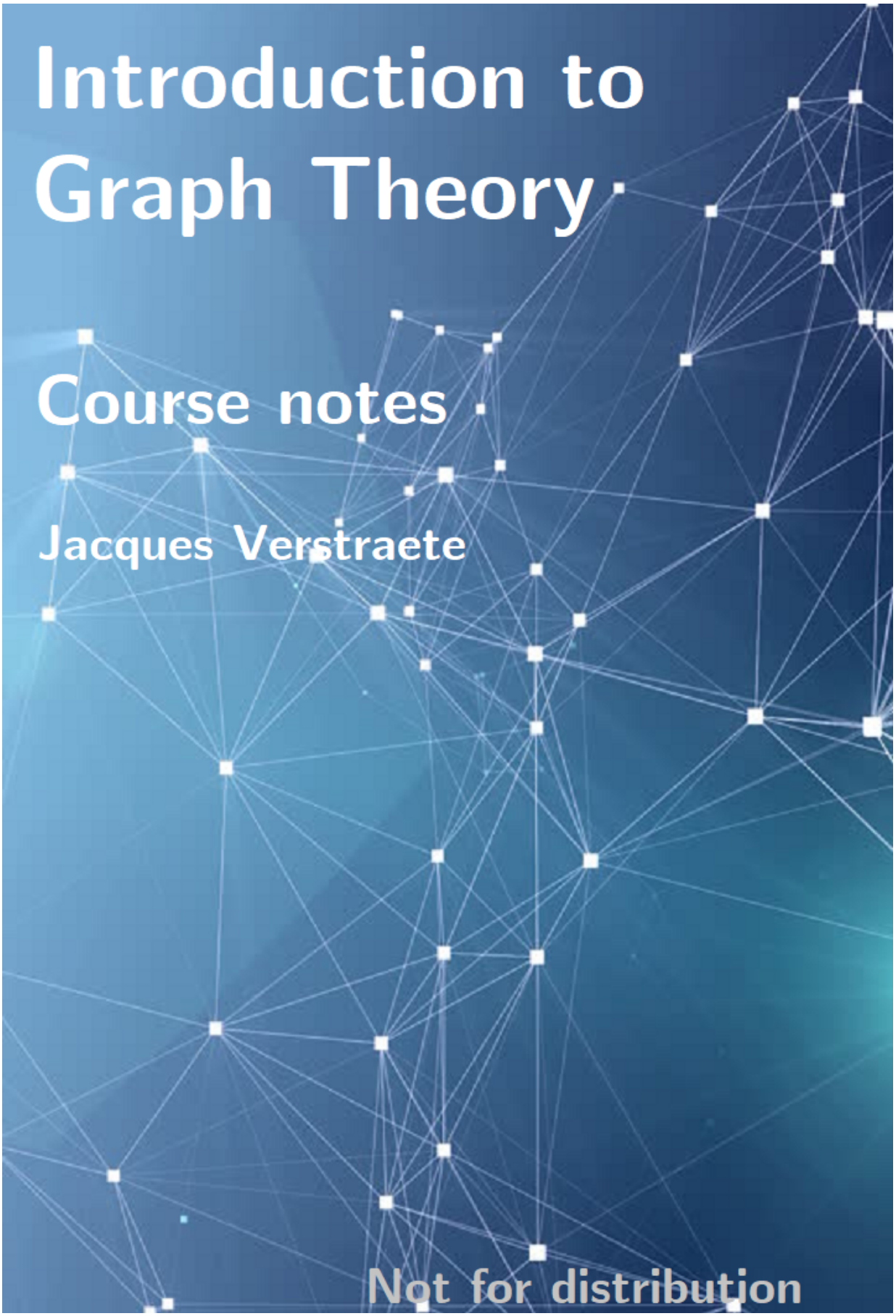
Introduction to Graph Theory

Course notes

Jacques Verstraete



Introduction to Graph Theory

A short course in graph theory at UCSD

February 3, 2021

Jacques Verstraete

Department of Mathematics
University of California at San Diego
California, U.S.A.

jacques@ucsd.edu

Contents

0 Annotation marks **5**

1 Introduction to Graph Theory **6**
1.1 Examples of graphs . . . 6
1.2 Graphs in practice* . . . 9
1.3 Basic classes of graphs . . . 15
1.4 Degrees and neighbourhoods . . . 16
1.5 The handshaking lemma . . . 17
1.6 Digraphs and networks . . . 18
1.7 Subgraphs . . . 19
1.8 Exercises . . . 21

2 Eulerian and Hamiltonian graphs **25**
2.1 Walks . . . 25
2.2 Connected graphs . . . 26
2.3 Eulerian graphs . . . 26
2.4 Eulerian digraphs and de Bruijn sequences . . . 28
2.5 Hamiltonian graphs . . . 30
2.6 Postman and Travelling Salesman Problems* . . . 32
2.7 Uniquely Hamiltonian graphs* . . . 33
2.8 Exercises . . . 35

3 Bridges, Trees and Algorithms **42**
3.1 Bridges and trees . . . 42
3.2 Breadth-first search . . . 43
3.3 Characterizing bipartite graphs . . . 46
3.4 Depth-first search . . . 47
3.5 Prim's and Kruskal's Algorithms . . . 47
3.6 Dijkstra's Algorithm* . . . 50
3.7 Exercises . . . 53

4 Structure of connected graphs **55**
4.1 Block decomposition* . . . 55
4.2 Structure of blocks : ear decomposition* . . . 57
4.3 Decomposing bridgeless graphs* . . . 60
4.4 Contractible edges* . . . 61
4.5 Menger's Theorems . . . 61
4.6 Vertex and edge connectivity . . . 65
4.7 Fan Lemma and Dirac's Theorem* . . . 66
4.8 Exercises . . . 69

5 Matchings and Factors **72**
5.1 Independent sets and covers . . . 72
5.2 Hall's Theorem . . . 73
5.3 Systems of distinct representatives . . . 75
5.4 Latin squares . . . 75
5.5 König-Ore Formula . . . 77
5.6 Tutte's 1-Factor Theorem . . . 77
5.7 Tutte-Berge Formula* . . . 80
5.8 Matching Algorithms . . . 81
5.9 Stable matchings* . . . 85
5.10 Exercises . . . 87

6 Vertex and Edge-Coloring **90**
6.1 König's Theorem . . . 91
6.2 Vizing's Theorem . . . 91
6.3 Brooks' Theorem . . . 93
6.4 Degenerate graphs . . . 94
6.5 Scheduling Problems . . . 95
6.6 Exercises . . . 96

7 Planar graphs **100**
7.1 Euler's Formula . . . 100
7.2 Platonic solids . . . 102
7.3 Coloring planar graphs . . . 103

7.4 Drawing planar graphs* . . . 105
7.5 The Art Gallery Theorem* . . . 106
7.6 Duality* . . . 108
7.7 Kuratowski's Theorem* . . . 110
7.8 Graphs on Surfaces* . . . 111
7.9 Exercises . . . 116

8 The Max-Flow Min-Cut Theorem 120
8.1 Flows . . . 120
8.2 Capacities . . . 121
8.3 Cuts . . . 121
8.4 Max-Flow Min-Cut Algorithm . . . 123
8.5 Proof of Hall's Theorem . . . 125
8.6 Proof of Menger's Theorems . . . 125
8.7 Exercises . . . 127

9 Introduction to Extremal Graph Theory* 130
9.1 Mantel's Theorem . . . 131
9.2 Turán's Theorem . . . 132
9.3 Kövari-Sós-Turán Theorem . . . 133
9.4 The Erdős-Gallai Theorem* . . . 134
9.5 Exercises . . . 136

A Appendix* 138
A.1 Sets and sequences . . . 138
A.2 Counting sets and sequences . . . 138
A.3 Multiplication and summation principles . . . 139
A.4 Inclusion-exclusion principle . . . 140
A.5 Bijections and combinatorial proofs . . . 140
A.6 Mathematical induction . . . 141
A.7 The pigeonhole principle . . . 141

Notation 143

Index 144

References 149

0 Annotation marks

- Statements marked ◁◁ in the margin are left to the reader to check/prove.

- Some section titles are starred (marked *). A one quarter class will typically cover most non-starred sections and optionally cover some starred sections, at the instructor's discretion.

- Questions are marked according to difficulty:

  Question 1.100. – Regular difficulty.
  Question 1.100* – Harder.
  Question 1.100° – Easier.

- For mathematical notation, see the Notation index at the end.

1 Introduction to Graph Theory

A ***graph*** G is a pair (V, E) where V is a set and E is a set of unordered pairs[1] of elements of V. The elements of V are called ***vertices*** and V is called the ***vertex set*** of the graph, and the elements of E are called ***edges***, and E is called the ***edge set*** of the graph. If G is a graph, we let $V(G)$ denote its vertex set and $E(G)$ its edge set. We note by $e(G)$ the number of edges in G. If u and v are two vertices of a graph $G = (V, E)$, then we say u and v are ***adjacent*** if $\{u, v\} \in E$ – in other words $\{u, v\}$ is an edge of G – and we say that vertex v is ***incident*** with edge e if $v \in e$. For convenience, since the edges are unordered pairs, it is traditional to write the edge $\{1, 2\}$ as the list 12. In general, it may be convenient to represent any graph $G = (V, E)$ by drawing V as a set of points in the plane, and draw a straight line between any two adjacent vertices in V.

We sometimes consider the following generalizations of graphs: a ***multigraph*** is a pair (V, E) where V is a set and E is a ***multiset*** of unordered pairs from V. In other words, we allow more than one edge between two vertices. A ***pseudograph*** is a pair (V, E) where V is a set and E is a ***multiset*** of unordered multisets of size two from V. A pseudograph allows ***loops***, namely edges of the form $\{a, a\}$ for $a \in V$. A ***digraph*** is a pair (V, E) where V is a set and E is a ***multiset*** of ordered pairs from V. In other words, the edges now have a direction: the edge (a, b) and edge (b, a) are different, and denoted in a digraph by putting an arrow from a to b or from b to a, respectively. An ***orientation*** of a graph G is a digraph $\vec{G}$ obtained by replacing each edge $\{a, b\} \in E(G)$ with either the arc (a, b) or the arc (b, a). The graph G is called the ***underlying graph*** of $\vec{G}$.

1.1 Examples of graphs

Example 1. Consider the graph $G = (V, E)$ where $V = \{1, 2, 3\}$ and $E = \{12, 13\}$. Then the drawing below represents this graph:

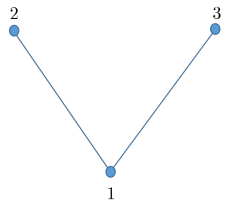


Figure 1.1: The graph $G = (\{1, 2, 3\}, \{12, 13\})$

[1] We denote sets using braces, for instance $\{1, 2, 3\}$ is the set whose elements are $1, 2$ and 3, and we write $1 \in \{1, 2, 3\}$ to say "1 is an element of the set $\{1, 2, 3\}$." Note that a set precludes "repeated elements".

Example 2. Let $V = \{p_1, p_2, p_3, p_4, p_5, p_6\}$ be a set of six people at a party, and suppose that p_1 shook hands with p_2 and p_4, p_3 shook hands with p_4, p_5 and p_6, and p_5 and p_6 shook hands. Let $G = (V, E)$ be the graph with edge set E consisting of pairs of people who shook hands. Then

$$E = \{p_1p_2, p_1p_4, p_3p_4, p_3p_5, p_3p_6, p_5p_6\}.$$

A drawing of G is given in Figure 1.2 below:

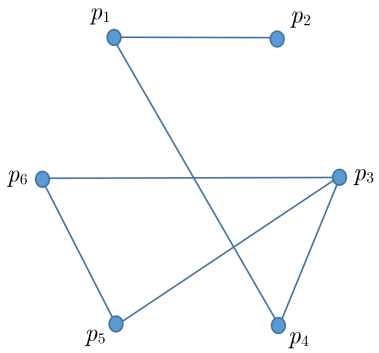


Figure 1.2: The handshake graph G

Example 3. Let $\mathbb{Z}$ denote the set of integers[2] and let

$$V = \{(x, y) \in \mathbb{Z} \times \mathbb{Z} : 0 \leq x \leq 2, 0 \leq y \leq 2\}.$$

Then V is just the set of points in the plane with integer co-ordinates between 0 and 2. Now suppose $G = (V, E)$ is the graph where E is the set of pairs of vertices of V at distance 1 from each other. In other words, (x, y) and (x', y') are adjacent if and only if $(x-x')^2+(y-y')^2 = 1$. We check that the edge set is

$$\begin{aligned} E \;=\; & \{(0,0)(0,1), (0,0)(1,0), (0,1)(0,2), (1,0)(2,0), (1,0)(1,1), (1,1)(1,2), \\ & (1,1)(2,1), (0,1)(1,1), (0,2)(1,2), (2,0)(2,1), (2,1)(2,2), (1,2)(2,2)\}. \end{aligned}$$

This is a cumbersome way to write the edge set of G, as compared to the drawing of G in Figure 1.3 below, which is much easier to absorb.

[2]Thus $\mathbb{Z} = \{\ldots, -2, -1, 0, 1, 2, 3, \ldots\}$. Then $\mathbb{Z} \times \mathbb{Z}$ is the ***Cartesian product***, which is the set of pairs (x, y) such that $x \in \mathbb{Z}$ and $y \in \mathbb{Z}$.

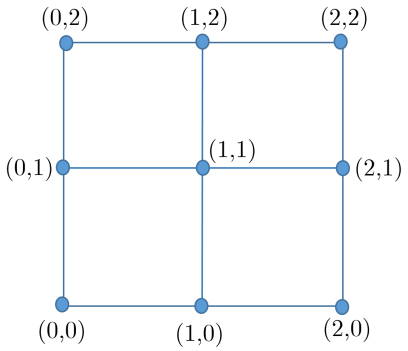


Figure 1.3: The grid graph G

Example 4. Let V be the set of binary strings of length three, so

$$V = \{000, 001, 010, 100, 011, 101, 110, 111\}.$$

Then let E be the set of pairs of strings which differ in one position. Then

$$E = \{\{000, 001\}, \{010, 000\}, \{100, 000\}, \ldots, \{111, 101\}, \{111, 110\}, \{111, 011\}\}.$$

The reader should fill in the rest of the edges as an exercise. Once again, this graph Q $\lhd\lhd$ actually has a very nice drawing (which explains why it is sometimes called the cube graph).

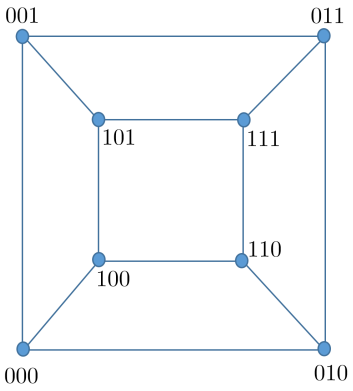


Figure 1.4: The cube graph Q

Example 5. Let G be the graph with vertex set $V = \{v_1, v_2, v_3, v_4, v_5, v_6, v_7\}$ and edge set

$$E = \{v_1v_4, v_1v_7, v_2v_3, v_2v_6, v_2v_7, v_3v_4, v_3v_5, v_3v_7, v_4v_5, v_4v_6, v_5v_6, v_6v_7\}.$$

In Figure 1.5, two drawings of G are shown (the reader should verify that they are both drawings of G).

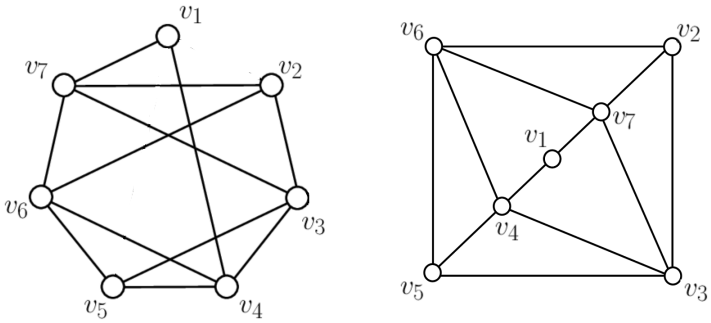


Figure 1.5: Two drawings of a graph with seven vertices

1.2 Graphs in practice*

Graphs appear in many theoretic and practical applications, including statistical physics, chemistry, broadcasting and networks, circuit design, computational complexity, coding and information theory, algorithm design, probability theory and markov chains, algebra, number theory and geometry, to mention a few. We give a few examples in this section:

The web graph. Let V denote the set of websites on the internet, and E the set of pairs of websites which are linked. The web graph is growing all the time, and due to its size, difficult to analyze. In Figure 1.6, two ***induced subgraphs*** of the web graph are shown.

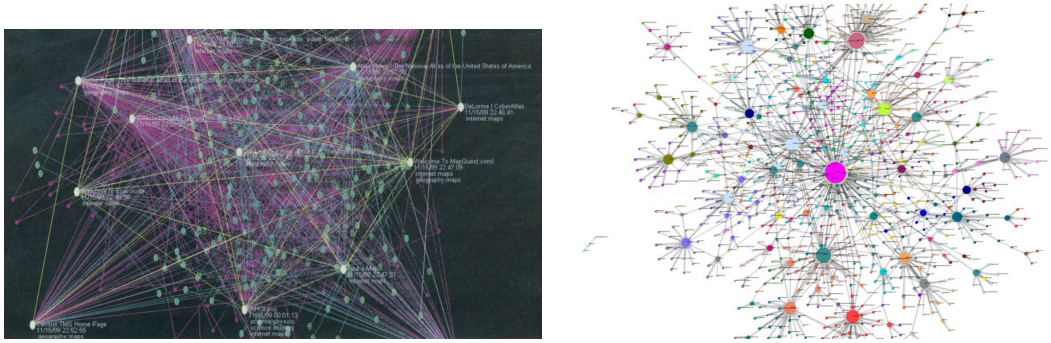


Figure 1.6: Induced subgraphs of the web graph

Natural questions related to searching are whether the web graph is ***connected***, the ***radius*** and ***diameter*** of the web graph, and so on. A famous graph-theoretic ingredient for searching the web is ***PageRank*** – see the book by Bonato [6].

Planar graphs and geometry. [Notes Part 7] A graph is planar if it can be "drawn" in the plane or on a sphere without any edges crossing. If we consider an abstract map, then we may represent it as a planar graph by representing each country by a vertex, and drawing an edge between countries which share a border. If we consider a three-dimensional polyhedron, then it has a natural embedding on a sphere without crossing edges. Similarly, we can consider planar lattices such as the integer lattice, hexagonal lattice (honeycomb lattice) and triangular lattice in the Euclidean plane.

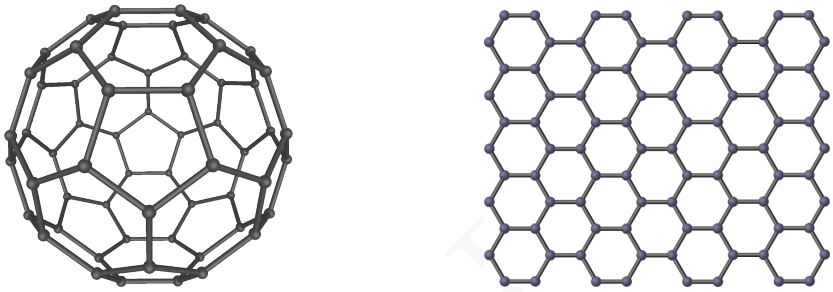


Figure 1.7: Carbon C_{60} fullerene and hexagonal lattice

One of the famous problems in graph theory is to color the regions of a map (in other words, color the vertices of a planar graph) so that no two adjacent regions receive the same color. A coloring of the world map with four colors is shown below:

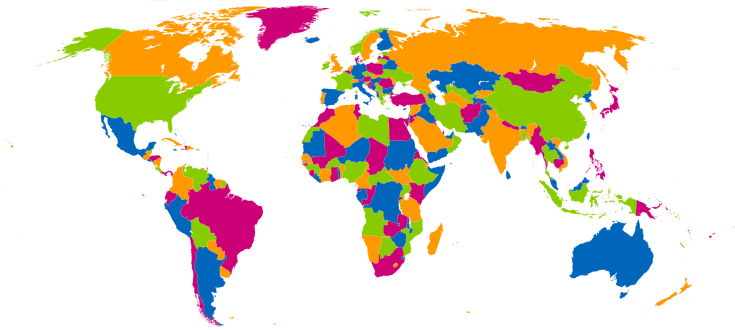


Figure 1.8: 4-Coloring of the world map

The famous ***4-color theorem*** says every map can be colored with at most four colors so that no adjacent countries have the same color. This was proved by Appel and Haken in [2] and [3] using the aid of a computer, and more recently by Robertson and Seymour [36].

The integer lattice is an example of a ***unit distance graph***: a graph whose vertices are points in the plane and whose edges are pairs of points at distance 1. Other examples of unit distance graphs are shown below. The big open problem of Erdős [13] is to determine the maximum number of edges in an n-vertex unit distance graph. The problem of determining the minimum number of distinct distances between n points in the plane [13] was recently solved asymptotically by Guth and Katz [20]. A famous open problem is to determine the minimum k such that there exists a map $f : \mathbb{R}^2 \to \{1, 2, \ldots, k\}$ such that whenever $x, y \in \mathbb{R}^2$ are at distance one, $f(x) \neq f(y)$. This is the ***Hadwiger-Nelson problem*** of finding the ***chromatic number*** of the plane, denoted $\chi(\mathbb{R}^2)$ – we are coloring an infinite unit distance graph whose vertex set is $\mathbb{R}^2$ and whose edge set is $\{\{x, y\} \subset \mathbb{R}^2 : d(x, y) = 1\}$. It was shown recently [11] that $\chi(\mathbb{R}^2) \geq 5$, whereas it is well-known that $\chi(\mathbb{R}^2) \leq 7$.

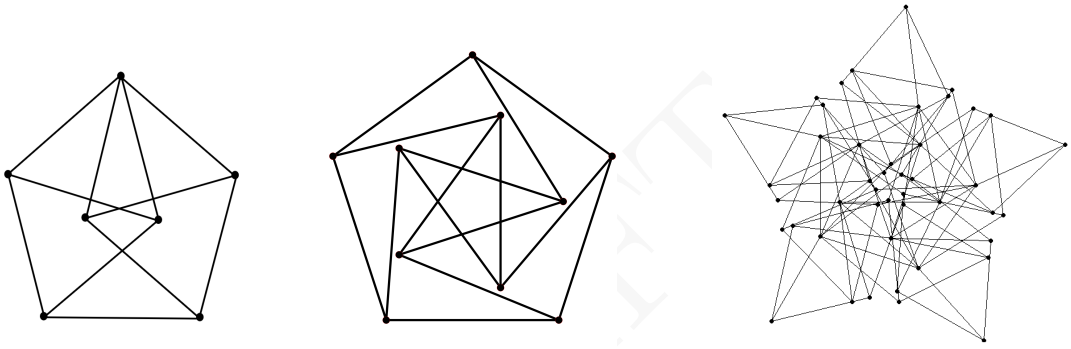


Figure 1.9: Unit distance graphs

Connectivity and matchings. [Notes Parts 2 and 3] Given a graph G, how many vertices or edges must be removed to disconnected the graph (split it into connected pieces)? This is the fundamental connectivity problem in graphs, addressed by Menger's Theorems [31]. Given a graph G, can we find a set of pairwise vertex-disjoint edges covering all the vertices (a ***perfect matching***) in the graph? This is addressed by ***Hall's Theorem*** for bipartite graphs [21], and ***Tutte's 1-Factor Theorem*** in general graphs [42]. Furthermore, the maximum matching can be found efficiently. The maximum matching problem, for instance, is very natural in practical applications, such as scheduling and job assignment. Given a set $A = \{a_1, a_2, \ldots, a_k\}$ of people and a set $B = \{b_1, b_2, \ldots, b_l\}$ of jobs, and for each person a list of jobs in B that they can do, we would like to assign as many people to jobs without having one job done by two people or two jobs done by one person. The natural graph has vertex set $A \cup B$, where a_i is joined to b_j by an edge if a_i can do job b_j. Then we are asking for a maximum matching, and an efficient algorithm exists, even if we put a weight on each edge $\{a_i, b_j\}$ to denote how much a_i would like to do job b_j. An example is shown below:

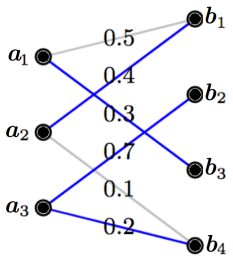


Figure 1.10: Maximum matching

The book by Lovász and Plummer [27] is an authority on the theory of matchings in graphs.

Flows in networks. Let $G = (V, E)$, and let $s, t \in V$ be vertices designated as ***source*** and ***sink***. Suppose each edge of the graph has a direction and a ***capacity***, denoting the maximum number of units of fluid that the edge can carry between its ends. If fluid flows through the network from the source to the sink, we assume that the flow in to each vertex other than s or t is equal to the flow out of the vertex. Given the capacities, the question is the maximum flow can be transmitted from s to t (flow occurs simultaneously in all edges). This is completely answered by the ***Max-Flow Min-Cut Theorem*** of Ford and Fulkerson [17], together with an efficient algorithm for finding a maximum flow (see the book by Ford and Fulkerson on flows in networks [18]). Many generalizations of this theorem exist, and it is a special case of duality in linear programming. The theorem has wide applicability, and the Max-Flow Min-Cut Theorem implies the afore-mentioned Hall's and Menger's Theorems on matchings and connectivity.

Random graphs. The classical ***Erdős-Rényi model*** of random graphs takes n vertices and then for each pair of vertices, we place an edge with probability p and no edge with probability $1 - p$ (in other words, the edge set are decided by $\binom{n}{2}$ coin flips). In this way we generate a graph $G_{n,p}$. When $p = 0$, it is the empty graph, and when $p = 1$, it is the complete graph. In the figure below, we show examples of $G_{64,p}$ when $p \in \{0, \frac{1}{256}, \frac{1}{64}, \frac{1}{16}, \frac{1}{4}, 1\}$.

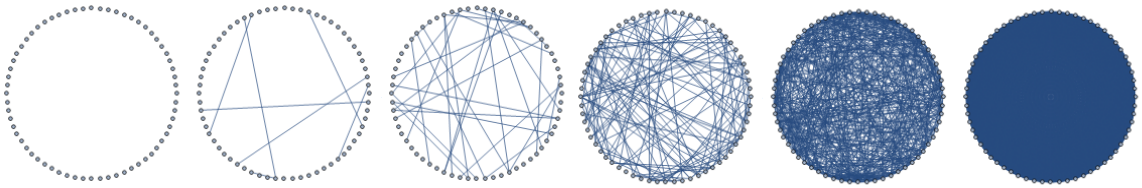


Figure 1.11: Random graphs

There are very many other types of random graphs, for example the ***preferential attachment*** graph used to model the web graph, or ***random regular graphs***, or ***random***

geometric graphs. The graph below is a random geometric graph in the unit square: the vertices are uniformly randomly chosen points in the unit square, and the edges correspond to pairs of points at most a certain distance from each other. Random graphs are beyond the scope of this course. The books by Bollobás [4] and Janson, Łuczak and Rucinski [22] are sources on the theory of random graphs, and Penrose studies random geometric graphs [33].

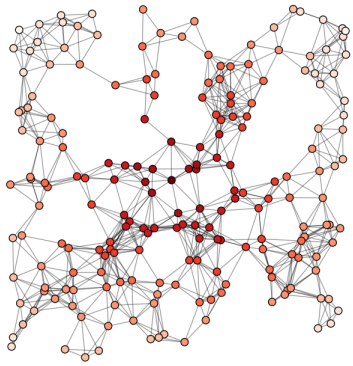


Figure 1.12: Random geometric graph

Percolation and automata. Let G be the graph whose vertex set is a set of organisms, and put an edge between two organisms if they can communicate a virus between them. For each vertex v in the graph, let $r(v)$ denote the minimum number of infected neighbors of v required in order for v to become infected. If X is the set of vertices initially infected, one may ask whether the infection spreads to the entire graph. This clearly depends on the graph, and in particular ***connectivity*** of the graph. In addition, perhaps after a certain time a vertex v becomes uninfected, and the same question remains. In fact, this is a very restricted instance of the famous ***Conway's game of life***. The game of life is on cells of the integer lattice, according to the following rules, with cells being in two states, infected or dormant:

- Infected cells with at most one/more than three infected neighbors becomes dormant
- Dormant cells with exactly three infected neighbors becomes infected.

In all other cases, the cells preserve their state. The question is whether the infection dies out or spreads forever, and what the set of infected cells looks like at any time. For example, if the cells initially infected form the white cells in the left frame of the picture below, it takes 130 generations for the infection to die out. Some of these generations are shown in the figure.

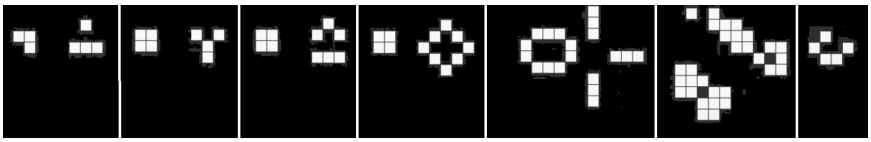


Figure 1.13: Conway's game of life

These kinds of questions fall into the realm of ***percolation on graphs*** and ***cellular automata***, which we do not study in this course. The books by Grimmett [19] and Bollobás and Riordan [5] offer comprehensive studies of percolation.

Coding and information theory. For the purpose of this remark, a ***code*** is a set C of binary strings called ***codewords***. A message is sent by first encoding it using a binary string, sending it over a channel, and then the receiver decodes the message. Both encoding and decoding should be done efficiently, while the channel may be noisy and bits may be corrupted or deleted, so the challenge is to design the code so that the original message can be recovered. A ***t-error-correcting code*** is a code C such that if a binary string $c \in C$ is sent over the channel and t bits are corrupted, so that a binary string c' is received, then c can be uniquely recovered. An easy guarantee is that any two distinct codewords $c_1, c_2 \in C$ differ in at least $2t+1$ positions : we say that C has ***minimum distance*** at least $2t+1$. By constructing certain graphs called ***expander graphs***, one can build good error-correcting codes : these codes are binary strings of length n that are able to correct linearly many errors (due to the minimum distance). We have touched extremely briefly on a point in coding theory, which is the tip of the iceberg. Two relevant books on coding theory are van Lint [43] and McEliece [30].

Search algorithms. Let G be a graph, and let v be a vertex of G. A person starts at an arbitrary vertex $u \in V(G)$, and wishes to walk to v in as few steps as possible. This in itself is not a hard problem to solve, via ***Dijkstra's Shortest Path Algorithm***. Two twists on the problem are (1) the use of only local information and (2) the addition of a few extra edges to speed up the walk. Local information means each vertex has only the information as to a direction which leads closer to the destination vertex v. The addition of an edge $\{u, w\}$ comes with the local information at u and at w. For a concrete example, let G be the n by n square grid graph. The typical distance between two vertices is roughly n, so it may take n steps to get from u to v. Can we add few edges (for instance, add one edge per vertex) so that the number of steps drops dramatically? Kleinberg [24] showed that using randomness, one can decrease the number of steps to roughly $(\log n)^2$, and called the algorithm ***decentralized search***. This is an example of a highly graph theoretic search algorithm, and it has been adapted to detect given a general graph G whether it is possible to speed up the search in a similar way.

1.3 Basic classes of graphs

There are some graphs which we shall encounter very frequently, and we describe these here.

Complete Graphs. The ***complete graph*** or ***clique*** on n vertices, denoted K_n is the graph consisting of all possible edges on n vertices (in other words, every pair of vertices is adjacent). The ***empty graph*** on n vertices has no edges. In Figure 1.14, drawings of K_n for $2 \leq n \leq 6$ are given:

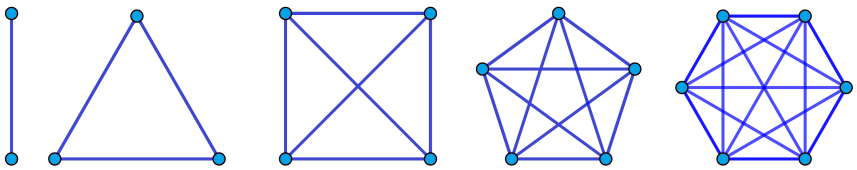


Figure 1.14: The complete graphs K_2 through K_6

Since the number of pairs of vertices in K_n is $\binom{n}{2}$, and every pair is an edge, the number of edges in K_n is $\binom{n}{2}$.

Bipartite graphs. Recall a ***partition*** of a set V consists of pairwise disjoint non-empty subsets whose union is V. A ***bipartite graph*** is a graph $G = (V, E)$ such that for some partition of V into two sets A and B such that every edge of G has the form $\{a, b\}$ with $a \in A$ and $b \in B$ (or in other words, no two vertices in A are adjacent, and no two vertices in B are adjacent). We call A and B the ***parts*** of G and refer to (A, B) as the ***bipartition of*** G. When $|A| = r$ and $|B| = s$ and all possible edges $\{a, b\}$ with $a \in A$ and $b \in B$ are included, then G is called the ***complete bipartite graph***, and denoted $K_{r,s}$. In Figure 1.15, we draw the graphs $K_{2,3}$ and $K_{2,5}$.

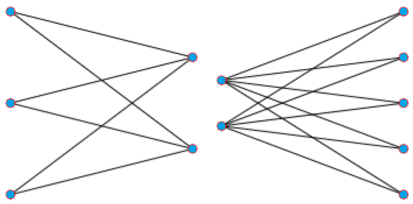


Figure 1.15: Complete bipartite graphs $K_{2,3}$ and $K_{2,5}$

Note that the number of edges in a complete bipartite graph $K_{r,s}$ is exactly rs.

For $k \geq 3$, a ***k-cycle*** is the graph C_k with vertex set $\{1, 2, \ldots, k\}$ and edge set

$$\{\{1,2\},\{2,3\},\{3,4\},\ldots,\{k-1,k\},\{k,1\}\}.$$

For $k \geq 1$, a ***k-path*** is the graph P_k with vertex set $\{1, 2, \ldots, k+1\}$ and edge set

$$\{\{1,2\},\{2,3\},\{3,4\},\ldots,\{k-1,k\},\{k,k+1\}\}.$$

Note that a k-cycle has k edges and a k-path has k edges, and we often refer to the number k as the ***length*** of the cycle or path. It is convenient to represent a path as a sequence of vertices, for instance $v_1v_2\ldots v_k$ represents a path with k vertices consisting of the edges v_iv_{i+1} for $i < k$. Similarly, a cycle can be represented as a circular sequence $v_1v_2\ldots v_kv_1$. In Example 1, P_2 is drawn, and in Figure 1.16, we draw C_3 and C_6.

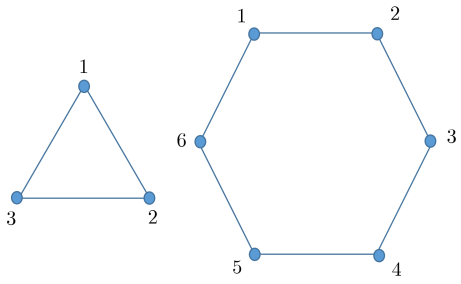


Figure 1.16: Cycles C_3 and C_6

1.4 Degrees and neighbourhoods

The ***neighborhood*** of a vertex v in a graph $G = (V, E)$, denoted $N_G(v)$, is the set of vertices of G which are adjacent to v. The ***degree*** of a vertex v in a graph G, denoted $d_G(v)$, is $|N_G(v)|$. When it is clear which graph G we are referring to, we write $d(v)$ and $N(v)$ instead of $d_G(v)$ and $N_G(v)$. The ***degree sequence*** of a graph G is the sequence of degrees of vertices of G in non-increasing order. For example, the degree sequence of the graph in Figure 1.1 is $(2, 1, 1)$, whereas the degree sequence of the graph in Figure 1.3 is $(4, 3, 3, 3, 3, 2, 2, 2, 2)$. A vertex of degree zero is called an ***isolated vertex***.

We write $\delta(G) = \min\{d_G(v) : v \in V\}$ and $\triangle(G) = \max\{d_G(v) : v \in V\}$ for the ***minimum degree*** and ***maximum degree*** of G, respectively. For the examples in the last section, we note $\delta(G) = 1$ and $\triangle(G) = 2$ for Figure 1.1, $\delta(G) = 2$ and $\triangle(G) = 4$ for Figure 1.3, $\delta(G) = 1$ and $\triangle(G) = 3$ for Figure 1.2, and $\delta(Q) = \triangle(Q) = 3$ for the cube graph in Figure 1.4. If all vertices in a graph have the same degree r, then the graph is said to be ***r-regular***.

For instance, the graph Q is 3-regular (all the degrees are 3). Sometimes, 3-regular graphs are also referred to as ***cubic*** graphs.

If G is a pseudograph, then the degree $d_G(v)$ of a vertex $v \in V(G)$ equals the number of edges $\{v, w\}$ with $w \neq v$ plus twice the number of loops $\{v, v\}$. The neighborhood of v is $N_G(v) = \{w \in V(G) : \{v, w\} \in E(G)\}$, however, for multigraphs G it is not true in general that $d_G(v) = |N_G(v)|$.

1.5 The handshaking lemma

An important fact involving the degrees of a graph G, which we will use on numerous occasions, is the ***handshaking lemma***:

Lemma 1.5.1 (Handshaking Lemma) *For any graph $G = (V, E)$,*

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

Proof ▷ When we add up the degrees of vertices of G, every edge of G is counted twice, so the sum of the degrees is twice the number of edges. □

The handshaking lemma gives an easy way to count the number of edges in a graph: it is just half the sum of the degrees of the vertices. The reader may verify that the same holds for pseudographs G, if we define the degree of a vertex $d_G(v)$ of a vertex $v \in V(G)$ to be the number of edges $\{v, w\}$ with $w \neq v$ plus twice the number of loops $\{v, v\}$. Note if G is r-regular and has n-vertices, then the number of edges in G is $nr/2$, by the handshaking lemma (check this for the cube graph Q in the last section). A consequence of the handshaking lemma is that the number of vertices of odd degree in any graph must be even – otherwise the sum on the left above would be odd whereas the right hand side is even:

Lemma 1.5.2 *For any graph $G = (V, E)$, the number of vertices of odd degree is even.*

The reader may check that this is satisfied for the graphs in Examples 1 – 4. Consider the complete graph K_n. Every vertex of K_n is adjacent to every other vertex of K_n, so the degree of every vertex of K_n is $n - 1$ – in other words, K_n is $(n-1)$-regular. By the handshaking lemma, the number of edges in K_n is $\frac{1}{2} \cdot n \cdot (n-1) = \binom{n}{2}$, as we already knew. Next, consider Figure 1.3 in the last section (the grid graph). The degree sequence of this graph is $(4, 3, 3, 3, 3, 2, 2, 2, 2)$. Therefore by the handshaking lemma, the number of edges in the grid graph is ◁◁

$$\frac{1}{2}(4 + 3 + 3 + 3 + 3 + 2 + 2 + 2 + 2) = 12.$$

A manual count of the edges in Figure 1.3 confirms this. The reader should check how many edges the n by n grid graph has (the vertex set is $V = \{(x, y) \in \mathbb{Z} \times Z : 0 \leq x < n, 0 \leq y < n\}$ and the edge set is the set of pairs of vertices at distance 1 from each other.) ◁◁

Example 6. The ***n-cube***, denoted Q_n, is the graph whose vertex set is the set of binary strings of length n, and whose edge set consists of all pairs of strings differing in one position. The cube graph Q_3 in Example 4 is the 3-cube. Let us see how many edges Q_n has as a formula in n. Since there are 2^n binary strings of length n, there are 2^n vertices in Q_n. Now each vertex v is adjacent to n other vertices – namely flip one position in the string v to get each string adjacent to v, and there are n possible positions in which to do a flip. So every vertex of the n-cube has degree n (in other words, it is n-regular), and so the number of edges in Q_n is

$$\frac{1}{2}\sum_{v \in V} d_{Q_n}(v) = \frac{1}{2} \cdot 2^n \cdot n = n2^{n-1}.$$

A manual count of the edges confirms this for Q_4, which is drawn below:

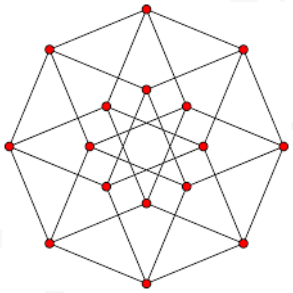


Figure 1.17: The 4-cube Q_4

1.6 Digraphs and networks

A ***digraph*** or ***network*** is a pair $\vec{G} = (V, \vec{E})$ where V is a set and $\vec{E}$ is a multiset of ordered pairs of elements of V, which we refer to in this section as ***arcs***. Note that two vertices can be joined by many arcs in either direction, and we allow ***loops***: a vertex may have an arc to itself. In a digraph $\vec{G} = (V, \vec{E})$, let $N^+(v)$ and $N^-(v)$ denote the sets of vertices adjacent ***from*** v and ***to*** v, respectively. These are the ***out-neighborhood*** of v and the ***in-neighborhood*** of v respectively. Thus

$$N^+(v) = \{u : (v, u) \in \vec{E}\} \qquad N^-(v) = \{u : (u, v) \in \vec{E}\}.$$

For example, in the digraph drawn below, we have $N^+(x) = \{u, v, w\}$, $N^-(x) = \{v\}$.

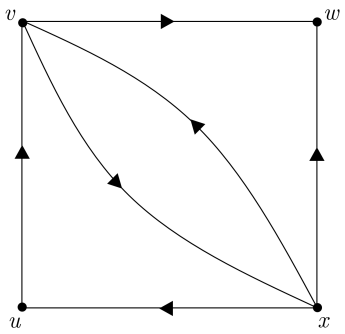


Figure 1.18: A digraph

The ***in-degree*** of a vertex v is $d^-(v) = |N^-(v)|$ and the ***out-degree*** is $d^+(v) = |N^+(v)|$.

1.7 Subgraphs

If H and G are graphs and $V(H) \subseteq V(G)$ and $E(H) \subseteq E(G)$, then H is called a ***subgraph*** of G. To denote that H is a subgraph of G, we write $H \subseteq G$. If in addition $V(H) = V(G)$ then H is called a ***spanning subgraph*** of G.

Example 7. For instance, the reader will check that the graph $G = P_2$ shown in Figure 1.1 is a subgraph of the graphs in Figures 1.2 – 1.4. The graph in Figure 1.2 is not a subgraph of any of the others, since it contains a triangle but none of the others contains a triangle. The graph G in Figure 1.3 is not a subgraph of the cube graph Q in Figure 1.4 since it has a vertex of degree four, whereas Q is 3-regular. We note that every graph with at most n vertices is a subgraph of K_n, and every graph with n vertices is a spanning subgraph of K_n. The path P_{k-1} is a spanning subgraph of C_k. ◁◁

We now define how to remove edges and vertices from a graph G. If X is a set of vertices of G, we denote by $G - X$ the graph with vertex set $V(G)\backslash X$ and edge set $E = \{e \in E(G) : e \cap X = \emptyset\}$. If $L \subseteq E(G)$, we denote by $G - L$ the graph with vertex set $V(G)$ and edge set $E(G)\backslash L$. Similarly, if L is a set of pairs vertices in G, then we denote by $G + L$ the graph with vertex set $V(G)$ and edge set $E(G) \cup L$. When $L = \{e\}$ for some pair e of vertices, then we write $G - e$ and $G + e$ instead of $G - \{e\}$ and $G + \{e\}$. We let $e(U, V)$ denote the number of edges of a graph with one end in U and one end in V.

Example 8. For instance, if we remove one edge e from a cycle C_k, we get the path P_{k-1}, which we write as $C_k - e = P_{k-1}$. If we remove one vertex v from a cycle C_k, we get the path P_{k-2}, which we write as $C_k - v = P_{k-2}$. If we remove the vertex 1 from the graph in Figure 1.1, we get a graph consisting of two ***isolated vertices***. If we remove $X = \{101, 100, 111, 110\}$

from the graph Q in Figure 1.4, we get C_4, so we may write $Q - X = C_4$. If instead we remove $X = \{001, 101, 110\}$ we get the graph shown below in Figure 1.19:

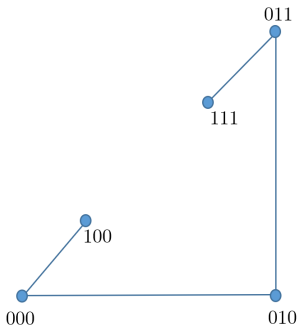


Figure 1.19: The graph $Q - \{001, 101, 110\}$

The subgraph of G ***induced*** by a set $X \subseteq V(G)$, denoted $G[X]$, is precisely $G - (V \backslash X)$. A subgraph H of G is an ***induced subgraph*** if for some $X \subseteq V(G)$, $H = G[X]$. If L is a set of edges of G, then the subgraph of G ***spanned by*** L is the graph with edge set L and vertex set $\bigcup_{e \in L} e$. The graph in Figure 1.19 is an induced subgraph of the cube graph Q_3, whereas P_{k-1} is not an induced subgraph of C_k. ◁◁

A more sophisticated operation on graphs is ***contraction***. If G is a graph and $X \subset V(G)$ and $Y = V(G) \backslash X$, then the ***contraction of*** X is the graph G/X obtained from $G - X$ by adding a new vertex x and all edges between x and $N(X)$. An example is shown below, where $X = \{1, 2, 3, 5, 7\}$.

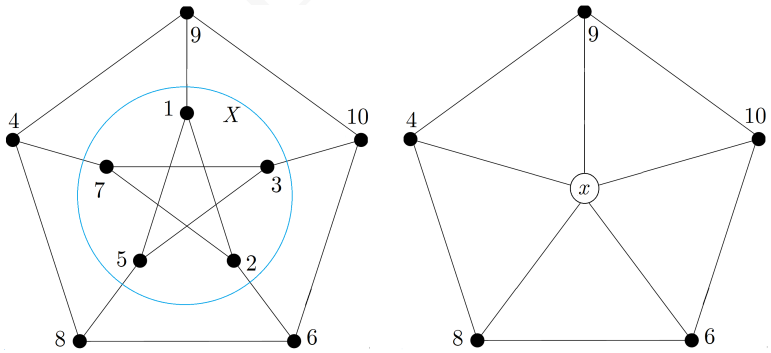


Figure 1.20: Contraction of $X = \{1, 2, 3, 5, 7\}$ to x

1.8 Exercises

Question 1.1° In Figure 1.21, a graph G with nine vertices is shown.

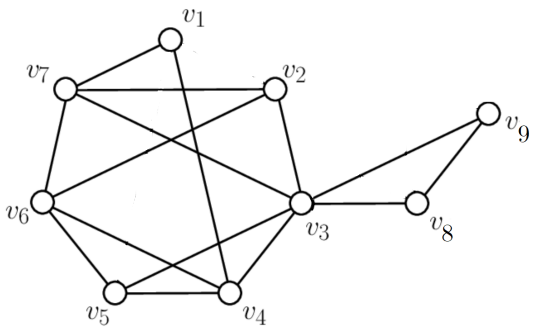


Figure 1.21: A graph with nine vertices

(a) How many edges does G have?
(b) Write down $N(v_1), N(v_5), d(v_1)$ and $d(v_5)$, and $\delta(G)$ and $\triangle(G)$.
(c) If $X = \{v_3, v_6\}$, how many edges does $G - X$ have?
(d) How many components does $G - \{v_1, v_3, v_7\}$ have?
(e) Are K_4, $K_{1,7}$ and $K_{2,3}$ subgraphs of G?
(f) What is the length of a longest cycle in G? What about a longest path?
(g) Draw G/Y where $Y = \{v_1, v_4, v_5, v_6, v_7\}$.

Question 1.2° For $n \geq 4$, a ***wheel graph*** W_n with n vertices consists of a cycle of length $n-1$ plus a vertex adjacent to every vertex in the cycle. Are any of the graphs in Figure 1.22 a drawing of the wheel graph W_7? If the graph is a drawing of W_7, label the vertices $v_1, v_2, \ldots, v_7$ so that the edges are $\{v_2, v_3\}$, $\{v_3, v_4\}$, ... , $\{v_7, v_2\}$, and $\{v_1, v_i\} : 2 \leq i \leq 7$. If the graph is not a drawing of W_7, prove that it is not a drawing of W_7.

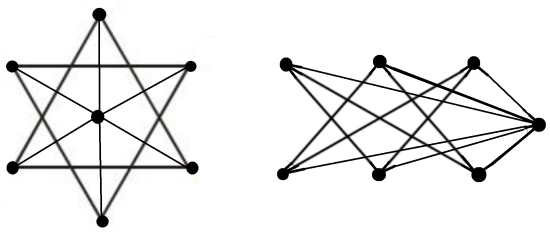


Figure 1.22: Two graphs with seven vertices

Question 1.3° For $n \geq 2$, let G_n be the ***grid graph***, whose vertex set is

$$V = \{(x, y) \in \mathbb{Z} \times \mathbb{Z} : 0 \leq x < n, 0 \leq y < n\}$$

and whose edge set is

$$E = \{\{(x, y), (x', y')\} : (x - x')^2 + (y - y')^2 = 1\}.$$

Determine the number of vertices and number of edges in G_n for each $n \geq 2$.

Question 1.4° In Figure 1.23, the ***wheel graphs*** W_n with n vertices are shown for $4 \leq n \leq 11$.

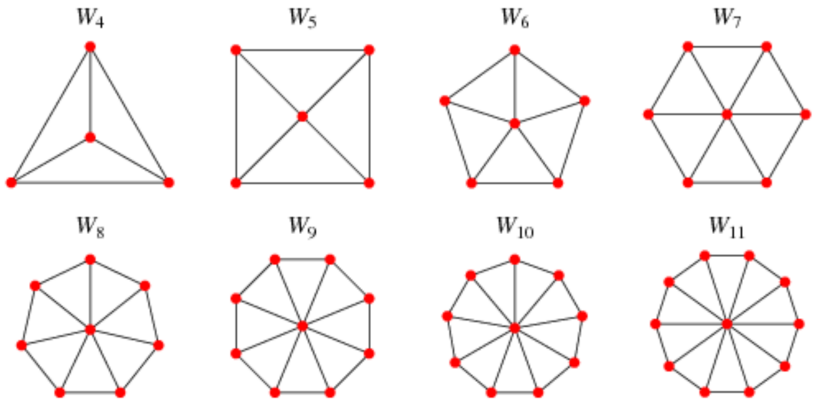


Figure 1.23: Wheel graphs W_n

(a) Write down the minimum and maximum degree of W_n for all $n \geq 4$.
(b) Write down the number of edges in W_n for all $n \geq 4$.
(c) For which $n \geq 4$ is $K_n \subset W_n$?
(d) Is there a spanning cycle in W_n for $n \geq 4$?
(e) How many cycles does W_n have for $n \geq 4$?
(f) For which m and n is W_m a subgraph of W_n?
(g) Is W_m ever an induced subgraph of W_n?

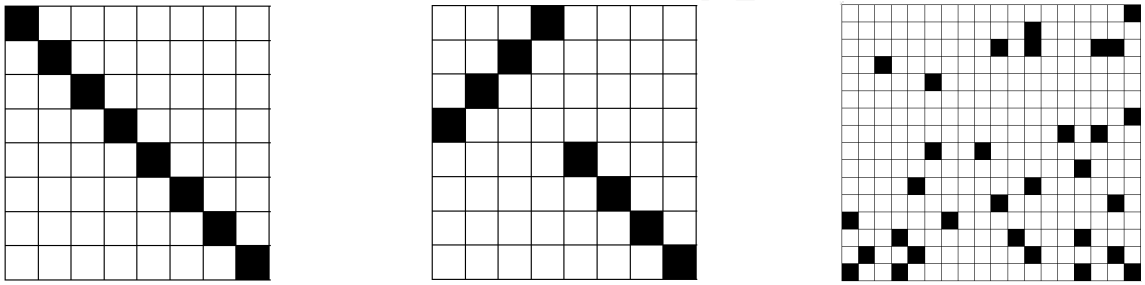
Question 1.5° Let G be a digraph such that every vertex has positive in-degree. Prove that G contains a ***directed cycle*** – a digraph with vertex set $\{v_1, v_2, \ldots, v_k\}$ and edges $(v_1, v_2), (v_2, v_3), \ldots, (v_{k-1}, v_k), (v_k, v_1)$.

Question 1.6. The ***line graph*** of a graph $G = (V, E)$ is the graph $L(G) = (E, F)$ whose vertex set is E and whose edge set is

$$F = \{\{e, f\} \subset E : e \cap f \neq \emptyset\}.$$

(a) Draw $L(C_4)$ and $L(K_4)$.
(b) How many edges does $L(G)$ have in terms of the degrees $d(v) : v \in V(G)$ of the vertices of G?

Question 1.7° Suppose initially a set I of squares in the grid is infected with a virus, and that at any stage in time, a square becomes infected if it has at least two infected neighbors (sharing two or more sides with infected squares). Determine for which sets I in the pictures below the virus (infected squares are black squares) spreads to the entire grid.



Question 1.8. Let G be a graph whose vertex set is a set $V = \{p_1, p_2, \ldots, p_6\}$ of six people. Prove that there exist three people who are all friends with each other, or three people none of whom are friends with each other.

Question 1.9. Let $K_{n:r}$ denote the ***Kneser graph***, whose vertex set is the set of r-element subsets of an n-element sets, and where two vertices form an edge if the corresponding sets are disjoint.

(a) Describe $K_{n:1}$ for $n \geq 1$.
(b) Draw $K_{4:2}$ and $K_{5:2}$.
(c) Determine $|E(K_{n:r})|$ for $n \geq 2r \geq 1$.

Question 1.10.

(a) Prove that every graph with at least two vertices contains two vertices with the same degree.

(b) Is (a) true for multigraphs?

(c) For each $n \geq 2$ give an example of a graph with n vertices which does not have three vertices of the same degree.

Question 1.11* Let G be an n-vertex digraph such that

$$|N^+(v)| > \frac{1}{2}(3 - \sqrt{5})n$$

for every $v \in V(G)$. Prove that G contains a directed cycle of length two or three.

Question 1.12* Consider n people possessing unique items $u_1, u_2, \ldots, u_n$ of information that they wish to share with each other. Two people can call each other and share all the items of information they currently have.

(a) For $n \leq 4$, determine the minimum number of calls that can be made so that all information is shared amongst all n people.

(b) Prove that for $n \geq 5$ the minimum number of calls so that all n people have all items of information is $2n - 4$.

2 Eulerian and Hamiltonian graphs

2.1 Walks

A ***walk*** in a graph $G = (V, E)$ is an alternating sequence of vertices and edges, whose first and last elements are vertices, and such that each edge joins the vertices immediately preceding it and succeeding it in the sequence. For example,

$$a\{a,d\}d\{d,e\}e\{e,a\}a\{a,d\}d$$

is a walk in the graph in Figure 2.1. Since there is no ambiguity, we denote a walk by a sequence of vertices, so the above walk is (a, d, e, a, d). Note that if the vertices of a walk are all distinct, then the walk is a path. The ***length*** of a walk is the number of steps taken in the walk. A ***closed walk*** is a walk whose first and last vertices are the same. If a closed walk has no repeated vertices except the first and the last, then we observe it is a cycle. If the first and last vertices of a walk are u and v, then we say the walk is a *uv-**walk***. We refer similarly to a *uv-**path***. The vertices u and v are called the ***ends*** of the path or walk.

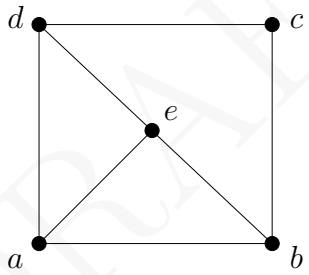


Figure 2.1: Walks

Lemma 2.1.1 *Let u, v be distinct vertices in a graph G, and let W be a shortest uv-walk in G. Then W is a path.*

Proof ▷ Suppose $W = v_0e_0v_1e_1 \dots v_{k-1}e_{k-1}v_k$, where $v_0, v_1, \dots, v_k$ are vertices of G with $v_0 = u$ and $v_k = v$, and $e_0, e_1, \dots, e_k$ are edges of G. If W is not a path, then $v_i = v_j$ for some $i < j$ with $(v_i, v_j) \neq (u, v)$. Define the new walk

$$W' = v_0e_0v_1e_1 \dots v_ie_jv_{j+1} \dots e_{k-1}v_k.$$

Then the length of W' is less than the length of W, a contradiction. So W is a path. □

2.2 Connected graphs

A graph is ***connected*** if any pair of vertices in the graph are the ends of at least one path. If a graph is not connected, we say it is ***disconnected***. The ***components*** of a graph $G = (V, E)$ are the maximal connected subgraphs of G – that is, the connected subgraphs such that no edge of G not already in the subgraph can be added while still preserving connectivity. For instance, the graph below in Figure 2.2 has three components:

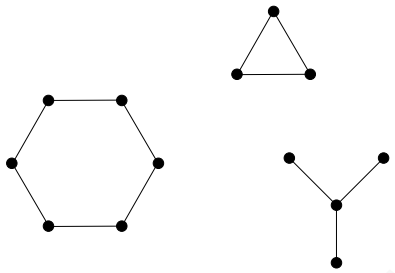


Figure 2.2: Components

2.3 Eulerian graphs

A multigraph is ***Eulerian*** if all its vertices have even degree. A ***trail*** in a graph is a walk with no repeated edges, and a ***tour*** in a graph is a closed walk with no repeated edges. An ***Euler tour*** or ***Eulerian tour*** in a graph G is a tour which contains every edge of G and an ***Euler trail*** or ***Eulerian trail*** is a trail that contains all the edges of G. In the graph shown below, an example of a tour is the walk $(v_1, v_2, v_4, v_1, v_5, v_6, v_1)$. This graph has an Euler tour, namely

$$(v_1, v_2, v_3, v_4, v_5, v_6, v_1, v_5, v_2, v_4, v_1).$$

Roughly speaking, the presence of an Euler tour in a graph means that the graph can be drawn on paper without lifting your pen and without retracing edges.

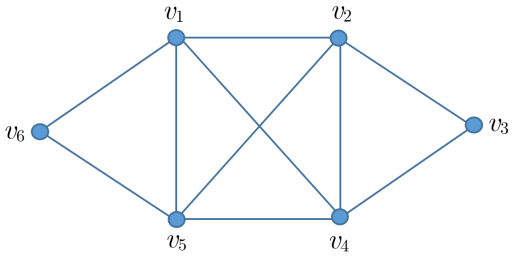


Figure 2.3: Euler tours

The problem of existence of Euler tours was first studied by Euler, in his famous ***bridges of Königsberg problem***. The following theorem [16] is responsible for the existence of an Euler tour in the above graph.

Theorem 2.3.1 *Let G be a connected multigraph and $u, v \in V(G)$. Then*

1. *G has an Euler tour if and only if all of the vertices of G have even degree.*
2. *G has an Euler uv-trail if and only if u and v have odd degree and all other vertices of G have even degree.*

Proof ▷ We prove the first statement and leave the second as an exercise. First we show ◁◁ that if G has an Euler tour, then every vertex of G has even degree. Proceed by induction on $m = |E(G)|$. For $m = 2$, we have an Euler tour (v_1, v_2, v_1), and so G is a double edge, and we are done: both vertices of G have degree two. For $m > 2$, if G has an Euler tour, say $(v_1, v_2, \ldots, v_m, v_1)$ (in this sequence, note that some vertices can be repeated), let $i = \min\{j > 1 : v_i = v_1\}$. Then the edges $\{v_1, v_2\}, \{v_2, v_3\}, \ldots, \{v_{i-1}, v_i\}, \{v_i, v_1\}$ form a cycle C in G (this cycle may be a double edge). If $i = m$, then $G = C$, and all vertices of G have degree two. Otherwise, $G - E(C)$ has the Euler tour $(v_1, v_{i+1}, v_{i+2}, \ldots, v_m, v_1)$, and therefore all degrees of $G - E(C)$ are even. Adding back the edges of C increases degrees by zero or two, so all degrees in G are even, as required.

Now suppose all vertices of G have even degree. Let $\tau = (v_1, v_2, \ldots, v_k)$ be the longest possible trail in G. If $v_k \neq v_1$, then as in the first part of the proof given above, the reader will check that an odd number of edges of τ contain each of v_1 and v_k, so there is an edge ◁◁ $\{v_k, v_{k+1}\}$ of G that is not traversed by τ. Now $(v_1, v_2, \ldots, v_k, v_{k+1})$ is a longer trail than τ, a contradiction. We conclude $v_k = v_1$ and τ is a tour in G. If τ traverses all edges in G, we are done. Suppose τ does not traverse all edges of G. Since G is connected, there is an edge e not in the tour τ, say $\{v_i, v\} \in E(G)$. If v is not a vertex of τ, then

$$(v_i, v_{i+1}, \ldots, v_k, v_1, v_2, \ldots, v_{i-1}, v_i)$$

is a tour of the same length as τ in G. If we add the edge $\{v_i, v\}$, we get the trail

$$(v_i, v_{i+1}, \ldots, v_k, v_1, v_2, \ldots, v_{i-1}, v_i, v)$$

which is longer than τ. If v is a vertex on the trail, say $v = v_j$ where $j < i$, then consider the trail $(v_i, v_{i+1}, \ldots, v_k, v_1, \ldots, v_{j-1}, v_j, v_i, v_{i-1}, \ldots, v_{j+1}, v_j)$. This trail uses the edge e and is therefore longer than τ. This contradiction completes the proof. □

There are a number of simple algorithms for finding Euler tours in Eulerian graphs; the one given by the proof of the above theorem is known as ***Hierholzer's Algorithm***. The running time of this algorithm is linear in the number of edges of the graph. Pseudocode for ◁◁

the algorithm is as follows:

```
1:  function Hierholzer(Graph)
2:  start ← arbitrary node
3:  tour ← ∅
4:  While there are any unvisited edges
5:      start ← node in tour with unvisited edge
6:      subtour ← {start}
7:      current = start
8:      Repeat
9:          {current, u} ← take unvisited edge leaving
                           current
10:         subtour ← subtour ∪ {u}
11:         current ← u
12:     while start ≠ current
13:     Integrate subtour in tour
14: return tour
```

2.4 Eulerian digraphs and de Bruijn sequences

An ***Eulerian digraph*** is a digraph $\vec{G} = (V, \vec{E})$ (with loops allowed) such that for every $v \in V$, $d^+(v) = d^-(v)$. In other words, the in-degree of every vertex equals the out-degree of every vertex. A ***directed Euler tour*** in a digraph $\vec{G}$ is a $(v_0, v_1, \ldots, v_m)$ of vertices with $m = |\vec{E}|$ such that $(v_i, v_{i+1}) \in \vec{E}$ for $0 \leq i < m$ and no edge is repeated. A digraph is connected if its underlying graph is connected. The same proof as for Eulerian graphs (see Theorem 2.3.1) shows the following:

Theorem 2.4.1 *A connected digraph $\vec{G} = (V, \vec{E})$ is Eulerian if and only if it has a **directed Euler tour**.*

This leads us to an application involving ***de Bruijn sequences***. Let k be a positive integer and let A be an alphabet of size n. A ***de Bruijn sequence*** is a cyclic sequence of letters $(a_0, a_1, a_2, \ldots, a_m)$ from A such that every word of length k appears exactly once as k cyclically consecutive letters in the sequence. For instance, if $n = k = 2$ and $A = \{0, 1\}$, then 0011 is a de Bruijn sequence, since $00, 01, 11, 10$ are the cyclically consecutive pairs of letters and each word appears once. If $n = 2$ and $k = 3$ and $A = \{0, 1\}$, then 00010111 is a de Bruijn sequence. It is convenient to put the letters on a circle:

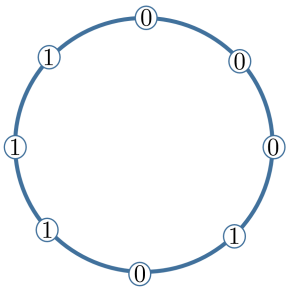


Figure 2.4: de Bruijn sequence

Since there are in general n^k words of length k from an alphabet of size n – see Theorem A.2.2 – a de Bruijn sequence must have length n^k.

A key way to generate de Bruijn sequences is using directed Euler tours. We define the k-dimensional ***de Bruijn digraph*** $\vec{G}(n,k)$ as follows: let the vertex set V of $\vec{G}(n,k)$ be the set of sequences of $k-1$ elements from the alphabet A. We place an arc (u,v) in $\vec{G}(n,k)$ if the last $k-2$ letters of u are the first $k-2$ letters of v. Note that it is possible that $u=v$ in this definition, in which case we are placing a loop from u to u. It is also possible to have both an arc from u to v and from v to u. These possibilities appear in the example below:

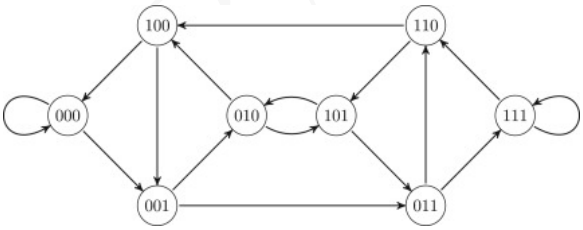


Figure 2.5: de Bruijn digraph $\vec{G}(2,4)$

Given a vertex u, the out-degree and in-degree of u are both exactly n: for the in-degree/out-degree we just have to pick a letter from the alphabet A to prepend/append to u and then remove the last/first letter of u. If n is even, then by Theorem 2.4.1, $\vec{G}(n,k)$ is Eulerian, and so it has a directed Euler tour $(v_1,\ldots,v_m)$ where m is the number of arcs in $\vec{G}(n,k)$. An arc (v_i,v_{i+1}) in this tour corresponds to two words v_i and v_{i+1} of length $k-1$ such that the last $k-2$ letters of v_i are the same as the first $k-2$ letters of v_{i+1}. In particular, we can add the last letter of v_{i+1} to v_i to get a word $w(v_i,v_{i+1})$ of length k. So each arc of the Euler tour corresponds to a word of length k. Since there are n^k arcs in $\vec{G}(n,k)$ by the handshaking

lemma, we have produced n^k words $w(v_1, v_2), w(v_2, v_3), \ldots, w(v_{m-1}, v_m)$ of length k using the Euler tour. No word can be produced more than once, since if $w(v_i, v_{i+1}) = w(v_j, v_{j+1})$, then $v_i = v_j$ and $v_{i+1} = v_{j+1}$. If a_i is the last letter of v_i, then the cyclic sequence $a_1 a_2 \ldots a_m$ of letters is a de Bruijn sequence.

Example 9. Consider the de Bruijn graph in Figure 2.5. An Euler tour $(v_1, v_2, \ldots, v_{16}, v_1)$ with $v_i = i$ is shown below:

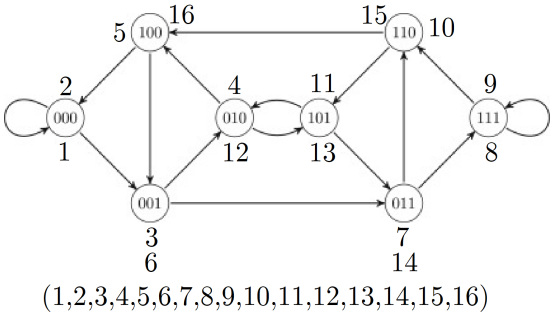


Figure 2.6: Euler tour in $\vec{G}(2, 4)$

To construct a de Bruijn sequence from this tour, let a_i be the last letter of $v_i = i$, and write down $a_1, a_2, \ldots, a_{16}$: we get the sequence 0010011110101100. A manual check reveals all 16 words of length four appear as cyclically consecutive letters in this sequence.

2.5 Hamiltonian graphs

A spanning cycle in a graph is called a ***Hamilton cycle*** and a spanning path in a graph is called a ***Hamilton path***. A graph is ***Hamiltonian*** if it contains a Hamilton cycle and ***traceable*** if it has a Hamilton path. While Theorem 2.3.1 gives a simple necessary and sufficient condition for a graph to have an Euler tour, no such simple condition is available for a graph to be Hamiltonian. In this section, we consider sufficient conditions for a graph to be Hamiltonian. The first is Dirac's Theorem [12].

Theorem 2.5.1 (Dirac) *Let $n \geq 3$, and let G be an n-vertex graph of minimum degree at least $n/2$. Then G is Hamiltonian.*

Proof ▷ Suppose, for a contradiction, that there is a non-Hamiltonian n-vertex graph of minimum degree at least $n/2$. Amongst all such graphs, let G be one with a maximum number of edges. If we add an edge $e = \{v_1, v_n\}$ between non-adjacent vertices of G, then we have a graph with a Hamilton cycle C, and so $P = C - e$ is a Hamilton $v_1 v_n$-path in

G, say $v_1v_2\dots v_n$. Let $N(v_n)^+ = \{v_{i+1} : v_i \in N(v_n)\}$ – this is the set of vertices which are immediately after neighbors of v_n on the path P. Then $N(v_n)^+ \cup N(v_1) \subseteq V(P)\backslash\{v_n\}$ as $\{v_1, v_n\} \notin E(G)$, so

$$|N(v_n)^+ \cup N(v_1)| \leq n-1.$$

On the other hand, $|N(v_n)^+| + |N(v_1)| \geq n$, since G has minimum degree at least $n/2$. Therefore

$$|N(v_n)^+ \cap N(v_1)| = |N(v_n)^+| + |N(v_1)| - |N(v_n)^+ \cup N(v_1)| > 0.$$

Let $v_{i+1} \in N(v_n)^+ \cap N(v_1)$. Then $v_1v_2\dots v_iv_nv_{n-1}\dots v_{i+1}v_1$ is a Hamilton cycle in G, as shown in Figure 2.7, a contradiction. So every n-vertex graph of minimum degree at least $n/2$ is Hamiltonian. □

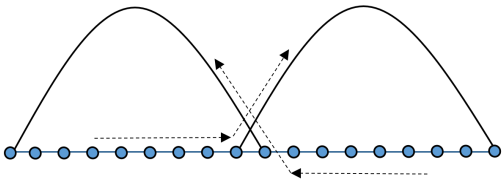


Figure 2.7: Finding a Hamilton cycle

Let $k = \lfloor (n-1)/2 \rfloor$ (round $(n-1)/2$ down to the nearest integer). Then $G = K_{k,n-k}$ is not Hamiltonian, while $\delta(G) = k < n/2$. These examples show that Theorem 2.5.1 is best possible – the condition on the minimum degree cannot be lowered. The reader can check as an exercise that for $n \geq 2$, every n-vertex graph of minimum degree at least $n/2 - 1$ is traceable. The ***closure*** of an n-vertex graph G, denoted $C(G)$, consists in adding edges between any two non-adjacent vertices whose sum of degrees is at least n. The proof of Theorem 2.5.1 actually gives the following result of Bondy and Chvatal [7]:

Theorem 2.5.2 (Bondy, Chvatal) *A graph G is Hamiltonian if and only if $C(G)$ is Hamiltonian.*

If G is an n-vertex graph of minimum degree at least $n/2$, then $C(G) = K_n$, which shows Theorem 2.5.1 follows from Theorem 2.5.2. A major challenge is to find non-trivial sufficient conditions for graphs of low minimum degree to be Hamiltonian or traceable – for instance, graphs of minimum degree at least three. It is true, however, that a graph of minimum degree k has a long cycle:

Theorem 2.5.3 *Let $k \geq 2$, and let G be a graph of minimum degree at least k. Then G contains a cycle of length at least $k+1$.*

Proof ▷ Let P be a longest path in G, say $v_1v_2\dots v_r$. Then $N(v_r) \subseteq V(P)$. Since $|N(v_r)| \geq \delta(G) \geq k$, $r \geq k+1$ and v_r has a neighbor v_i for some $i \leq r-k$. Now the cycle $v_iv_{i+1}\dots v_rv_i$ has length at least $k+1$, as required. □

2.6 Postman and Travelling Salesman Problems*

Let G be a connected graph whose edges represent roads between points, with a ***weight function*** $\omega : E(G) \to \mathbb{R}$ denoting the cost of travelling that road. The ***Postman Problem*** or ***route inspection problem*** is to finding a minimum cost closed walk in the graph that traverses every road at least once. In the event that the graph is Eulerian, this problem is solved by Theorem 2.3.1: an Euler tour is a minimum cost closed walk. Otherwise, the graph has a set X of vertices of odd degree, and $|X|$ is even by Lemma 1.5.2. If P is a path with ends x and y in X, then by doubling every edge of P, the ends of P now have even degree and all other vertices of P still have even degree. By repeating this for every remaining pair of vertices of odd degree, we arrive at a multigraph G' all of whose vertices have even degree. By Theorem 2.3.1, this multigraph has an Euler tour. The strategy is to add the paths P such that the cost of that Euler tour is a minimum. If we suppose that all the roads have the same length, and $X = \{x_1, x_2, \dots, x_{2k}\}$, then we are asking for a partition of X into pairs $\{v_1, w_1\}, \{v_2, w_2\}, \dots, \{v_k, w_k\}$ such that the sum of the lengths of the shortest v_iw_i-paths is a minimum (the shortest paths can be found via ***Dijkstra's Algorithm***). This is a problem in the theory of ***matchings*** – finding a minimum weight matching – which we return to later. An algorithm for finding an Euler tour is described in Question 2.5 in the exercises. Putting everything together gives an algorithm which runs in time at most roughly n^3.

The ***Travelling Salesman Problem*** or ***TSP*** is a generalization of the problem of finding a Hamilton cycle in a graph. Specifically, we consider a complete graph K_n and a ***weight function*** $\omega : E(K_n) \to \mathbb{R}$. The ***weight*** of a Hamilton cycle C in K_n is defined to be

$$\sum_{e \in E(C)} \omega(e).$$

Given a weight function ω, the Travelling Salesman Problem asks for a Hamilton cycle of K_n of minimum weight. One might interpret the weights as the cost of travelling across edges for a salesperson who would like to visit every vertex of K_n once and return to the starting point, in which case one is looking for a Hamilton cycle of minimum cost. If G is any graph, then we could define a weight function ω on K_n by defining $\omega(e) = 0$ if $e \in E(G)$ and $\omega(e) = 1$ otherwise. Then there is a Hamilton cycle of zero cost if and only if the graph G is Hamiltonian. Methods for solving the TSP are often based on ***polyhedral combinatorics*** and ***cutting-plane algorithms*** in ***integer linear programming***, which are beyond the

scope of this course. The current fastest algorithm for solving TSP on an n-vertex graph runs in time roughly $n^2 2^n$.

2.7 Uniquely Hamiltonian graphs*

A graph is ***uniquely Hamiltonian*** if it has exactly one Hamilton cycle. The notion of uniquely Hamiltonian graphs first arose in the context of coloring planar graphs, a topic we return to in Section 7.3. In this section, we prove the following theorem, which is known as Smith's Theorem [41] for cubic graphs. The statement of the theorem here is due to Thomason [39]:

Theorem 2.7.1 *Let G be a graph all of whose vertices have odd degree. Then there exist an even number of Hamilton cycles containing any edge $e \in E(G)$. In particular, G is not uniquely Hamiltonian.*

One of the main ideas in the proof of this theorem is ***rotation***. If P is a Hamilton uv-path in a graph G, $\{w, v\} \in E(G) \backslash E(P)$, and x is the vertex closer to v adjacent to w on P, then $Q = P - \{w, x\} + \{w, v\}$ another Hamilton path in G ending at the vertex x. Note also that $P = Q - \{w, v\} + \{w, x\}$. We say that P and Q are obtained from one another by ***rotation***. The key to the proof of Dirac's Theorem was to find a rotation $Q = P - \{w, x\} + \{w, v\}$ of a Hamilton path P such that $x \in N(u)$, which implies $Q + \{u, x\}$ is a Hamilton cycle. In the proof of Theorem 2.7.1, we consider all possible rotations from a given Hamilton path:

Proof ▷ Let $e = \{u, v\}$. If there are no Hamilton cycles containing e, we are done. Suppose there is a Hamilton cycle C containing e, and let $N_C(u) = \{v, w\}$ and $f = \{u, w\}$. Consider the Hamilton uw-path $P = C - f$, ordered from u to w. Form a new graph $H = H(G, C, P)$ whose vertices are the Hamilton paths of $G - f$ starting with the edge e, where two Hamilton paths in G form an edge of H if they are obtained from one another by ***rotation*** (in Figure 2.8 below, the path P is shown in bold black edges). We observe that if Q is any Hamilton path in H, starting with e and ending at some vertex t, then there are exactly $d_G(t) - 1$ possible rotations, one for each edge containing t and not already used by Q, unless $\{u, t\}$ is an edge, in which case there are $d_G(t) - 2$ rotations (see Figure 2.8). In the latter case, Q together with $\{u, t\}$ forms a Hamilton cycle in G containing e. Since $d_G(t)$ is odd for every vertex t, $d_G(t) - 2$ is also odd. The number of vertices of H of odd degree is even, by Lemma 1.5.2, so there must be an even number of Hamilton paths Q in $G - f$ which end at a neighbor of u (in Figure 2.8, H has three vertices, one corresponding to the path in bold black edges, one in dashed black edges, and one in dashed red edges, and H is a path of length two). Therefore G contains an even number of Hamilton cycles containing e. □

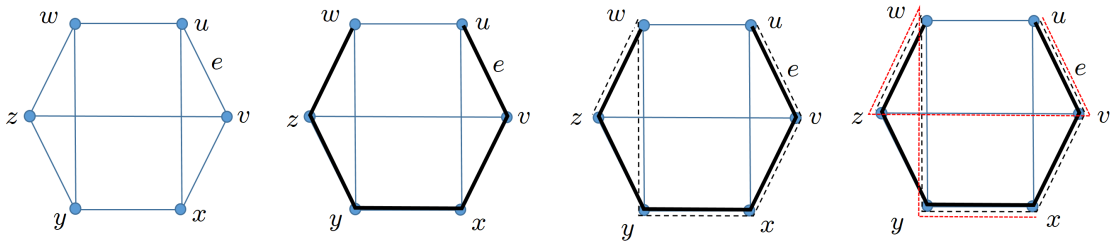
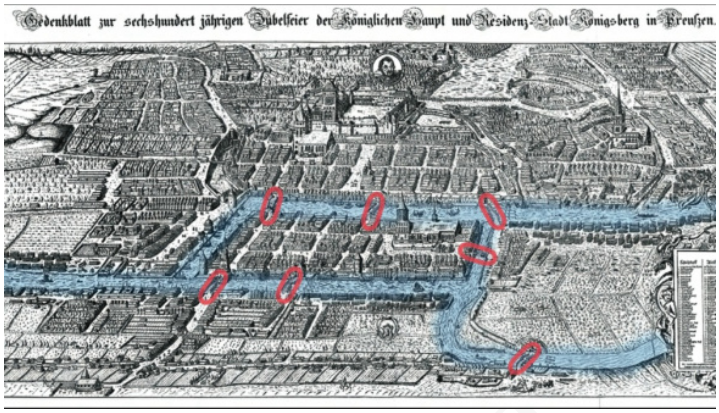


Figure 2.8: Finding a second Hamilton cycle

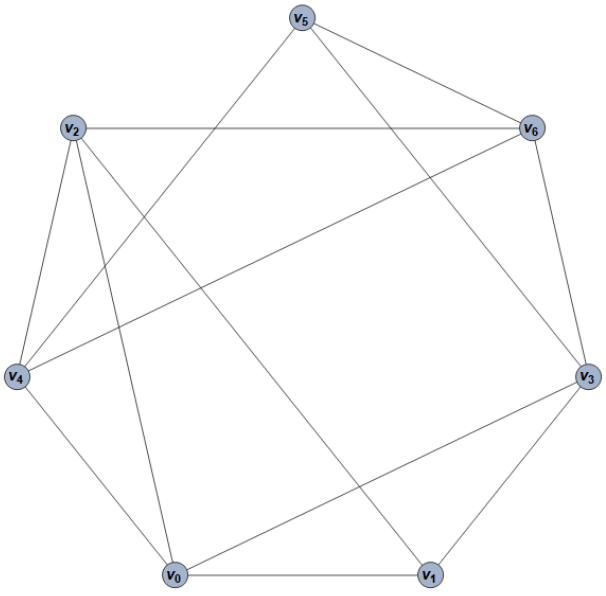
The proof of the above theorem is a ***parity argument***, based on Lemma 1.5.2 applied to the graph $H(G, C, P)$. One of the main open conjectures due to Sheehan [37] is that there are no uniquely Hamiltonian 4-regular graphs. Another interesting question is the algorithmic complexity of finding a second Hamiltonian cycle can be found in a graph G with a given Hamiltonian cycle C. It turns out as was shown by Cameron [8] that there are n-vertex cubic graphs where exponentially many rotations in n are required to find a Hamiltonian cycle different from a given Hamiltonian cycle C. The use of multiple rotations was first exploited to great effect by Pósa [34] in finding Hamiltonian cycles in random graphs, and is key in many modern contexts for finding long paths and cycles in graphs.

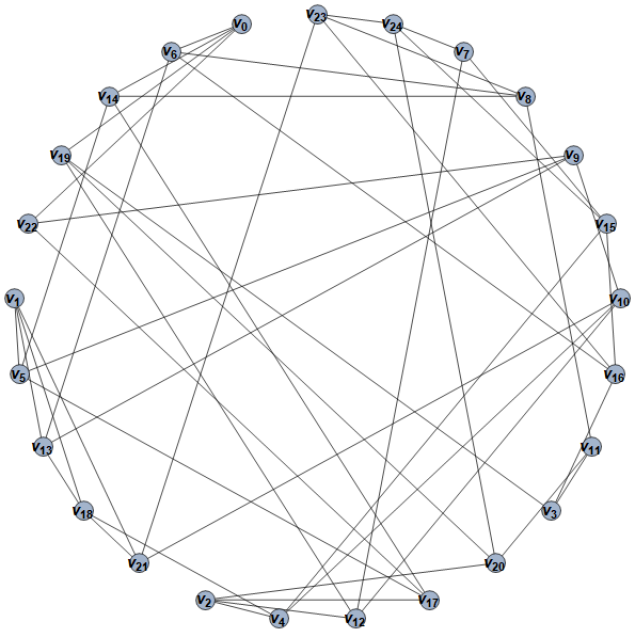
2.8 Exercises

Question 2.1° The ***Bridge of Königsberg Problem*** is to devise a route through the city that crosses each of the bridges in the map below exactly once. The starting and ending point of the route do not need to be the same. Does such a route exist?

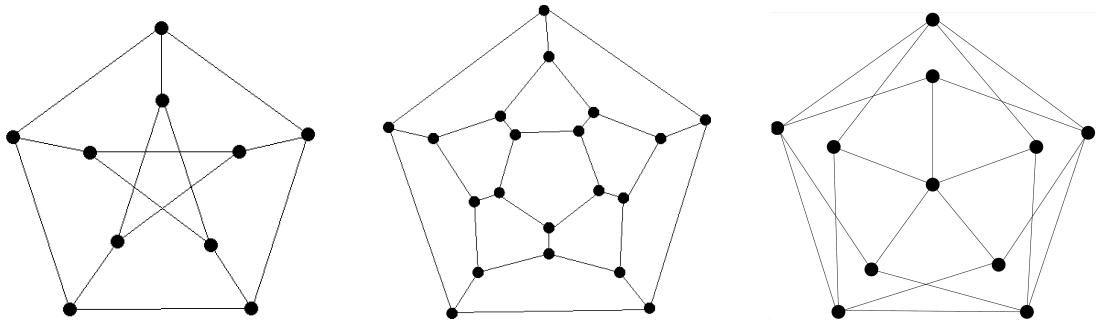


Question 2.2° Prove that if G is a connected graph such that two vertices $u, v \in V(G)$ have odd degree and all other vertices have even degree, then there is an Euler uv-trail in G. Then find an Euler trail in the graphs below, or state why no Euler trail exists:





Question 2.3° Find a Hamilton cycle in each graph below, or state that none exists:



Question 2.4° The ***line graph***[3] of a graph $G = (V, E)$ is the graph $L(G) = (E, F)$ whose vertex set is E and whose edge set is

$$F = \{\{e, f\} \subset E : e \cap f \neq \emptyset\}.$$

Prove that if G is connected and regular, then $L(G)$ is Eulerian.

Question 2.5. ***Fleury's Algorithm*** for finding Euler tours is described as follows. Let G be a graph and $v_1 \in V(G)$. Having chosen vertices $v_1, v_2, \ldots, v_k$ in G such that $\{v_1, v_2\}, \{v_2, v_3\}, \ldots, \{v_{k-1}, v_k\}$ are edges of G, select an edge $e = \{v_k, v_{k+1}\}$ such that $(G - \{v_1, v_2, \ldots, v_k\}) - e$ is connected[4] if such an edge e exists, otherwise stop. Prove that if G is Eulerian, then the algorithm terminates with an Euler tour $(v_1, v_2, \ldots, v_m)$ of G.

Question 2.6° A driver starting in San Francisco wishes to drive on each road between pairs of the following major cities in California and end in Sacramento using as short as possible a routing: Fresno, Los Angeles, Sacramento, San Diego, and San Francisco. The lengths of the roads the driver wishes to cover are indicated in the table below.[5] Determine the total length of a shortest route. If instead the driver wishes to visit each city exactly once, what then is the length of a shortest route?

[3] See Question 1.6.

[4] An edge e in a connected graph G such that $G - e$ is disconnected is called a ***bridge*** of G. So here e is not a bridge in $G - \{v_1, v_2, \ldots, v_k\}$.

[5] Distances are in both directions.

City Pair	Distance
Fresno-Los Angeles	221
Fresno-Sacramento	171
Fresno-San Diego	345
Fresno-San Francisco	187
Los Angeles-San Diego	120
Los Angeles-San Francisco	382
Sacramento-San Francisco	88

Figure 2.9: City Pairs

Question 2.7°
(a) Draw the de Bruijn graph $\vec{G}(3,2)$.
(b) Find a de Bruijn sequence for words of length two over the alphabet $\{0,1,2\}$.
(c) In a brute-force attach on a 2 digit pin code with digits from $\{0,1,2\}$, 18 key presses may be required to guess the code, since there are $3^2 = 9$ possible 2 digit pin codes. If the keypad has no enter key[6], show that 10 key presses are sufficient to guess the code.

Question 2.8° Is it possible to write down a sequence $a_1a_2\ldots a_m$ of letters from an alphabet Σ of size n so that every word of length k appears exactly once as a sequence of k consecutive letters of $a_1a_2\ldots a_m$?

Question 2.9° Let P be a longest path in a connected graph G, and suppose there exists a cycle C such that $P \subseteq C \subseteq G$. Prove that G is Hamiltonian.

Question 2.10° Prove that if G is a connected graph with m edges such that two vertices $u, v \in V(G)$ have odd degree and all other vertices have even degree, then there exists a sequence $(v_0, v_1, v_2, \ldots, v_m)$ of vertices of G with $u = v_0$ and $v = v_m$ such that every edge of G appears exactly once as a pair $\{v_i, v_{i+1}\}$ (an ***Euler trail*** or ***Eulerian trail***).

Question 2.11. A ***tournament*** is an orientation of a complete graph. Prove that every tournament contains a directed path containing all of its vertices.

Question 2.12° For the ***Heawood graph*** shown below, draw the graph H from the proof of Theorem 2.7.1 where P is the Hamilton path $(1,2,3,\ldots,14)$. Then find a Hamilton cycle different from $(1,2,3,\ldots,14,1)$.

[6]This means that the code is guessed any time the correct 2 consecutive keys are pressed.

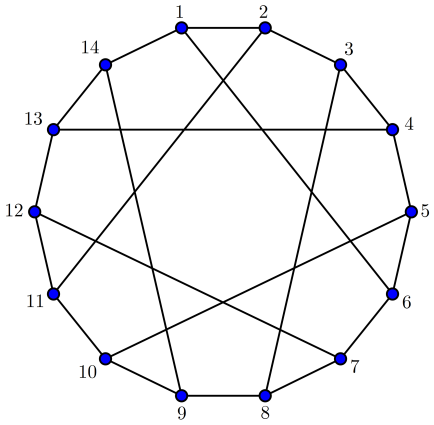


Figure 2.10: Finding a second Hamilton cycle

Question 2.13. Solve the Postman Problem and Travelling Salesman Problem for the weighted graph below.

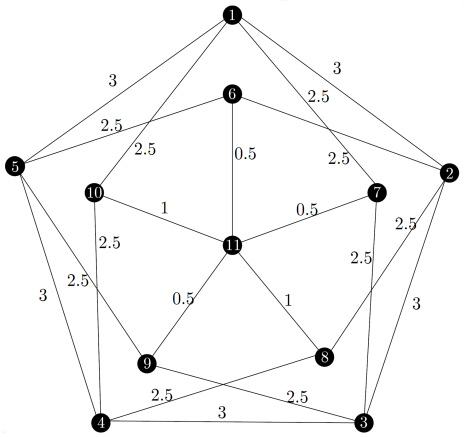


Figure 2.11: Postman Problem

Question 2.14. Let P and Q be longest paths in a finite connected graph G.

(a) Prove that $V(P) \cap V(Q) \neq \emptyset$.

(b) Is it true that if P, Q and R are longest paths then $V(P) \cap V(Q) \cap V(R) \neq \emptyset$? Give a proof of this statement, or a counterexample to this statement.

Question 2.15. Let $P_1, P_2, \ldots, P_k$ be longest paths in a tree T. Prove that

$$V(P_1) \cap V(P_2) \cap \cdots \cap V(P_k) \neq \emptyset.$$

Is it true that $E(P_1) \cap E(P_2) \cap \cdots \cap E(P_k) \neq \emptyset$?

Question 2.16. Prove that a graph of minimum degree at least $k \geq 2$ containing no triangles contains a cycle of length at least $2k$.

Question 2.17. Let G be an n-vertex graph such that for any non-adjacent vertices $u, v \in V(G)$, $d(u) + d(v) \geq n$. Prove that G is Hamiltonian.

Question 2.18. Let $n \geq 2$.
(a) Prove that an n-vertex graph with at least $\binom{n-1}{2} + 1$ edges is ***traceable***.
(b) Give an example of an n-vertex graph with $\binom{n-1}{2}$ edges that is not traceable.

Question 2.19. Prove that every graph has an orientation such that the difference between in and out degrees at each vertex is at most 1.

Question 2.20. Prove that a graph of minimum degree at least $k \geq 2$ containing no triangles or quadrilaterals contains a cycle of length at least $3k - 1$.

Question 2.21. The ***closure*** of an n-vertex graph G, denoted $C(G)$, consists in adding edges between any two non-adjacent vertices u and v such that $d_G(u) + d_G(v) \geq n$. Prove that a graph G is Hamiltonian if and only if $C(G)$ is Hamiltonian.

Question 2.22° Let $n \geq 3$ How many Hamilton cycles does K_n contain? Find an algorithm for the travelling salesman problem on n vertices which runs in time at most $n!/2$.

Question 2.23. Let $n \geq 2$.
(a) Prove that an n-vertex graph with at least $\binom{n-1}{2} + 2$ edges is ***Hamiltonian***.
(b) Give an example of an n-vertex graph with $\binom{n-1}{2} + 1$ edges that is not Hamiltonian.

Question 2.24* Let G be a connected graph with an even number of vertices. Prove that there is a spanning subgraph $H \subseteq G$ such that all vertices of H have odd degree.

Question 2.25* Prove that for every graph G, there exists a partition (X, Y) of $V(G)$ such that every component of $G[X]$ and every component of $G[Y]$ is Eulerian.

Question 2.26* A graph is ***1-tough*** if for every set $X \subseteq V(G)$, the number of components of $G - X$ is at most $|X|$.

(a) Prove that if G is a Hamiltonian graph, then G is 1-tough.

(b) Find a graph that is 1-tough but not Hamiltonian.

Question 2.27. The ***Möbius ladder*** graph M_n has $V(M_n) = \{v_0, v_1, v_2, \ldots, v_{2n-1}\}$ and edge set

$$E(M_n) = \{v_0v_1, v_1v_2, \ldots, v_{2n-1}v_0\} \cup \{v_iv_{i+n} : 0 \leq i \leq n-1\}.$$

The graph M_8 is shown in Figure 2.12. Determine the number of Hamiltonian cycles in M_n for each $n \geq 2$.

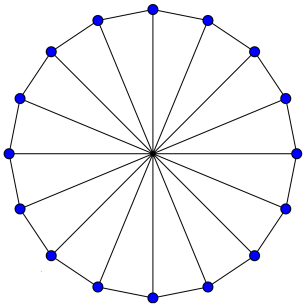


Figure 2.12: Möbius ladder M_8

Question 2.28. This question refers to the graph G below.

(a) Identify a Hamilton cycle $C \subseteq G$ and a Hamilton path $P \subseteq C$.

(b) Draw the graph $H(G, C, P)$ from the proof of Theorem 2.7.1.

(c) Use rotation to find a second Hamilton cycle in G.

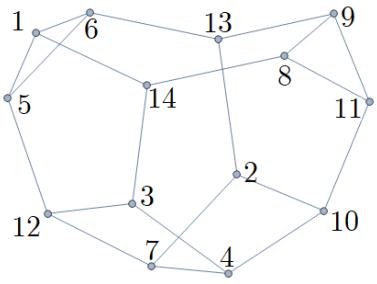


Figure 2.13: A cubic graph

Question 2.29* Let G be a Hamiltonian bipartite graph of minimum degree at least three. Prove that G contains at least two Hamilton cycles.

Question 2.30* Let C be a Hamilton cycle in an Eulerian graph G, and suppose every component of $G - E(C)$ has an even number of vertices. Prove that there exists a Hamilton cycle C' in G such that $C' \neq C$.[7]

Question 2.31. Let G be any ***cubic multigraph*** and $v \in V(G)$. The ***Y△-transform of G at v*** is the multigraph $Y\triangle(G, v)$ obtained as shown in the illustration in Figure 2.14 below. The ***Y△-transform of G*** is the multigraph $Y\triangle(G)$ obtained by applying the $Y\triangle$-transform of G at v for each $v \in V(G)$.

(a) Define $Y\triangle(G, v)$ and $Y\triangle(G)$ precisely in mathematical terms.

(b) Draw $Y\triangle(G)$ when G is a triple edge and when $G = K_4$.

(c) Prove that G is Hamiltonian if and only if $Y\triangle(G)$ is Hamiltonian.

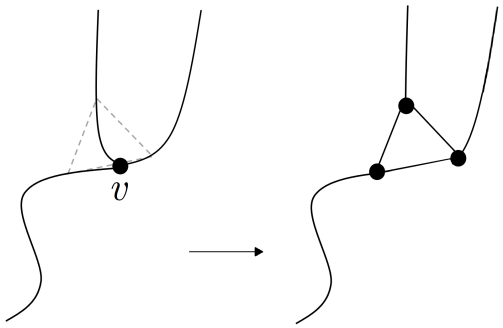


Figure 2.14: $Y\triangle$-transform at v

Question 2.32* Let G be an r-regular graph with $2r + 1$ vertices, where $r \geq 2$. Prove that G is Hamiltonian. Then give an example for each $r \geq 2$ of an r-regular graph with $2r + 2$ vertices that is not Hamiltonian.

Question 2.33* Let G be an r-regular graph with at most $3r$ vertices, where $r \geq 2$. Suppose $G - \{v\}$ is connected for every $v \in V(G)$. Prove that G is Hamiltonian.

Question 2.34* Prove that if G is an Eulerian digraph with m edges and n vertices, then G contains a directed path of length at least $\sqrt{m/n}$.

[7] Question 2.24 may be helpful.

3 Bridges, Trees and Algorithms

3.1 Bridges and trees

Recall a graph G is ***connected*** if for every $u, v \in V(G)$, there exists a uv-walk (equivalently, a uv-path by Lemma 2.1.1). A ***tree*** is a connected graph without cycles – a connected ***acyclic*** graph. An acyclic graph is called a ***forest*** – thus all components of a forest are trees. To describe the structure of trees, we define the notion of a bridge. A ***bridge*** of a graph G is an edge $e \in E(G)$ such that $G - e$ has more components than G. For example, in Figure 1.2, the edges $\{p_1, p_2\}, \{p_1, p_4\}$ and $\{p_4, p_3\}$ are bridges, whereas the remaining edges are not bridges. It is easy to spot the bridges of a graph, using the following lemma:

Lemma 3.1.1 *An edge $e \in E(G)$ is a bridge of G if and only if e is not contained in any cycle in G.*

Proof ▷ If e is contained in a cycle C of G, then $C - e$ is a path joining the ends of e. But that means $G - e$ is connected, so e could not have been a bridge. □

Since a tree has no cycles, every edge of a tree must be a bridge. We can now characterize which graphs are trees in a few ways.

Lemma 3.1.2 *Let $n \geq 1$ and let T be an n-vertex tree. Then T has $n - 1$ edges.*

Proof ▷ We proceed by strong induction on n. If $n = 1$, then $|E(T)| = 0 = n - 1$. Suppose the statement is true for all trees with few than n vertices, and let T be a tree with n vertices. If $e \in E(T)$, then e is a bridge of T so $T - e$ is disconnected, with two components T_1 and T_2. Since T_1 and T_2 are acyclic, they are trees, say with n_1 and n_2 vertices resepectively. By ◁◁ induction, $|E(T_1)| = n_1 - 1$ and $|E(T_2)| = n_2 - 1$. We note $n = n_1 + n_2$ and

$$|E(T)| = |E(T_1)| + |E(T_2)| + 1 = (n_1 - 1) + (n_2 - 1) + 1 = n - 1.$$

This completes the proof. □

Proposition 3.1.3 *Each of the following is equivalent to a graph G being a tree:*

1. *The graph G is connected and acyclic.*
2. *The graph G is connected and every edge of G is a bridge.*
3. *The graph G is connected and has $|V(G)| - 1$ edges.*

Proof ▷ Clearly Part 1 of Proposition 3.1.3 is the definition of G being a tree. Since a connected graph is acyclic if and only if every edge of the graph is a bridge, by the last lemma, Part 1 and Part 2 of Proposition 3.1.3 are equivalent. Part 1 implies Part 3 by Lemma 3.1.2.

It remains to show that Part 3 of Proposition 3.1.3 implies Part 1 of Proposition 3.1.3. To see that, if G is connected with $|V(G)| - 1$ edges, we remove an edge of any cycle and that does not disconnect G, by Lemma 3.1.1. We continue removing edges of G in cycles until all the cycles are gone. But then the remaining graph T is connected and acyclic, so must be a tree. Since it has $|V(G)|$ vertices, we know it must have $|V(G)| - 1$ edges. But G itself has $|V(G)| - 1$ edges, so $G = T$. □

The last part of the proof of this proposition is important. It says that in any connected graph G, while there is a cycle, pick an edge of the cycle and remove it. By Lemma 3.1.1, we did not disconnect the graph, so if we repeat this procedure we eventually obtain a spanning subgraph of G which is acyclic and connected – a tree. We call this a ***spanning*** tree of the graph. A spanning tree of the cube graph Q is given below in bold edges (the reader should find other spanning trees of Q): ◁◁

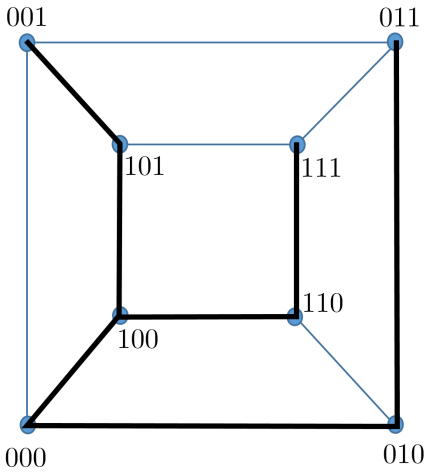


Figure 3.1: Spanning tree

In general, a graph has many spanning trees.

Proposition 3.1.4 *Any connected graph contains a spanning tree.*

The proof gives a fairly quick way to find a spanning tree of a graph: search for a cycle and remove an edge of the cycle, and repeat until there are no cycles left. We next discuss an algorithmic way for finding a spanning tree.

3.2 Breadth-first search

One of the simplest things to check is whether a connected graph is bipartite. Namely, pick any vertex of the graph and place it in A. Then, all the neighbors of that vertex are forced

to be in B. Then all their neighbors must be in A, and so on. We repeat this procedure on all components of the graph until all the vertices of the graph have been placed in A and in B. Applying this procedure to the graphs in Examples 1 – 4, the reader may check that only the graph in Example 2 is not bipartite. There is a systematic way to check whether a graph is bipartite, at the same time as producing a spanning tree in G. To describe this algorithm, we need the notion of ***distance*** in graphs. ◁◁

The ***distance*** between vertices u and v in a connected graph G, denoted $d_G(u, v)$, is the length of a shortest uv-path. For instance, we check in Figure 2.1 that

$$d_G(a, c) = d_G(e, c) = d_G(b, d) = 2$$

and any two other vertices are adjacent, so they are at distance 1. The maximum distance between any two vertices in a connected graph is called the ***diameter*** of G:

$$\text{diam}(G) = \max\{d_G(u, v) : u, v \in V(G)\}.$$

The minimum r such that every vertex of G is at distance at most r from some vertex of G is called the ***radius*** of G:

$$\text{rad}(G) = \min\{\max\{d_G(u, v) : u \in V(G)\} : v \in V(G)\}.$$

For instance, the graph in Figure 1.3 has radius 2 but diameter 4, since every vertex is at distance at most 2 from $(1, 1)$, whilst the shortest path from $(0, 0)$ and $(2, 2)$ has length four. Similarly, the graph in Figure 1.2 has radius 2 and diameter 4. For complete graphs, the radius and diameter are both 1.

If v is a vertex in a connected graph G, we let $N_i(v)$ denote the set of vertices at distance exactly i from v, so that $N_1(v)$ is exactly the neighborhood of v and $N_0(v) = \{v\}$. Order the vertices of G. We build a ***breadth-first search tree*** T in G as follows. First we add v to T – the ***root*** of the tree. At any stage of the construction, we have a tree T whose vertex set is ordered $(v_0, v_1, v_2, \ldots, v_k)$ where $v = v_0$. If $V(T) = V(G)$, stop. Otherwise, since G is connected, there exists a smallest integer i such that some neighbor w of v_i is not in T. Choose w to be the smallest neighbor of v_i not in T in the ordering of the vertices of G, and then add the edge $\{v_i, w\}$ to T. The running time of the breadth-first search algorithm is roughly $|V(G)| + |E(G)|$.

Example 10. An example is given in Figure 3.2, where the vertices are in increasing order $1, 2, 3, 4, 5, 6, 7, 8, 9$ and $v = 1$.

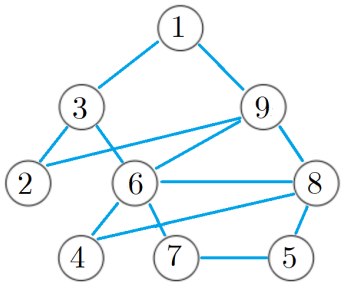


Figure 3.2: Breadth-first search

The edges with arrows on them in the figure below denote the edges in the breadth-first search tree T rooted at v. Then T can be encoded by the sequence $(1, 3, 9, 2, 6, 8, 4, 7, 5)$, where the layers of the tree are $N_0(v) = \{1\}$, $N_1(v) = \{3, 9\}$, $N_2(v) = \{2, 6, 8\}$ and $N_3(v) = \{4, 5, 7\}$.

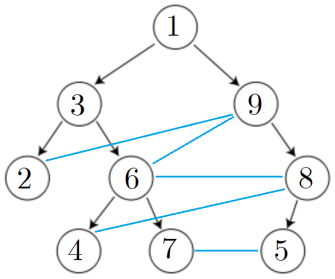


Figure 3.3: Breadth-first search

Lemma 3.2.1 *Let G be a connected graph and $v \in V(G)$. Then T is a spanning tree of G such that $d_T(v, w) = d_G(v, w)$ for all $w \in V(G)$.*

The last statement in this lemma says that T preserves distances from v to all other vertices. The tree T is called a ***breadth-first search tree rooted at*** v. The sets $N_i(v)$ are sometimes called the ***layers*** of T, and the ***height*** of T is the maximum distance of any vertex from v. In Figure 3.3, we have a tree with four layers and height three.

Example 11. The famous ***Petersen graph*** is drawn below, with vertices labelled 1 through 10. Let us apply the breadth-first search algorithm to find a spanning tree in G rooted at vertex 1. Of course, we start by adding 1 to the tree.

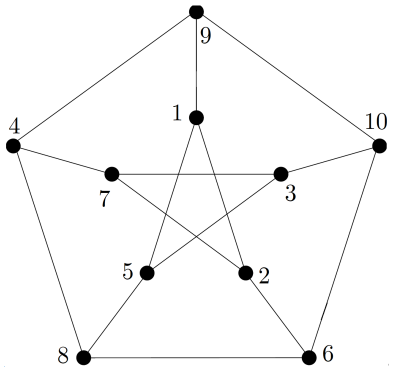


Figure 3.4: Petersen graph

We then add the neighbors of 1 in increasing order, namely, 2, 5 and then 9. So far our tree has edges $\{1,2\}, \{1,5\}$ and $\{1,9\}$. Now we move on to the first vertex added in $N_1(v)$, namely 2. We add first the vertex 6 and then the vertex 7, with edges $\{2,6\}$ and $\{2,7\}$. Then move to the next added vertex in $N_1(v)$, namely 5. We add 3 and 8 and the edges $\{5,3\}$ and $\{5,8\}$. Finally, we move to the vertex 9, and add the vertices 4 and 10 and the edges $\{9,4\}$ and $\{9,10\}$. Then we stop since there are no vertices left to add. The tree has edge set $\{\{1,2\},\{1,5\},\{1,9\},\{2,6\},\{2,7\},\{5,3\},\{5,8\},\{9,4\},\{9,10\}$, and the order in which vertices were added is $(1,2,5,9,6,7,3,8,4,10)$. The layers of the tree are $N_0(v) = \{1\}$, $N_1(v) = \{2,5,9\}$, and $N_2(v) = \{6,7,3,8,4,10\}$. The reader can check that the Petersen graph has diameter and radius equal to two. ◁◁

3.3 Characterizing bipartite graphs

We now use breadth-first search to prove a lemma characterizing bipartite graphs. Via this lemma, the Petersen graph in Figure 3.4 is not bipartite, since it contains a cycle of length five, for instance with vertex set $\{1,2,7,3,5\}$ (in fact there are many cycles of length 5, 7 and 9). ◁◁

Lemma 3.3.1 *A graph G is bipartite if and only if it does not contain any odd cycles.*

Proof ▷ Since an odd cycle is not bipartite, bipartite graphs cannot contain odd cycles. Conversely, suppose a graph G has no odd cycles. Since a graph G is bipartite if and only if all of its components are bipartite, we may assume the graph G is connected. Let T be a breadth-first search tree in G, rooted at some vertex v. We claim that $A = N_0(v) \cup N_2(v) \cup \ldots$ and $B = N_1(v) \cup N_3(v) \cup \ldots$ do not contain any edges of G, and therefore they are the parts in a bipartition of G. Suppose there exists an edge $\{x,y\}$ in A. Since edges of T connect consecutive layers, $\{x,y\}$ is not in T. Let P be an xy-path in T. Then P

together with $\{x, y\}$ forms a cycle C. On the other hand, P must have even length, since if $x \in N_{2i}(v)$ and $y \in N_{2j}(v)$, for if $N_h(v)$ is the lowest layer that P intersects, then P has length $(2i-h)+(2j-h) = 2i+2j-2h$. But then C has odd length, which is a contradiction (this is evident for instance in Figure 3.3, with $x = 6$ and $y = 8$, $i = j = 2$ and $h = 0$, and P is the path with edge set $\{\{6, 3\}, \{3, 1\}, \{1, 9\}, \{9, 8\}\}$). Similarly, B does not contain any edges of G, so A and B are the parts of G. □

3.4 Depth-first search

Another way to generate a spanning tree in a connected graph G is the ***depth-first search*** algorithm. Let the vertices of G be ordered, and identify a vertex $v \in V(G)$ which will be the ***root*** of the ***depth-first search tree***. At any stage, we have a tree T rooted at v. We stop if $V(T) = V(G)$. Otherwise, we pick a vertex $x \in V(T)$ as far as possible from v in T, such that x has a neighbor w not in the tree. Select the first such neighbor w in the ordering of the vertices of G, and add the edge $\{x, w\}$ to the tree. In words, the algorithm looks for a vertex that is as far from the root as possible at each stage and has a neighbor that is not in the tree which is then added to the tree.

Example 12. We consider again the graph in Figure 3.2, where the vertices are ordered $1, 2, 3, 4, 5, 6, 7, 8, 9$ and $v = 1$. For depth-first search, we first add the edge $\{1, 3\}$ and then the edge $\{3, 2\}$, followed by $\{2, 9\}$, $\{9, 6\}$, $\{6, 4\}$, $\{4, 8\}$, $\{8, 5\}$ and $\{5, 7\}$. So in this case the depth-first search tree is a spanning path in the graph. The vertices were added in the order $(3, 2, 9, 6, 4, 8, 5, 7)$.

If we consider the Petersen graph in Figure 3.4, with 1 being the root vertex, then the depth-first search algorithm gives the ordering $(1, 2, 6, 8, 4, 7, 3, 5, 10, 9)$. Note that after we add the edge $\{3, 5\}$ to the tree, the furthest vertex from the root and that has a neighbor not in the tree is 3, and we add $\{3, 10\}$. Then the next furthest vertex from the root which has a neighbor outside of the tree so far is 4 and we add the edge $\{4, 10\}$ to complete the tree.

3.5 Prim's and Kruskal's Algorithms

Let G be a connected graph and let ω be a ***weight function*** or ***cost function*** on the edges of the graph: this is a function $\omega : E(G) \to \mathbb{R}$, where $\mathbb{R}$ is the set of real numbers. In this section, we give algorithms for finding a ***minimum weight spanning tree*** or ***minimum cost spanning tree*** in the graph, namely a spanning tree whose sum of edge weights is as small as possible. Consider for instance the problem of laying cable to connect different locations on a map. If the cost of laying the cable is proportional to the distance between

locations, then we may seek to find a minimum cost spanning tree in the complete graph whose vertices are the locations, and where ω represents the distance between two locations.

A very simple algorithm, known as ***Prim's Algorithm***, starts with a single vertex, and then at any stage, given a tree T constructed so far, adds an edge $\{u, v\}$ such that $u \in V(T)$ and $v \notin V(T)$ and $\omega(u, v)$ is a minimum, until $V(T) = V(G)$. When the algorithm terminates, T is a minimum weight spanning tree in the graph:

Theorem 3.5.1 *The output of Prim's Algorithm is a minimum weight spanning tree.*

Proof ▷ We prove by induction on m that a tree T constructed at stage m of Prim's Algorithm is a subtree of some minimum weight spanning tree. This is clearly true if $m = 0$, so we suppose $m \geq 1$. We then have a tree T with m edges contained in some minimum weight spanning tree T^*, and an edge $e = \{u, v\}$ that will be added to T by Prim's Algorithm with $u \in V(T)$ and $v \notin V(T)$. We have to show $T \cup \{e\}$ is contained in some minimum weight spanning tree. If $T \cup \{e\}$ is contained in T^*, we are done. So suppose $T \cup \{e\}$ is not contained in T^*. Then $T^* \cup \{e\}$ contains a cycle C which contains e. Since $v \notin V(T)$ and $u \in V(T)$, there exists an edge $e' = \{u', v'\} \in E(C) \backslash \{e\}$ such that $u' \in V(T)$ and $v' \notin V(T)$. Therefore $\omega(e') \geq \omega(e)$. Now the weight of the spanning tree T' consisting of the edges $E(T^*) \cup \{e\} \backslash \{e'\}$ is the weight of T^* minus $(\omega(e') - \omega(e))$. But T^* was of minimum weight, so we conclude $\omega(e') = \omega(e)$, and T' has the same weight as T^*. Therefore T' is a minimum weight spanning tree containing $T \cup \{e\}$, as required. □

Another similar algorithm, ***Kruskal's Algorithm***, maintains two disjoint sets E and F of edges in the graph, starting with $E = E(G)$ and $F = \emptyset$. At any stage of the algorithm, F is the edge-set of a forest and E is some subset of the edges of the graph. At any stage where $E \neq \emptyset$, We select an edge of minimum weight in E, and remove it from E. If this edge connects different components of the forest spanned by F, we add it to F. Otherwise we select the next edge of E of minimum weight and repeat. The algorithm terminates when $E = \emptyset$ and F is a spanning tree of the graph, and in this case, F is a minimum weight spanning tree. The proof of the correctness of this algorithm is similar to the proof of correctness of Prim's Algorithm, and we leave it as an exercise. Both algorithms run in at most quadratic time in the number of vertices; the time complexity can be further improved to roughly $|E(G)| \log |V(G)|$ using ***heaps***. ◁◁

Example 13. Let S be a finite set of points in the plane. A ***Euclidean minimum spanning tree*** for the set S of points is a minimum weight spanning tree of the complete graph G with vertex set S where the weight function $\omega : E(G) \to \mathbb{R}$ is given by

$$\omega(\{x_1, y_1\}, \{x_2, y_2\}) = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}.$$

In other words, the cost function for the edge $\{\{x_1, y_1\}, \{x_2, y_2\}\}$ is the Euclidean length of the edge. We consider an example of both Prim's and Kruskal's algorithm to find a ***Euclidean minimum spanning tree*** for the example in Figure 3.5 of a set of points in the plane. The points are labelled P, Q, R, S, T, and the some of the distances between them are shown on the right.

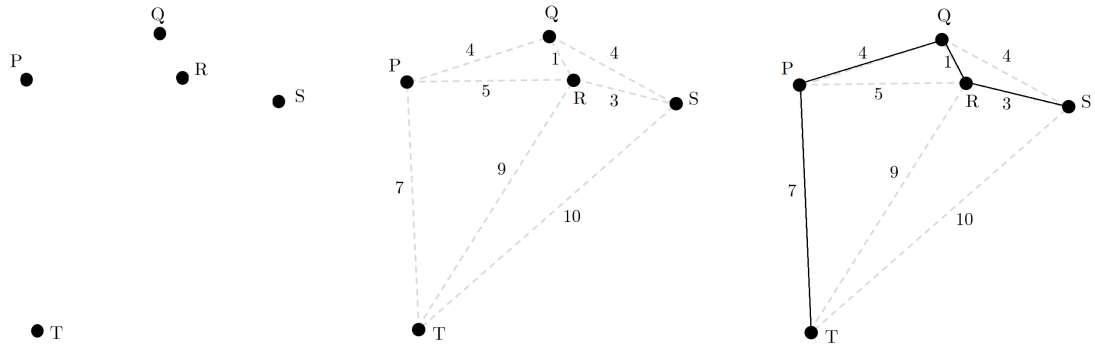


Figure 3.5: Euclidean minimum spanning tree

Prim's algorithm first selects the minimum cost edge in G, which in this case is $\{Q, R\}$. Then at any stage, we seek an edge of minimum cost which has one vertex in the tree so far, and one vertex not in the tree, and we add that edge until a spanning tree is obtained. So in the example, we must consider the edges $\{Q, P\}, \{Q, S\}, \{Q, T\}$ and $\{R, S\}, \{R, P\}, \{R, T\}$, and add an edge of minimum cost. An edge of minimum cost is $\{R, S\}$, so our tree so far has edges $\{Q, R\}$ and $\{R, S\}$. Next we consider the edges $\{Q, P\}, \{Q, T\}, \{R, P\}, \{R, T\}, \{S, P\}, \{S, T\}$ and add one of minimum cost. The edge $\{Q, P\}$ has minimum cost, so now our tree has edges $\{Q, R\}, \{R, S\}, \{Q, P\}$. The reader will check that $\{P, T\}$ is added next, and now we have a minimum cost spanning tree with edges $\{Q, R\}, \{R, S\}, \{Q, P\}, \{P, T\}$, with a total cost of $1 + 3 + 4 + 7 = 15$. This tree is shown in the rightmost image in Figure 3.5.

Kruskal's algorithm starts with two sets of edges, $E = E(G)$ and $F = \emptyset$. We add $\{Q, R\}$ to F, and remove $\{Q, R\}$ from E, since $\{Q, R\}$ has minimum cost. At any stage, we add an edge of E of minimum cost to F out of all edges $e \in E$ such that $F \cup \{e\}$ is a forest (i.e. contains no cycles). The reader will check that we add edges to F in exactly the same order as we did when building the tree in Prim's algorithm above.

There are some surprising open problems on Euclidean spanning trees: for instance, it is believed that if S is a finite set of points inside a unit square, then a Euclidean minimum spanning tree for S always has total length at most 3.

3.6 Dijkstra's Algorithm*

Let G be a graph and let ω be a ***weight function*** on the edges of the graph: this is a function $\omega : E(G) \to [0, \infty)$. For a vertex $v \in V(G)$ (the ***source***), we seek to find for every $u \in V(G)$ a shortest path from u to v, where the length of a path is the sum of the weights on its edges. When all weights equal 1, we are finding paths of shortest length from v to every other vertex, and the breadth-first search algorithm gives all such paths. In general, one may use ***Dijkstra's Shortest Path Algorithm*** to find the shortest paths in weighted graphs. There are also algorithms allowing negative weights, such as the ***Floyd-Warshall Algorithm*** and ***Bellman-Ford Algorithm***, but we do not discuss these here.

Dijkstra's Algorithm on a weighted graph G is as follows. Initially, let $P = \{v\}$ and $Q = V(G)\backslash\{v\}$, and define $\ell(v) = 0$ and $\ell(u) = \infty$ for $u \in Q$. As we proceed, the label $\ell(u)$ represents the smallest weight of a path discovered so far from v to u.

At any stage, we have the last vertex u added to P, and we have $Q = V(G) \setminus P$. For each neighbor $w \in Q$ of u, replace $\ell(w)$ with $\min\{\ell(w), \ell(u) + \omega(u, w)\}$. Then add the neighbor $w \in Q$ of u with the smallest label to P (and remove it from Q) and repeat. If no vertex of P has a neighbor in Q, then set $P = V(G)$. The algorithm terminates when $P = V(G)$. The pseudocode is as follows:

```
1:  function Dijkstra(Graph, source):
2:  for each vertex u in Graph:
3:      dist[u] ← infinity
4:      previous[u] ← undefined
5:  dist[source] ← 0
6:  Q ← the set of all nodes in Graph
7:  while Q is not empty:
8:      u ← node in Q with smallest dist[]
9:      remove u from Q
10:     for each neighbor w of u:
11:         alt ← dist[w] + ω(u,w)
12:         if alt < dist[w]
13:             dist[w] ← alt
14:             previous[w] ← u
15: return previous[], dist[]
```

Example 14. Use Dijkstra's Algorithm to find shortest paths from vertex d to all other vertices in the weighted graph below:

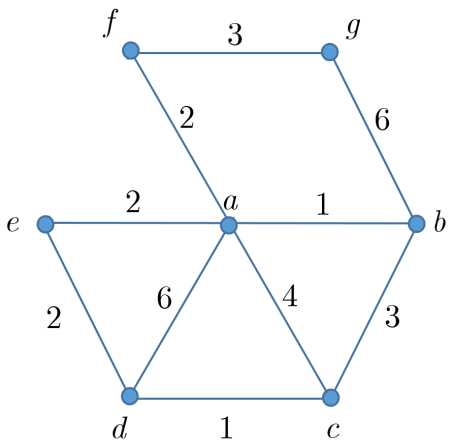


Figure 3.6: Dijkstra's Algorithm

We first assign all vertices except d the label ∞, whereas $\ell(d) = 0$. We also initialize the 'previous' array to $p(v) =$ undefined for all vertices, and set $P = \{d\}$.

Then we change the labels of the neighbors of d in $Q = V(G) \setminus P$ as follows:

$$\ell(a) = 6 \quad \ell(c) = 1 \quad \ell(e) = 2$$

and we set the 'previous' array to $p(a) = p(c) = p(e) = d$.

Since c has the smallest label, let $P = \{c, d\}$. We change the labels of the neighbors of c in Q as follows:

$$\ell(b) = 4 \quad \ell(a) = 5 \qquad \text{and we set } p(b) = p(a) = c.$$

Since e has the smallest label, let $P = \{e, c, d\}$. We change the labels of the neighbors of e in Q as follows:

$$\ell(a) = 4 \qquad \text{and we set } p(a) = c.$$

Since a has the smallest label, let $P = \{a, c, d, e\}$. We change the labels of the neighbors of a in Q as follows:

$$\ell(f) = 6 \qquad \text{and we set } p(f) = a.$$

Since b has the smallest label, let $P = \{a, b, c, d, e\}$. We change the labels of the neighbors of b in Q as follows:

$$\ell(g) = 10 \qquad \text{and we set } p(g) = b.$$

Since f has the smallest label, let $P = \{a, b, c, d, e, f\}$. We finally let $\ell(g) = 8$, $p(g) = f$, and $P = \{a, b, c, d, e, f, g\} = V(G)$.

This completes the proof, with the labelling of all the vertices being

$$\ell(a) = 4,\ \ell(b) = 4,\ \ell(c) = 1,\ \ell(d) = 0,\ \ell(e) = 2,\ \ell(f) = 6,\ \ell(g) = 10.$$

These are the weights of the shortest paths from d to each vertex. The 'previous' array is

$$p(a) = e,\ p(b) = c,\ p(c) = d,\ p(d) = \text{undefined},\ p(e) = d,\ p(f) = a,\ p(g) = f.$$

The shortest paths can be listed by backtracking using this array. For the shortest path from d to f, work backwards starting from f:

$$p(f) = a; \quad p(a) = e; \quad p(e) = d.$$

Reversing those gives the path *deaf*. Here is a table of the shortest paths:

Pair of vertices	Shortest path
d to a	$d\,e\,a$
d to b	$d\,c\,b$
d to c	$d\,c$
d to d	d
d to e	$d\,e$
d to f	$d\,e\,a\,f$
d to g	$d\,e\,a\,f\,g$

Figure 3.7: Shortest paths

We have not invested time in investigating the running time of implementations of Dijsktra's Algorithm. Using ***minimum priority queues***, the running time can be made to be of order $|E(G)| + |V(G)| \log |V(G)|$.

3.7 Exercises

Question 3.1° Prove that every tree is bipartite.

Question 3.2° Determine which of the graphs below is bipartite, and find a bipartition of each bipartite graph.

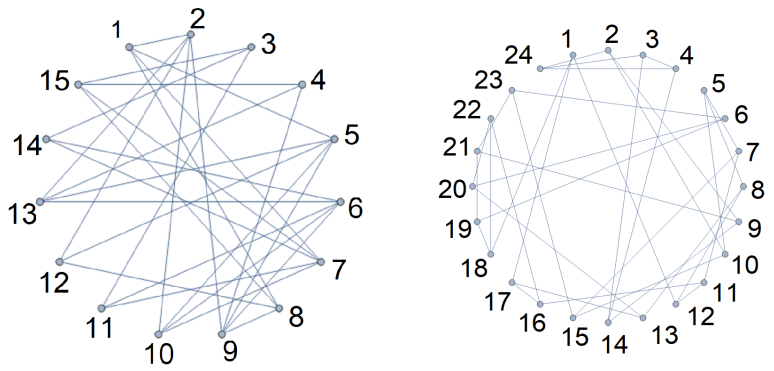


Figure 3.8: Testing bipartiteness

Question 3.3°

(a) Use the breadth-first and depth-first search algorithms to find spanning trees of the graph in Figure 1.21, with the root of the tree at v_1 and the ordering of the vertices being $(v_1, v_2, \ldots, v_9)$. Show all your work.

(b) Write down the heights of the BFS and DFS trees.

(c) Write down the radius of the graph.

(d) Write down the diameter of the graph.

Question 3.4° Apply Prim's and Kruskal's Algorithms to the graph in Figure 3.6 to determine minimum weight spanning trees in the graph. Apply Dijkstra's Algorithm to find all shortest paths from vertex g.

Question 3.5° Find breadth-first search and depth-first search trees in the ***dodecahedron graph*** shown below.

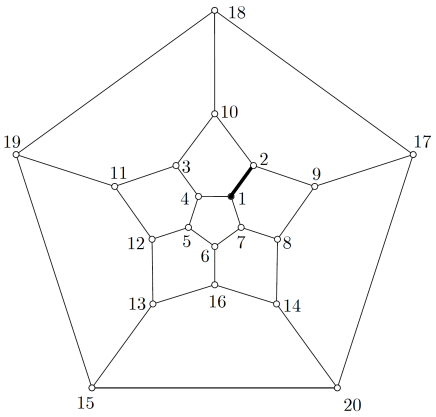


Figure 3.9: Dodecahedron graph

Question 3.6° Let $G = (V, E)$ be a graph and $d(x, y)$ the distance between two vertices. Prove that (V, d) is a ***metric space***.

Question 3.7° Give a description of all graphs of radius 1 and diameter 2.

Question 3.8. Prove that the vertices of an n-vertex connected graph can be ordered $(v_1, v_2, \ldots, v_n)$ so that for $i > 1$, v_i has at least one neighbor v_j with $j < i$.

Question 3.9. Let G be a digraph such that every vertex has in-degree at least $k \geq 1$. Prove that G contains a ***directed cycle*** of length at least $k + 1$.

Question 3.10. Prove any n-vertex graph with $m \geq n$ edges has at least $m - n + 1$ cycles.

Question 3.11. Let $G = (V, E)$ be a connected graph. Prove that for $1 \leq k \leq |V(G)|$, G has a connected subgraph with exactly k vertices.

Question 3.12. Prove that for every graph G with radius r and diameter d, $r \leq d \leq 2r$. For each pair of positive integers r and d with $r \leq d \leq 2r$, give an example of a graph with radius r and diameter d.

Question 3.13. Let G be an n-vertex graph with $n \geq 2$ and $\delta(G) \geq (n-1)/2$. Prove that G is connected and that the diameter of G is at most two.

Question 3.14. Let T be an tree, and let Z be the set of vertices of T such that every vertex of T is at distance at most $\mathrm{rad}(T)$ from some vertex in Z[8]. Prove that $|Z| \leq 2$.

[8]This is called the ***center*** of T.

4 Structure of connected graphs

We just gave three equivalent characterizations of trees in Proposition 3.1.3. In general, we would like to describe how to build connected graphs: what are the basic building blocks of graphs? In this section, we visit basic theorems of ***structural graph theory***, including notions of block decomposition, ear decomposition, and Menger's Theorems.

4.1 Block decomposition*

The main result in this section will be the block decomposition theorem. We require some definitions. A ***cut vertex*** of a graph G is a vertex x such that $G-\{x\}$ has more components than G. A ***block*** of a graph is a maximal connected subgraph with no cut vertex – a subgraph with as many edges as possible and no cut vertex. So a block is either K_2 or is a graph which contains a cycle. For example in a tree, every block is K_2. The block decomposition of a graph is just the set of all the blocks of the graph. An example of a block decomposition is shown below.

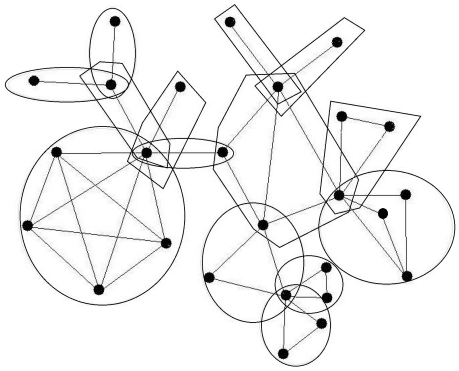


Figure 4.1: Blocks

In the picture, there are fourteen blocks. Seven of the blocks are K_2, four of the blocks are triangles, one of the blocks is K_5, and there are two other blocks. The block decomposition theorem says that block decompositions of graphs have a "tree-like structure". To make this precise, given a graph G, we form a new graph $\mathcal{B}$ where the vertices of $\mathcal{B}$ consist of all cut vertices of G and also all blocks of G, and where a block is joined to all cut vertices of G contained in it. An example of this graph is shown below for the figure above:

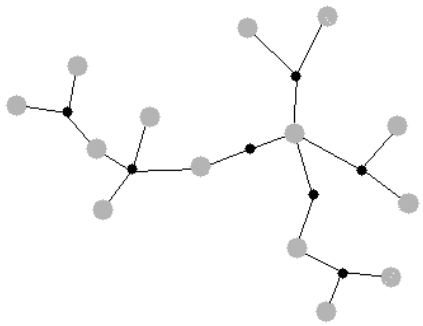


Figure 4.2: The graph $\mathcal{B}$

In the figure, the black vertices represent cut vertices of G, and the grey vertices represent blocks of G. Here is the block decomposition theorem:

Theorem 4.1.1 *Let G be a connected graph. Then $\mathcal{B}$ is a tree.*

Proof ▷ By adding edges inside the blocks of G, we do not change $\mathcal{B}$, so we can assume every block of G is a complete graph. Since G is connected, clearly $\mathcal{B}$ is connected too. Now we show $\mathcal{B}$ has no cycles. The vertices of a cycle $\mathcal{C} \subseteq \mathcal{B}$ are alternately blocks of G and cut vertices of G, by definition of $\mathcal{B}$. This is shown in the figure below:

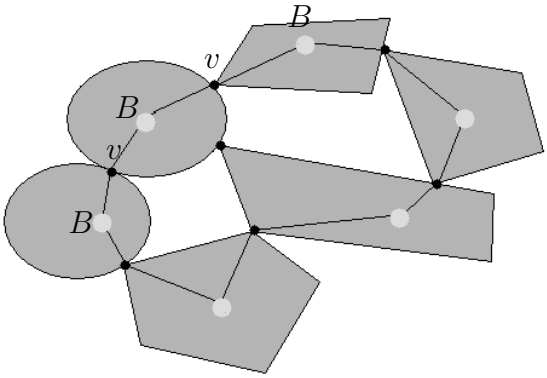


Figure 4.3: Cycle in $\mathcal{B}$

In the figure, the blocks are shown as grey dots and labelled B and the cut vertices are black dots labelled v. Let the cut vertices of G in order along $\mathcal{C}$ be $v_0, v_1, \ldots, v_k, v_0$. Then $v_0v_1v_2 \ldots v_kv_0$ is a cycle $C \subseteq G$. If $B \in \mathcal{C}$, then $B \cup C$ is a subgraph of G consisting of the

complete graph B together with the cycle C containing an edge of B and at least one edge not in B. Therefore $B \cup C$ has no cut vertex, and must be a block of G. However, this contradicts the definition that B is block. □

Using this theorem, we give a first example of a ***structure theorem*** in graph theory. We say that a uv-path P in a graph G is ***internally disjoint*** from a subgraph H of G if $V(P) \cap V(H) = \{u, v\}$. Define a ***theta graph*** to be any graph consisting of the union of three pairwise internally disjoint paths between two vertices.

Proposition 4.1.2 *Let G be a connected graph containing no theta graph. Then every block of G is a cycle or K_2 and G is a tree of cycles and K_2s.*

Proof ▷ Let B be a block of G. If $B \neq K_2$, then B contains a cycle, C. If $B \neq C$, then there is a path P in B such that $P \cup C$ is a theta graph: namely, pick a shortest path in $B - E(C)$ between two vertices of C. Therefore $B = K_2$ or B is a cycle. We know by the last result that G is then a tree of cycles and K_2s. □

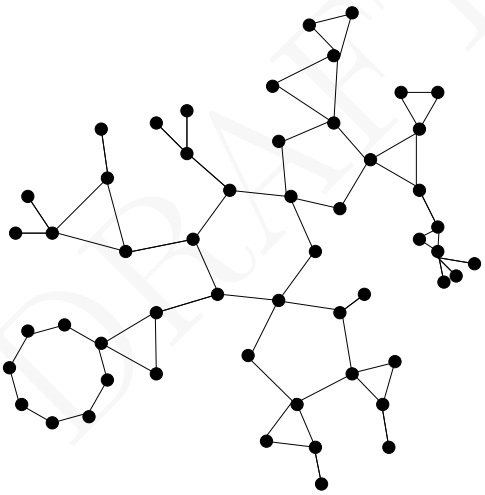


Figure 4.4: Tree of cycles and K_2s

4.2 Structure of blocks : ear decomposition*

In this section we will give method for decomposing blocks, called ***ear-decomposition***. Let $G \neq K_2$ be a block and $P \subset G$ a path all of whose internal vertices have degree two in G and whose ends have degree at least three in G. Then P is called an ear of G (see Figure 4.5). Note that an ear can be a single edge.

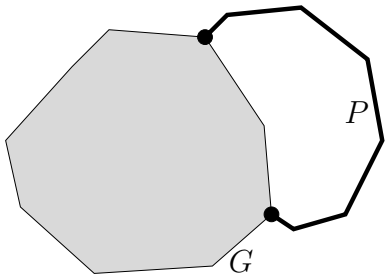


Figure 4.5: Ear decomposition

The main theorem in this section says that blocks can be built from a cycle by adding ears. More precisely, a graph G has an ear decomposition if there is sequence of subgraphs of G, say $G_0 \subset G_1 \subset \cdots \subset G_t$ such that G_0 is a cycle, $G_t = G$, and G_i is obtained from G_{i+1} by removing the internal vertices of some ear in G_{i+1} or, if the ear is a single edge, deleting this edge.

Theorem 4.2.1 (Ear decomposition)
A graph $G \neq K_2$ is a block if and only if it has an ear-decomposition.

We prove Theorem 4.2.1 using the notion of ***equivalence relations***.

Definition 4.2.2 *An equivalence relation on a set S is a set R of ordered pairs of elements of S with the following properties:*

1. $(a, a) \in R$
2. *if* $(a, b) \in R$ *then* $(b, a) \in R$.
3. *if* $(a, b), (b, c) \in R$ *then* $(a, c) \in R$.

The properties 1, 2 and 3 of an equivalence relation are called ***reflexivity***, ***symmetry*** and ***transitivity***, respectively. If $(a, b) \in R$, we say that a and b are equivalent under R.

Example 15. For instance, if $G = (V, E)$ is a graph and

$$R = \{(u, v) \in V \times V : u \text{ and } v \text{ are joined by a path}\},$$

then R is an equivalence relation, and any two vertices in a component of G are equivalent under R. To prove this, the main thing to check is transitivity: that if u and v are joined by a path and v and w are joined by a path then also u and w are joined by a path. It is convenient, rather than writing $(u, v) \in R$, to write $u \sim v$.

For the proof of Theorem 4.2.1, we define an equivalence relation $\sim$ on the edge set of a graph $G = (V, E)$ as follows: for $e, f \in E$, $e \sim f$ if and only if $e = f$ or some cycle in G contains both e and f. The following lemma says that $\sim$ is indeed an equivalence relation:

Lemma 4.2.3 *For any graph G, the relation $\sim$ is an equivalence relation on $E(G)$.*

Proof ▷ By definition we know $e \sim e$ for any edge $e \in E(G)$, and $e \sim f$ is clearly the same as $f \sim e$. It remains to verify transitivity: we have to prove that if some cycle $C \subset G$ contains e and f, and some cycle $D \subset G$ contains f and g, then some cycle in G contains both e and g. Consider the path $P = D - f$. then there is a path $Q \subseteq P$ containing g whose first and last vertices u, v are in $V(C)$ but with no other vertices in C. Clearly $C \cup Q$ is a theta graph containing e and g, consisting of internally disjoint uv-paths Q, R and S such that $R \cup S = C$. Then $Q \cup S$ or $Q \cup R$ is the required cycle containing both e and g. □

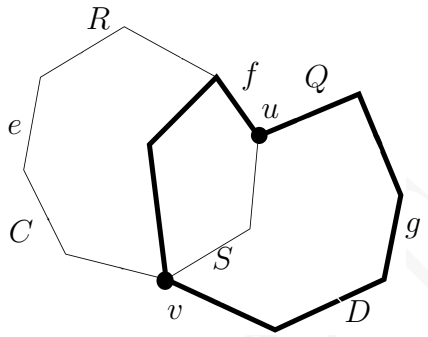


Figure 4.6: Transitivity of $\sim$

Theorem 4.2.4
For a graph G with at least three vertices and no isolated vertices, the following three statements are equivalent:

1. *G is a block*
2. *every two edges of $E(G)$ are in a common cycle*
3. *any two vertices of $V(G)$ are in a common cycle.*

Proof ▷ We show first that 1 implies 2. Let $e_0 = \{x_0, x_1\}$ and $e_k = \{x_k, x_{k+1}\}$ be edges of G. We have to show $e_0 \sim e_k$. Since G is connected, there is a path $P \subset G$ of the form $x_0e_0x_1e_1x_2e_2 \ldots x_ke_kx_{k+1}$. Since $G - \{x_i\}$ is connected, there is a path in $G - \{x_i\}$ from x_{i-1} to x_{i+1}, which means that e_{i-1} and e_i are contained in a common cycle in G, for all i. In other words, $e_{i-1} \sim e_i$ for all i. But by transitivity, this means $e_0 \sim e_k$, as required. So 1 implies 2. To prove that 2 implies 3, let $u, v \in V(G)$ and select an edge e containing u and an edge $f \neq e$ containing v (this edge exists because G has at least three vertices). Then $e \sim f$ by assumption, so some cycle in G contains both u and v, as required. So 2 implies 3. Finally, to show 3 implies 1, $G - \{x\}$ is connected for any $x \in V(G)$, otherwise two vertices in different components of $G - \{x\}$ are not on a cycle in G, a contradiction. □

Proof ▷ OF THEOREM 4.2.1. Suppose G is a block, and let H be a maximal subgraph of G with an ear decomposition. Since G contains a cycle, H certainly exists. Suppose, for a contradiction, that $H \neq G$. Then there exists an edge $e \in E(G) \backslash E(H)$. If e joins two vertices of H, then $H + e$ has an ear decomposition, contradicting the maximality of H. Therefore e has an end not in H. Let f be any edge of H. Then e and f are contained in a common cycle, C, by Theorem 4.2.4 part 2. In particular, C contains at least two vertices of H, so there is a path $P \subset C$, internally disjoint from H, and with both ends in H. But then P is an ear of $H \cup P$, contradicting the maximality of H. We conclude that $H = G$. The proof of the converse statement is left as an exercise. □ ◁◁

The theorem on ear decomposition is very useful for proving statements about blocks by induction.

4.3 Decomposing bridgeless graphs*

Here we prove an ear-decomposition theorem for graphs with no bridges. It cannot be the same as for blocks, since the graph consisting of the union of two cycles sharing exactly one vertex is not a block and does not have an ear decomposition in the sense of the last section. The new ear decomposition is described as follows: an ear decomposition of a graph G is a sequence of subgraphs of G, say $G_0 \subset G_1 \subset \cdots \subset G_t$ such that G_0 is a cycle, $G_t = G$, and $G_{i+1} = G_i \cup P$ for a path P internally disjoint from G_i with both ends in $V(G_i)$, or $G_{i+1} = G_i \cup C$ for a cycle C with exactly one vertex in common with G_i.

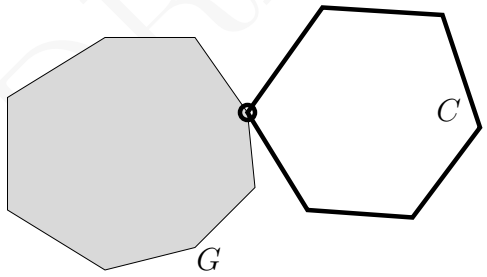


Figure 4.7: New ear decomposition

The proof of the ear-decomposition theorem is similar to that of Theorem 4.2.1:

Theorem 4.3.1
A graph is bridgeless if and only if it has an ear decomposition.

Proof ▷ Suppose G has an ear decomposition. Let e be an edge of G. It is sufficient to prove that e is contained in a cycle – then e cannot be a bridge of G by Lemma 3.1.1. If e is

not in a cycle, then e is a bridge of G. Let P be an ear of G. Since e is a bridge of G, P can never contain e, since there is a cycle in G containing all edges of P. Therefore e survives our procedure, but then e must be in a cycle, a contradiction. So G must be 2-edge-connected. Now suppose that G is 2-edge-connected. Then G contains a cycle, so G has a subgraph with an ear decomposition. So we can take a maximal subgraph H of G so that H has an ear decomposition. We'll show $H = G$. If $H \neq G$, then there is an edge e of G joining a vertex of H to a vertex of G not in H, otherwise $H + e$ has an ear decomposition. This edge is in a cycle C, by Lemma 3.1.1. If C contains only one vertex of H, then $H \cup C$ has an ear-decomposition, a contradiction. So C contains two vertices in H, and we find a shortest path between two vertices of H in G to add to H, a contradiction. So $H = G$. □

4.4 Contractible edges*

A ***vertex cut*** of a graph G is a set of vertices whose removal from G gives a disconnected graph. A graph G is ***k-connected*** if every vertex cut has size at least k. In the last section, we gave a structural characterization of connected graphs via blocks. Using the ear decomposition theorem, it is possible to prove the following statement. We recall that if G is a graph and $e \in E(G)$, then G/e is the graph obtained by ***contraction*** of the edge e. If G is a k-connected graph and $e \in E(G)$, and G/e is also k-connected, then e is called a ***contractible edge*** of G.

Theorem 4.4.1 *Let $G \neq K_3$ be 2-connected, and let $e \in E(G)$. Then G/e or $G - e$ is 2-connected. Furthermore, G contains a contractible edge.*

Tutte proved a similar theorem for 3-connected graphs:

Theorem 4.4.2 *Let $G \neq K_4$ be 3-connected, and let $e \in E(G)$. Then G/e or $G - e$ is 3-connected. Furthermore, G contains a contractible edge.*

This theorem cannot be extended to k-connected graphs with $k \geq 4$: there are infinitely many k-connected graphs with no contractible edges. In general, there is no good structural characterization of k-connected graphs. The next main topic is Menger's Theorem, which describes connectivity in terms of internally disjoint paths. ◁◁

4.5 Menger's Theorems

Let G be a graph and let $A, B \subseteq V(G)$. An ***AB-path*** is a uv-path $P \subseteq G$ with $V(P) \cap A = \{u\}$ and $V(P) \cap B = \{v\}$. A set $W \subset V(G)$ is called an ***AB-separator*** if there exists no

AB-path P in $G - W$. Note that we include the possibility that a path consists of a single vertex. Let $\kappa(A, B)$ denote the minimum size of an AB-separator.

Example 16. We consider some general examples of separators. By definition, every vertex of $A \cap B$ is contained in an AB-separator, so $\kappa(A, B) \geq |A \cap B|$ for any two sets $A, B \subset V(G)$. Furthermore, $\kappa(A, B) \leq \min\{|A|, |B|\}$ for any sets $A, B \subseteq V(G)$, and so $\kappa(A, A) = |A|$ for any set $A \subseteq V(G)$. As another example, if A and B are in different components of a graph G, then clearly $\kappa(A, B) = 0$, since there are no AB-paths in the graph – in other words, $W = \emptyset$ is an AB-separator.

Example 17. Now we consider a specific example. For the graph G shown below with $A, B, C \subset V(G)$, and $W = \{w\}$, W is an AB-separator, since there is no AB-path in $G - W$. It is also the case that $W = B$ is an AB-separator, so $\kappa(A, B) \leq 1$. On the other hand, G itself has plenty of AB-paths, so $\kappa(A, B) \geq 1$. We conclude $\kappa(A, B) = 1$. Similarly, $\kappa(A, A) = 2$ since $W = A$ is an AA-separator, and $\kappa(B, B) = 1$, since $W = B$ is a BB-separator. Now let $A' = C \cup \{x\}$ and $B' = B \cup \{w\}$. Then $\kappa(A', B') \leq \min\{|A'|, |B'|\} = 2$. On the other hand, we have to remove a vertex from the $A'B'$-path through w and a vertex from the $A'B'$-path through x in order that there are no remaining $A'B'$-paths, and therefore $\kappa(A', B') \geq 2$. We conclude $\kappa(A', B') = 2$.

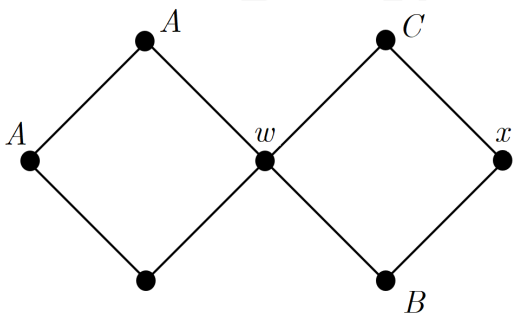


Figure 4.8: AB-separators

An ***AB-path*** is a uv-path $P \subseteq G$ such that $V(P) \cap A = \{u\}$ and $V(P) \cap B = \{v\}$. Paths $P, Q \subseteq G$ are ***vertex disjoint*** if $V(P) \cap V(Q) = \emptyset$. Here is Menger's Theorem:

Theorem 4.5.1 *Let G be a graph and $A, B \subseteq V(G)$. Then $\kappa(A, B)$ equals the maximum number of pairwise vertex disjoint AB-paths.*

Proof ▷ We proceed by induction on $|E(G)|$. If $|E(G)| = 0$, then $\kappa(A, B) = |A \cap B|$, and each vertex of $A \cap B$ is an AB-path, as required. If $|E(G)| > 0$, let $e = \{x, y\} \in E(G)$ and suppose $\kappa(A, B) = k$. By induction, the minimum size of an AB-separator W in $G - e$

equals the maximum number of vertex disjoint AB-paths in $G-e$. Since $W\cup\{x\}$ is an AB-separator in G, $|W\cup\{x\}|\geq k$ and therefore $|W|\geq k-1$. If $|W|=k$, then there are k vertex disjoint AB-paths in $G-e$, and hence in G, so in that case we are done. If $|W|=k-1$ and $W=\{w_1,w_2,\ldots,w_{k-1}\}$, let $S=W\cup\{x\}$ and $T=W\cup\{y\}$. Then an AS-separator in $G-e$ is an AB-separator in G, so by induction there are k vertex disjoint AS-paths $P_1,P_2,\ldots,P_k$ in $G-e$. Similarly, there are k vertex disjoint SB-paths $Q_1,Q_2,\ldots,Q_k$ in $G-e$. We may assume $V(P_i)\cap W=V(Q_i)\cap W=\{w_i\}$ for $1\leq i\leq k-1$ and $V(P_k)\cap S=\{x\}$ and $V(Q_k)\cap T=\{y\}$. Now as shown in Figure 4.9, $V(P_i)\cap(V(Q_1)\cup V(Q_2)\cup\cdots\cup V(Q_k))=\{w_i\}$ for $1\leq i\leq k-1$. Therefore the paths $R_i=P_i\cup Q_i$ for $1\leq i\leq k-1$ and $P_k\cup Q_k+e$ are vertex disjoint AB-paths. □

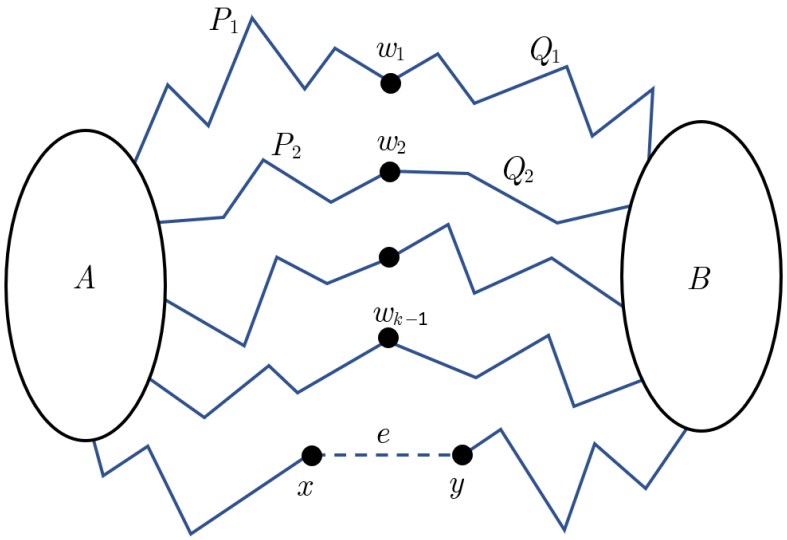


Figure 4.9: Constructing AB-paths

The reader should test the validity of this theorem on Figure 4.8. ◁◁

A set $W\subseteq V(G)$ is called a ***vertex cut*** if $G-W$ is disconnected. For example, if $W=\{w\}$ in Figure 4.8, then W is a vertex cut. A graph G is ***k-connected*** if and only if G has at least $k+1$ vertices and no vertex cut has size less than k. The complete graph K_n with $n\geq k+1$ is therefore k-connected. If P is a path, let $\text{end}(P)$ denote the pair of end vertices of P, and let $\text{int}(P)=V(P)\backslash\text{end}(P)$ denote the set of ***internal vertices*** of P. Paths $P,Q\subset G$ are ***internally disjoint*** if $V(P)\cap V(Q)\subseteq\text{end}(P)\cap\text{end}(Q)$ – in other words, the only vertices that P and Q may share are vertices which are end vertices of both P and Q.

If there are at least k pairwise internally disjoint uv-paths for every $u,v\in V(G)$, then for

any set $W \subseteq V(G)\backslash\{u,v\}$ such that u and v are in different components of $G-W$, $|W| \geq k$, since W must contain at least one vertex from each path in a collection of k internally disjoint paths. In particular, G is k-connected. We show using Menger's Theorem that the converse is true. Let $\kappa(u,v)$ denote the minimum size of a set $W \subseteq V(G)\backslash\{u,v\}$ such that u,v are in different components of $G-W$. Let $k(u,v)$ denote the maximum number of internally disjoint uv-paths. Here is the so-called vertex form of Menger's Theorem:

Theorem 4.5.2 (Menger's Theorem - Vertex Form) *Let G be a graph and let $u,v \in V(G)$ be non-adjacent vertices. Then $\kappa(u,v) = k(u,v)$. In particular, if G has at least $k+1$ vertices, then G is k-connected if and only if for every pair u,v of non-adjacent vertices of G, there are at least k internally disjoint uv-paths.*

Proof ▷ To disconnect u from v, we must remove at least one vertex from each of a set of $k(u,v)$ internally disjoint uv-paths, so $\kappa(u,v) \geq k(u,v)$. Now we show $\kappa(u,v) \leq k(u,v)$ by finding $\kappa(u,v)$ internally disjoint uv-paths using Theorem 4.5.1. Form a new graph H from $G-\{u\}-\{v\}$ by adding a set A of $\kappa(u,v)$ vertices adjacent to all vertices in $N_G(u)$, and a set B of $\kappa(u,v)$ vertices adjacent to all vertices in $N_G(v)$, as shown in Figure 4.10. We claim in H that $\kappa(A,B) = |A|$. Since A is an AB-separator, $\kappa(A,B) \leq |A|$. Now suppose W is an AB-separator in H and let $X = W\backslash(A \cup B)$. We aim to show $|W| \geq |A|$. Suppose, for a contradiction, that $|W| < |A|$ – say $a \in A\backslash W$ and $b \in B\backslash W$. If there exists a uv-path P in $G-X$, then there exists an ab-path in $H-W$, since $N_H(a) = N_G(u)$ and $N_H(b) = N_G(v)$. This contradicts that W is an AB-separator in H, so there is no uv-path in $G-X$. In particular, u and v are in different components of $G-X$, so $|X| \geq \kappa(u,v)$. Since $|W| \geq |X|$, we conclude $|W| \geq |A|$, and so in H, $\kappa(A,B) = |A|$. By Theorem 4.5.1, there exist $|A| = \kappa(u,v)$ vertex disjoint AB-paths in H. For each path, we remove the ends and then add u and v to obtain $\kappa(u,v)$ internally disjoint uv-paths in G, as required. □

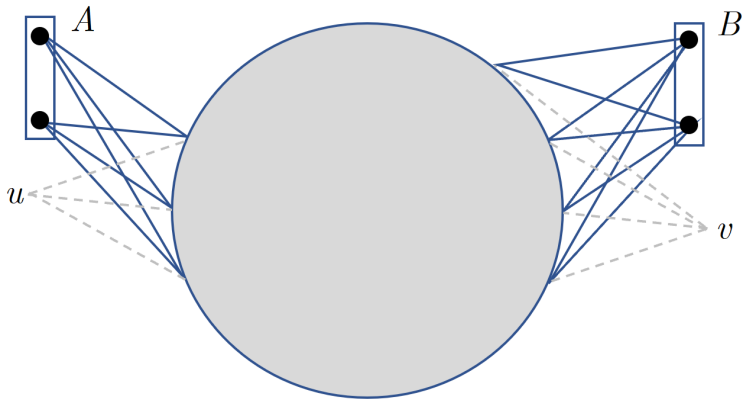


Figure 4.10: Proof of Menger's Theorem

An ***edge cut*** of a graph G is a set of edges whose removal from G gives a disconnected graph. A graph G is ***k-edge-connected*** if every edge cut of G has size at least k. For vertices $u, v \in V(G)$, let $\lambda(u, v)$ denote the minimum size of a set $L \subseteq E(G)$ such that u and v are in different components of $G - L$. Then in a k-edge-connected graph, $\lambda(u, v) \geq k$ for all $u, v \in V(G)$. Let $\ell(u, v)$ denote the maximum number of edge-disjoint uv-paths. The following is another consequence of Menger's Theorem, sometimes called the edge form of Menger's Theorem:

Theorem 4.5.3 (Menger's Theorem - Edge Form) *Let G be a graph and $u, v \in V(G)$. Then $\lambda(u, v) = \ell(u, v)$. In particular, G is k-edge-connected if and only if for every $u, v \in V(G)$, there are at least k pairwise edge-disjoint uv-paths.*

We will also derive Menger's theorems from the Max-Flow Min-Cut Theorem later in the course. The above edge-form of Menger's Theorem given above may also be derived by applying the vertex-form of Menger's theorem in the ***line graph*** – see Question 1.6.

4.6 Vertex and edge connectivity

Let G be a graph. We define $\lambda(G)$, the ***edge-connectivity*** of G, to be the minimum size of an edge cut in G i.e. the minimum size of $L \subset E(G)$ such that $G - L$ is disconnected. In particular,

$$\lambda(G) = \min\{\lambda(u, v) : u, v \in V(G)\}$$

where the minimum is over distinct vertices $u, v \in V(G)$. Thus a graph is ℓ-edge-connected if and only if $\lambda(G) \geq \ell$. If G is not a complete graph, then we define $\kappa(G)$, the ***vertex-connectivity*** of G, to be the minimum size of a vertex cut in G i.e. the minimum size of a set $S \subset V(G)$ such that $G - S$ is disconnected. In particular,

$$\kappa(G) = \min\{\kappa(u, v) : u, v \in V(G), \{u, v\} \notin E(G)\}.$$

Note here the minimum is over distinct non-adjacent vertices $u, v \in V(G)$. Thus a graph is k-edge-connected if and only if $\kappa(G) \geq k$. If $G = K_n$, we define $\kappa(G) = n - 1$. It should be intuitively clear that $\kappa(G) \leq \lambda(G)$ for every graph G, since we do more "damage" by removing vertices than by removing edges. The quickest proof is via Menger's Theorem:

Corollary 4.6.1 *For any graph G, $\kappa(G) \leq \lambda(G) \leq \delta(G)$.*

Proof ▷ Since the edges containing a vertex of minimum degree form an edge cut, $\lambda(G) \leq \delta(G)$. Now we prove $\kappa(G) \leq \lambda(G)$. For $u, v \in V(G)$, let $k(u, v)$ be the maximum number of pairwise internally disjoint uv-paths, and $\ell(u, v)$ be the maximum number of pairwise

edge-disjoint uv-paths. Then by the edge form of Menger's Theorem:

$$\lambda(G) = \min\{\lambda(u,v) : u,v \in V(G)\} = \min\{\ell(u,v) : u,v \in V(G)\}.$$

Now $k(u,v) \leq \ell(u,v)$ for all $u,v \in V(G)$, since internally disjoint paths are also edge-disjoint paths, and therefore taking minima over distinct nonadjacent $u,v \in V(G)$:

$$\begin{aligned} \kappa(G) &= \min\{\kappa(u,v) : u,v \in V(G), \{u,v\} \notin E(G)\} \\ &= \min\{k(u,v) : u,v \in V(G)\} \\ &\leq \min\{\ell(u,v) : u,v \in V(G)\} \ = \ \lambda(G). \end{aligned}$$

The reader should check why the first two lines are equal here. □ ◁◁

This corollary can be proved directly, without Menger's Theorems. It is also the case that for any three positive integers $d \geq \ell \geq k$, there exists a graph G with $\kappa(G) = k$, $\lambda(G) = \ell$ and $\delta(G) = d$. ◁◁

Example 18. Consider the graph G shown in Figure 4.8. If v is the leftmost vertex, then we notice $\lambda(v,x) \geq 2$, since removing a single edge does not disconnect the graph (no edge of the graph is a bridge since every edge is in a cycle), whereas $\lambda(v,x) = 2$ since we could remove two edges on w to disconnect v from x. In fact, $\lambda(a,b) = 2 = \ell(a,b)$ for any two vertices a and b in this graph, so $\lambda(G) = 2$. Since the graph is connected, $\kappa(G) \geq 1$. On the other hand, $G - \{w\}$ is disconnected, so $\kappa(G) = 1$. If a and b are in different components of $G - \{w\}$, then $\kappa(a,b) = k(a,b) = 1$, otherwise $\kappa(a,b) = k(a,b) = 2$, since a,b are in a common cycle of length four.

4.7 Fan Lemma and Dirac's Theorem*

We leave the following lemma as an exercise (see the proof of the vertex-form of Menger's Theorem): ◁◁

Lemma 4.7.1 *Let G be a k-connected graph and let A be a set of at least k vertices in G. Then the graph obtained from G by adding a new vertex adjacent to all vertices in A is k-connected.*

If A and B are sets of vertices in a graph G, then an ***AB-path*** is a path with one end in A and the other in B. For $x \in V(G)$ and $|A| = k$, an ***xA-fan*** is a set of paths $P_1, P_2, \ldots, P_k$ such that $V(P_i) \cap V(P_j) = \{x\}$ for $i \neq j$ and each P_i has one end equal to x and the other end in A. This is shown in Figure 4.11 with $A = \{a,b,c,d,e\}$:

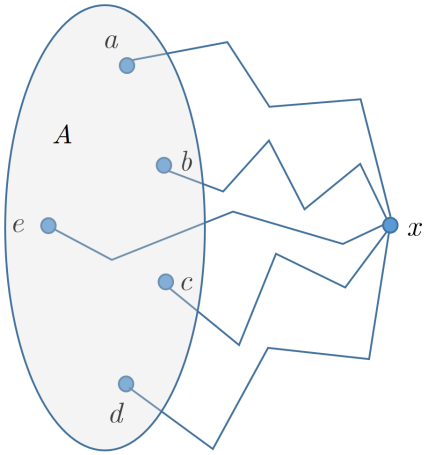


Figure 4.11: An xA-fan

We derive the following from the vertex form of Menger's Theorem:

Corollary 4.7.2 (Fan Lemma) *Let G be k-connected with at least $k+1$ vertices. Then*

1. *for any $A \subset V(G)$ of size k and $x \in V(G) \backslash A$, there exists an xA-fan in G.*
2. *for any $A, B \subset V(G)$ of size k, there exist k pairwise vertex-disjoint AB-paths.*

Proof ▷ To prove (2), let G^* be the graph obtained from G by adding a vertex y adjacent to all vertices in A. Since $|A| \geq k$, Lemma 4.7.1 shows G^* is k-connected. By Menger's Theorem, there exist k pairwise internally disjoint paths between x and y in G^*. Removing y from all of these paths, we have k paths from x to A with only x in common – an xA-fan.

To prove (3), let G^{**} be obtained from G by adding a vertex a adjacent to all vertices in A and a vertex b adjacent to all vertices in B. Since $|A| \geq k$ and $|B| \geq k$, G^{**} is k-connected, via two applications of Lemma 4.7.1. By Menger's Theorem, there are k pairwise internally disjoint ab-paths in G^{**}. Removing a and b from these paths gives k pairwise vertex-disjoint AB paths in G. □

Dirac's Theorem says that through any k vertices in a k-connected graph we can find a cycle, when $k \geq 2$.

Theorem 4.7.3 (Dirac's Theorem) *Let G be a k-connected graph, where $k \geq 2$, and let X be a set of k vertices of G. Then there exists a cycle in G containing X.*

Proof ▷ By induction on k. If $k = 2$, then every pair of vertices of G is joined by two internally disjoint paths by Menger's Theorem, so every pair of vertices is contained in a cycle (this is also Theorem 2.2.4 (3)).

Now let G be a k-connected graph, where $k > 2$, and let $X = \{x_1, x_2, \ldots, x_k\}$ be a set of k vertices of G. Since G is also $k-1$ connected, there is a cycle C containing $\{x_1, x_2, \ldots, x_{k-1}\}$. We can assume that the order in which these vertices appear on C is $x_1, x_2, \ldots, x_{k-1}$. We consider first the case that C has length $k-1$. Since $|V(G)| \geq k+1$, there is a vertex $x \in V(G)\backslash X$. By the Fan Lemma (2), there are k paths from x_k to $\{x_1, x_2, \ldots, x_{k-1}, x\}$ with only the vertex x_k in common. Now if P_i is the path from x_k to x_i, then $C \cup P_i \cup P_{i+1}$ is a cycle containing X, as required. Next we consider the case $|V(C)| \geq k$. If $x_k \in V(C)$, we are done, so we assume $x_k \notin V(C)$. Then for any $x \in V(C)\backslash\{x_1, x_2, \ldots, x_{k-1}\}$, there are k paths from x_k to $\{x_1, x_2, \ldots, x_{k-1}, x\}$ with only the vertex x_k in common, by the Fan Lemma (2). Let these paths be $P_1, P_2, \ldots, P_{k-1}, P_k$. Let y_i denote the first vertex of P_i on C and let $Q_i \subset P_i$ denote the path from x_k to y_i. For some i, j, there is a path $P \subset C$ joining y_i to y_j containing none of the vertices $\{x_1, x_2, \ldots, x_{k-1}\}$ (see Figure 4.12). Now delete the vertices of P between y_i and y_j from C to get a path $Q \subset C$. Then $Q \cup Q_i \cup Q_j$ is a cycle containing X (define $Q_{k+1} = Q_1$). This proves the result. □

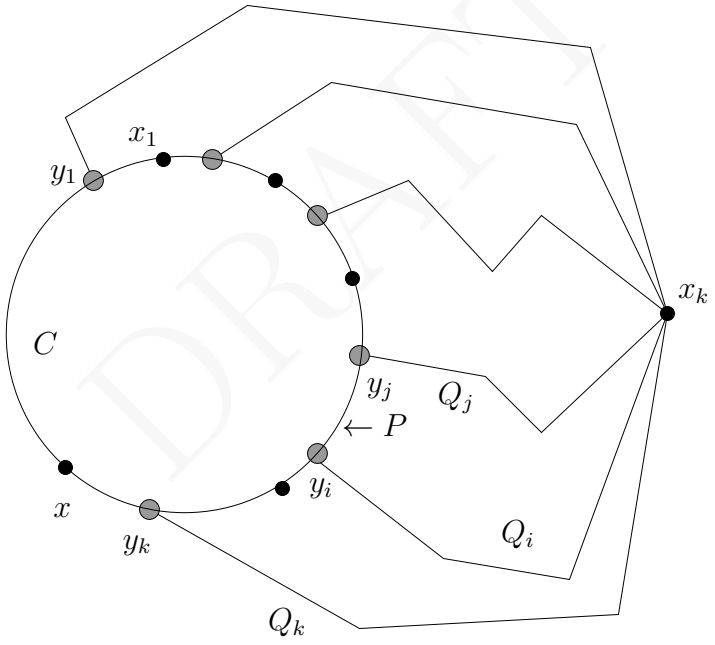


Figure 4.12: Dirac's Theorem

4.8 Exercises

Question 4.1° In Figure 1.21, a graph G with nine vertices is shown.

(a) Give the block decomposition of G.

(b) Find an ear decomposition of G.

(c) Determine $\lambda(G)$ and $\kappa(G)$.

(d) Is there a subgraph of G with larger vertex-connectivity than G?

(e) Is there a subgraph of G with larger edge-connectivity than G?

Question 4.2° Find the values of $\kappa(u, v)$ and $\lambda(u, v)$ for the graph in Figure 4.13.

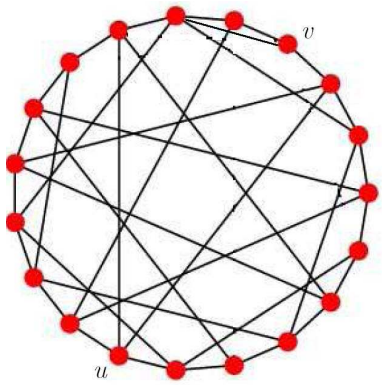
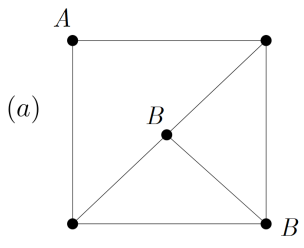
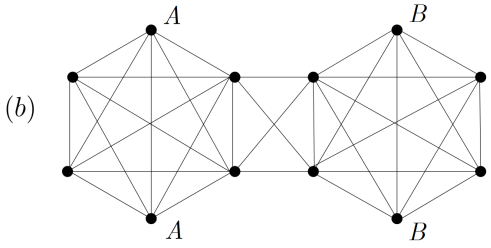


Figure 4.13: Find $\kappa(u, v)$

Question 4.3° Determine $\kappa(A, B)$ and $\lambda(A, B)$, as well as the minimum degree $\delta(G)$, edge connectivity $\lambda(G)$ and vertex connectivity $\kappa(G)$ for each of the following two graphs. You do not need to justify your answers.





Question 4.4° Let $A, B \subset V(G)$ and $A' \subset A$ and $B' \subset B$. Prove that $\kappa(A, B) \geq \kappa(A', B')$ and $\lambda(A, B) \geq \lambda(A', B')$.

Question 4.5° Let G be a connected k-regular bipartite graph where $k \geq 2$. Prove that G is 2-connected.

Question 4.6. Let G be a connected graph with at least two vertices. Prove that if L is a minimum edge cut of G, then $G - L$ has exactly two components.

Question 4.7. Let G be an Eulerian graph. Prove that $\lambda(G)$ is even.

Question 4.8. Let G be a graph.

(a) Prove that if G is 3-connected, then G has a cycle of even length.

(b) Prove that if G has maximum degree at most three, then $\lambda(G) = \kappa(G)$.

(c) Give an example to show $\lambda(G) > \kappa(G)$ when G has maximum degree four.

Question 4.9. Let G be an n-vertex graph with $n \geq k+1$ and $\delta(G) \geq (n+k-2)/2$. Prove that G is k-connected.

Question 4.10.

(a) Let G be an n-vertex block where $n \geq 4$, and let $a, b \geq 2$ with $a + b = n$. Prove that there is a partition $A \cup B$ of $V(G)$ such that $|A| = a$ and $|B| = b$ and $G[A]$ and $G[B]$ are connected.

(b) Prove that for $n \geq 2$, if G and H are two graphs whose union is K_n, then G is connected or H is connected.

Question 4.11. Prove that if G is a k-connected graph and A is a set of at least k vertices in G, then the graph obtained from G by adding a new vertex adjacent to all vertices in A is k-connected.

Question 4.12. Let $T_1, T_2, \ldots, T_k$ be spanning trees in a graph G, such that $E(T_i) \cap E(T_j) = \emptyset$ for all $i, j : 1 \leq i < j \leq k$ and $E(G) = E(T_1) \cup E(T_2) \cup \cdots \cup E(T_k)$. Prove that G is k-edge-connected, but not $2k$-edge-connected.

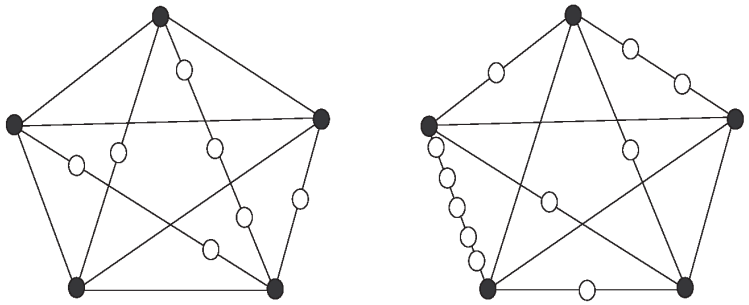
Question 4.13. Prove that if $G \neq K_2$ is a block such that for all $v \in V(G)$, $G - \{v\}$ is not a block, then G has a vertex of degree exactly two.

Question 4.14. An edge $e = \{u, v\}$ of a 4-connected graph G is ***contractible*** if $G/\{u, v\}$ is 4-connected. Prove that there exist infinitely many 4-connected graphs with no contractible edges.

Question 4.15. Prove that if $G \neq K_3$ is a 2-connected graph, then for every edge $e \in E(G)$, G/e or $G - e$ is 2-connected.

Question 4.16*
(a) A ***critically k-connected*** graph is a graph G with $\kappa(G) = k$ such that $\kappa(G - \{v\}) < k$ for every $v \in V(G)$. Prove that every critically k-connected graph has a vertex of degree exactly k.
(b) A ***critically k-edge-connected*** graph is a graph G with $\lambda(G) = k$ such that $\lambda(G-e) < k$ for every $e \in E(G)$. Prove that every critically k-edge-connected graph has a vertex of degree exactly k.

Question 4.17* A ***subdivision*** of a graph G is obtained by replacing each edge of G with a path between the ends of the edge, such that all the paths are pairwise internally disjoint. Two subdivisions of K_5 are shown below:



(a) Prove that for $1 \leq k \leq 3$, every k-connected graph contains a subdivision of K_{k+1}.
(b) Find a 4-connected graph with at least five vertices that does not contain a subdivision of K_5.

Question 4.18*
(a) Prove that for each tree T with $n \geq 2$ edges, there exist trees T_1 and T_2 such that $T_1 \cup T_2 = T$, $n/3 \leq |E(T_1)| \leq 2n/3$ and $n/3 \leq |E(T_2)| \leq 2n/3$, and $|V(T_1) \cap V(T_2)| = 1$.
(b) Prove for each n a multiple of three that there exists a tree T as above such that every pair of trees T_1 and T_2 as above has $|V(T_1)| = 2n/3$ and $|V(T_2)| = 2n/3$.

Question 4.19* Prove that for any block G with n vertices and m edges, any two vertices $u, v \in V(G)$ are the ends of at least $m - n + 2$ distinct paths.

Question 4.20* Let $k \geq 3$ and let G be a k-connected graph such that every set of at least k vertices of the graph contains an edge of the graph. Prove that G is Hamiltonian.

5 Matchings and Factors

A ***matching*** in a graph is a set of pairwise vertex-disjoint edges of the graph. In this section we are interested in determining the size of a ***maximum matching*** in a given graph and when a graph has a ***perfect matching*** or ***1-factor*** – that is, a matching covering all its vertices. For bipartite graphs, this question will be completely answered by Hall's Theorem, and for general graphs, by Tutte's 1-Factor theorem.

5.1 Independent sets and covers

An ***independent set*** in a graph G is a set X of vertices no pair of which is an edge of G – in other words the subgraph $G[X]$ induced by X has no edges. The maximum size of an independent set in a graph G is denoted $\alpha(G)$. The maximum size of a matching in a graph G is denoted $\alpha'(G)$. A ***vertex cover*** of G is a set of vertices $X \subset V(G)$ such that $e \cap X \neq \emptyset$ for every edge $e \in E(G)$ – in other words, a set of vertices which intersects every edge of G. The minimum size of a vertex cover of G is denoted $\beta(G)$. An ***edge cover*** of G is a set of edges covering every vertex of G – that is a set $E \subset E(G)$ such that for every vertex $v \in V(G)$, there is an edge of E containing v. The minimum size of an edge-cover is denoted $\beta'(G)$.

Example 19. The Petersen graph P is shown in the figure below. This graph has a perfect matching, for instance the edges $\{1,9\},\{3,10\},\{2,6\},\{5,8\},\{7,4\}$ form a perfect matching. Therefore $\alpha'(P) = 5$. An example of a maximum independent set is $\{2,4,5,10\}$, and therefore $\alpha(P) = 4$. A minimum vertex cover is $\{1,3,6,7,8,9\}$ and so $\beta(P) = 6$. Finally, a perfect matching is by definition a minimum edge cover, so $\beta'(P) = 5$.[9]

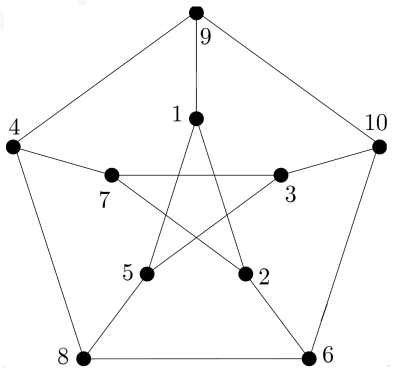


Figure 5.1: Covers, matchings and independent sets

[9]Find examples of other perfect matchings, maximum independent sets, and minimum covers in the Petersen graph.

Lemma 5.1.1 *For any graph G, $\alpha(G) + \beta(G) = |V(G)|$.*

Proof ▷ If I is an independent set of vertices in G, then $V(G)\backslash I$ is a vertex cover: every edge of G has at least one end in $V(G)\backslash I$ since no edges have both ends in I. Conversely, if C is a vertex cover, then every edge is incident with C so no edges have both ends in $V(G)\backslash C$. Therefore $V(G)\backslash C$ is an independent set of G. We conclude $\alpha(G) + \beta(G) = |V(G)|$. □

Lemma 5.1.2 (Gallai's Lemma) *Let G be a graph with no isolated vertices. Then $\alpha'(G) + \beta'(G) = |V(G)|$.*

Proof ▷ Let M be a matching in G of size $\alpha'(G)$ – a maximum matching. Then no edge of G has both ends in $V(G)\backslash V(M)$, so $V(G)\backslash V(M)$ is an independent set of vertices. Now let us choose one edge incident with each vertex in $V(G)\backslash V(M)$ and all edges of M. The set of edges we get, say F, is an edge-cover of size $|E(M)| + |V(G)\backslash V(M)| = |V(G)| - \alpha'(G)$. Therefore $\beta'(G) \leq |V(G)| - \alpha'(G)$.

Conversely, let F be an edge-cover of G of size $\beta'(G)$ – a minimum edge-cover. Then $F - e$ is not an edge cover for any $e \in E(F)$. This means that each edge of F must cover one of its ends uniquely, so every edge of F has an end of degree one in F. In particular, every component of F is a star – a $K_{1,t}$ for some $t \geq 1$ (see Figure 5.2). Pick one edge from each component of F to get a matching M with $|E(M)|$ equal to the number of components of F. Since all components of F are stars,

$$\beta'(G) = |E(F)| = |V(F)| - |E(M)| = |V(G)| - |E(M)| \geq |V(G)| - \alpha'(G).$$

This completes the proof. □

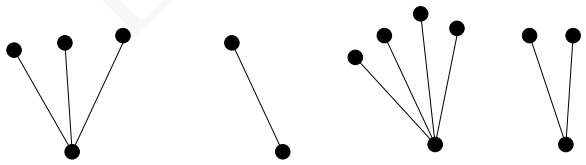


Figure 5.2: Structure of minimal edge-covers

5.2 Hall's Theorem

Let X be a set of vertices in a graph G. We define $N(X)$ to be the ***neighborhood of X***, namely

$$\{y \in V(G)\backslash X : \{x, y\} \in E(G) \text{ for some } x \in X\}.$$

In other words, it is the set of vertices not in X adjacent to some vertex in X. Hall's Theorem gives a necessary and sufficient condition for a bipartite graph to have a perfect matching – and in fact a matching covering all vertices of one part. There are many proofs of Hall's Theorem; we give two proofs, one here and one in Chapter 8.5.

Theorem 5.2.1 (Hall's Theorem) *Let $G(A, B)$ be a bipartite graph. Then G has a perfect matching if and only if for every set $X \subset A$ and every set $X \subset B$,*

$$|N(X)| \geq |X| \quad \text{(Hall's Condition)}$$

Proof ▷ If G has a perfect matching M, then for every $X \subseteq A$ and $X \subseteq B$, there are $|X|$ neighbors of X in M. Therefore the number of neighbors of X in G is at least $|X|$, which is Hall's condition. Now we suppose Hall's Condition is true, and show how to get a perfect matching in G. We proceed induction on $|A|$: we prove that if $|N(X)| \geq |X|$ for every set $X \subset A$ in a graph G, then G contains a matching saturating all vertices of A. Note that this proves Hall's Theorem, since we could apply the same statement to B to get a matching saturating all vertices of B. If $|A| = 1$, then the statement is true. Suppose $|A| > 1$. We consider two cases. The first case is that $|N(X)| > |X|$ for all non-empty $X \subset A$. In this case, pick any edge of G and remove both ends of that edge, say a and b. Then we obtain the bipartite graph $H(A\backslash\{a\}, B\backslash\{b\})$. In this bipartite graph, Hall's Condition holds in $A\backslash\{a\}$, and therefore H has a matching, M, saturating all vertices of $A\backslash\{a\}$. Now $M \cup \{a, b\}$ is the required matching in G. The second case is that for some proper subset X of A or B. $|N(X)| = |X|$. Let $Y = N(X)$. In this case, consider the graph $G_1(A_1, B_1)$ obtained from G by removing all vertices of $X \cup Y$, and the graph $G_2(X, Y)$ consisting of all edges between X and Y. Then Hall's Condition holds in G_1 and also in G_2. To see that it holds in G_1, take any set $S \subset A_1$. Then:

$$|N_{G_1}(S)| + |N_{G_2}(X)| \geq |N_G(X \cup S)| \geq |X \cup S| = |X| + |S|$$

and since $|N_{G_2}(X)| = |N_G(X)| = |X|$, we have $|N_{G_1}(S)| \geq |S|$ for any $S \subset A_1$. By induction G_1 and G_2 have matchings, say M_1 and M_2, saturating all their vertices in A, and $M_1 \cup M_2$ is a matching in G saturating all vertices of A. □

A ***1-factorization*** of a graph G is a collection of pairwise edge-disjoint 1-factors $M_1, M_2, \ldots, M_r$ such that $G = M_1 \cup M_2 \cup \cdots \cup M_r$. For example, for even values of n, the complete graph K_n has a 1-factorization.

Corollary 5.2.2 *Let $G(A, B)$ be a k-regular bipartite graph, where $k \geq 1$. Then G has a 1-factorization.*

Proof ▷ It suffices to prove that G has a perfect matching. To see this, we apply Hall's Theorem. For a set $X \subseteq A$, $e(X, B) = k|X|$ since every vertex of X has degree k. We also have $e(N(X), A) = k|N(X)|$ for the same reason. The set of edges between X and B is contained in the set of edges between $N(X)$ and A, and therefore $e(X, B) \leq e(N(X), A)$. It follows that $k|X| \leq k|N(X)|$ and so $|X| \leq |N(X)|$ for all $X \subseteq A$. Similarly, $|X| \leq |N(X)|$ for all $X \subseteq B$. Therefore, by Hall's Theorem, G has a perfect matching. ◁◁ □

5.3 Systems of distinct representatives

Let $S_1, S_2, \ldots, S_n$ be sets. Then the sets have a ***system of distinct representatives*** or ***transversal*** if we can select $s_1 \in S_1, s_2 \in S_2, \ldots, s_n \in S_n$ such that $s_1, s_2, \ldots, s_n$ are all different. The problem of determining whether sets $S_1, S_2, \ldots, S_n$ have a system of distinct representatives can be solved via Hall's Theorem, as follows.

Let G be a bipartite graph with parts $A = \{S_1, S_2, \ldots, S_n\}$ and $B = S_1 \cup S_2 \cup \cdots \cup S_n$, and where $\{a, b\} \in E(G)$ with $a \in A$ and $b \in B$ if $b \in a$. In other words, join a set to all the elements it contains. Then G has a matching covering A if and only if $S_1, S_2, \ldots, S_n$ have a system of distinct representatives: the edges of the matching tell us which set each element is a representative for.

Example 20. The sets below have a system of distinct representatives, since the graph G we construct from these sets is the ***cube graph*** Q_3, which has a perfect matching: ◁◁

$$S_1 = \{1, 2, 3\} \quad S_2 = \{2, 3, 4\} \quad S_3 = \{1, 3, 4\} \quad S_4 = \{1, 2, 4\}$$

Halls' Theorem gives a necessary and sufficient condition for distinct representatives:

Theorem 5.3.1 *Sets $S_1, S_2, \ldots, S_n$ have a system of distinct representatives if for every set $I \subseteq \{1, 2, \ldots, n\}$,*

$$\left|\bigcup_{i \in I} S_i\right| \geq |I|.$$

5.4 Latin squares

A ***latin square*** is a square array of symbols such that every symbol appears exactly once in every row and exactly once in every column. If the array has n rows and n columns, then the number of symbols is exactly n, and each column and each row is a permutation of the symbols. We call this a ***latin square of order*** n. An example of a latin square of order 7 is shown in Figure 5.3.

1	4	7	2	6	5	3
4	2	5	1	3	7	6
7	5	3	6	2	4	1
2	1	6	4	7	3	5
6	3	2	7	5	1	4
5	7	4	3	1	6	2
3	6	1	5	4	2	7

Figure 5.3: A latin square of order 7

There are many ways to construct latin squares – the reader should construct a latin square of order n for each $n \geq 1$. Note also that the ***multiplication table of a group*** is a latin square.[10] ◁◁

A natural question is whether we can ever get stuck constructing a latin square of order n after partially filling the array with numbers such that no number appears more than once in any row or column. For $1 \leq m \leq n$, a ***latin rectangle*** is an $m \times n$ array of n numbers such that every number appears at most once in every row and column. In particular, can we always extend a latin $m \times n$ rectangle to a latin square of order n? The answer is given by Hall's Theorem:

Theorem 5.4.1 *Let $n \geq m \geq 0$. Then a latin $m \times n$ rectangle can always be extended to a latin square of order n.*

Proof ▷ If $m = n$, then the latin rectangle is a latin square, so we can assume $m < n$. We show that a latin $m \times n$ rectangle R can be extended to a latin $(m+1) \times n$ rectangle by induction on m. If $m = 0$ then we just take a permutation of the n symbols to get a $1 \times n$ rectangle. If $m > 0$, form a bipartite graph with parts A and B where A represents the set of n symbols, and B represents the entries of row $m+1$, and where $\{a, b\}$ is an edge of the graph with $a \in A$ and $b \in B$ if symbol a can be placed in the bth entry of row $m+1$ – in other words, symbol a is not used by R in the first m rows of column b. Then each $b \in B$ has degree $n - m$ in the graph, since there are exactly m symbols used by R in column b. Each $a \in A$ has degree $n - m$, since symbol a is used by exactly m columns of R, so there are $n - m$ remaining columns such that a can be placed in that column and row $m + 1$. In other words, the graph is $(n - m)$-regular. By Corollary 5.2.2, the graph has a perfect matching $\{a_1, b_1\}, \{a_2, b_2\}, \ldots, \{a_n, b_n\}$. Now place a_i in position b_i of row $m + 1$ to get a latin $(m + 1) \times n$ rectangle. □

[10] The reader should check that not every latin square is the multiplication table of a group.

The problem of completing a latin square becomes harder if we are given an arbitrary set of filled entries in the array. For instance, in an $n \times n$ array, it is possible to fill in only n entries thereby preventing completion to a latin square. It turns out this is the minimum: if we fill in any $n-1$ entries, then completion to a latin square is possible. ◁◁

5.5 König-Ore Formula

A vertex not contained by any edge of a given matching is called ***unsaturated*** or ***exposed*** by the matching, and those vertices which are contained in edges of the matching are called ***saturated*** by the matching. Hall's Theorem gives a formula for finding $\alpha'(G)$ in a bipartite graph. For a bipartite graph $G(A,B)$, define $\mathrm{x}(G,A) = |A| - \alpha'(G)$: this is the number of vertices of A exposed by a maximum matching. Hall's Theorem gives a formula for $\mathrm{x}(G,A)$:

Theorem 5.5.1 (König-Ore Formula) *Let $G(A,B)$ be a bipartite graph. Then*

$$\mathrm{x}(G,A) = \max_{S \subset A}\{|S| - |N(S)|\}.$$

Proof ▷ Let d be the right hand side of the identity above. Add d vertices to B, all adjacent to all vertices of A. Then Hall's Condition – namely $|N(X)| \geq |X|$ for all $X \subseteq A$ – is satisfied in this new graph, so it has a matching covering all vertices of A, by Hall's Theorem. It follows that G has a matching of size at least $|A|-d$. Therefore $\mathrm{x}(G,A) \leq d$. Conversely, if M is a matching of size $|A|-\mathrm{x}(G,A)$, then each set $S \subseteq A$ has at least $|S|-\mathrm{x}(G,A)$ neighbours in B. In other words, $|N(S)| \geq |S| - \mathrm{x}(G,A)$ so $d = \max\{|S| - |N(S)|\} \leq \mathrm{x}(G,A)$. □

As an exercise, one can prove that a bipartite graph $G(A,B)$ of minimum degree δ and maximum degree $\triangle$ contains a matching of size at least $\delta|A|/\triangle$. Another consequence of Hall's Theorem is the following theorem: ◁◁

Theorem 5.5.2 (König-Egeváry Theorem) *If $G(A,B)$ is a bipartite graph, then $\alpha'(G) = \beta(G)$ and if G has no isolated vertices, then $\beta'(G) = \alpha(G)$.*

To prove this, it is sufficient to show $\alpha'(G) = \beta(G)$, by Lemmas 5.1.1 and 5.1.2.

5.6 Tutte's 1-Factor Theorem

There is a natural condition for a graph G to have a perfect matching: if S is a set of vertices of G and $H_1, H_2, \ldots, H_r$ are the ***odd components*** of $G - S$ – that is the components with an odd number of vertices – then none of the H_i can have a perfect matching, so each sends at least one edge of a perfect matching to S (see Figure 5.4). In particular $|S| \geq r$, so we have for all $S \subset V(G)$, denoting by odd(G - S) the number of odd components of $G - S$,

$$|S| \geq \mathrm{odd}(G - S).$$

Note that if $S = \emptyset$, this asserts that G has an even number of vertices. Tutte's Theorem shows, remarkably, that this is also a sufficient condition:

Theorem 5.6.1 (Tutte's 1-Factor Theorem)
Let G be a graph. Then G has a perfect matching if and only if for every set $S \subset V(G)$,

$$|S| \geq \text{odd}(G - S) \quad \text{(Tutte's Condition)}$$

Proof ▷ If G has a perfect matching M, then for any $S \subseteq V(G)$, every odd component F of $G - S$, there is at least one exposed vertex of F for the matching $M \cap E(F)$. Each exposed vertex is adjacent in M to a vertex of S, so $|S| \geq \text{odd}(G - S)$, which is Tutte's Condition. Now suppose G satisfies Tutte's Condition; we show how to find a perfect matching in G.

The proof we give is by induction on $|V(G)|$, the case $|V(G)| = 2$ holds since $G = K_2$ in that case. Suppose $|V(G)| > 2$, and let S be the largest subset of G such that equality holds in Tutte's Condition. Such a set S exists, because $|V(G)|$ is even, and so $G - \{s\}$ has at least one odd component for each $s \in V(G)$. Let F and H denote generic odd and even components of $G - S$.

Claim 1. *The graph H has a 1-factor.*
For any $R \subset V(H)$, we note

$$\text{odd}(G - (R \cup S)) = \text{odd}(H - R) + \text{odd}(G - S)$$

since every odd component of $G - S$ is an odd component of $G - (R \cup S)$. By Tutte's Condition, $\text{odd}(G - R \cup S) \leq |R| + |S|$. Since $\text{odd}(G - S) = |S|$, we conclude $\text{odd}(H - R) \leq |R|$ for all $R \subset V(H)$. By induction H has a 1-factor.

Claim 2. *The graph $F' = F - \{v\}$ has a 1-factor for any $v \in V(F)$.*
By induction, if this is false, then there exists a set $Q \subset V(F')$ such that $\text{odd}(F' - Q) > |Q|$. Now for any set $R \subset V(F)$,

$$\text{odd}(F - R) + |R| \equiv |V(F)| \equiv 1 \mod 2$$

since F has an odd number of vertices (this step is really key to the proof). Therefore $\text{odd}(F' - Q) \geq |Q| + 2$. We also observe

$$\text{odd}(G - S \cup \{v\} \cup Q) = \text{odd}(G - S) - 1 + \text{odd}(F' - Q)$$

since F is an odd component of $G - S$ but not of $G - S \cup \{v\} \cup Q$. If $T = S \cup \{v\} \cup Q$, then

by Tutte's Condition, we get

$$\begin{aligned} |T| &\geq \text{odd}(G-T) \\ &= \text{odd}(G-S)-1+\text{odd}(F'-Q) \quad \geq |S|+|Q|+1. \end{aligned}$$

This shows $\text{odd}(G-T) = |T|$, contradicting the maximality of S, and the claim is proved.

Claim 3. *Let $G(S,C)$ be the bipartite graph formed from G by contracting each odd component of $G-S$ to a single vertex, and taking all edges with one end in S and one end in the set C of contracted vertices. Then $G(S,C)$ has a perfect matching.*

To prove this, we use Hall's Theorem: for every set $X \subset C$,

$$|X| = \text{odd}(G - N(X)) \leq |N(X)|$$

as required. Since $|S| = |C| = \text{odd}(G-S)$, there is a 1-factor in $G(S,C)$.

To complete the proof of Tutte's 1-Factor Theorem, put together all the 1-factors that we found in Claims 1–3. Let $M_1, M_2, \ldots, M_r$ be 1-factors in the even components of G. Now let M be a 1-factor in $G(S,C)$. Then the edges of M form a matching in G, and for each odd component H_i of $G-S$, for $i \in \{1,2,\ldots,s\}$ where $s = \text{odd}(G-S)$, there is exactly one vertex of H, say v_i, incident with an edge of M. Now Claim 2 gives a 1-factor N_i in $H - v_i$. Then

$$M \cup M_1 \cup \cdots \cup M_r \cup N_1 \cup N_2 \cup \cdots \cup N_s$$

is a perfect matching of G. □

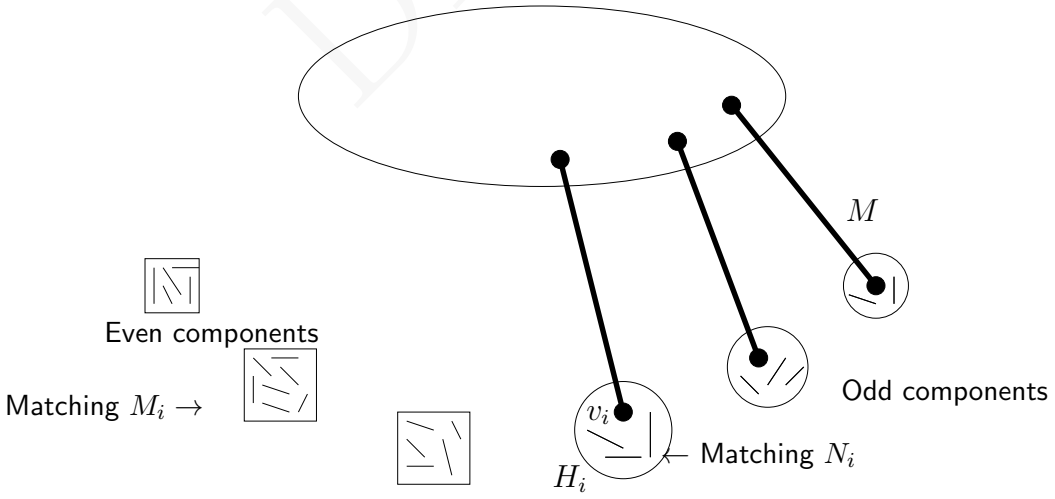


Figure 5.4: The proof of Tutte's Theorem

From Tutte's 1-Factor Theorem, we obtain the following condition for a ***cubic graph*** (3-regular graph) to have a perfect matching:

Theorem 5.6.2 (Petersen's Theorem) *Any cubic bridgeless graph has a 1-factor.*

Proof ▷ We have to check Tutte's Condition. Pick a set $S \subset V(G)$. If $S = \emptyset$, then Tutte's Condition holds since G has an even number of vertices and is connected. Then there are at least two edges from S to each odd component of $G - S$. If H is an odd component of $G - S$, then it contains an even number of vertices of degree three, so it must send to S an odd number of edges. It must send at least three edges. So we have $3r$ edges out of odd components. On the other hand, G is cubic so $|S| \geq r$, as required. □

5.7 Tutte-Berge Formula*

The ***Tutte-Berge Formula*** is the analog of the ***König-Ore Formula*** for non-bipartite graphs, and gives a method for finding $\alpha'(G)$ i.e. the size of a largest matching in the graph. We define $\mathrm{x}(G)$ to be the minimum number of vertices of G exposed by a matching of G – thus $\mathrm{x}(G) = |V(G)| - 2\alpha'(G)$.

Theorem 5.7.1 (Tutte-Berge Formula) *For any graph G,*

$$\mathrm{x}(G) = \max_{S \subset V(G)} \{\mathrm{odd}(G - S) - |S|\}.$$

The proof of this theorem is left as an exercise. The theorem can be used to give lower bounds on $\alpha'(G)$ for various graphs. For example, we apply the Tutte-Berge Formula to cubic graphs – graphs where all the vertices have degree three – to get a lower bound on $\alpha'(G)$: ◁◁

Theorem 5.7.2
Let G be a cubic graph on n vertices. Then G has a matching of size at least $\frac{7n}{16}$.

Proof ▷ It may be assumed that G is connected, otherwise we pass to the components of G. We have to find an upper bound for $\mathrm{x}(G)$, namely $\mathrm{x}(G) \leq n/8$. By the Tutte-Berge formula, this is the same as showing $\mathrm{odd}(G - X) - |X| \leq n/8$ for all sets $X \subset V(G)$. Let $X \subset V(G)$ have size γ, and let α be the number of odd components of $G - X$ with at most three vertices, and β be the number of odd components of $G - X$ with at least five vertices. Let's call these α-components and β-components, respectively. Then $\mathrm{odd}(G - X) - |X| = \alpha + \beta - \gamma$. Now each α-component H of G is K_1 or K_3 or a path on three vertices. In each case, since G is

cubic, $e(V(H), X) \geq 3$. Each β-component F of G has $e(V(F), X) \geq 1$. On the other hand, $e(X, V(G)\backslash X) \leq 3|X|$, since every vertex of X has degree three. Therefore

$$3\alpha + \beta \leq 3\gamma.$$

Next we observe that there are $n - \gamma$ vertices in $G - X$, but also at least $\alpha + 5\beta$ vertices in $G - X$, so

$$\alpha + 5\beta \leq n - \gamma.$$

We want to maximize $\alpha + \beta - \gamma$ subject to the above two inequalities. It is not hard to see that we must have $\alpha = 0$, $\beta = 3n/16$ and $\gamma = n/16$, in which case $\mathrm{ex}(G) = \alpha + \beta - \gamma = n/8$, as required. □

Theorem 5.7.2 is best possible: the graph shown in Figure 5.5 is cubic with $n = 16$ vertices with no matching of size more than $7 = 7n/16$.

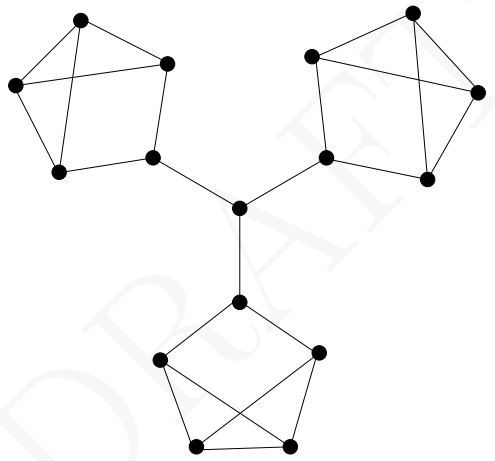


Figure 5.5: A cubic graph with no perfect matching

5.8 Matching Algorithms

In bipartite graphs, the König-Egeváry Theorem gives a practical way to find a maximum matching, using the notion of an augmenting path. An ***alternating path*** in a graph G with a matching M is a path whose every alternate edge is in M – we call this ***M-alternating***. An ***augmenting path*** for a matching M is an alternating path whose ends are exposed by M – we call this ***M-augmenting***.[11] The following theorem is key to many matching algorithms:

[11] Recall this means that these vertices are not in any edge of M.

Theorem 5.8.1 (BERGE) *A matching M in a graph G is a maximum matching if and only if M does not admit any augmenting paths.*

Proof ▷ If M is a maximum matching, it clearly admits no augmenting path, for if P is an augmenting path, then the matching M' with

$$E(M') = E(M) \triangle E(P)$$

is larger than M: $|E(M')| = |E(M)| + 1$.

Conversely, suppose M is a matching which does not admit an augmenting path, and $|M| < |N|$ for some maximum matching N. Then $M \cup N$ is a graph of maximum degree at most two, and so all the components of $M \cup N$ are paths or cycles (this is called a ***linear forest***). However, since $|N| > |M|$, and any cycle in $M \cup N$ has as many edges of M as of N, there must be a path P such that

$$|E(P) \cap M| < |E(P) \cap N|.$$

This means that the first and last edge of the path are in N, and so the path augments M, a contradiction. □

A version of the ***Hopcroft-Karp Algorithm*** for finding a maximum matching in bipartite graphs G with parts A and B is as follows. Using Berge's Theorem above, the key is to start with a given matching M in a bipartite graph, and to try to find an M-augmenting path. If no such path exists, then M is a maximum matching, otherwise, we can use the augmenting path P to find a matching M' with $E(M') = E(M) \cup (E(P) \backslash E(M))$ which has one more edge than M i.e. $|E(M'| = |E(M)| + 1$ i.e. we take the edges of P in M out, and add the edges of P not in M to get M'. Pseudocode for the algorithm is given below:

```
1: function Hopcroft-Karp(BipartiteGraph, A, B)
2: M ← ∅
3: U ← the set of exposed vertices with respect to M.
4: From each vertex of U, grow M-alternating path
5: If some M-alternating path u_1 u_2 ... u_{2k} is M-augmenting,
6:     Let M ← M ∪ {u_1u_2, u_3u_4, ..., u_{2k-1}u_{2k}} \ {u_2u_3, u_4u_5, ..., u_{2k-2}u_{2k-1}}.
7:     Return to Step 3.
8: Else M ← maximum matching.
9: return M
```

A key step is step 4, which is to grow all M-alternating paths starting with U, the set of exposed vertices with respect to M. To achieve this, we may do a similar procedure to the ***breadth-first search algorithm***. For each $u \in U$, we build a ***layered graph*** as follows. First add u and let $L_0 = \{u\}$ be the zeroth layer. At any stage, given a layered graph T with layers $L_0, L_1, \ldots, L_i$, we consider two cases. If i is odd, then there exists a set L_{i+1} of vertices not in T connected by edges of M to vertices of L_i, and we add those edges of M to T to get a new layered graph. If i is even, let L_{i+1} be the set of vertices not in T connected by any edge to vertices in L_i, and add to T a set of edges between L_i and L_{i+1} to get a new layered graph. Such a layered graph might look like the graph in Figure 5.6, where the edges of the matching M are in bold.

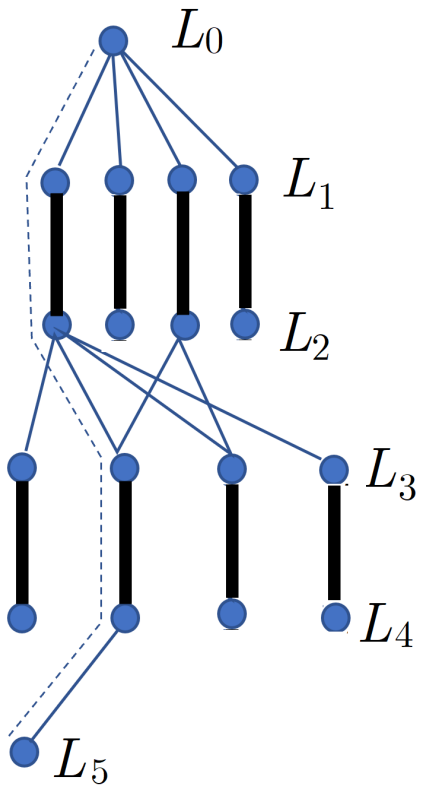


Figure 5.6: Layered graph

We continue until a vertex v of $U\backslash\{u\}$ is added, in which case there is an M-augmenting path in the layered graph T from u to v (see the dotted path in Figure 5.6), and we proceed to step 6, or until every vertex of the graph not in U is added, in which case we go to the next vertex of U and repeat the procedure. If for every $u \in U$, the layered graph rooted at u contains no vertices of $U\backslash\{u\}$, then there is no M-augmenting path and we go to step 8.

Example 21. Consider the grid graph below. We use the matching algorithm to find a maximum matching in the grid, starting with the given matching $\{1,2\},\{5,6\}$ shown in bold.

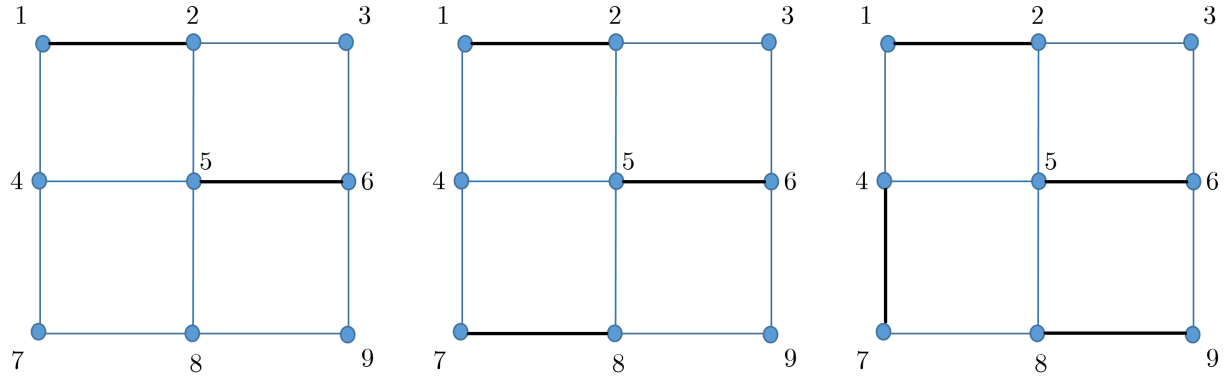


Figure 5.7: A matching in the grid graph

First we identify the parts A and B of the grid graph. We may let $A = \{1,3,5,7,9\}$ and $B = \{2,4,6,8\}$, and we are starting with the matching $M = \{\{1,2\},\{5,6\}\}$.

In step 3 of the pseudocode, the set U of exposed vertices with respect to M is $U = \{3,4,7,8,9\}$. For step 4, we grow a layered graph from each vertex of U to try to find an augmenting path, as outlined above. We could start with $7 \in U$, so the root of the layered graph is 7, and this is L_0. Since 0 is even, we seek edges of the graph between 7 and the rest of the graph: $\{7,8\}$ for instance, and now we stop since $8 \in U$. So we have steps 5 and 6 done: since $\{7,8\}$ is an edge contained in U, it is an M-augmenting path. So we can add $\{7,8\}$ to the matching M to get a new matching $M = \{\{1,2\},\{5,6\},\{7,8\}\}$. This is shown in the center panel in Figure 5.7. [12] Now we return to step 3.

Now for step 3, the set U of exposed vertices is $\{3,4,9\}$. For step 4, we grow layered graphs starting at vertices of U, as outlined above. Starting at $9 \in U$, we add the edges $\{9,6\}$ and $\{9,8\}$ to the layered graph to get the first layer $L_1 = \{6,8\}$. Now since 1 is odd, we add edges of M to the layered graph and $L_2 = \{5,7\}$ and so we add $\{8,7\}$ and $\{6,5\}$ to the layered graph. Since 2 is even, we add edges with one end in L_2 and the other end not in the layered graph so far. We start with the edge $\{7,4\}$; and $L_1 = \{1,7\}$. Then $L_2 = \{2,8\}$ and $L_3 = \{3,5,9\}$. Since $9 \in U$, we have the augmenting path with edges $\{9,7\},\{7,8\}$ and $\{7,4\}$. So we take $\{7,8\}$ out of M and add the edges $\{9,8\},\{7,4\}$ to M for step 6. So now $M = \{\{9,8\},\{7,4\},\{1,2\},\{5,6\}\}$ (see the right panel in Figure 5.7), and we return to step 3.

[12] We could equally have added $\{7,4\}$ or even $\{8,9\}$ if our layered graph had been rooted at 8.

We restart the algorithm with this matching M. In step 3, the set U of exposed vertices is just $U = \{3\}$. For step 4, we grow a layered graph starting at 3: first we add the edges $\{3, 2\}$ and $\{3, 6\}$ and the first layer is $L_1 = \{2, 6\}$. Since 1 is odd, we add edges of M to the layered graph so far, namely $\{6, 5\}$ and $\{2, 1\}$. So the layered graph so far has edges $\{3, 2\}, \{3, 6\}, \{6, 5\}, \{2, 1\}$ and $L_2 = \{5, 1\}$. Now we add edges with one end in L_2 and the other not in the layered graph so far – so we add $\{1, 4\}$, $\{5, 4\}$ and $\{5, 8\}$, and $L_3 = \{4, 8\}$. Since 3 is odd, we add edges of M to the layered graph, and the only edge we can add is $\{4, 7\}$. There are no vertices of U left (as we knew at the start), so there are no M-augmenting paths. So we are at step 8, and therefore M is a maximum matching (it was clear since there are 9 vertices in the graph, so at least one must be exposed by every maximum matching).

The runtime complexity of the Hopcroft-Karp Algorithm is polynomial time in n . An ◁◁
algorithm of Micali and Vazirani for maximum matchings in an n-vertex bipartite graphs with m edges runs in time roughly $m\sqrt{n}$, and is the fastest known deterministic algorithm for maximum matching. In addition, the ***Hungarian Algorithm*** and ***Kuhn-Munkres Algorithm*** are more general than the Hopcroft-Karp Algorithm, and use matrices to find maximum weighted matchings in bipartite graphs with weights on the edges. There is an algorithm for maximum matchings in general graphs, called ***Edmonds' Matching Algorithm***, but it is beyond the scope of this course.

5.9 Stable matchings*

Suppose we have a set of n candidates for n jobs, with each candidate listing in order of preference the jobs they would like to do and each job having a hiring committee which lists candidates in order of preference. Let G be the bipartite graph whose parts are A, the set of candidates, and B the set of jobs/hiring committees. We join $a \in A$ to $b \in B$ if candidate a is able to do job b. If this bipartite graph has a perfect matching, then of course we can suitably assign all candidates to all jobs. However, this takes no account of the preferences of the candidates. A ***stable matching*** is a perfect matching of candidates to jobs such that no two candidates would prefer to switch jobs. In other words, if a is matched to job b and c is matched to job d, then matching a to d and b to c leads to both a and c being matched to jobs they prefer less than the original jobs they were assigned to. One of the classical examples is solving medical school students assignments to internships, called the ***National Residency Matching Program***.[13]

The ***Gale-Shapley Algorithm*** solves the stable matching problem, by showing that there is always a stable matching and it can be found efficiently, namely with runtime complexity roughly n^2. The algorithm runs as follows: first each candidate chooses a job they most

[13] See https://en.wikipedia.org/wiki/NRMP and the references therein.

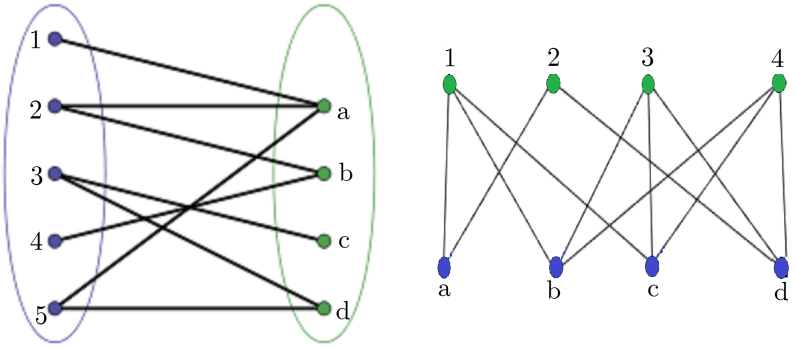
prefer, and each job is initially assigned the best candidate that has chosen that job. In subsequent rounds of the algorithm, each candidate chooses a job they most prefer amongst all the jobs they have not yet chosen in previous rounds, and then each job is assigned the best candidate, choosing between the current candidate choosing that job, or the candidate they were provisionally assigned in the preceding round. The process is repeated until all candidates have been assigned a job and all jobs have been assigned a candidate. The pseudocode is as follows:

```
1:  function StableMatching(Graph):
2:  Initialize M to the empty matching
3:  While (some candidate a is unmatched and there
          remains a job a is never chosen)
4:      b ← first job on a's list which a has not
            yet chosen
5:      If (b is unmatched)
6:          Add {a,b} to matching M.
7:      Else if (hiring committee for b prefers a to
                 current candidate a')
8:          Replace {a',b} with {a,b} in matching M.
9:      Else hiring committee b rejects a.
10: Return stable matching M.
```

5.10 Exercises

Question 5.1° For the graphs in Figures 5.5 and 5.7, determine $\alpha(G), \alpha'(G), \beta(G)$ and $\beta'(G)$.

Question 5.2° For each graph in Figure 5.8, determine a maximum matching using the Hungarian Matching Algorithm, starting with the given matching M.



$M = \{\{1, a\}, \{4, b\}, \{3, d\}\}$ $M = \emptyset$

Figure 5.8: Two bipartite graphs

Question 5.3° In Figure 1.21, a graph G with nine vertices is shown.

(a) How many components does $G - \{v_3\}$ have?
(b) What is $\text{odd}(G - \{v_3\})$?
(c) Find a set S of vertices such that $\text{odd}(G - S) > |S|$.
(d) Find $\alpha(G), \alpha'(G), \beta(G)$ and $\beta'(G)$.

Question 5.4° A school with 20 professors forms 10 committees, each containing 6 professors, such that every professor is on exactly 3 committees. Prove that it is possible to select a distinct representative from each committee.

Question 5.5° A ***tiling*** of an $m \times n$ chess board is a set of dominoes which cover all the squares on the chess board exactly once (each domino covers two adjacent squares).

(a) For which $m \geq 1$ and $n \geq 1$ does an $m \times n$ chess board having a tiling?
(b) If we remove two squares from an $m \times n$ chessboard, when do the remaining squares have a tiling?

Question 5.6° Let e be an edge of a connected cubic graph such that $G-e$ is disconnected. Prove that every perfect matching of G contains e.

Question 5.7. Given an example of an $n \times n$ array with n cells filled in with numbers from $\{1, 2, \ldots, n\}$, so that no column or row has two of the same number and the array cannot be completed to a latin square.

Question 5.8. A ***tiling*** of an $m \times n$ chess board with some squares removed is a set of dominoes which cover all the squares on the chess board exactly once (each domino covers two adjacent squares). Prove that if the number of white and black squares are equal, and for each set S of black squares (respectively, white squares), there are at least as many white squares (respectively, black squares) adjacent to squares in S, then there is a tiling of the board.

Question 5.9. Prove that for any graph G, $\alpha'(G) \leq \beta(G) \leq 2\alpha'(G)$

Question 5.10. Let G be an n-vertex bipartite graph G with $\delta(G) \geq d$ and $\triangle(G) \leq \triangle$. Prove that $\alpha'(G) \geq nd/\triangle$.

Question 5.11. Prove that if $k \geq 1$ is odd and G is a k-regular $(k-1)$-edge-connected graph, then G has a perfect matching.

Question 5.12. Let $k \geq 1$, and let A be an $n \times n$ 0-1 matrix such that every row and every column has exactly k 1s. Prove that we can pick n entries of A, no two in the same row or column, such that each entry is a 1.

Question 5.13. Check that the Gale-Shapley Algorithm for a bipartite graph with $2n$ vertices runs in time at most n^2.

Question 5.14. Check that the Hungarian Matching Algorithm for a bipartite graph with n vertices runs in polynomial time in n, and give an explicit upper bound on the runtime complexity.

Question 5.15.

(a) Prove that a tree has at most one perfect matching.

(b) Show that a tree has a perfect matching if and only if $\text{odd}(T-x) = 1$ for every $x \in V(T)$.

Question 5.16.

(a) Let G be an n by n bipartite graph of minimum degree more than $n/2$. Prove that G has a perfect matching.

(b) Let G be a $2n$-vertex graph of minimum degree at least n. Prove that G has a perfect matching.

Question 5.17. Let A_k be the set of subsets of $\{1, 2, \ldots, n\}$ of size k. Prove that for $k < n/2$, there is an injective function $f : A_k \to A_{k+1}$ such that $a \subseteq f(a)$ for all $a \in A_k$. For instance, if $k = 1$ and $n = 3$ then the function

$$f(\{1\}) = \{1, 2\} \quad f(\{2\}) = \{2, 3\} \quad f(\{3\}) = \{1, 3\}$$

is an example of such a function $f : A_1 \to A_2$.

Question 5.18. Let G be an n-vertex 4-regular multigraph. Prove that G has a matching with at least $n/3$ edges, and when n is a multiple of 3, describe 4-regular multigraphs with no larger matchings.

Question 5.19* Suppose we fill in fewer than $n/2$ cells in an $n \times n$ array with symbols from $\{1, 2, \ldots, n\}$ so that no symbol appears more than once in every row or column. Prove that the array can be completed to a latin square.

Question 5.20* Prove that a bipartite graph with minimum degree at least d containing a perfect matching contains at least $d!$ perfect matchings. Is this best possible?

Question 5.21* Prove that a cubic $n \times n$ bipartite graph contains at least $(4/3)^n$ perfect matchings.

Question 5.22* Let $n = 2k + 1$ and let A_k be the family of subsets of $\{1, 2, \ldots, n\}$ of size k. Define an injective function $f : A_k \to A_{k+1}$ such that $f(a) \subseteq a$ for all $a \in A_k$.

6 Vertex and Edge-Coloring

A ***proper k-edge-coloring*** of a graph G is a function $c : E(G) \to \{1, 2, \ldots, k\}$ such that if $e, f \in E(G)$ intersect, then $c(e) \neq c(f)$. In other words, any two edges which share a vertex must receive different colors (it is convenient to refer to the elements of $\{1, 2, \ldots, k\}$ as colors). The minimum k for which G has a proper k-edge-coloring is denoted $\chi'(G)$, and referred to as the ***edge-chromatic number of G***. Another way of saying it is: $\chi'(G)$ is the minimum number of matchings which partition $E(G)$, since the set of edges of any particular color is a matching. A graph G is ***k-edge colorable*** if $\chi'(G) \leq k$, and ***k-edge-chromatic*** if $\chi'(G) = k$. It is left as an exercise to verify that $\chi'(K_n) = n - 1$ when n is even and $\chi'(K_n) = n$ if n is odd. The main theorems we prove on edge coloring are ***König's Theorem*** and ***Vizing's Theorem***. ◁◁

A ***proper k-coloring*** of a graph G is a function $c : V(G) \to \{1, 2, \ldots, k\}$ such that if $u, v \in V(G)$ are adjacent, then $c(u) \neq c(v)$. So we color the vertices with k colors in such a way that no two adjacent vertices have the same color. The ***chromatic number*** of G is denoted $\chi(G)$, and is the minimum k for which G has a proper k-coloring. Thus $\chi(G)$ is the minimum number of independent sets which partition $V(G)$. For example, $\chi(K_n) = n$, and a graph G is bipartite if and only if $\chi(G) \leq 2$. We say that a graph is ***k-colorable*** if $\chi(G) \leq k$ and ***k-chromatic*** if $\chi(G) = k$. The main theorem on vertex coloring is ***Brooks' Theorem***, which states that $\chi(G) \leq \triangle(G)$ when G is not an odd cycle or a complete graph (for those graphs one has $\chi(G) = \triangle(G) + 1$). ◁◁

Example 22. Consider the ***Grötsch graph*** G below.

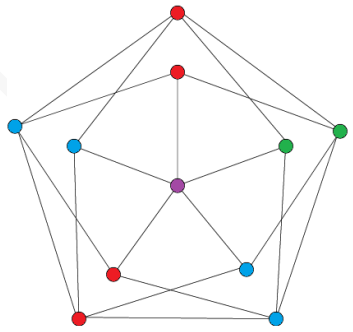


Figure 6.1: Proper coloring of the Grötsch graph

We prove that $\chi(G) = 4$. A proper 4-coloring is shown, so $\chi(G) \leq 4$. To show that 4 colors are needed, we proceed as follows. Consider the "outer" cycle of length five. We know that 3 colors are needed to color this cycle, and we may assume that the colors around the cycle are red, blue, red, blue, green. If we are only allowed three colors, then the color of each vertex adjacent to the central vertex must be the same as its partner on the outer cycle.

However, that means we used three colors in the neighborhood of the central vertex, so the central vertex must have a fourth color (purple in the picture).

6.1 König's Theorem

For any graph, it is clear that $\chi'(G) \geq \triangle(G)$ – all the edges incident with a vertex of degree $\triangle(G)$ must have different colors in a proper coloring. The main theorem we prove on edge-coloring is Vizing's Theorem. Before proceeding to Vizing's Theorem, we discuss edge-colorings of bipartite graphs. König's Theorem states that $\chi'(G) = \triangle(G)$ for any bipartite graph G – thus determining $\chi'(G)$ in bipartite graphs is easy:

Theorem 6.1.1 (König's Theorem) *For any bipartite graph G, $\chi'(G) = \triangle(G)$.*

Proof ▷ The first proof we give relies on Hall's Theorem: we know by Corollary 5.2.2 that every k-regular bipartite multigraph has a k-coloring. So if we can show that G is contained in a $\triangle(G)$-regular bipartite graph, then we are done. To prove this, take two copies of G, say $G_1(A, B)$ and $G_2(A, B)$, and if $y \in A \cup B$ has degree d, add $\triangle(G) - d$ multiple edges between the vertex of $G_1(A, B)$ corresponding to y and the vertex in $G_2(A, B)$ corresponding to y. Then we obtain a graph J which is $\triangle(G)$-regular, so $\chi'(J) = \triangle(G) = \chi'(G)$. ◁◁ □

Proof ▷ The second proof we give is by induction on $|E(G)|$. If $|E(G)| = 0$ then the theorem is clear. Suppose $|E(G)| > 0$ and let $e = \{x, y\} \in E(G)$. By induction, the graph $G - e$ is $\triangle(G)$-edge-colorable. If there is a color i which is not used on any edges incident with x or y, then we can assign color i to $\{x, y\}$ to get a $\triangle(G)$-edge-coloring of G. So we may assume that the colors at x are $1, 2, \ldots, \triangle(G) - 1$ and the colors at y are $2, 3, \ldots, \triangle(G)$. Let H be the subgraph of G spanned by edges of colors 1 and $\triangle(G)$. Then the component of H containing x is a path or a cycle. It cannot be a cycle, otherwise x would be incident with an edge of color 1 and color $\triangle(G)$ in the cycle, contradicting that $\triangle(G)$ is missing at x. So the component of H containing x is a path, P. If P ends at y, then since P has odd length we would have an edge of color 1 at y, a contradiction. So P ends at a vertex $z \neq y$. Now z is not incident with any edge of color 1 or $\triangle(G)$ in $G - E(P)$, otherwise we could extend the path or the edge is incident with a vertex w of the path, but then the coloring would not be a proper edge-coloring. Now interchange colors 1 and $\triangle(G)$ along the path P, to obtain a proper coloring of $G - e$ where the color 1 does not appear at x. Finally, assign e color 1 to get a proper coloring of G. □

6.2 Vizing's Theorem

The next remarkable theorem tells us that $\chi'(G)$ is either the maximum degree of G or one more than that. For example, for the complete graph K_n, we have $\chi'(K_n) = n - 1$ if n is

even and $\chi'(K_n) = n$ if n is odd (the first statement does require a proof – it is equivalent to saying we can partition K_n into $n-1$ pairwise edge-disjoint matchings when n is even – this is left as an exercise). Perhaps surprisingly, it is known to be difficult to determine whether $\chi'(G) = \triangle(G)$ or $\chi'(G) = \triangle(G)+1$ for a given graph G. The graphs G with $\chi'(G) = \triangle(G)$ are called ***class 1*** graphs and those with $\chi'(G) = \triangle(G)+1$ are called ***class 2*** graphs. ◁◁

Theorem 6.2.1 (VIZING'S THEOREM) *For every graph G of maximum degree $\triangle$, $\chi'(G) = \triangle$ or $\chi'(G) = \triangle + 1$.*

Proof ▷ Since $\triangle$ different colors are needed at a vertex of degree $\triangle$ in G, $\chi'(G) \geq \triangle$. Now we prove by induction on $|E(G)|$ that G is $(\triangle+1)$-colorable, which gives $\chi'(G) \leq \triangle + 1$. If $|E(G)| = 0$, then the theorem is clearly true. Suppose $|E(G)| > 0$, and let $\{x, y_1\} \in E(G)$ be any edge of G. By induction, $G_1 = G - \{x, y_1\}$ is $(\triangle+1)$-colorable. Now if there is a color, say color c_1, missing at y_1 and missing at x, then we can assign edge $\{x, y_1\}$ the color c_1. So we can assume that an edge on x, say $\{x, y_2\}$ has color c_1. Let c be a color missing at x – we know c appears on y_1 otherwise $\{x, y_1\}$ could be colored with color c. In general, we construct a maximal sequence $y_1, y_2, \ldots, y_k$ of neighbors of x such that c_i is missing at y_i and $\{x, y_{i+1}\}$ has color c_i for all $i < k$, and color c_k is missing at y_k and does not appear on any edge $\{x, y\}$ for $y \notin \{y_1, y_2, \ldots, y_k\}$.

Case 1. For all $i < k$, $c_k \neq c_i$. In this case, a proper edge-coloring of G is found by recoloring $\{x, y_j\}$ with color c_j for all $j \leq k$. Note that the coloring is proper since color c_j is missing at y_j for all $j \leq k$. An illustration is provided in Figure 6.2.

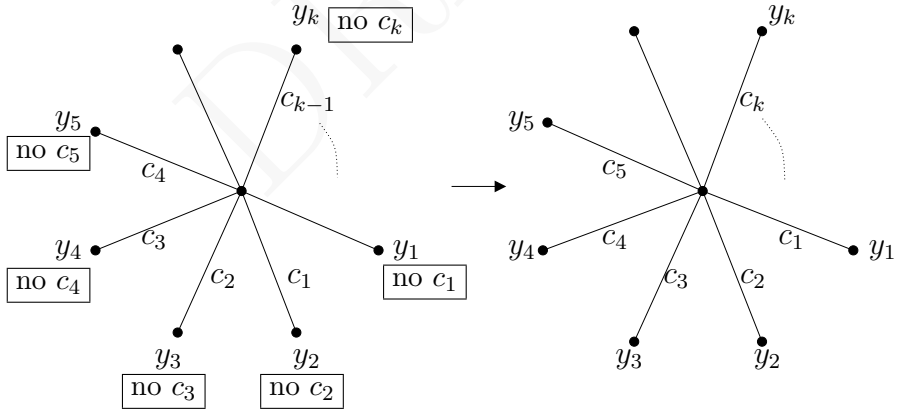


Figure 6.2: Rotate colors around x

Case 2. For some $i < k$, $c_k = c_i$. In this case, recolor all edges $\{x, y_j\}$ for $j \leq i$ with color c_j – so far we still have a proper coloring since color c_j is missing at y_j for $j \leq i$. This

is shown for $i = 4$ in the left diagram in Figure 6.3. Then $\{x, y_{i+1}\}$ is the new uncolored edge, since the edge $\{x, y_1\}$ has now received color c_1. Now let H denote the subgraph of G consisting of edges of color c and edges of color c_k. Then the components of H are paths and cycles, since H has maximum degree at most two. Also x, y_{i+1}, y_k all have degree one in H, so either x, y_{i+1} are in different components of H or x, y_k are in different components of H. We consider these cases separately. If x, y_{i+1} are in different components of H, then we interchange colors c and c_i in the component of H containing y_{i+1}. In this new coloring, color c is missing at x and missing at y_{i+1}, so we can assign the edge $\{x, y_{i+1}\}$ the color c (see Figure 6.3). If x, y_k are in different components of H, then recolor the edge $\{x, y_j\}$ for $i < j < k$ with color c_j, so that $\{x, y_k\}$ is the new uncolored edge. Then H is unchanged (we never recolored edges of color c or c_i) so we may interchange the colors c and c_k in the component of H containing y_k. In doing so, c becomes a missing color at x and y_k, so the uncolored edge $\{x, y_k\}$ can be colored with color c. This completes the proof. □

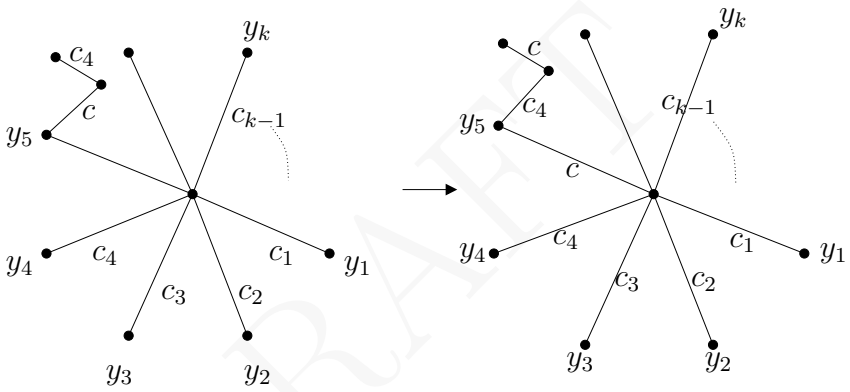


Figure 6.3: Interchanging colors in components of H

6.3 Brooks' Theorem

The chromatic number of a graph G is the minimum number of colors which can be assigned to the vertices of G so that no two adjacent vertices have the same color. This number is denoted $\chi(G)$. Unlike in the case of edge-coloring, $\chi(G)$ can be arbitrarily small relative to $\triangle(G)$: for example $\chi(G) \leq 2$ if and only if G is a bipartite graph. One also notices that $\chi(G) = \triangle(G) + 1$ is possible, since $\chi(K_n) = n$ and $\chi(C) = 3$ when C is an odd cycle. In fact these are the only cases where $\chi(G) = \triangle(G) + 1$:

Theorem 6.3.1 (Brooks' Theorem) *Let G be a connected graph of maximum degree $\triangle$. Then $\chi(G) \leq \triangle$, unless G is an odd cycle or a complete graph.*

Proof ▷ If G has no subgraph of minimum degree at least $\triangle$, then we may repeatedly remove vertices $v_1, v_2, \ldots, v_n$ such that v_i has degree at most $\triangle - 1$ in $G_i = G - \{v_1, v_2, \ldots, v_{i-1}\}$

until no vertices are left. Let the colors be $1, 2, \ldots, \triangle$. Now color v_n with color 1, and in general, color v_i with the first available color not used on its neighbors in G_i. Since v_i has degree at most $\triangle - 1$ in G_i, there is always a color from $1, 2, \ldots, \triangle$ available to properly color v_i. This completes the proof in this case, so we assume G has a subgraph of minimum degree at least $\triangle$. Since G is connected and has maximum degree $\triangle$, this implies G is $\triangle$-regular. If $\triangle = 2$, the proof is complete since even cycles have chromatic number $\triangle = 2$. Suppose $\triangle > 2$.

If G is not complete, then it is possible to find vertices x, y, z in G such that $\{x, y\}$ and $\{x, z\}$ are edges in G but $\{y, z\}$ is not an edge in G. We order the vertices of G so that x is first and y and z are last. If $G - \{y, z\}$ is connected, then we can order the vertices of $G - \{y, z\}$ as $v_1, v_2, \ldots, v_{n-2}$ where $v_1 = x$ and for $i > 1$, v_i has at least one neighbor v_j with $j < i$. Let $v_{n-1} = y$ and $v_n = z$. Then we color v_n and v_{n-1} with color 1, and color v_i for $i > 1$ with the first available color from $\{1, 2, \ldots, \triangle\}$. Such a color is always available, since there are at most $\triangle - 1$ colored neighbors v_j of v_i with $j > i$. To color v_1, we note that the number of colors used on the neighbors of v_1 is at most $\triangle - 1$, since v_n and v_{n-1} both received color 1. Therefore there is an available color for v_1 from $\{1, 2, \ldots, \triangle\}$ to complete the proper coloring.

Now suppose $G - \{y, z\}$ is not connected. If there is a vertex $v \in V(G)$ such that $G - \{v\}$ is disconnected, then $G = G_1 \cup G_2$ where $V(G_1) \cap V(G_2) = \{v\}$, and we color G_1 and G_2 each with at most $\triangle$ colors, making sure the colors match on v. If no such vertex v exists, then $G = H \cup I$ where H and I are induced subgraphs of G such that $V(H) \cap V(I) = \{y, z\}$, and y and z both have degree at least 1 in H and I. Let e denote the edge $\{y, z\}$ (not an edge of G). Then $H + e$ and $I + e$ both have maximum degree at most $\triangle$, so both these graphs can be properly colored with $\triangle$ colors, so that y and z receive colors 1 and 2. This means the union of the proper colorings of $H + e$ and $I + e$ is a proper coloring of the whole graph G with $\triangle$ colors. □

6.4 Degenerate graphs

A graph is called ***d*-degenerate** if it has no subgraph of minimum degree more than d. In this section, we prove a proposition which often gives a better bound for $\chi(G)$ than $\triangle(G)$. The idea is to remove vertices of small degree from the graph and to notice that whenever we remove a vertex v of degree at most k from a graph G and obtain a graph with a proper $(k + 1)$-coloring, then we can reinsert v and color it with a color not used on any of its neighbors to obtain a proper $(k + 1)$-coloring of G.

Proposition 6.4.1 *Let G be a d-degenerate graph. Then $\chi(G) \leq d + 1$.*

Proof ▷ Order the vertices $(v_1, v_2, \ldots, v_n)$ so that v_i has at most d neighbors v_j with $j < i$. Then assign v_1 color 1, and in general assign v_i the first color from $\{1, 2, \ldots, d+1\}$ that has not appeared on a neighbor v_j of v_i with $j < i$. Then this is a proper $(d+1)$-coloring of G, so $\chi(G) \leq d + 1$. □

6.5 Scheduling Problems

The ***Scheduling Problem*** (also known as the ***Timetable Problem*** or ***Storage Problem***) is a natural application of vertex coloring: we have a number of events to be scheduled and a number of participants in those events. We form a graph G whose vertex set is the set of events, and whenever two events have a participant in common, we put an edge between those events. The question is to determine the minimum number of time slots into which the events can be scheduled. This is equivalent to determining $\chi(G)$. The ***Storage Problem*** is defined by a number of items which have to be stored in containers, however some pairs of items cannot be stored in the same container (for instance, chemicals which might react). We want to minimize the total number of containers that can be used to store the items.

Example 23. Suppose five students $\{s_1, s_2, s_3, s_4, s_5\}$ have to write some exams from a set $\{t_1, t_2, t_3, t_4\}$. The first three students must write exams t_1, t_2, t_3. Then s_4 must write t_3 and t_4 and s_5 must write t_2 and t_4. The corresponding graph is shown below:

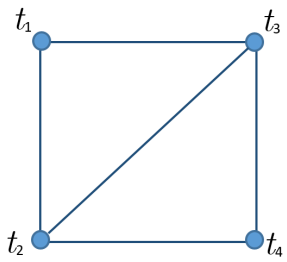


Figure 6.4: Scheduling

This graph is easily 3-colorable: we assign time slot 1 to exams t_1 and t_4, time slot 2 to exam t_2 and time slot 3 to exam t_3.

6.6 Exercises

Question 6.1° Determine $\chi'(G)$ and $\chi(G)$ for each of the graphs shown below.

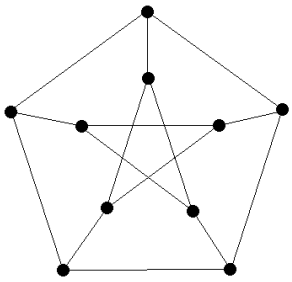


Figure 6.5: The Petersen graph

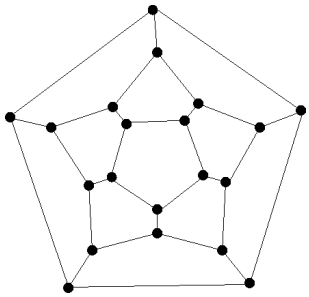


Figure 6.6: The dodecahedron graph

Question 6.2° A factory wishes to store the following chemicals in storage containers: hydrogen, helium, oxygen, chlorine, sulfur, and iron. The chemicals must be stored in separate containers if they are liable to react with one another. Determine the minimum number of containers to store all these chemicals.

Question 6.3° Is it possible for a cubic Hamiltonian graph to have exactly one 3-edge-coloring?

Question 6.4° Classify as class 1 or class 2 all connected graphs with at most five vertices.

Question 6.5. Determine $\chi'(G)$ when G is the ***Grötsch graph***, shown in Figure 6.1.

Question 6.6. Let $\omega(G)$ be the maximum number of vertices in a complete subgraph of a graph G.

(a) Prove that for every graph G, $\chi(G) \geq \omega(G)$.
(b) Prove that for every graph G, $\chi(G) \geq |V(G)|/\alpha(G)$.
(c) For each $k \geq 2$, find a graph G such that $\chi(G) = k + 1$ and $\omega(G) = k$.

Question 6.7. Prove that $\chi'(K_n) = n$ if n is odd and $\chi'(K_n) = n - 1$ if n is even without using Vizing's Theorem.

Question 6.8.

(a) Prove that $\chi'(G)\alpha'(G) \geq |E(G)|$ for every graph G.
(b) Let G be a graph obtained by removing less than $(n-1)/2$ edges from K_n, where $n \geq 3$ is odd. Show that $\chi'(G) = n$.

Question 6.9. Let G be a bipartite graph of maximum degree $\triangle$. Prove that there exists a $\triangle$-regular bipartite graph H containing G as a subgraph.

Question 6.10. Let $k \geq 2$ and let G be a graph of chromatic number k such that $\chi(G - \{v\}) < k$ for every $v \in V(G)$ (these are called ***k-critical graphs***).

(a) If $k = 2, 3$, describe the graph G.
(b) Prove that $\delta(G) \geq k - 1$.
(c) Show that G is a block.

Question 6.11. Show that the maximum number of edges in an n-vertex graph of chromatic number k is at most $(k-1)n^2/2k$.

Question 6.12. Let $n > k \geq 1$, and let G be a k-degenerate n-vertex graph. Prove that $|E(G)| \leq k(n-k) + \binom{k}{2}$. Is this best possible?

Question 6.13. Let G_1 and G_2 be graphs with vertex set V, and define the graph $G_1 \cup G_2$ to consist of vertex set V and edge-set $E(G_1) \cup E(G_2)$. Let $c_1 : V \to \{1, 2, \ldots, k\}$ and $c_2 : V \to \{1, 2, \ldots, \ell\}$ be proper vertex colorings of G_1 and G_2, respectively. Let $c : V \to \{(i, j) : 1 \leq i \leq k, 1 \leq j \leq \ell\}$ be a vertex coloring of $G_1 \cup G_2$ defined by $c(v) = (c_1(v), c_2(v))$ for $v \in V$.

(a) Prove that c is a proper coloring of $G_1 \cup G_2$.
(b) Prove that $\chi(G_1 \cup G_2) \leq \chi(G_1)\chi(G_2)$.
(c) Is it possible to express K_9 as a union of three bipartite graphs?

Question 6.14. Let G be a bipartite graph and let H be the graph with $V(H) = V(G)$ and $E(H) = \{\{x, y\} : \{x, y\} \notin E(G)\}$ – this is the ***complement*** of G. Prove that $\chi(H) = \max\{r : K_r \subseteq H\}$.

Question 6.15. Let G be a 3-regular graph.

(a) Show that if G is Hamiltonian, then $\chi'(G) = 3$.
(b) Prove that $\chi'(G) = 3$ if and only if there exists a set $\mathcal{C}$ of pairwise vertex-disjoint even cycles such that $V(G) = \bigcup_{C \in \mathcal{C}} V(C)$.
(c) Find a 3-regular graph with $\chi'(G) = 4$.

Question 6.16. Let $k \geq 1$. Prove that if G is a $2k$-regular graph, then there exists a set $\mathcal{C}$ of pairwise vertex-disjoint cycles such that $V(G) = \bigcup_{C \in \mathcal{C}} V(C)$.

Question 6.17. Let G be a graph of chromatic number four such that for any vertices $x, y \in V(G)$, $\chi(G - \{x, y\})$ is bipartite. Prove that $G = K_4$.

Question 6.18. Prove that if G is any graph and $a, b \geq 1$ satisfy $a + b = \chi(G)$, then there exists a partition (X, Y) of $V(G)$ such that $\chi(G[X]) = a$ and $\chi(G[Y]) = b$.

Question 6.19. Prove that if $\chi(G) = k+1$ and $\chi(G-\{e\}) < \chi(G)$ for every edge $e \in E(G)$, then G is k-edge-connected.

Question 6.20. Construct for each $n \geq 1$ a graph G with $4n$ vertices and at least n^2 edges and $\chi(G) = 4$ such that $\chi(G - \{x\}) = 3$ for every $x \in V(G)$.

Question 6.21* Find for each $\triangle \geq 4$ a multigraph G such that $\chi'(G) = \lfloor 3\triangle/2 \rfloor$ and $\triangle(G) = \triangle$. Then prove that $\chi'(G) \leq \lfloor 3\triangle(G)/2 \rfloor$ for every multigraph G.

Question 6.22* Let $k \geq 2$ and $n \geq 1$, and let $G_1, G_2, \ldots, G_k$ be pairwise vertex-disjoint complete graphs on n vertices. Let $G_{n,k}$ be the graph obtained by joining every vertex of G_i to every vertex of G_{i+1} for $i < k$ and every vertex of G_k to every vertex of G_1. The graph $G_{2,13}$ is shown below. Determine $\chi'(G_{n,k})$ and $\chi(G_{n,k})$.

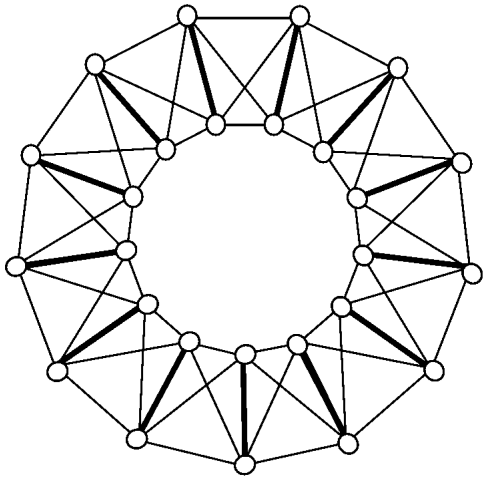


Figure 6.7: Graph $G_{2,13}$

Question 6.23* Let G be a graph. An ***orientation*** of G is a digraph $\vec{G}$ obtained by replacing each edge $\{a, b\} \in E(G)$ with either the arc (a, b) or the arc (b, a). Prove that if $\chi(G) \geq k$, then every orientation $\vec{G}$ of G contains a ***directed path*** of length at least k – a digraph with vertex set $\{v_1, v_2, \ldots, v_\ell\}$ and edges $(v_1, v_2), (v_2, v_3), \ldots, (v_{\ell-1}, v_\ell)$ where $\ell \geq k + 1$.

7 Planar graphs

Roughly speaking, a graph is planar if and only if it can be drawn in the plane without any two of its edges crossing. More formally, an ***embedding*** of a graph $G = (V, E)$ is a function $f : V \cup E \to \mathbb{R}^2 \cup \mathcal{C}$, where $\mathcal{C}$ is the set of continuous curves in $\mathbb{R}^2$, such that f is one-to-one, $f(v)$ is a point in $\mathbb{R}^2$ for each $v \in V$, and $f(\{u, v\})$ is a continuous curve in $\mathbb{R}^2$ with ends u and v when $\{u, v\} \in E$. The graph G is ***planar*** if we can choose f so that the curves $f(e) : e \in E$ meet only at their ends – that is no curve meets itself and any point in the intersection of two distinct curves is an endpoint of both of the curves. A drawing of G without crossings is called a ***plane embedding*** of G, or a ***plane graph***. Thus a graph is planar if and only if it has a plane embedding.

The main theorem of this section, due to Kuratowski [25], is a necessary and sufficient condition for a graph to be planar – and a characterization of planar graphs. A ***subdivision*** of a graph G is any graph obtained from G by replacing each edge of G with a path with the same ends as the edge, such that paths may meet only at their ends.

Theorem 7.0.1 (Kuratowski's Theorem) *A graph is planar if and only if it contains no subdivision of K_5 and no subdivision of $K_{3,3}$.*

7.1 Euler's Formula

Throughout this section, we deal only with connected graphs. If G is a plane graph, then $\mathbb{R}^2 \backslash G$ consists of a union of disjoint connected plane regions, which are called ***faces*** of G. The ***boundary*** of a face F of G is the set of points in the topological closure of F which are not in the interior of F. Each plane graph has a face which is infinite, which we refer to as the ***infinite face***. The ***boundary walk*** of a face F with a connected boundary, denoted by ∂F, is the shortest closed walk consisting of edges and vertices in the boundary of F. We denote by F(G) the set of faces of a plane graph G. The ***degree*** of a face $F \in F(G)$ is the length of the walk ∂F, and denoted $\deg(F)$.

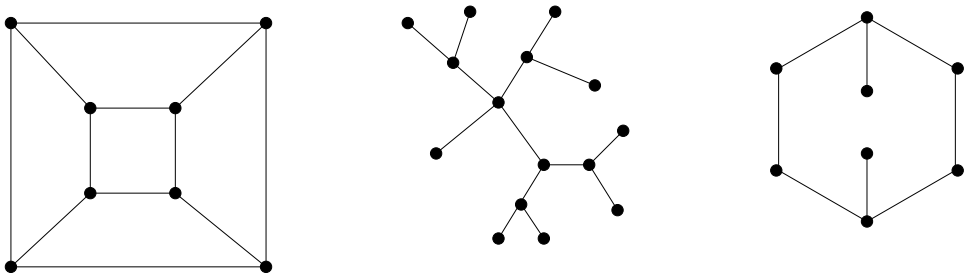


Figure 7.1: Faces of a plane graph

The graph on the left in Figure 7.1 has six faces, all boundary walks of which are cycles of length four – so every face has degree four. The tree in the centre has only one face – the infinite face – and since a tree on n vertices has $n-1$ edges and the boundary walk goes through each edge twice, the degree of the infinite face is $2(n-1)$. In the graph on the right, there are two faces, one of degree six and one of degree ten.

The degrees of the faces in a plane graph depend very much on the way the graph is drawn in the plane: for example, the graph on the right in Figure 7.1 can be redrawn as a new plane graph by flipping one of the bridges into the infinite face, thereby producing two new faces, both of degree eight.

There is a very useful analog of the handshaking lemma for face degrees in a plane graph. If we add up the degree of every face $F \in F(G)$, we observe that every edge of the graph is counted exactly twice. This is true since an edge in a cycle is counted once for each of the faces on either side of it, and an edge which is not in a cycle is a bridge (Lemma 3.1.1), and therefore counted twice in one boundary walk. These observations give the following useful fact, which is the analog of the handshaking lemma for face degrees:

Lemma 7.1.1 *Let G be a plane graph. Then*

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

In general, note the a bridge on the boundary of a face is counted twice in the boundary walk of that face, whereas all other edges in the boundary are counted once in the boundary walk.

Lemma 7.1.1 is very useful in conjunction with Euler's Formula and the handshaking lemma for proving non-existence of planar graphs with given face and vertex degrees. Euler's Formula [9] relates $|F(G)|$, $|E(G)|$ and $|V(G)|$ as follows:

Theorem 7.1.2 (Euler's Formula) *Let G be a connected plane graph. Then*

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

Proof ▷ Proceed by induction on $|E(G)|$. The minimum value of $|E(G)|$ is $|V(G)| - 1$, by Proposition 3.1.3. In that case, $|F(G)| = 1$ and Euler's Formula is satisfied. So we may assume that $|E(G)| > |V(G)| - 1$ and G contains a cycle C. Let e be an edge of C. By Lemma 3.1.1, $G - e$ is connected, since e is not a bridge. By induction,

$$|V(G-e)| - |E(G-e)| + |F(G-e)| = 2.$$

We now observe $|E(G-e)| = |E(G)| - 1$ and $|F(G-e)| = |F(G)| - 1$ and $|V(G-e)| = |V(G)|$. It follows that ◁◁

$$|V(G)| - (|E(G)| - 1) + (|F(G)| - 1) = 2$$

and this gives Euler's Formula. □

A useful application is to give a sufficient condition for non-planarity:[14]

Theorem 7.1.3 *Let G be a planar graph containing a cycle. Then $|E(G)| \leq \frac{g}{g-2}(|V(G)|-2)$, where g is the length of a shortest cycle in G. In particular, for any planar graph G, $|E(G)| \leq 3|V(G)| - 6$, and therefore G is 5-degenerate.*

Proof ▷ Since every face has degree at least g, Theorem 7.1.1 gives $g|F(G)| \leq 2|E(G)|$. Putting this in Euler's Formula, we get

$$|V(G)| - |E(G)| + \frac{2}{g}|E(G)| \geq 2$$

which, rearranged, gives the required bound on $|E(G)|$. The right side of the formula is maximized when $g = 3$, in which case we get $|E(G)| \leq 3|V(G)| - 6$ for all planar graphs G. By the handshaking lemma, if all vertices of G had degree at least six, then $|E(G)| \geq 3|V(G)|$, a contradiction to what we just proved. So every planar graph has a vertex of degree at most five. Since every subgraph of G is also planar, this means that every subgraph of G has a vertex of degree at most five, so G is 5-degenerate. □

By Theorem 7.1.3, any graph satisfying $|E(G)| > \frac{g}{g-2}(|V(G)| - 2)$ can't be planar. In particular, K_5 is not planar since $|E(K_5)| = 10$ and $g = 3$, and $K_{3,3}$ is not planar since $|E(K_{3,3})| = 9$ and $g = 4$. A ***maximal planar*** graph is a graph that is planar but the addition of any edge results in a non-planar graph. A ***maximal plane*** graph is a plane drawing of a maximal planar graph. Evidently, every face of a maximal plane graph with at least three vertices is a triangle, and using Lemma 7.1.1 one can show that a maximal planar graph with $n \geq 3$ vertices has exactly $3n - 6$ edges and $2n - 4$ faces. ◁◁

7.2 Platonic solids

A ***platonic solid*** is a connected plane graph where all vertices have the same degree r and all faces have the same degree s. It has been known for millenia that there are only five platonic solids, as shown in Figure 7.2. We use Euler's Formula to determine the five possible pairs (r, s) for which such graphs exist:

[14] Trying to draw the graph in every possible way is inefficient.

Theorem 7.2.1 *Let $r \geq 3$ and $s \geq 3$. Then there exists a connected plane graph with all vertices of degree r and all faces of degree s if and only if $(r, s) \in \{(3,3), (3,4), (4,3), (5,3), (3,5)\}$.*

Proof ▷ Figure 7.2 shows that the plane graphs exist for those values of r and s. Now we show that these are the only values for which they exist. By the handshaking lemma, if G is such a plane graph then $|E(G)| = r|V(G)|/2$. By Lemma 7.1.1, $|E(G)| = s|F(G)|/2$. By Euler's Formula, and since G is connected,

$$|V(G)| - |E(G)| + |F(G)| = 2|E(G)|/r - |E(G)| + 2|E(G)|/s = 2.$$

So $|E(G)| = 2rs/(2s - rs + 2r)$. Since $|E(G)|$ is positive, $rs < 2(s + r)$ and this is only possible for the pairs (r, s) listed in the theorem. □

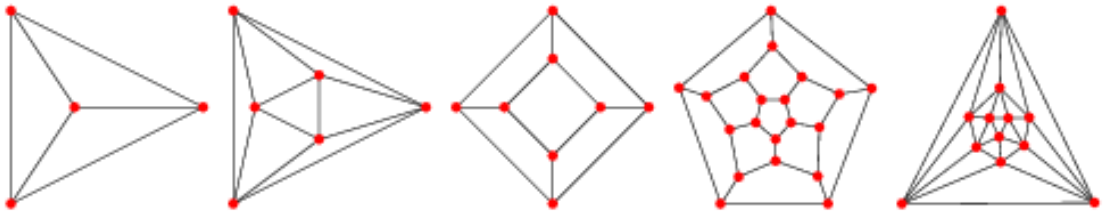


Figure 7.2: The platonic solids

7.3 Coloring planar graphs

Euler's Formula also can be applied to vertex-coloring of planar graphs. Recall that a graph is d-degenerate if every subgraph of G (including G itself) has minimum degree at most d. Also, any d-degenerate graph is $(d + 1)$-colorable, by Proposition 6.4.1. By Theorem 7.1.3, every planar graph is 5-degenerate, so this means that every planar graph is 6-colorable. Here is another example: suppose we have a planar graph G of girth at least six. Then $|E(G)| \leq \frac{3}{2}(|V(G)| - 2)$ by Theorem 7.1.3, so every subgraph of G must have a vertex of degree at most two, by the handshaking lemma. Therefore G is 2-degenerate, which means that G is 3-colorable. We prove the 5-color theorem here, using the notion of contraction of edges.

Recall the ***contraction*** of a pair of vertices $\{a, b\} \subset V(G)$ is the graph $G/\{a, b\}$ obtained from G by identifying the vertices a and b and joining the new vertex ab to all neighbors of a and all neighbors of b. It is not hard to show that if G is a planar graph and $\{a, b\} \in E(G)$, then $G/\{a, b\}$ is planar. A key point in the proof of the 5-color theorem is that if $\{a, b\} \notin E(G)$ and if there is a proper coloring $c : V(G/\{a, b\}) \to \{1, 2, \ldots, k\}$, then the coloring ◁◁

$c' : V(G) \to \{1, 2, \ldots, k\}$ defined by $c'(v) = c(v)$ if $v \notin \{a, b\}$ and $c'(a) = c'(b) = c(ab)$ is a proper coloring of G.

Theorem 7.3.1 *Every planar graph is 5-colorable.*

Proof ▷ Proceed by induction on $|V(G)|$. If $|V(G)| \leq 5$, then the theorem is true: just assign all vertices different colors. Now suppose $|V(G)| > 5$. If G has a vertex v of degree at most four, then $G - \{v\}$ is 5-colorable by induction, and we can extend this coloring to v by assigning to v a color which does not appear on any of its neighbors, since there were five colors but at most four neighbors of v. So now we assume G has no vertex of degree at most four.

Since G is 5-degenerate, by Theorem 7.1.3, G has a vertex v of degree exactly five. If all neighbors of v are adjacent to each other, then they form a K_5, but K_5 is not planar (as we saw from Theorem 7.1.3), so that is a contradiction. Therefore we can pick neighbors a and b of v which are not adjacent. We contract the set $\{a, b, v\}$ to a new vertex w: consider the graph $H = G/\{a, b, v\}$ and let w be the vertex of H joined to every vertex in $N(a) \cup N(b) \cup N(v)$ – see Figure 7.3. This graph H is still planar. By induction, H has a 5-coloring $c : V(H) \to \{1, 2, 3, 4, 5\}$. For each $u \in V(G)\backslash\{a, b, v\}$, let $c'(u) = c(u)$. Let $c'(a) = c'(b) = c(w)$ – we can do that since a and b are not adjacent. Finally, the number of colors used by neighbors of v in G in the coloring c' is at most four, since a and b got the same color. So there is a color i not used by any neighbor of v, and we let $c'(v) = i$. Then c' is a proper 5-coloring of G. □

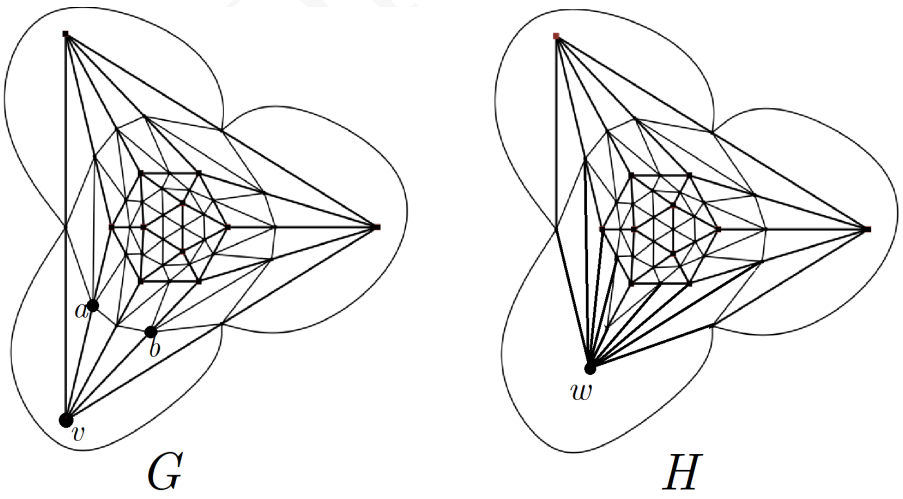


Figure 7.3: Proof of the 5-Color Theorem

Perhaps the most famous theorem in graph theory is the 4-color theorem, proved by Appel and Haken (1976): every planar graph is 4-colorable. Unfortunately, there is no proof known which is not computer assisted. The shortest proof is currently the one in Robertson and Seymour (1997).

Theorem 7.3.2 (4-Color Theorem)
Every planar graph is 4-colorable.

7.4 Drawing planar graphs*

We remarked earlier that there are many plane embeddings for a given planar graph G; even the degrees of the faces can change with different embeddings (see Figure 7.1). In fact, we can go from any plane embedding of G to any other plane embedding of G using the notion of stereographic projection. In particular, we can make any face of a plane embedding the infinite face:

Proposition 7.4.1 *Every face of a plane embedding G_0 of a graph G is the infinite face of some plane embedding of G. Furthermore, if every edge of G_0 is a straight line, then we can ensure that every edge of the new embedding is also a straight line.*

Proof ▷ Let $\mathbb{S}$ denote a sphere of diameter one placed so that the xy-plane is tangent to $\mathbb{S}$ at the origin. Then wrap the plane embedding G_0 of G around the sphere. Formally, consider the function f which maps a point (x, y) to the point $(x, y, z) \in \mathbb{S}$ which is at height $z = 1 - 1/(1 + x^2 + y^2)$ in the plane defined by the line through the origin and (x, y) and the z-axis. Note that f is a bijection between $\mathbb{R}^2$ and $\mathbb{S}\backslash N$, where $N = (0, 0, 1)$ denotes the north pole of $\mathbb{S}$. Let H be the image of G_0 under f. Keeping H fixed, rotate the sphere until some face F of H contains the north pole of $\mathbb{S}$. Now apply f^{-1} to get an embedding of G, namely $f^{-1}(H)$, with the property that the face F of $f^{-1}(H)$ is the infinite face. The second statement of the theorem is left as an exercise. □ ◁◁

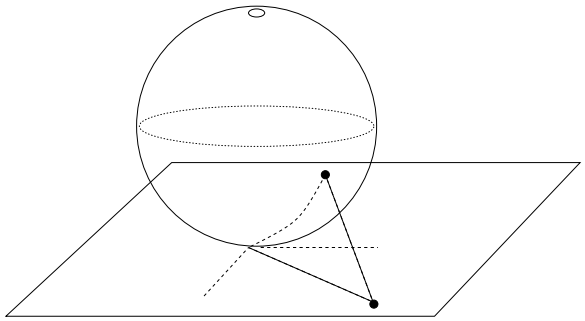


Figure 7.4: Stereographic projection

Perhaps the most natural embedding would be to try to draw the edges as striaght lines. This can be done, by the following theorem:

Theorem 7.4.2 (Fary's Theorem) *Every planar graph has a plane embedding in which all edges are straight line segments.*

Proof ▷ By Theorem 7.1.3, every planar graph is 5-degenerate. Now we proceed by induction on $|V(G)|$, the number of vertices in a planar graph G. If $|V(G)| \leq 3$, then the result is obvious. Suppose G is a planar graph with $|V(G)| > 3$. We may assume that G is maximal planar – so $G + e$ is not planar anymore for any edge e. Then if G_0 is a plane embedding of G, all faces of G_0 have degree three, otherwise we could add a diagonal edge in some face. Now let v be a vertex of degree at most five in G_0. Then $G_0 - v$ has a plane embedding, call it H, such that all the edges are straight lines. Let $v_1, v_2, \ldots, v_k$ be the neighbors of v, where $k \leq 5$. Then there is a cycle $C \subset H$ such that $V(C) = \{v_1, v_2, \ldots, v_k\}$ – since every face of G_0 is of degree three, every face of H containing non-neighbors of v on its boundary is a triangle. This means that C is the boundary of a face of H. By Proposition 7.4.1, we can make C the boundary of a finite face of H. Now place v in the interior of C so that v sees all vertices of C – that is, we can draw a straight line segment from v to each vertex $v_1, v_2, \ldots, v_k$. This is an embedding of G in which all edges are straight lines. □

Concerning properties of the drawing of a planar graph, we have seen that there are, in general, plane embeddings with different face degrees (see Figure 7.1). Furthermore, we can't ensure that the faces are convex, even for 2-connected planar graphs, for example $K_{2,3}$ has no embedding in which all faces are convex. However, Tutte showed that every 3-connected planar graph can be drawn with convex faces and straight line edges, and Whitney's Theorem states that the embedding is unique.

Theorem 7.4.3 (Tutte-Whitney Theorem)
Every 3-connected planar graph has a unique embedding in the plane in which every face is a convex polygon and every edge is a straight line segment.

A natural physical interpretation is to nail down the edges of a cycle which is a face in a plane embedding of G, and replace the edges with rubber bands. Then, allowing this dynamical system to reach equilibrium in terms of the laws of physics, Tutte proved that the plane embedding at equilibrium is a convex straight line embedding. We do not prove this or Theorem 7.4.3 here.

7.5 The Art Gallery Theorem*

Let R be a connected region in the plane bounded by an n-sided polygon (we include here the boundary in R, so R is a closed region). Two points of R are ***mutually visible*** if there

exists a straight line segment between the two points that is entirely contained in R. The ***art gallery problem*** is to determine the minimum size $f(R)$ of a set S of points in R such that for any point $x \in R$, there is a point $y \in S$ such that x and y are mutually visible. We say that every point in R is visible from S. Evidently, if S is the set of vertices of the n-sided polygonal boundary of R, then every point in R is visible from S, so $f(R) \leq n$ for any region R bounded by an n-sided polygon. The art gallery theorem of Chvatal [10] gives the optimal value of $f(R)$:

Theorem 7.5.1 (Art Gallery Theorem)
Let $n \geq 3$. For every n-sided polygonal region R, $f(R) \leq \lfloor n/3 \rfloor$. Furthermore, there exists an n-sided polygonal region R such that $f(R) = \lfloor n/3 \rfloor$.

Proof ▷ Let us ***triangulate*** the region R. In other words, we add straight lines between vertices of the boundary to obtain a plane graph G where every face other than the infinite face is a triangle inside R. We prove by induction on n that $\chi(G) = 3$. For $n = 3$, this is clear, since $G = K_3$. Suppose $n \geq 4$. Then some edge $e = \{u, v\}$ of G is not on the boundary of the infinite face of G. Then $\{u, v\}$ is a 2-vertex cut of G, so $G = G_1 \cup G_2$ where G_1 is a triangulation of a region R_1 and G_2 is a triangulation of a region R_2 such that $R_1 \cup R_2 = R$. Since both G_1 and G_2 have at least three vertices, $\chi(G_1) = 3$ and $\chi(G_2) = 3$. If $c_i : V(G_i) \to \{1, 2, 3\}$ is a proper 3-coloring of G_i, we can ensure $c_i(u) = 1$ and $c_i(v) = 2$ for $i \in \{1, 2\}$. Now let $c(x) = c_i(x)$ if $x \in V(G_i)$. Then $c : V(G) \to \{1, 2, 3\}$ is a proper 3-coloring of G, so $\chi(G) \leq 3$. Since G contains a triangle, we conclude $\chi(G) = 3$.

If $c : V(G) \to \{1, 2, 3\}$ is a proper 3-coloring of G, then for some $i \in \{1, 2, 3\}$, there are at most $\lfloor n/3 \rfloor$ vertices of color i. If S is the set of vertices of that color, then $|S| \leq \lfloor n/3 \rfloor$ and every triangle in G contains exactly one vertex of S, since each triangle uses all three colors on its vertices. Now in a triangular region, any two points are mutually visible, so since R is a union of triangular regions, every point in R is visible from S, as required.

To find R such that $f(R) = \lfloor n/3 \rfloor$, consider first the region R bounded by a polygon with vertices at

$$(0,0), (1,1), (2,0), (3,1), (4,0), (5,1), \ldots, (4k-1,1), (4k,0) \quad \text{and} \quad (0,-0.1), (4k,-0.1)$$

– see Figure 7.5. If $S = \{(2,0), (6,0), (10,0), \ldots, (4k-2,0)\}$ then every point in R is visible from S and $f(R) = |S| = k$ whereas $n = 3k+2$. Therefore $f(R) = \lfloor n/3 \rfloor$. By removing one or two boundary points from R, we get a region R' bounded by an n-gon with $n = 3k+1$ and $n = 3k$ and $f(R') = k$. □ ◁◁

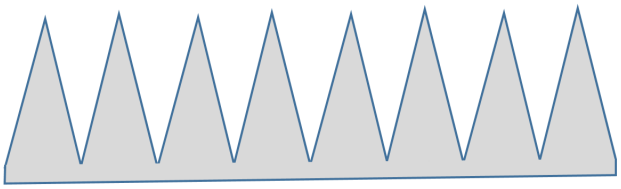


Figure 7.5: Art gallery R with $3k+2$ sides and $f(R) = k$

7.6 Duality*

Let G be a plane graph, and let G^* denote the pseudograph obtained by placing a vertex v_f in the interior of each face $f \in F(G)$ and whose edges are defined as follows: (1) for each bridge on the boundary of a face $f \in F(G)$, join v_f to v_f with a loop in G^* passing through the bridge. (2) for each edge $e \in E(G)$ on the boundary of distinct faces $f, g \in F(G)$, join v_f and v_g by an edge in G^* which crosses e. Then G^* is referred to as the ***plane dual*** or ***combinatorial dual*** of G. Examples of duals are shown in Figure 7.6:

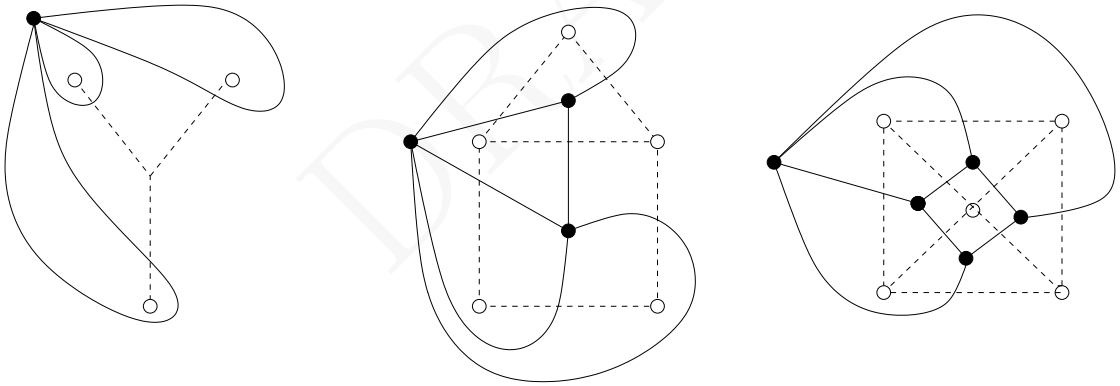


Figure 7.6: Duality

There are many uses of duality in planar graphs, but for brevity we mention one example in coloring. The map coloring problem is to color the faces of a plane graph in such a way that whenever two faces share an edge, they have different colors. Now by drawing the dual of a planar graph, we see that the map coloring problem on a plane graph is equivalent to the vertex coloring problem in the dual, except that we have to remove all loops in the

dual. By the 4-color theorem, this means that the regions of any map can be colored in four colors in such a way that adjacent regions have different colors. In fact, even more is true: if we want to prove the 4-color theorem for plane graphs, it is sufficient to consider maximal plane graphs on at least three vertices (i.e. if we add any edge we get a non-planar graph). In a maximal plane graph, all faces are bounded by triangles, and therefore the dual of a maximal plane graph is a cubic graph. It can also be checked that every maximal plane graph, except a triangle, is 3-connected, and that the dual is therefore also 3-connected. The oldest approach to the 4-color theorem is to try to prove that every cubic graph is 3-edge-colorable: in fact this is equivalent to the 4-color theorem. ◁◁

Theorem 7.6.1 *Every planar graph is 4-colorable if and only if every cubic planar 3-connected graph has edge-chromatic number three.*

Proof ▷ Let G be a planar graph and let G_0 be a plane embedding of G. Then G_0 is contained in a maximal plane graph G_1. If every planar graph is 4-colorable, then G_1 is 4-colorable which means that the map G_1^* is 4-face-colorable and cubic. Since G_1 is 3-connected, no edge of G_1^* is a bridge so every edge of G_1^* is on the boundary of exactly two faces. Now assign edge-color 1 to those edges of G_1^* on the boundary of faces of color 1 and 2, or color 3 and 4, assign edge-color 2 to those edges of G_1^* on the boundary of faces of colors 1 and 3, or colors 2 and 4, and assign edge-color 3 to all remaining edges of G. One checks that this is a proper 3-edge-coloring of G^*, as required.

Define G, G_0, G_1, G_1^* as in the first part of the proof. If every cubic planar graph is 3-edge-colorable, then G_1^* has a proper 3-edge-coloring, with colors 1, 2 and 3. That is to say that $G_1^* = M_1 \cup M_2 \cup M_3$ where M_i is the perfect matching consisting of edges of color i. Then $H_1 = M_1 \cup M_2$ is a plane graph and $H_2 = M_1 \cup M_3$ is a plane graph. Color the faces of H_1 with colors 1 and 2, and color the faces of H_2 with colors 1' and 2'. To get a coloring of the faces of G_1^*, and hence a color of G, color a face F with color (i, j') if it is contained in a region of color i in H_1 and a region of color j' in H_2. Then the number of colors we used is four, and one checks that this a proper coloring of the faces of G_1^*. □

Tait [38] conjectured in the 19th century that all 3-connected cubic planar graphs are Hamiltonian – but this is false, as a counterexample of Tutte [41] on forty-six vertices showed (Figure 7.7). Tutte's counterexample is shown below, as well as a smallest counterexample, with 38 vertices (there are six examples with 38 vertices). If Tait's conjecture had been true, then we could color the Hamilton cycle red and blue, and the remaining matching with green to get a proper 3-coloring of every cubic 3-connected graph.

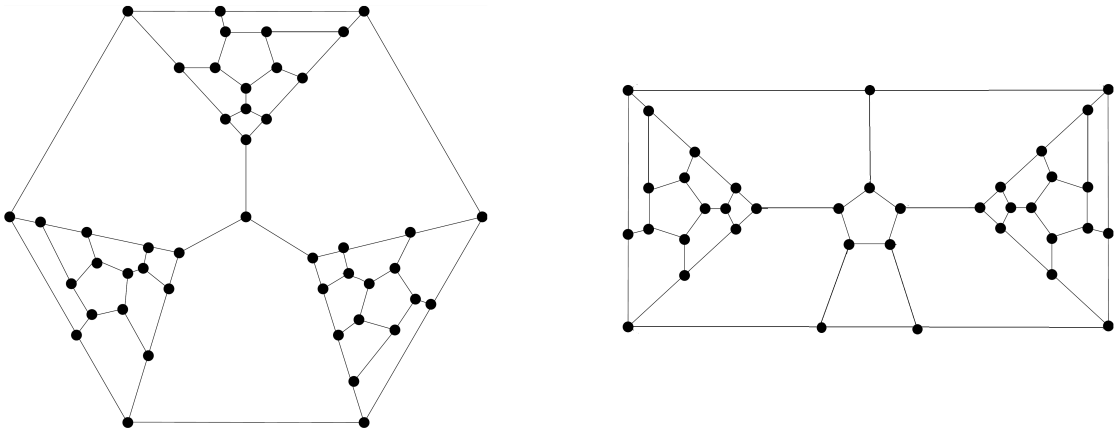


Figure 7.7: Non-Hamiltonian cubic planar graphs

7.7 Kuratowski's Theorem*

In this section, we'll give a proof of Kuratowski's Theorem [25]. There are many proofs of this theorem; we give a fairly short proof by Makarychev [28]. First, recall from Proposition 4.1.2 that a graph G which does not contain a theta-graph is a tree of cycles and bridges. We refer to a cycle or bridge containing only one cutvertex of G as an ***endblock*** – these correspond to the leaves of the tree in the block decomposition of G – see Theorem 4.1.1. The second ingredient is the following lemma, whose slightly technical proof is left as an exercise. Let $\mathcal{S}$ denote the set of all graphs containing a subdivision of K_5 or $K_{3,3}$. ◁◁

Lemma 7.7.1 *For any graph G and $e \in E(G)$, $G/e \in \mathcal{S}$ implies $G \in \mathcal{S}$.*

The final notion is the following: if C is a cycle in a plane graph G, then $\text{int}(C)$ denotes the set of vertices of G inside C, and $\text{ext}(C)$ denotes the set of vertices of G outside C.

Proof ▷ OF KURATOWSKI'S THEOREM. If G is planar then $G \notin \mathcal{S}$. Now let $G \notin \mathcal{S}$ be a minimal non-planar graph. By case checking, we see that $|V(G)| > 6$. Furthermore, every proper subgraph of G is planar, and G/e is planar, by Lemma 7.7.1. Clearly $d(v) \geq 2$ for all $v \in V(G)$, otherwise $G - \{v\}$ is planar which implies G is planar. Also, $d(v) > 2$ for all $v \in V(G)$: otherwise with $N(v) = \{u, w\}$, the graph $G/\{u, v\}$ is planar. Since G is a subdivision of $G/\{u, v\}$ – we insert v into $\{u, w\}$ – G is also planar, a contradiction.

Part 1 *For $\{u, v\} \in E(G)$, $G_{uv} := G - \{u\} - \{v\}$ contains no theta-graph.*
Suppose $T \subseteq G_{uv}$ is a theta-graph and let $C \subset T$ be a cycle. By Proposition 7.4.1, $G/\{u, v\}$ has a plane embedding H with $u, v \in \text{int}(C)$ and $\text{ext}(C) \neq \emptyset$. Now $H - \text{int}(C) = G - \text{int}(C)$ is a plane graph in which C is a face boundary. Also $H - \text{ext}(C)$ is a plane graph with C

as the infinite face boundary, and since $G - \text{ext}(C)$ is planar, there is a plane embedding I of $G - \text{ext}(C)$ in which C is the infinite face boundary (careful : this key step is subtle). Gluing I and $H - \text{int}(C)$ along C, we get a plane embedding of G, a contradiction.

Part 2 *For $\{u, v\} \in E(G)$, G_{uv} has at most one leaf.*
Let X be a set of two leaves of G_{uv} and $Y = V(G_{uv}) \backslash X$. Notice that $e(X, Y) = 2$ and since $|V(G)| > 6$, $|Y| > 2$. Since $d(x) \geq 3$ for $x \in X$, $u, v \in N(x)$ for $x \in X$. This implies $G - Y$ contains a theta-graph, and Part 1 shows $E(G[Y]) = \emptyset$. Since $d(y) \geq 3$ for $y \in Y$, and y sends at most two edges to $\{u, v\}$, $e(X, Y) \geq |Y| > 2$, a contradiction.

Now we complete the proof. Part 1 and Proposition 4.1.2 show that G_{uv} is a tree of cycles and bridges. By Part 2, G_{uv} has at most one leaf, so some endblock $C \subseteq G_{uv}$ is a cycle. Let $P \subseteq C$ be a path of length two; if $|V(C)| > 3$ then P can be chosen to contain no cutvertex of G_{uv}. Since $|V(G)| > 6$ and G_{uv} has at most one leaf, we can find an edge $\{w, x\} \in G_{uv}$ (see Figure 7.8) vertex-disjoint from P. Now each vertex of P is adjacent to u or v, since G has minimum degree at least three. This implies $G[V(P) \cup \{u, v\}]$ contains a theta-graph, vertex-disjoint from $\{w, x\}$. This contradicts Part 1 applied to G_{wx}. □

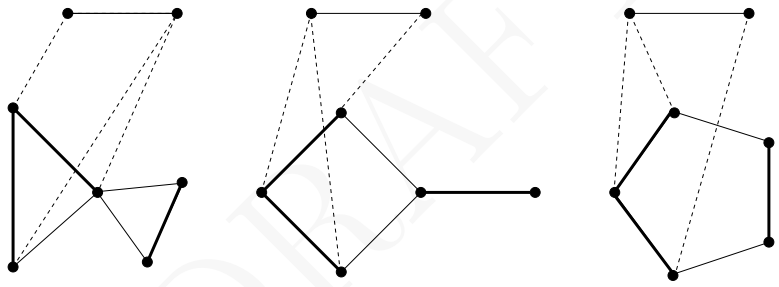


Figure 7.8: The path $P \subset C$ and edge $\{w, x\}$

7.8 Graphs on Surfaces*

In this section, we study drawing graphs on general surfaces without crossings. Rather than assume background in general topology, we will define everything in elementary terms, keeping in mind that everything can be made rigorous through topology. The surfaces we look at will all be ***orientable***: these are closed surfaces which consist of a sphere with a finite number of ***handles*** (or tubes) attached. For example, the ***torus consists*** in attaching one handle to the sphere, and the ***double torus*** consists in attaching two handles to the sphere. The number of handles attached to $\mathbb{S}$ is called the ***genus*** of $\mathbb{S}$, and denoted $\gamma(\mathbb{S})$. The ***Euler characteristic*** of $\mathbb{S}$ is $\chi(\mathbb{S})$, defined by $2 - 2\gamma(\mathbb{S})$. For example, the Euler characteristic of the sphere is two, whereas the euler characteristic of the torus is zero. Let $G = (V, E)$ be a

graph. An embedding of G without crossings on a surface $\mathbb{S}$ is a function $f : V \cup E \to \mathbb{S} \cup \mathcal{C}$, where $\mathcal{C}$ is the set of continuous curves in $\mathbb{S}$, such that f is one-to-one, $f(v)$ is a point in $\mathbb{S}$ for each $v \in V$, and $f(\{u, v\})$ is a continuous curve in $\mathbb{S}$ with ends u and v when $\{u, v\} \in E$, and none of the curves in $\mathcal{C}$ cross internally. Generally we identify G with the image of $V \cup E$ under f. Throughout this section, we assume that the connected regions (the faces) of $\mathbb{S} \backslash G$ are homeomorphic to discs. For example, an embedding of K_5 on the torus is shown below, where all faces are homeomorphic to discs. It is convenient to call a graph a ***toroidal graph*** if it can be embedded on the torus without crossings. As an exercise, find an embedding of K_4 on the torus with two faces, but such that one of the faces is not homeomorphic to an open disc. ◁◁

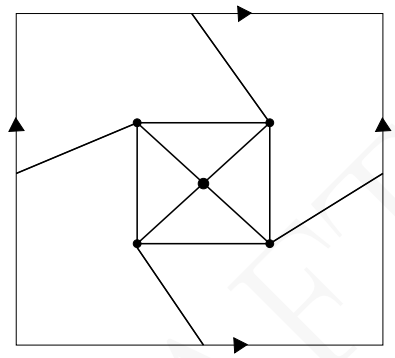


Figure 7.9: Toroidal embedding of K_5

Thus the definition of faces in a drawing of a graph on a surface $\mathbb{S}$ without crossings is the same as in plane graphs. The degree of a face is again the shortest closed walk which traverse every edge on the boundary of the face (since the faces are essentially discs these definitions are the same as for plane graphs, but we could define it even if the faces were not discs). For example, in the toroidal embedding of K_5 in Figure 7.9, there are five faces, four having degree three and one having degree four. It is a good exercise to try to embed other graphs, for example $K_{4,4}$, in the torus. A note on drawings: take a square $[0, 1] \times [0, 1]$ and identify $(0, a)$ with $(1, a)$ and identify $(b, 0)$ with $(b, 1)$ for all $a, b \in [0, 1]$. Then we obtain the torus (the reverse procedure corresponds to cutting a torus along a cross section and then cutting the resulting cylinder along its height). A drawing of a graph without crossings in the rectangle with opposite points identified then gives a toroidal drawing of the graph. Roughly speaking, an edge can pass through the top (respectively, left) of the square and emerge from the vertically opposite (respectively, horizontally opposite) point at the bottom (respectively, right). ◁◁

The genus of a graph G, denoted $\gamma(G)$, is the minimum possible value of γ such that G embeds without crossings in a surface of genus γ. So $\gamma(K_5) = 1$. The Euler characteristic of G is $\chi(G) = 2 - 2\gamma(G)$ (not to be confused with the chromatic number). The ***Euler-Poincaré Formula*** is an analog of Euler's Formula for surfaces, and it says that the number of faces in any embedding of a graph on a surface of characteristic χ does not depend on the embedding (again with the proviso that the faces behave like discs):

Theorem 7.8.1 (Euler-Poincaré Formula) *Let G be a connected graph of Euler characteristic χ embedded without crossings in a surface $\mathbb{S}$ of characteristic χ, where all faces of G are homeomorphic to discs. Then*

$$|V(G)| - |E(G)| + |F(G)| = \chi.$$

The proof of this theorem can be achieved by induction on e, similar to Euler's Formula. Actually it is convenient rather to prove that $|V(G)| - |E(G)| - |F(G)| = \chi(G) + c(G) - 1$ where $c(G)$ is the number of components of G. For instance, if G has a bridge e, then $\chi(G) = \chi(G - e)$ and $|F(G)| = |F(G - e)|$ and $c(G - e) = c(G) + 1$ so the induction step works. Now suppose G has no bridges. Then for an edge $e \in E(G)$ such that $\chi(G - e) = \chi(G)$, we have $|F(G - e)| = |F(G)| - 1$ and again the induction step works. Finally if $\chi(G - e) = \chi(G) - 1$, then e must have been the only edge on some handle of $\mathbb{S}$. Then one verifies that $|F(G-e)| = |F(G)|+1$, so the induction step works. The details are left to the reader. The formula is called the Euler-Poncaré formula since Poincaré gave a very general topological generalization of it, where edges are replaced by simplices. The first proof of the Euler-Poincaré Formula for general χ was given by Lhuillier [26].

A natural consequence of the Euler-Poincaré Formula is that $|E(G)| \leq 3|V(G)|-3\chi$ whenever G is a graph which can be embedded in a surface of characteristic χ without crossings. This shows, for example, that K_8 cannot be embedding on the torus without crossings, since $|E(K_8)| = \binom{8}{2} = 28$ whereas $3|V(K_8)| - 3\chi = 24$. Another consequence is that the degeneracy of a graph G embedded in $\mathbb{S}$ without crossings satisfies

$$d(G) \leq \left\lfloor 6 - \frac{6\chi}{|V(G)|} \right\rfloor.$$

So we can extend our result about planar graphs being 6-colorable to higher genus surfaces, via ***Heawood's Map Coloring Theorem***:

Theorem 7.8.2 (Heawood's Map Coloring Theorem)
The chromatic number of a graph embedding without crossings in a surface of characteristic $\chi \leq 0$ is at most

$$h(\chi) = \left\lfloor \frac{1}{2}(7 + \sqrt{49 - 24\chi}) \right\rfloor$$

Furthermore, if G is a minimal $h(\chi)$-chromatic graph drawn on a surface of characteristic $\chi \neq -2$, then $G = K_{h(\chi)}$.

Proof ▷ First we prove the upper bound $h(\chi) \leq \lfloor \frac{1}{2}(7 + \sqrt{49 - 24\chi}) \rfloor$. Let G be embedded on a surface $\mathbb{S}$ of characteristic χ without crossings. We may assume that G is a minimal k-chromatic graph on $\mathbb{S}$, where $k = h(\chi)$. By Proposition 6.4.1 (or by just deleting a vertex of small degree) it follows that $\delta(G) \geq k - 1$. Therefore $|E(G)| \geq \frac{k-1}{2}|V(G)|$, by the handshaking lemma. On the other hand $|E(G)| \leq 3|V(G)| - 3\chi$, by the Euler-Poincaré Formula. Therefore

$$\frac{k-1}{2}|V(G)| \leq 3|V(G)| - 3\chi$$

which is the same as

$$(k-7)|V(G)| + 6\chi \leq 0.$$

Since we assumed $\chi \leq 0$, $k \geq 7$ follows from the fact that K_7 can be embedded on the torus. Now $|V(G)| \geq k$ since $\delta(G) \geq k - 1$, so

$$k^2 - 7k + 6\chi \leq 0.$$

This gives $2k \leq 7 \pm \sqrt{49 - 24\chi}$, and we clearly must take the positive square root. This proves the upper bound on $h(\chi)$.

The second part of the proof is to show $G = K_k$. If $G \neq K_k$ then $|V(G)| \geq k + 2$, and it is an exercise to prove that if $|V(G)| = k+2$, then G consists of a pentagon disjoint from K_{k-3} together with all edges between the pentagon on the K_{k-3}. In particular, $|E(G)| = \binom{k+2}{2} - 5$. This is greater than $3(|V(G)| - \chi)$, contradicting the Euler-Poincaré Formula. Therefore $|V(G)| \geq k + 3$. Now $\delta(G) \geq k - 1 \geq 6$, and by Brook's Theorem, G is not $k - 1$ regular, otherwise it would be $k - 1$ colorable. Therefore

$$|E(G)| > \frac{|V(G)|(k-1)}{2}.$$

This gives $k^2 - 4k - 20 + 6\chi \leq 0$ and so $k \leq 2 + \sqrt{24 - 6\chi}$. This is not possible unless $\chi = -2$. □

Note that we do not include the case of plane graphs, where $h(\chi) = 4$. Surprisingly, this case also agrees with the above formula for $h(\chi)$. Also, we avoided $\chi = -2$. The full classification, for all χ and even for all surfaces (not only orientable ones) was given by Ringel and Youngs [35]. It should be noted that better results can be obtained for a graph of girth $g > 3$: in that case Euler's Formula can be used to give

$$|E(G)| \leq \frac{g}{g-2}(|V(G)| - \chi)$$

and then one can color these graphs with fewer colors, by repeating the pattern of the proof above.

We give the following quote from the book of Mohar and Thomassen [32]:

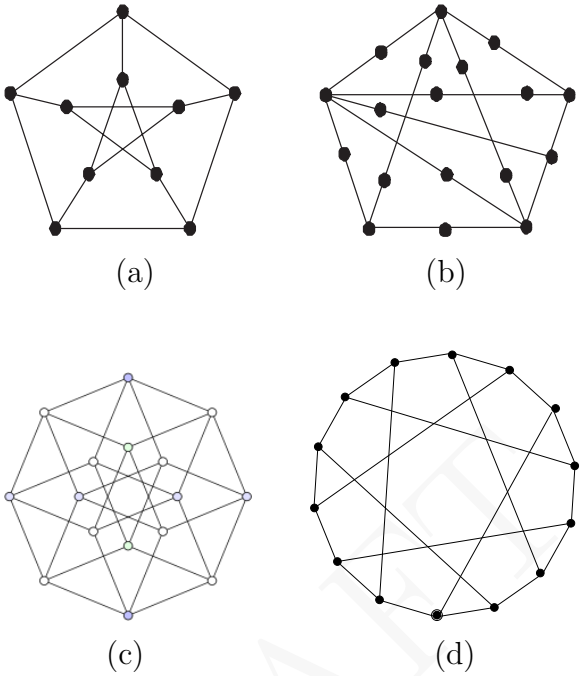
> *"Graphs on surfaces form a natural link between discrete and continuous mathematics. They enable us to understand both graphs and surfaces better. It would be difficult to prove the celebrated classification theorem for (compact) surfaces without the use of graphs. Map color problems are usually formulated and solved as problems concerning graphs."*

Which graphs can be embedded in a surface of genus γ without crossings? A consequence of one of the deepest theorems in mathematics, called the Graph Minors Theorem, due to Robertson and Seymour, is that for any γ, there exists a finite list of graphs $\mathcal{H}_\gamma$ such that any graph which does not embed on a surface of genus γ contains a graph in $\mathcal{H}_\gamma$ as a minor. Kuratowski's Theorem [25] is such a result for planar graphs, where $\mathcal{H}_0 = \{K_5, K_{3,3}\}$. Unfortunately, for surfaces of higher genus, the known list is prohibitively long, and the smallest possible list is not known. The shortest proof that there is a finite list is due to Thomassen (it is much simpler than the graph minors theorem).

A very natural question on embedding graphs is the following: in the last section, we gave embeddings of graphs on surfaces where some of the faces were a bit unorthodox: they are not homeomorphic to plane disks. While it is true that every 2-connected planar graph has an embedding where every face is homeomorphic to a disk, it is not known if every 2-connected graph can be embedded in some surface in such a way that all faces are homeomorphic to disks. This is known as the strong embeddability conjecture, and it remains open It has many other consequences in graph theory; for example, does every 2-connected graph contain a set of cycles such that every edge is in either one or two of the cycles? For planar graphs, this is clear: the boundaries of the faces will do, and then every edge is in exactly two cycles. The conjecture also has implications for the existence of nowhere zero 5-flows. For more information on graphs on surfaces, see Archdeacon or Mohar and Thomassen.

7.9 Exercises

Question 7.1° Which of the graphs in (a) – (d) below is planar? Justify your answers.



Question 7.2. Show that there is no cubic bipartite planar graph with ten vertices.

Question 7.3° Prove or disprove the existence of a non-empty plane pseudograph with

(a) all vertices having different degrees
(b) minimum degree at least two whose faces all have different degrees
(c) minimum degree at least three whose faces all have different degrees?

Question 7.4. A maximal plane graph is a plane graph $G = (V, E)$ with $n \geq 3$ vertices such that if we join any two non-adjacent vertices in G, we obtain a non-plane graph.

(a) Draw a maximal plane graph on six vertices.
(b) Show that a maximal plane graph on n points has $3n - 6$ edges and $2n - 4$ faces.
(c) A ***triangulation*** of an n-gon is a plane graph G whose vertex set is the vertex set of a convex n-gon, whose infinite face boundary is the convex n-gon, and whose other faces are all triangles. Determine $|E(G)|$.

Question 7.5. Show that every triangle-free planar graph is 4-colorable.

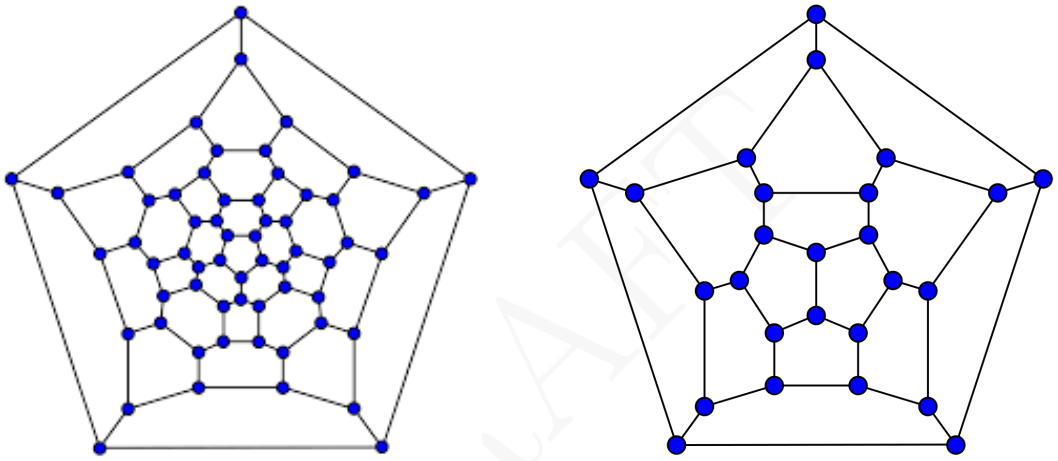
Question 7.6.

(a) Give an example of a connected cubic planar graph that is not Hamiltonian.

(b) Give an example of a cubic planar multigraph with no 3-edge-coloring.

(c)* Prove that the graphs in Figure 7.7 are not Hamiltonian.

Question 7.7. For which r does there exist a 3-regular plane graph with r faces of degree five and all other faces of degree six? Examples of such plane graphs are drawn below.



Question 7.8. Suppose a person is standing in a room which has a painting on each of its walls. Prove that if the room has at most five walls, then the person can find a place to stand so as to see all the paintings at once. Prove that if the room has six walls or more, then it is possible that the person cannot find a place to stand so as to see all the paintings at once.

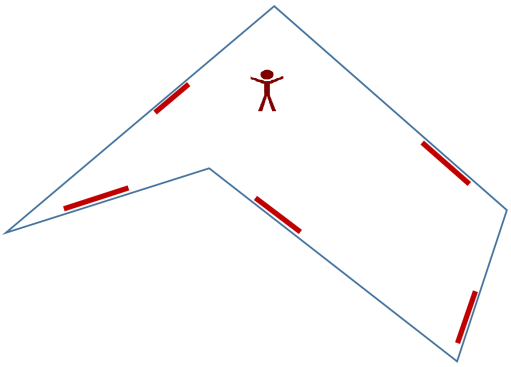


Figure 7.10: A five sided room

Question 7.9.
(a) Give an example of a 2-connected cubic planar graph that is not Hamiltonian.
(b) Give an example of a 3-connected cubic graph that is not Hamiltonian.

Question 7.10. The ***crossing number*** of a graph G is the minimum number $\nu(G)$ such that the graph can be drawn in the plane with $\nu(G)$ pairs of crossing edges. Determine $\nu(G)$ when G is the Petersen graph (see Figure 7.11).

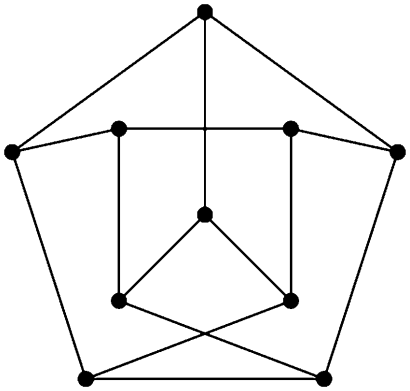


Figure 7.11: A drawing of the Petersen graph

Question 7.11. Prove that a maximal planar graph with at least three vertices is 3-colorable if and only if it is Eulerian.

Question 7.12* Let G be a planar graph with degree sequence $(d_1, d_2, \ldots, d_n)$ where $n \geq 3$.

(a) Prove that
$$\sum_{i=1}^{n}(6 - d_i) \geq 12.$$

(b) Prove that a planar graph of minimum degree five has at least twelve vertices of degree five.

Question 7.13* Let G be a planar graph with degree sequence $(d_1, d_2, \ldots, d_n)$ where $n \geq 3$. Prove that for $3 \leq m \leq n$.
$$\sum_{i=1}^{m} d_i \leq 2n + 6m - 16.$$

Question 7.14* Prove that $n/4$ guards suffice to guard a rectilinear art gallery with n sides.

Question 7.15* Prove that the Petersen graph is toroidal.

8 The Max-Flow Min-Cut Theorem

8.1 Flows

Let $\vec{G} = (V, \vec{E})$ be a digraph and $s, t \in V$. We shall refer to s as the source vertex and t as the sink vertex in what follows. Let $\mathbb{R}_{\geq 0}$ denote the non-negative real numbers. We want to define what it means for the network to have a flow from s to t. A ***static st-flow*** is a function $f : \vec{E} \to \mathbb{R}_{\geq 0}$ such that for all $x \in V \backslash \{s, t\}$:

$$\sum_{y \in N^+(x)} f(x, y) = \sum_{y \in N^-(x)} f(y, x)$$

The flow in arc (u, v) is denoted $f(u, v)$. This last requirement is known as ***Kirchoff's Law***, in words it states that the flow into a vertex must be equal to the flow out of a vertex, and that this must be true for all vertices apart from s and t. The flow in an arc $e \in \vec{E}$ is denoted $f(e)$. The ***value of a flow*** f from s to t is defined by

$$v(f) = \sum_{y \in N^+(s)} f(s, y) - \sum_{y \in N^-(s)} f(y, s).$$

This is the net amount of flow leaving the source s. An example of all these concepts is given below in Figure 8.1. The flows f through the arcs in the networks equals the number of arrows on each arc. So $f(s, v) = 2 = f(v, w) = f(u, x)$, $f(s, u) = 3$, $f(w, t) = 4$, $f(u, w) = f(x, w) = f(x, t) = 1$, $f(v, u) = 0$. The value of the flow is

$$v(f) = f(s, v) + f(s, u) = 5.$$

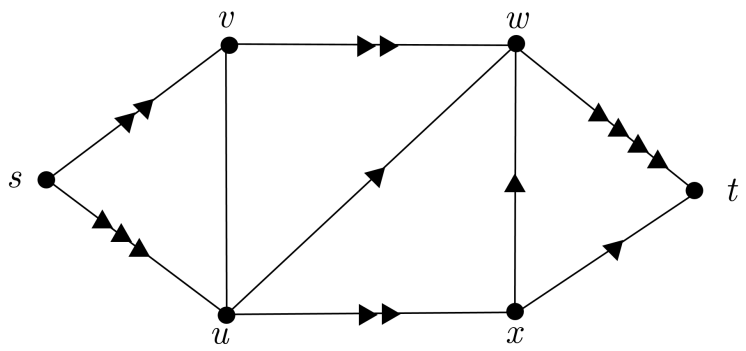


Figure 8.1: Flows

8.2 Capacities

The ***capacity*** of an arc $e \in \vec{E}$ is a non-negative real number and denoted $c(e)$, and is the maximum possible amount of flow in the arc. An example of a network with capacities is shown below. Here $c(s,u) = c(v,w) = c(x,w) = c(v,u) = 1$ and $c(s,v) = c(u,w) = 4$ and $c(w,t) = 3$ and $c(u,x) = c(x,u) = c(x,t) = 2$.

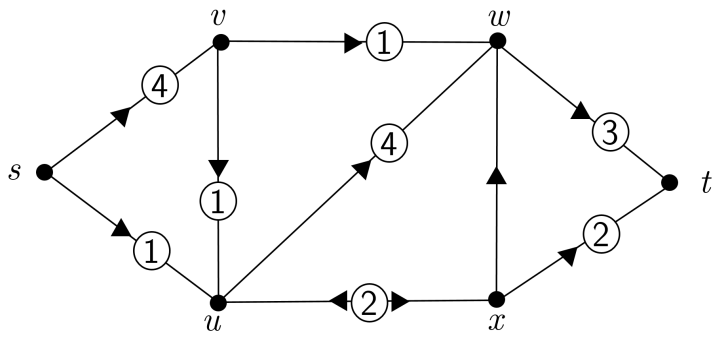


Figure 8.2: Capacities

We require that if f is a flow in the network, then $f(e) \leq c(e)$ for every arc $e \in \vec{E}$. The flow in Figure 8.1 is impossible in Figure 8.2, since $f(s,u) = 3 > c(s,u) = 1$. However a valid flow would be to define $f(s,u) = f(v,w) = f(v,u) = 1$ and $f(s,v) = f(u,w) = f(w,t) = 2$, and the value of this flow is 3. A ***maximum flow*** in a network with capacities is a flow f^* such that for every flow f, $v(f^*) \geq v(f)$ – it is a flow with largest value.

8.3 Cuts

If $S \subset V$ and $s \in S$ and $t \notin S$, then the ***cut induced by*** S is the set of arcs from S to $V \backslash S$ – this is the set of arcs leaving S. This set of arcs is denoted $(S, \overline{S})$, and is called an ***st-cut*** or simply a cut. For example, $\{(s,u),(v,u),(v,w)\}$ in Figure 8.2 is the cut induced by $S = \{s,v\}$. The ***capacity*** of a cut $(S, \overline{S})$ is defined by

$$c(S, \overline{S}) = \sum_{e \in (S, \overline{S})} c(e).$$

It is clear that if f is a flow in the network, then $v(f) \leq c(S, \overline{S})$ for any cut $(S, \overline{S})$. In Figure 8.2, we found a flow with $v(f) = 3$, and also a cut $(S, \overline{S})$ with $c(S, \overline{S}) = 3$ – namely when $S = \{s,v\}$. A cut $(S, \overline{S})$ ***minimum cut*** is a cut $(S, \overline{S})$ such that $c(T, \overline{T}) \geq c(S, \overline{S})$ for every cut $(T, \overline{T})$ – so it is a cut with a minimum value of $c(S, \overline{S})$. Our main theorem says that

the minimum capacity of a cut equals the maximum value of a flow – as already verified for the specific network in Figure 8.2. The proof will give us a way of finding a maximum flow. First we need a lemma:

Lemma 8.3.1 *For any flow f and any cut $(S, \overline{S})$,*

$$v(f) = \sum_{x \in S} \sum_{y \notin S} f(x, y) - \sum_{x \in S} \sum_{y \notin S} f(y, x).$$

Proof ▷ By Kirchoff's Law, for all $x \notin \{s, t\}$,

$$\sum_{y \in N^+(x)} f(x, y) \quad = \quad \sum_{y \in N^-(x)} f(y, x)$$

Summing over $x \in S \backslash \{s\}$ we get

$$\sum_{x \in S \backslash \{s\}} \sum_{y \in N^+(x)} f(x, y) \quad = \quad \sum_{x \in S \backslash \{s\}} \sum_{y \in N^-(x)} f(y, x).$$

All those $f(x, y)$ with $x, y \in S$ cancel out: they are counted once in the left sum, and again once in the right sum. So we get

$$\sum_{x \in S \backslash \{s\}} \sum_{y \notin S} f(x, y) \quad = \sum_{x \in S \backslash \{s\}} \sum_{y \notin S} f(y, x).$$

By definition of $v(f)$, this means

$$\sum_{x \in S} \sum_{y \notin S} f(x, y) - \sum_{x \in S} \sum_{y \notin S} f(y, x) - v(f) = 0$$

and this is the required result. □

The definition of the value of the flow is for the specific cut $(S, \overline{S})$ with $S = \{s\}$, but this lemma says we can measure the value of a flow in any network by just looking at the net flow across a cut in the network. The main theorem we prove is the following:

Theorem 8.3.2 (Max-Flow Min-Cut Theorem)
In any network with capacities, the maximum value of a flow equals the minimum value of a cut.

Proof ▷ Let f be a maximum flow.[15] Then $v(f) \leq c(S, \overline{S})$ for every cut $(S, \overline{S})$ so $v(f) \leq \min c(S, \overline{S})$. To prove the theorem, we define a set $S \subset V$ with $c(S, \overline{S}) = v(f)$. First put

[15] Why does a maximum flow even exist? Prove that a maximum flow exists.

s, the source, into S. Then for every arc (x, y) such that $x \in S$ and $c(x, y) > f(x, y)$, put $y \in S$, and for every arc (y, x) with $x \in S$ and $f(y, x) > 0$, put $y \in S$. We claim $t \notin S$ and $c(S, \overline{S}) = v(f)$. Suppose that $t \in S$. Then there exists a path $x_1 x_2 \dots x_r$ where $x_i \in S$ for all i and $x_1 = s$ and $x_r = t$ and, by definition of S,

$$c(x_i, x_{i+1}) - f(x_i, x_{i+1}) > 0 \quad \text{or} \quad f(x_{i+1}, x_i) > 0$$

for each i. Let ε be the smallest of all these positive numbers. Define a new flow g by taking $g(x_i, x_{i+1}) = f(x_i, x_{i+1}) + \varepsilon$ if $c(x_i, x_{i+1}) - f(x_i, x_{i+1}) > 0$, and taking $g(x_{i+1}, x_i) = f(x_{i+1}, x_i) - \varepsilon$ if $f(x_{i+1}, x_i) > 0$. Then $v(g) = v(f) + \varepsilon$, contradicting the maximality of f. We conclude $t \notin S$. Finally, since $f(y, x) = 0$ for every $x \in S$ and $y \notin S$, by Lemma 8.3.1 we have

$$v(f) = \sum_{x \in S} \sum_{y \notin S} f(x, y) - \sum_{x \in S} \sum_{y \notin S} f(y, x) = \sum_{(x,y) \in (S, \overline{S})} c(x, y) \quad = \quad c(S, \overline{S}).$$

This completes the proof. □

8.4 Max-Flow Min-Cut Algorithm

As we stated, the proof of the Max-Flow Min-Cut Theorem gives an algorithm for finding a maximum flow as well as a minimum cut. To construct a maximum flow f^* and a minimum cut $(S^*, \overline{S}^*)$, proceed as follows: start by letting f be the zero flow and $S = \{s\}$ where s is the source. Construct a set S as in the theorem: whenever there is an arc (x, y) such that $f(x, y) < c(x, y)$ and $x \in S$ and $y \notin S$, or an arc (y, x) such that $f(y, x) > 0$ and $x \in S$ and $y \notin S$, add y to S. If at the end of this procedure, $t \notin S$, then let $S^* = S$ to get a minimum cut and the current flow is a maximum flow. If at the end of this procedure $t \in S$, then there must be a path $x_0 x_1 x_2, \dots x_r$ where $s = x_0$ and $t = x_r$, along which f can be augmented by some value $\varepsilon > 0$. The value of ε is given in the proof above: it is

$$\varepsilon = \min\{c(x_i, x_{i+1}) - f(x_i, x_{i+1}), f(x_{i+1}, x_i) | 0 \leq i < r\}.$$

Now restart with the augmented flow which is $f(x_i, x_{i+1}) + \varepsilon$ and $c(x_i, x_{i+1}) > f(x_i, x_{i+1})$ and $f(x_{i+1}, x_i) - \varepsilon$ if $f(x_{i+1}, x_i) > 0$, for each $i : 0 \leq i < r$. Now we start again with $S = \{s\}$ and the new flow as input.

Example 24. Consider the network with capacities in Figure 8.2. According to the algorithm, start by letting f be the zero flow and $S = \{s\}$:

arc	flow	capacity
(s,u)	0	1
(s,v)	0	4
(v,w)	0	1
(x,u)	0	2
(u,x)	0	2
(u,w)	0	4
(x,t)	0	2
(w,t)	0	3

Since $f(s,v) = 0 < c(s,v) = 4$, we put v into S. Since $c(v,w) > f(v,w)$ we put w into S. Then put t into S since $c(w,t) > f(w,t)$. We stop since we have placed t in S. By the algorithm, there is a way to augment f: we consider the path $svwt$. We have the smallest difference between capacities and flows in the arcs of this path equal to 1. So we augment f to $f(s,v) = f(v,w) = f(w,t) = 1$. Now we start again with $S = \{s\}$ and the new flow.

arc	flow	capacity
(s,u)	0	1
(s,v)	1	4
(v,w)	1	1
(v,u)	0	1
(x,u)	0	2
(u,x)	0	2
(u,w)	0	4
(x,t)	0	2
(w,t)	1	3

Since $c(s,u) = 1$ and $f(s,u) = 0$, we add $u \in S$. Since $c(u,x) = 2$ and $f(u,x) = 0$, add $x \in S$. Since $c(x,t) = 2$ and $f(x,t) = 0$, add $t \in S$. So $S = \{s,u,x,t\}$ and since $t \in S$, we stop and we augment f by $\min\{1,2,2\} = 1$ along the path $suxt$ to get

arc	flow	capacity
(s,u)	1	1
(s,v)	1	4
(v,w)	1	1
(v,u)	0	1
(x,u)	0	2
(u,x)	1	2
(u,w)	0	4
(x,t)	1	2
(w,t)	1	3

Let $S = \{s\}$. Since $c(s,v) = 4$ and $f(s,v) = 1$, we can put $v \in S$. We cannot put $w \in S$ since $c(v,w) = 1 = f(v,w)$. But we can put $u \in S$ since $c(v,u) = 1$ and $f(v,u) = 0$. Then we can put $x \in S$ since $c(u,x) = 2$ and $f(u,x) = 1$. Finally we put $t \in S$ since $f(x,t) = 1$ and $c(x,t) = 2$. Since $t \in S$, we stop and augment f by $\min\{3,1,1,1\} = 1$ along the path $svuxt$ to get

arc	flow	capacity
(s,u)	1	1
(s,v)	2	4
(v,w)	1	1
(v,u)	1	1
(x,u)	0	2
(u,x)	2	2
(u,w)	0	4
(x,t)	2	2
(w,t)	1	3

Let $S = \{s\}$. Since $c(s,v) = 4$ and $f(s,v) = 2$, we put $v \in S$. But now $c(v,w) = f(v,w) = 1$, $c(v,u) = f(v,u) = 1$ and $c(s,u) = f(s,u) = 1$. So $S = \{s,v\}$ and this induces a minimum cut $(S, \overline{S})$. The flow we have just defined is a maximum flow, with value three, and notice $c(S,\overline{S}) = 3$, as expected.

8.5 Proof of Hall's Theorem

The Max-Flow Min-Cut Theorem gives an alternative proof of Hall's Theorem, as follows. We have a bipartite graph G with parts A and B satisfying Hall's Condition, $|N(X)| \geq |X|$ for all $X \subseteq A$ and all $X \subseteq B$. Orient all edges of G from A to B, add a vertex a joined to all vertices in A and a vertex b joined from all vertices in B, and assign all arcs capacity 1. Here a and b play the rôle of source and sink. Let $S \subset V(G) \cup \{a,b\}$ contain a but not b. We claim $c(S,\overline{S}) = |A|$. Let $X = A \backslash S$ and $Y = B \backslash S$. Then $|N(A\backslash S)| \geq |A| - |S \cap A|$ and $|N(B\backslash S)| \geq |B| - |S \cap B|$ by Hall's Condition. It follows that

$$c(S,\overline{S}) \geq |A| - |S \cap A| + |B| - |S \cap B| \geq |A| + |B| - |S| \geq |A|.$$

A minimum cut therefore has $S = \{a\}$, and by max-flow min-cut, there is a flow of value $|A|$. The edges of G with unit flow form a perfect matching of G. □ ⋖⋖

8.6 Proof of Menger's Theorems

The Max-Flow Min-Cut Theorem gives a short proof of Menger's Theorems. Here we give a proof of the edge-form of Menger's Theorem: let G be a graph and let s and t be distinct

vertices of G. Then the minimum size of an st-edge-cut equals the maximum number of pairwise edge-disjoint st-paths.

To apply the Max-Flow Min-Cut Theorem, we replace each edge of G with two arcs in both directions, and designate s and t as the source and sink vertices, respectively. We assign every arc capacity 1 to this digraph $\vec{G}$. A key point in the proof of the Max-Flow Min-Cut Theorem with positive integer capacities is that in a maximum flow f, the flow in each arc is an integer, and therefore if the value of f is k, then there are k paths carrying unit flow from source s to sink t. This follows from the fact that the value ε by which successive flows are augmented in the Max-Flow Min-Cut Algorithm is an integer.

To prove Menger's Theorem, the minimum capacity of an st-cut in $\vec{G}$ equals the value of a maximum st-flow in $\vec{G}$. Let the value of this flow be k. Then every st-edge-cut $L \subseteq E(G)$ in G has at least k edges, since the capacity of the st-cut $L = \{(a, b) : \{a, b\} \in L\}$ is exactly $|L|$. Since some st-cut has capacity k, a minimum st-edge-cut in G has size exactly k. Since the value of a maximum st-flow in $\vec{G}$ is k, the above remarks show there are k paths from s to t each carrying unit flow. Furthermore, no arc is shared by any two of these paths, since each arc has capacity 1. If there exists a path P and a path Q such that P uses an arc (a, b) and Q uses the arc (b, a), then we can remove (a, b) and (b, a) from P and Q to obtain two new paths P' and Q', each carrying unit flow from s to t. By repeatedly removing such pairs (a, b) and (b, a), we arrive at a set of k paths $P_1, P_2, \ldots, P_k$ in $\vec{G}$, each carrying unit flow from s to t, and such that the paths $Q_i = \{\{a, b\} : (a, b) \in P_i\}$ are edge-disjoint in G. Consequently, we have found k edge-disjoint paths $Q_1, Q_2, \ldots, Q_k$ in G, and the proof is complete. $\square$

8.7 Exercises

Question 8.1. Find a maximum st-flow in the network shown in Figure 8.3, starting with the given flow f consisting of unit flow in the st-path of length four at the top of the diagram. Also find a minimum cut in the network. The capacities of the arcs are denoted by numbers next to each arc.

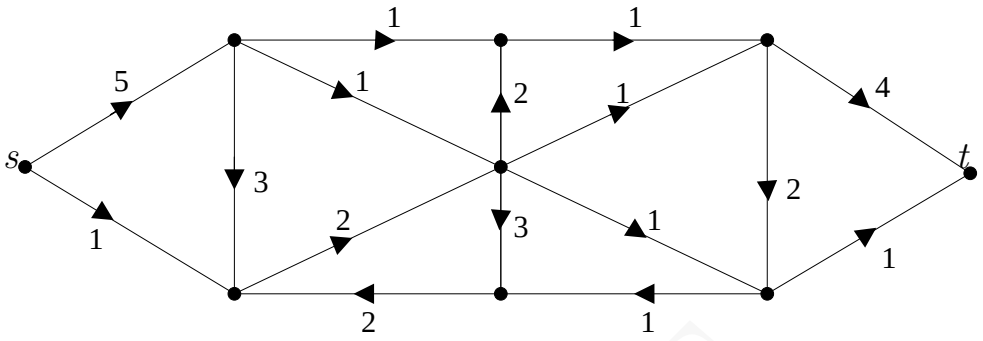


Figure 8.3: Network

Question 8.2. Find a maximum st-flow in the network shown in Figure 8.4, starting with the zero flow. Also find a minimum cut in the network. The capacities of the arcs are shown as numbers next to each arc.

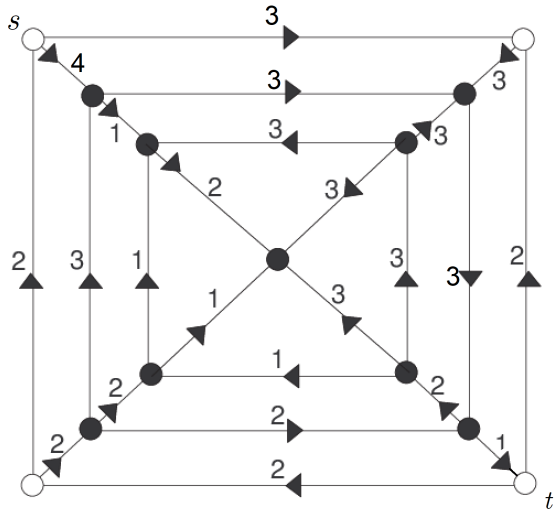


Figure 8.4: Network

Question 8.3. Find a maximum st-flow and minimum st-cut in the network below with source s and sink t by applying the Max-Flow Min-Cut Algorithm. The capacities of each arc are shown alongside the arcs as numbers below and the current flow is zero.

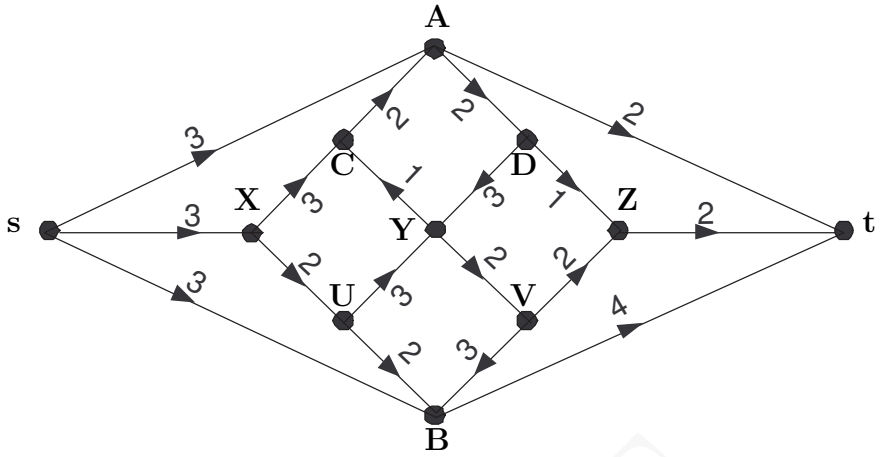


Figure 8.5: Network

Question 8.4. Find a maximum st-flow and minimum st-cut in the network shown below using the Max-Flow Min-Cut Algorithm. The current flow in the network is zero, and the capacities are shown as numbers next to the arcs.

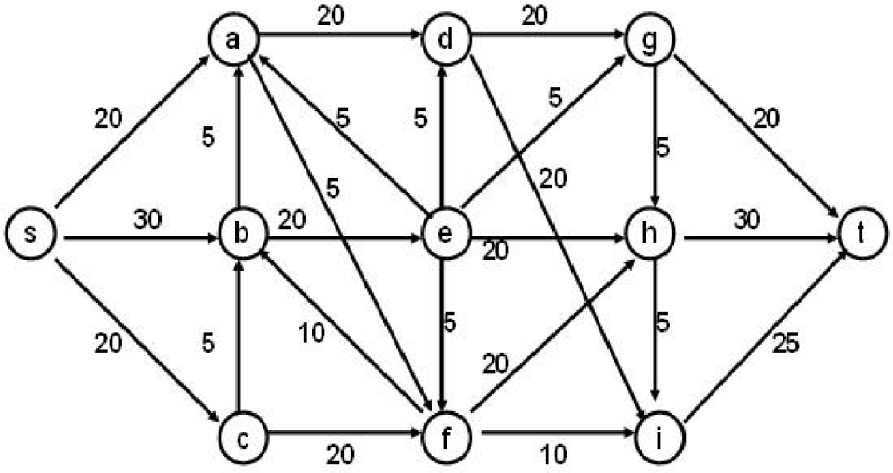
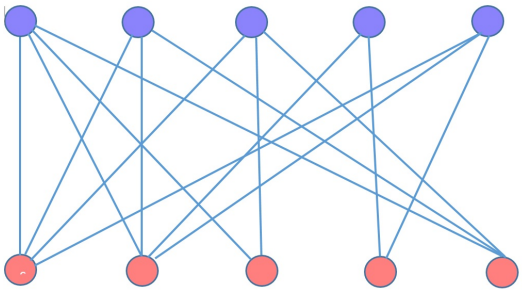


Figure 8.6: Network

Question 8.5. In a network with a set Σ of sources and a set T of sinks, explain how you would find a maximum flow and minimum cut from Σ to T.

Question 8.6. Explain how to use the Max-Flow Min-Cut Theorem to find a maximum matching in a bipartite graph $G = (A \cup B, E)$.

Question 8.7. Use Max-Flow Min-Cut to find a maximum matching in the bipartite graph below.



Question 8.8. Let G be a digraph with source s and sink t and integer capacities. Prove that there exists a maximum flow f such that $f(e)$ is an integer for each arc e of G.

9 Introduction to Extremal Graph Theory*

If F and G are graphs, then we say G is ***F-free*** if G does not contain F as a subgraph. The central problem in extremal graph theory is to determine the maximum number of edges in an n-vertex graph that does not contain F has a subgraph. Let $\mathrm{ex}(n, F)$ denote the maximum number of edges that an n-vertex F-free graph can have: these are known as the ***extremal numbers*** or ***Turán numbers*** for F. An F-free graph on n vertices with exactly $\mathrm{ex}(n, F)$ edges is called an ***extremal graph***. We begin with some basic examples.

Example 25. Let $F = K_{1,2}$. Any F-free n-vertex graph G consists of a set of isolated vertices plus a matching. Therefore $|E(G)| \leq \lfloor n/2 \rfloor$, and so $\mathrm{ex}(n, F) = \lfloor n/2 \rfloor$, and the extremal graphs G_n on n vertices are matchings of size $\lfloor n/2 \rfloor$ plus an isolated vertex if n is odd. These are shown for $n \leq 5$ in the figure below.

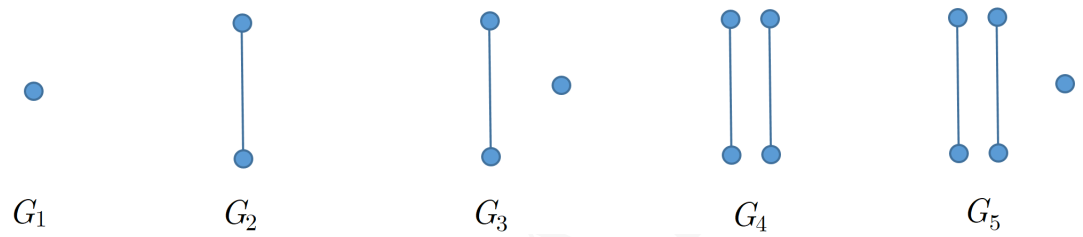


Figure 9.1: The $K_{1,2}$-free extremal graphs

If F is a graph with r vertices, then we observe $\mathrm{ex}(n, F) = \binom{n}{2}$ for $1 \leq n < r$, since the complete graph K_n does not contain F, and is the unique extremal F-free graph.

Example 26. Let us consider another example, where F is a matching of size two. As stated above, if $n \leq 3$, then $\mathrm{ex}(n, F) = \binom{n}{2}$. If $n \geq 4$, then we claim $\mathrm{ex}(n, F) = n-1$ and the unique extremal n-vertex F-free graphs G_n are isomorphic to $K_{1,n-1}$. The complete picture is therefore shown below for $1 \leq n \leq 5$:

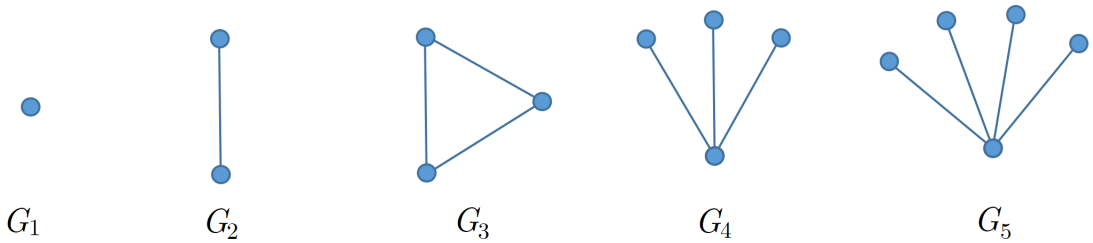


Figure 9.2: Extremal graphs with no two disjoint edges

We prove $\text{ex}(n, F) \leq n - 1$ for $n \geq 4$. If G is an n-vertex graph and $|E(G)| \geq n \geq 4$, then by Proposition 3.1.3, G is not a tree so G contains a cycle, C. If C has length at least four, then C contains F. So C has length three. Since $n \geq 4$, there is an edge e in G that is not on the cycle C. But then we may pick an edge of the cycle disjoint from e, so that e and f form a copy of F in G. Therefore $F \subset G$, and so $\text{ex}(n, F) \leq n - 1$.

9.1 Mantel's Theorem

How many edges can a graph on n vertices have if it contains no triangle? Evidently, for $1 \leq k \leq n-1$, a complete bipartite graph $K_{k,n-k}$ does not contain a triangle and has n vertices and exactly $k(n-k)$ edges. First year calculus shows the maximum value of $k(n-k)$ for $1 \leq k \leq n-1$ is $\lfloor n^2/4 \rfloor$ which occurs when the parts of the complete bipartite graph have size $k = \lfloor n/2 \rfloor$ and $n-k = \lceil n/2 \rceil$.[16] Therefore the maximum number of edges in an n-vertex graph with no triangle is at least $\lfloor n^2/4 \rfloor$. Mantel [29] showed more than one hundred years ago that in fact this is the answer: Mantel's Theorem shows $\text{ex}(n, K_3) = \lfloor n^2/4 \rfloor$ for $n \geq 2$, and the extremal K_3-free graphs are complete bipartite graphs whose parts have sizes as equal as possible.

Theorem 9.1.1 (Mantel's Theorem) *Let $n \geq 2$ and let G be an n-vertex triangle-free graph. Then $|E(G)| \leq \lfloor n^2/4 \rfloor$. Furthermore, equality holds if and only if $G = K_{k,n-k}$ where $k = \lfloor n/2 \rfloor$.*

Proof ▷ We prove by induction on n that if G is a triangle-free n-vertex graph with at least $\lfloor n^2/4 \rfloor = |E(K_k, n-k)|$ edges, then $G = K_{k,n-k}$. For $n = 2$, this is clear since $G = K_2 = K_{1,1}$. Now suppose $n > 2$. Let H be a subgraph of G with exactly $\lfloor n^2/4 \rfloor$ edges. By the handshaking lemma,

$$\lfloor n^2/4 \rfloor = |E(H)| = \frac{1}{2} \sum_{v \in V(G)} d(v)$$

and therefore $\delta(H) \leq 2\lfloor n^2/4 \rfloor / n$. If n is even, this gives $\delta(H) \leq n/2$. If n is odd, this gives $\delta(H) \leq (n-1)/2$. We conclude $\delta(H) \leq k$. Let v be a vertex of H of minimum degree. Then we check ◁◁

$$|E(H - \{v\})| = |E(H)| - \delta(H) \geq |E(K_{k,n-k})| - k = |E(K_{\ell,n-\ell})|$$

where $\ell = \lfloor (n-1)/2 \rfloor$. Note that removing a vertex of degree k from $K_{k,n-k}$ gives $K_{k,n-k-1} = K_{\ell,n-\ell}$. By induction, $H - \{v\} = K_{\ell,n-\ell}$ and $d(v) = k$. Let X and Y be the parts of $H - \{v\}$,

[16] For a real number x, $\lfloor x \rfloor$ is the largest integer that is at most x, and $\lceil x \rceil$ is the smallest integer that is at least x.

where $|X| = \ell$ and $|Y| = n - \ell$. If there exist vertices $x \in N(v) \cap X$ and $y \in N(v) \cap Y$, then $\{v, x, y\}$ is the vertex set of a triangle in G, a contradiction. Therefore $N(v) \subseteq X$ or $N(v) \subseteq Y$. Since $d(v) = k$, $N(v) = X$ or $N(v) = Y$, and this implies $H = K_{k,n-k}$. Now $H \subseteq G$, and since the addition of any edge to H gives a triangle, $G = H$. □

9.2 Turán's Theorem

Turán's Theorem generalizes Mantel's Theorem to determining $\mathrm{ex}(n, K_r)$ for all $r \geq 3$. To go about constructing a K_{r+1}-free graph on n vertices with many edges, take disjoint sets $V_1, V_2, \ldots, V_r$, where $|V_1| + |V_2| + \ldots |V_r| = n$ and join all vertices in V_i to all vertices in V_j for all $i \neq j$ and $i, j \in \{1, 2, \ldots, r\}$. This graph is called a ***complete r-partite graph*** or ***Turán graph***. For $r = 2$ it is a complete bipartite graph as in Mantel's Theorem. Note that a complete r-partite graph cannot possibly contain K_{r+1}, since $\chi(K_{r+1}) = r + 1$. The number of edges an r-partite graph is

$$\sum_{i \neq j} |V_i||V_j|.$$

Since $|V_1| + |V_2| + \cdots + |V_r| = n$, this expression is maximized when all the V_is are as equal in size as possible, so $|V_i| = \lfloor n/r \rfloor$ or $|V_i| = \lceil n/r \rceil$ for all $i \in \{1, 2, \ldots, r\}$.

The ***Turán graph***, denoted $T_r(n)$, is the unique r-partite graph all of whose parts have sizes as equal as possible. For $r = 2$, this corresponds to a complete bipartite graph with parts of size $\lfloor n/2 \rfloor$ and $\lceil n/2 \rceil$. So the number of edges in $T_2(n)$ is exactly $\lfloor n^2/4 \rfloor$. In general, we let $t_r(n)$ denote the number of edges in $T_r(n)$ – it is not a very nice number to determine, but it is roughly $(1 - \frac{1}{r})\binom{n}{2}$. Turán's Theorem [40] states that $\mathrm{ex}(n, K_{r+1}) = t_r(n)$ when $n \geq r$. ⋈ But it says even more: the only graph with $t_r(n)$ edges and no K_{r+1} is $T_r(n)$ – so $T_r(n)$ is the unique extremal graph. The inductive proof we give is fairly subtle. It relies on two facts: first that we can't add any edges to $T_r(n)$ without creating a K_{r+1}, and second, the degrees of $T_r(n)$ are as close together as possible amongst all n-vertex graphs with $t_r(n)$ edges.

Theorem 9.2.1 (Turán's Theorem) *Let $n \geq 1$ and let G be an n-vertex graph containing no K_{r+1}. Then $|E(G)| \leq t_r(n)$, with equality if and only if $G = T_r(n)$.*

Proof ▷ We prove the theorem by induction on n, the number of vertices in G, starting with $n = r$. The statement we prove is that if G is an n-vertex graph with no K_{r+1} and $|E(G)| \geq t_r(n)$, then $G = T_r(n)$. This proves the theorem: if $|E(G)| > t_r(n)$ then delete edges until $t_r(n)$ edges remain, but then the graph is $T_r(n)$, and we can't add any edges to $T_r(n)$ without creating a K_{r+1}. If $n = r$, then $t_r(n) = t_r(r) = \binom{r}{r}$, and clearly $|E(G)| \geq \binom{r}{2}$ implies $G = K_r = T_r(r)$, as required. Now suppose $n > r$, and let G be a graph on n vertices containing no K_{r+1} and with at least $t_r(n)$ edges. Delete edges from G until $|E(G)| = t_r(n)$.

Now $T_r(n)$ is a graph with $t_r(n)$ edges with the largest minimum degree amongst all graphs with $t_r(n)$ edges. Therefore $\delta(G) \leq \delta(T_r(n))$. Now every vertex of $T_r(n)$ has degree $n-\lfloor n/r \rfloor$ or $n - \lceil n/r \rceil$, so $\delta(T_r(n)) = n - \lceil n/r \rceil$. Furthermore, if x has degree $\delta(T_r(n))$ in $T_r(n)$, then $T_r(n) - \{x\} = T_r(n-1)$. Therefore $t_r(n) - \delta(T_r(n)) = t_r(n-1)$. This shows that if v is a vertex of smallest degree in G, then

$$|E(G - \{v\})| = |E(G)| - \delta(G) \geq t_r(n) - \delta(T_r(n)) = t_r(n-1).$$

By induction, $G-\{v\} = T_r(n-1)$. This means that v has degree exactly $\delta(T_r(n)) = n-\lceil n/r \rceil$. Now since G has no K_{r+1}, v is joined to vertices in $r-1$ parts of $T_r(n-1)$. But since v has degree exactly $n - \lceil n/r \rceil$, v must be joined to the $r-1$ smallest parts of $T_r(n-1)$, which means that $G = T_r(n)$, as required. □

9.3 Kövari-Sós-Turán Theorem

We consider the extremal function for $F = K_{r,s}$, the complete bipartite graph with parts of sizes r and s. Before we prove the main theorem, we state a special case of a real number inequality, called ***Jensen's Inequality***. It is based on the fact that the function $f(x) = \binom{x}{r} = x(x-1)\dots(x-r+1)/r!$ for $x \geq r-1$ and $f(x) = 0$ for $x < r-1$ is convex on $\mathbb{R}$, and therefore $nf(a) \geq f(a_1) + f(a_2) + \cdots + f(a_n)$ where $a = (a_1 + a_2 + \cdots + a_n)/n$ and the a_i are real numbers.

Lemma 9.3.1 *Let $a_1, a_2, \dots, a_n$ and r be positive integers and let $a = \frac{1}{n}\sum_{i=1}^{n} a_i$. If $a \geq r-1$ then*

$$\sum_{i=1}^{n} \binom{a_i}{r} \leq n\binom{a/n}{r}.$$

The following was proved by Kövari, Sós and Turán [23]:

Theorem 9.3.2 (Kövari-Sós-Turán Theorem) *Let r, s be positive integers, and suppose $r \leq s$. Then*

$$\mathrm{ex}(n, K_{r,s}) \leq (\tfrac{s-1}{2})^{1/r} n^{2-1/r} + \tfrac{1}{2}(r-1)n.$$

Proof ▷ Let G be an n-vertex graph not containing $K_{r,s}$. If $|E(G)| \leq (r-1)n/2$, then we are done, so we assume $|E(G)| \geq (r-1)n/2$. The number of sets of r vertices of G is exactly $\binom{n}{r}$. Suppose the vertices are the elements of $\{1, 2, \dots, n\}$ and the degree of vertices i is a_i. Then there are exactly $\binom{a_i}{r}$ sets of size r in the neighborhood of i. So the total number of sets of size r which are in the neighborhood of some vertex is

$$\sum_{i=1}^{n} \binom{a_i}{r}.$$

Note that we might have counted some sets of size r more than once in this sum. But what we do know is that no set of size r could have been counted at least s times, otherwise that set of size r would be in the neighborhood of s vertices in $V(G)$, and that would give a $K_{r,s}$ in G. So

$$\sum_{i=1}^{n} \binom{a_i}{r} \leq (s-1)\binom{n}{r}.$$

Applying Lemma 9.3.1, and noting $a = 2|E(G)|/n \geq r-1$ via the handshaking lemma,

$$n\binom{a}{r} \leq (s-1)\binom{n}{r}.$$

We now use the fact that for $x \geq r$,

$$\frac{(x-r)^r}{r!} \leq \binom{x}{r} = x(x-1)\dots(x-r+1)/r! \leq \frac{(x-r+1)^r}{r!}.$$

It follows that

$$n\frac{(a-r+1)^r}{r!} \leq (s-1)\frac{n^r}{r!}.$$

This gives $(an-(r-1)n)^r \leq (s-1)n^{2r-1}$ and therefore since $an = 2|E(G)|$,

$$|E(G)| \leq (\tfrac{s-1}{2})^{1/r} n^{2-1/r} + \tfrac{1}{2}(r-1)n.$$

This proves the theorem. □

9.4 The Erdős-Gallai Theorem*

The famous ***Erdős-Sós Conjecture*** [14] states that if T is any tree with k edges and G is any graph not containing T, then G has average degree at most $k-1$:

Conjecture 9.4.1 (Erdős-Sós Conjecture)
If T is any tree with k edges, then $\mathrm{ex}(n,T) \leq (k-1)n/2$.

A clique with k vertices does not contain any tree T with k edges, and therefore according to the conjecture a graph whose components are all cliques with k vertices is an extremal T-free graph, and $\mathrm{ex}(n,T) = (k-1)n/2$ when $k|n$. A proof of this conjecture was claimed by Ajtai, Komlós, Simonovits and Szemerédi [1]. The special case where the tree is a path is the ***Erdős-Gallai Theorem***, which we study next. Erdős and Gallai [15] proved the following theorem, which verifies the Erdős-Sós Conjecture for paths:

Theorem 9.4.2 (Erdős-Gallai Theorem) *Let $k \geq 1$ and let G be an n-vertex P_k-free*

graph. Then $e(G) \leq (k-1)n/2$, with equality if and only if $k|n$ and every component of G is K_k.

Proof ▷ Let G be an n-vertex graph with at least $(k-1)n/2$ edges. We prove by induction on $n+k$ that G contains P_k unless every component of G is K_k. If G is disconnected, then some component of G contains a path of length k or equals K_k, by induction. If some component is K_k, we remove it and get a graph with $n-k$ vertices and $(k-1)(n-k)/2$ edges. By induction, that graph is a union of K_k or contains a path of length k, and we are done. Therefore we may assume G is connected. The theorem is clear for $k=1$, since a single edge forms a path P_1, and if there are no edges then every component of G is K_1. Suppose $k \geq 2$. If G contains a vertex v of degree less than $k/2$, then $G - \{v\}$ has at least $(k-1)(n-1)/2$ edges. By induction, $G - \{v\}$ contains a path of length k, unless $k|(n-1)$ and $G - \{v\}$ is a union of cliques K_k. Since G is connected, v has a neighbor in one of these cliques, and this gives a path of length k ending with v. So we may assume every vertex of G has degree at least $k/2$. By induction, G contains a path P of length $k-1$. The ends u and w of that P have all their neighbors on the path. As in Dirac's Theorem, there exists a neighbor of u that comes after a neighbor of w on the path from u to w, and this gives a cycle C of length k containing P. If there is a vertex x not in C, then since G is connected, there is a path from x to C, and in particular by adding an edge $\{w, x\}$ with $w \in V(C)$, we get a path of length k ending with x. So $V(C) = V(G)$, and so $|V(G)| = k$ and $G = K_k$, as required. □

9.5 Exercises

Question 9.1° Prove that an n-vertex graph G with at least $3n/2$ edges contains a cycle of length at least four.

Question 9.2. Let $k \geq 1$. Prove that an n-vertex bipartite graph containing no matching of size k has at most $(k-1)(n-k+1)$ edges for $n \geq 2k$.

Question 9.3. A ***bowtie*** is a graph B consisting of two triangles sharing exactly one vertex. Determine $\mathrm{ex}(n, B)$ for all $n \geq 1$.

Question 9.4. Let G be a bipartite graph with parts of sizes m and n, not containing a 4-cycle. Prove that

$$|E(G)| \leq m\sqrt{n} + n.$$

Question 9.5. Let G be an n-vertex graph not containing a 4-cycle. Prove that

$$|E(G)| \leq \frac{n}{2}(1 + \sqrt{4n-3}).$$

Question 9.6. Let G be an n-vertex d-regular graph. Prove that the number of triangles in G is at least

$$\frac{1}{3}d^2 n - \frac{1}{6}dn^2.$$

Question 9.7. Let r be a positive integer, and let $f : \mathbb{R} \to [0, \infty)$ be defined by $f(x) = 0$ if $x < r-1$ and $f(x) = x(x-1)\dots(x-r+1)$ if $x \geq r-1$. Prove that f is convex on $\mathbb{R}$ and then show if $a = \frac{1}{n}(a_1 + a_2 + \cdots + a_n) \geq r-1$ for positive integers $a_1, a_2, \dots, a_n$, then

$$\frac{1}{n}\sum_{i=1}^{n} \binom{a_i}{r} \geq \binom{a}{r}.$$

Question 9.8. Let q be an odd prime number, and let G be the pseudograph with vertex set $\mathbb{Z}_q \times \mathbb{Z}_q$ such that (x_1, x_2) is adjacent to (y_1, y_2) whenever

$$x_2 + y_2 = x_1 y_1.$$

Prove that G does not contain any quadrilaterals. Suppose the equation $x^2 = 2$ has no solution. Determine the number of edges of G after loops are removed.

Question 9.9* Draw $n+1$ line segments between a set of n points in the plane. Prove that two of the line segments do not intersect. Is this best possible?

Question 9.10* A ***pentagon*** is a cycle C of length five. Prove that the extremal C-free graphs are either complete bipartite graphs with $\lfloor n^2/4 \rfloor$ edges or $n \leq 7$ and the extremal C-free graphs consist of a clique of size $\min\{4, n\}$ and a clique of size $n - \min\{4, n\} + 1$ sharing exactly one vertex.

Question 9.11* Let G be an n-vertex d-regular graph. Prove that the number of cycles of length four in G is at least

$$\frac{1}{4}n(n-1)\binom{\frac{d(d-1)}{n-1}}{2}.$$

Question 9.12* Let $n \geq r \geq 1$ and let G be an n-vertex graph with $\frac{1}{r}\binom{n}{2}$ edges. Prove that G contains a subgraph with $m \geq \sqrt{n/r}$ vertices and minimum degree at least $\frac{1}{r}(m-1)$.

A Appendix*

A.1 Sets and sequences

Sets. A set is an unordered collection of distinct objects. The objects are called ***elements*** of the set. We use braces to denote a set, for example, the set with elements $1, 2$ and 3 is denoted $\{1, 2, 3\}$. Since the elements are not ordered, we can rearrange the elements in the representation to get the same set, so $\{1, 2, 3\}$ and $\{3, 2, 1\}$ are the same set. The set with no elements is denoted $\{\}$ or $\emptyset$, and is called the ***empty set***. A set A is a ***subset*** of a set B, denoted $A \subset B$, if every element of A is also an element of B. We write $a \in A$ to denote that a is an element of set A. If A is a set with finitely many elements, we write $|A|$ for the number of elements of the set A. Some standard infinite sets include $\mathbb{Z}$, the set of integers, $\mathbb{Z}_{\geq 0}$ the set of non-negative integers, and $\mathbb{R}$, the set of real numbers. Recall that if A and B are sets, then $A \cap B = \{a : a \in A \text{ and } a \in B\}$, and $A \cup B = \{a : a \in A \text{ or } a \in B\}$. These are the ***intersection*** and ***union*** of the sets A and B respectively. Two sets A and B are ***disjoint*** if $A \cap B = \emptyset$. Sets $A_i : i \in S$ are ***pairwise disjoint*** if $A_i \cap A_j = \emptyset$ for all $i, j \in S$ with $i \neq j$. We write $\bigcup_{i \in S} A_i$ to denote the union of all sets A_i such that $i \in S$. For sets A and B, $A \times B = \{(a, b) : a \in A, b \in B\}$ is the ***Cartesian product*** of A and B.

Sequences. A sequence is an ordered collection of (not necessarily distinct) objects. The objects are called entries of the sequence. We use brackets to denote a sequence, for example $(1, 1, 2)$ denotes the sequence with entries $1, 1$ and 2. Since the entries are ordered, we can rearrange the elements in the representation to get a new sequence, so $(1, 1, 2)$ and $(1, 2, 1)$ are different sequences. When the entries are required to be distinct, the sequence is called a ***permutation*** of the set of its entries. For example, $(1, 2, 3)$ and $(2, 3, 1)$ are permutations of $\{1, 2, 3\}$. If a and b are sequences, then a is a subsequence of b if we can delete entries of b to get a. For example, $(1, 2, 3, 4)$ is a subsequence of $(1, 1, 2, 1, 3, 1, 4)$ obtained by deleting 1s. The ***length*** of a sequence with finitely many entries is the number of entries in the sequence. Here is some notation involving products and sums of elements of sets and sequences: when we want to sum up the values of a function $f(i)$ for $i \in S$, where S is a set or a sequence, we write $\sum_{i \in S} f(i)$. The symbol we use for products is Π, so the product of $f(i)$ over $i \in S$ is denoted $\prod_{i \in S} f(i)$. We will be making extensive use of this notation.

A.2 Counting sets and sequences

Basic combinatorial questions involve counting sequences of finite length and sets of finite size. The following theorem tells us the total number of subsets of an n-element set:

Theorem A.2.1 *The number of subsets of an n-element set is 2^n.*

The next natural question is how many sequences of length n can be formed from a k-element set? For example, from the set $\{a, b\}$, we can form the sequences $(a, a), (a, b), (b, a)$ and (b, b) of length two. The answer is as follows:

Theorem A.2.2 *The number of sequences of length n from a k-element set is k^n.*

It should already be plain why this theorem is true: there are k choices for each entry of the sequence, and so k^n choices to fill up the sequence. By this logic, counting permutations is just as easy:

Theorem A.2.3 *The number of permutations of a set of size n is $n! := n(n-1)(n-2)\dots 1$. There are $(n)_k := n(n-1)\dots(n-k+1)$ sequences of k distinct elements in a set of size n.*

The notation $n!$ is read n ***factorial***, and denotes the product of all integers from 1 to n. Again, there are n choices for the first entry of a permutation, but then only $n-1$ for the next, $n-2$ for the next, and so on until the last entry, since all the entries are distinct. The last thing to count is the number of subsets of size k in an n-element set. The answer in general is given by the following theorem

Theorem A.2.4 *The number of sets of size k in an n-element set is*

$$\binom{n}{k} := \frac{n(n-1)(n-2)\dots(n-k+1)}{k!} = \frac{(n)_k}{k!} = \frac{n!}{k!(n-k)!}.$$

The numbers $\binom{n}{k}$ defined in this theorem are called ***binomial coefficients***, for reasons which we shall see shortly.

A.3 Multiplication and summation principles

All of the basic theorems in the last section have the same organizing principle, known as the ***multiplication principle***. Informally, the multiplication principle says that if we want to know how many sequences $(x_1, x_2, \dots, x_k)$ there are given that the number of choices for x_i is known, all we have to do is multiply together the number of choices (or decisions) for each x_i when $x_1, x_2, \dots, x_{i-1}$ have already been chosen.

Principle A (The Multiplication Principle) *The number of sequences $(x_1, x_2, \dots, x_k)$ with a_i choices for x_i after having chosen $x_1, x_2, \dots, x_{i-1}$ for each $i = 1, 2, \dots, n$ is exactly $a_1 a_2 \dots a_n$.*

The proofs of Theorems A.2.1, A.2.2 and A.2.3 come from this principle. Our argument for proving Theorem A.2.1 uses the multiplication principle with two choices for each x_i, namely

$x_i \in \{0,1\}$ for all i, in which case the number of choices is 2^n. The assignment $x_i = 1$ means i is placed in the set, and $x_i = 0$ means i is not placed in the set. So we have represented sets by binary sequences and there are 2^n binary sequences.

A second principle we use often is to break down a counting problem into a number of disjoint parts which are easier to deal with. We will refer to this as the summation principle:

Principle B (The Summation Principle) *Let $A_1, A_2, \ldots, A_n$ be pairwise disjoint finite sets. Then*

$$\left|\bigcup_{i=1}^{n} A_i\right| = \sum_{i=1}^{n} |A_i|.$$

A.4 Inclusion-exclusion principle

A basic course in mathematics confirms $|A \cup B| = |A| + |B| - |A \cap B|$. This is a special instance of the ***inclusion-exclusion formula***, or ***combinatorial sieve***. Let $[n] = \{1, 2, \ldots, n\}$.

Principle C (Inclusion-Exclusion) *Let $A_1, A_2, \ldots, A_n$ be sets of finite size. Then*

$$\left|\bigcup_{i=1}^{n} A_i\right| \quad = \quad \sum_{\emptyset \neq S \subseteq [n]} (-1)^{|S|+1} \left|\bigcap_{i \in S} A_i\right|.$$

Note that the sum is over all non-empty subsets of $[n]$, and when $n = 2$ it reduces to $|A_1 \cup A_2| = |A_1| + |A_2| - |A_1 \cap A_2|$. When the sets A_i are pairwise disjoint (they share no elements – meaning $A_i \cap A_j = \emptyset$ for all i, j), we get the summation principle.

A.5 Bijections and combinatorial proofs

Let A and B be sets. A function $f : A \to B$ is called an ***injection*** (or ***one-to-one***) if whenever $x, y \in A$ are distinct, then $f(x) \neq f(y)$. The function f is a ***surjection*** (or ***onto*** B) if for every $b \in B$ there exists $x \in A$ such that $f(x) = b$. Finally, $f : A \to B$ is a ***bijection*** if f is an injection and a surjection. How would we check, given two sets A and B of finite size, that $|A| = |B|$? For each element $a \in A$, we would associate in some way an element $b \in B$, so that no other element of A is associated with b. In other words, we would element-by-element find a matching or pairing of the elements of A with the elements of B. This is exactly a bijection $f : A \to B$.

Combinatorial identities are often proved by interpreting each side combinatorially, and then establishing a bijection between the two objects. For instance, the number of subsets of an

n-element set is 2^n, so by the summation principle and Theorem A.2.4,

$$\sum_{k=0}^{n} \binom{n}{k} = 2^n.$$

Combinatorial identities can generally be hard to prove by finding bijections, since it often hard to see what each side of the identity actually represents. Another general example is the ***binomial theorem***:

Theorem A.5.1 *Let a be a rational number and $x \in (-1, 1)$ a real number. Then*

$$(1+x)^a = \sum_{k=0}^{\infty} \binom{a}{k} x^k.$$

A.6 Mathematical induction

Let $P(n)$ denote a logical statement for each positive integer n. Thus for each integer n, we can determine whether $P(n)$ is true or $P(n)$ is false. For example, $P(n)$ might be the statement that there is a prime larger than n, and so on. In the most basic form, the principle of mathematical induction can be stated as follows:

Principle D (Mathematical Induction) *Let $P(n)$ be a statement for each positive integer n, and suppose that $P(1)$ is true, and $P(n) \to P(n+1)$ for each positive integer n. Then $P(n)$ is true for every positive integer.*

There are two steps in any induction. First one establishes the base case (in the terms above, one proves $P(0)$). Then, under the assumption that $P(n)$ is true, one attempts to prove that $P(n+1)$ must also be true (it is very important to get the order correct here – we are to show $P(n) \to P(n+1)$ and not $P(n+1) \to P(n)$).

In some instances, the following stronger form of induction is necessary:

Principle E (Strong Induction) *Let $P(n)$ be a statement for each positive integer n, and suppose that $P(1)$ is true, and $P(n) \wedge P(n-1) \wedge \cdots \wedge P(1) \to P(n+1)$ for each positive integer n. Then $P(n)$ is true for every positive integer.*

A.7 The pigeonhole principle

One of the most fundamental principles in combinatorics is the ***pigeonhole principle***. It is entirely straightforward:

Principle F (Pigeonhole Principle) *Let A and B be finite sets and $f : A \to B$ an injection. Then $|B| \geq |A|$.*

If we have a set of $n+1$ objects of n different types, then the pigeonhole principle says there exist two objects of the same type. We simply note that if A is the set of objects and $f(a)$ is the type of object $a \in A$, then $f : A \to B$ where B is the set of types and A is the set of objects. Since $|A| = n + 1$ and $|B| = n$, f is not an injection, which means $f(a) = f(a')$ for some $a, a' \in A$. It is often in this form that the pigeonhole principle is used.

Notation

$(S, \overline{S})$ 119
$A \cap B$ 136
$A \cup B$ 136
$A \subset B$ 136
$A \times B$ 136
C_k 14
$F(G)$ 98
$G - L$ 17
$G - X$ 17
G/X 18
$G[X]$ 18
K_n 13
$K_{r,s}$ 13
$N^+(v)$ 16
$N^-(v)$ 16
$N_G(v)$ 14
$N_i(v)$ 42
P_k 14
Q_n 16
$T_r(n)$ 130
$[n]$ 138
$\alpha'(G)$ 70
$\alpha(G)$ 70
$\beta'(G)$ 70
$\beta(G)$ 70
$\chi'(G)$ 88
$\chi(G)$ 88
$\delta(G)$ 14
$\ell(u, v)$ 66
$\emptyset$ 136
$\mathrm{ex}(n, F)$ 128
$\gamma(\mathbb{S})$ 109
$\kappa(G)$ 65
$\lambda(G)$ 65
$\lceil x \rceil$ 129
$\lfloor x \rfloor$ 129
$\mathbb{R}$ 136
$\mathbb{Z}$ 136
$\mathbb{Z}_{\geq 0}$ 136
$\mathrm{x}(G, A)$ 75
$\nu(G)$ 116
∂F 98
$\triangle(G)$ 14
$a \in A$ 136
$c(S, \overline{S})$ 119
$c(e)$ 119
$d_G(u, v)$ 42
$d_G(v)$ 14
$e(G)$ 4
$e(U, V)$ 17
$f(u, v)$ 118
$k(u, v)$ 65
$\deg(F)$ 98
$\mathrm{odd}(G - S)$ 75
$\binom{n}{k}$ 137
$|A|$ 136

Index

AB-path, 59, 60, 63
AB-separator, 59
F-free, 128
M-alternating, 79
M-augmenting, 79
$Y\triangle$-transform of G, 39
$Y\triangle$-transform of G at v, 39
d-degenerate, 92
k-chromatic, 88
k-colorable, 88
k-connected, 59, 61
k-critical graphs, 95
k-cycle, 14
k-edge colorable, 88
k-edge-chromatic, 88
k-edge-connected, 62
k-path, 14
n factorial, 137
n-cube, 16
r-regular, 14
st-cut, 119
t-error-correcting code, 12
uv-path, 23
uv-walk, 23
xA-fan, 63
1-factor, 70
1-factorization, 72
1-tough, 38
4-color theorem, 8

acyclic, 40
adjacent, 4
alternating path, 79
arcs, 16
art gallery problem, 105
augmenting path, 79

Bellman-Ford Algorithm, 48
bijection, 138
binomial coefficients, 137
binomial theorem, 139
bipartite graph, 13
bipartition of G, 13
block, 53
boundary, 98
boundary walk, 98
bowtie, 134
breadth-first search algorithm, 81
breadth-first search tree T, 42
breadth-first search tree rooted at v, 43
bridge, 34, 40
Bridge of Königsberg Problem, 33
bridges of Königsberg problem, 25
Brooks' Theorem, 88

capacity, 10, 119
Cartesian product, 5, 136
cellular automata, 12
center, 52
chromatic number, 9, 88
class 1, 90
class 2, 90
clique, 13
closed walk, 23
closure, 29, 37
code, 12
codewords, 12
combinatorial dual, 106
combinatorial sieve, 138
complement, 95
complete r-partite graph, 130
complete bipartite graph, 13
complete graph, 13

components, 24
connected, 8, 24, 40
connectivity, 11
contractible, 68
contractible edge, 59
contraction, 18, 59, 101
contraction of X, 18
Conway's game of life, 11
cost function, 45
critically k-connected, 69
critically k-edge-connected, 69
crossing number, 116
cube graph, 73
cubic, 15
cubic graph, 78
cubic multigraph, 39
cut induced by S, 119
cut vertex, 53
cutting-plane algorithms, 30

de Bruijn digraph, 27
de Bruijn sequence, 26
de Bruijn sequences, 26
decentralized search, 12
degree, 14, 98
degree sequence, 14
depth-first search, 45
depth-first search tree, 45
diameter, 8, 42
digraph, 4, 16
Dijkstra's Algorithm, 30
Dijkstra's Shortest Path Algorithm, 12, 48
directed cycle, 20, 52
directed Euler tour, 26
directed path, 97
disconnected, 24
disjoint, 136
distance, 42
dodecahedron graph, 51
double torus, 109

ear-decomposition, 55
edge cover, 70
edge cut, 62
edge set, 4
edge-chromatic number of G, 88
edge-connectivity, 65
edges, 4
Edmonds' Matching Algorithm, 83
elements, 136
embedding, 98
empty graph, 13
empty set, 136
endblock, 108
ends, 23
equivalence relations, 56
Erdős-Gallai Theorem, 132
Erdős-Rényi model, 10
Erdős-Sós Conjecture, 132
Euclidean minimum spanning tree, 46
Euler characteristic, 109
Euler tour, 24
Euler trail, 24, 35
Euler-Poincaré Formula, 111
Eulerian, 24
Eulerian digraph, 26
Eulerian tour, 24
Eulerian trail, 24, 35
expander graphs, 12
exposed, 75
extremal graph, 128
extremal numbers, 128

faces, 98
Fleury's Algorithm, 34
Floyd-Warshall Algorithm, 48
forest, 40
from v, 16

Gale-Shapley Algorithm, 83
genus, 109
Grötsch graph, 88, 94
graph, 4
grid graph, 20

Hadwiger-Nelson problem, 9
Hall's Theorem, 9
Hamilton cycle, 28
Hamilton path, 28
Hamiltonian, 28, 37
handles, 109
handshaking lemma, 15
heaps, 46
Heawood graph, 35
Heawood's Map Coloring Theorem, 111
height, 43
Hierholzer's Algorithm, 25
Hopcroft-Karp Algorithm, 80
Hungarian Algorithm, 83

in-degree, 17
in-neighborhood, 16
incident, 4
inclusion-exclusion formula, 138
independent set, 70
induced, 18
induced subgraph, 18
induced subgraphs, 7
infinite face, 98
injection, 138
integer linear programming, 30
internally disjoint, 55, 61
intersection, 136
isolated vertex, 14
isolated vertices, 17

Jensen's Inequality, 131

König's Theorem, 88
König-Ore Formula, 78
Kirchoff's Law, 118
Kneser graph, 21
Kruskal's Algorithm, 46
Kuhn-Munkres Algorithm, 83

latin rectangle, 74
latin square, 73
latin square of order n, 73
layered graph, 81
layers, 43
length, 14, 23, 136
line graph, 21, 34, 62
linear forest, 80
loops, 4, 16

Möbius ladder, 38
matching, 70
matchings, 30
Max-Flow Min-Cut Theorem, 10
maximal planar, 100
maximal plane, 100
maximum degree, 14
maximum flow, 119
maximum matching, 70
metric space, 52
minimum cost spanning tree, 45
minimum cut, 119
minimum degree, 14
minimum distance, 12
minimum priority queues, 50
minimum weight spanning tree, 45
multigraph, 4
multiplication principle, 137
multiplication table of a group, 74
multiset, 4
mutually visible, 104

National Residency Matching Program, 83
neighborhood, 14

neighborhood of X, 71
network, 16

odd components, 75
one-to-one, 138
onto, 138
orientable, 109
orientation, 4, 97
out-degree, 17
out-neighborhood, 16

PageRank, 8
pairwise disjoint, 136
parity argument, 32
partition, 13
parts, 13
pentagon, 135
percolation on graphs, 12
perfect matching, 9, 70
permutation, 136
Petersen graph, 43
pigeonhole principle, 139
planar, 98
plane dual, 106
plane embedding, 98
plane graph, 98
platonic solid, 100
polyhedral combinatorics, 30
Postman Problem, 30
preferential attachment, 10
Prim's Algorithm, 46
proper k-coloring, 88
proper k-edge-coloring, 88
pseudograph, 4

radius, 8, 42
random geometric graphs, 10
random regular graphs, 10
reflexivity, 56
root, 42, 45
rotation, 31
route inspection problem, 30

saturated, 75
Scheduling Problem, 93
sink, 10
source, 10, 47
spanned by L, 18
spanning, 41
spanning subgraph, 17
stable matching, 83
static st-flow, 118
Storage Problem, 93
structural graph theory, 53
structure theorem, 55
subdivision, 69, 98
subgraph, 17
subset, 136
surjection, 138
symmetry, 56
system of distinct representatives, 73

theta graph, 55
tiling, 85, 86
Timetable Problem, 93
to v, 16
toroidal graph, 110
torus consists, 109
tour, 24
tournament, 35
traceable, 28, 37
trail, 24
transitivity, 56
transversal, 73
Travelling Salesman Problem, 30
tree, 40
triangulate, 105
triangulation, 114
TSP, 30
Turán graph, 130

Turán numbers, 128
Tutte's 1-Factor Theorem, 9
Tutte-Berge Formula, 78

underlying graph, 4
union, 136
uniquely Hamiltonian, 31
unit distance graph, 9
unsaturated, 75

value of a flow, 118
vertex cover, 70
vertex cut, 59, 61
vertex disjoint, 60
vertex set, 4
vertex-connectivity, 65
vertices, 4
Vizing's Theorem, 88

walk, 23
weight, 30
weight function, 30, 45, 47
wheel graph, 19
wheel graphs, 20

References

[1] Ajtai, M., Komlós, J., Simonovits, M., and Szemerédi, E. Exact solution of the Erdös-Sós conjecture. *Manuscript* (2015).

[2] Appel, K., and Haken, W. Every planar map is four colorable. I. Discharging. *Illinois J. Math. 21*, 3 (1977), 429–490.

[3] Appel, K., Haken, W., and Koch, J. Every planar map is four colorable. II. Reducibility. *Illinois J. Math. 21*, 3 (1977), 491–567.

[4] Bollobás, B. *Random graphs*, second ed., vol. 73 of *Cambridge Studies in Advanced Mathematics*. Cambridge University Press, Cambridge, 2001.

[5] Bollobás, B., and Riordan, O. *Percolation*. Cambridge University Press, New York, 2006.

[6] Bonato, A. *A course on the web graph*, vol. 89 of *Graduate Studies in Mathematics*. American Mathematical Society, Providence, RI; Atlantic Association for Research in the Mathematical Sciences (AARMS), Halifax, NS, 2008.

[7] Bondy, J. A., and Chvátal, V. A method in graph theory. *Discrete Math. 15*, 2 (1976), 111–135.

[8] Cameron, K. Thomason's algorithm for finding a second Hamiltonian circuit through a given edge in a cubic graph is exponential on Krawczyk's graphs. *Discrete Math. 235*, 1-3 (2001), 69–77. Combinatorics (Prague, 1998).

[9] Cauchy, A. L. Recherche sur les polyèdres - premier mémoire. *J. École Polytechnique. 9*, 16 (1813), 66–86.

[10] Chvátal, V. A combinatorial theorem in plane geometry. *J. Combinatorial Theory Ser. B 18* (1975), 39–41.

[11] de Grey, A. D. N. J. The chromatic number of the plane is at least 5. *Geombinatorics 28*, 1 (2018), 18–31.

[12] Dirac, G. A. Some theorems on abstract graphs. *Proc. London Math. Soc. (3) 2* (1952), 69–81.

[13] Erdős, P. On sets of distances of n points. *Amer. Math. Monthly 53* (1946), 248–250.

[14] Erdős, P. On some extremal problems in graph theory. *Israel J. Math. 3* (1965), 113–116.

[15] Erdős, P., and Gallai, T. On maximal paths and circuits of graphs. *Acta Math. Acad. Sci. Hungar 10* (1959), 337–356 (unbound insert).

[16] Euler, L. Solutio problematis ad geometriam situs pertinentis. *Comment. Academiae Sci. I. Petropolitanae 8* (1736), 128–140.

[17] Ford, Jr., L. R., and Fulkerson, D. R. Maximal flow through a network. *Canadian J. Math. 8* (1956), 399–404.

[18] Ford, Jr., L. R., and Fulkerson, D. R. *Flows in networks*. Princeton Landmarks in Mathematics. Princeton University Press, Princeton, NJ, 2010. Paperback edition.

[19] Grimmett, G. *Percolation*, second ed., vol. 321 of *Grundlehren der Mathematischen Wissenschaften [Fundamental Principles of Mathematical Sciences]*. Springer-Verlag, Berlin, 1999.

[20] Guth, L., and Katz, N. H. On the Erdös distinct distances problem in the plane. *Ann. of Math. (2) 181*, 1 (2015), 155–190.

[21] Hall, Jr., M. Distinct representatives of subsets. *Bull. Amer. Math. Soc. 54* (1948), 922–926.

[22] Janson, S., Ł uczak, T., and Rucinski, A. *Random graphs*. Wiley-Interscience Series in Discrete Mathematics and Optimization. Wiley-Interscience, New York, 2000.

[23] Kővari, T., Sós, V. T., and Turán, P. On a problem of K. Zarankiewicz. *Colloquium Math. 3* (1954), 50–57.

[24] Kleinberg, J. Complex networks and decentralized search algorithms. In *International Congress of Mathematicians. Vol. III*. Eur. Math. Soc., Zürich, 2006, pp. 1019–1044.

[25] Kuratowski, K. Sur le problème des courbes gauches en topologie. *Fund. Math.* (1930), 271–283.

[26] Lhuillier, S.-A.-J. Mémoire sur la polyèdrométrie. *Ann. Math. 3* (1861), 169–189.

[27] Lovász, L., and Plummer, M. D. *Matching theory*. AMS Chelsea Publishing, Providence, RI, 2009.

[28] Makarychev, Y. A short proof of Kuratowski's graph planarity criterion. *J. Graph Theory 25*, 2 (1997), 129–131.

[29] Mantel, W. Problem 28. *Wiskundige Opgaven 10* (1907), 60–61.

[30] McEliece, R. J. *The theory of information and coding*, student ed., vol. 86 of *Encyclopedia of Mathematics and its Applications*. Cambridge University Press, Cambridge, 2004.

[31] Menger, K. Zur allgemeinen kurventheorie,. *Fund. Math. 10* (1927), 96–115.

[32] Mohar, B., and Thomassen, C. *Graphs on surfaces*. Johns Hopkins Studies in the Mathematical Sciences. Johns Hopkins University Press, Baltimore, MD, 2001.

[33] Penrose, M. *Random geometric graphs*, vol. 5 of *Oxford Studies in Probability*. Oxford University Press, Oxford, 2003.

[34] Pósa, L. Hamiltonian circuits in random graphs. *Discrete Math. 14*, 4 (1976), 359–364.

[35] Ringel, G., and Youngs, J. W. T. Solution of the Heawood map-coloring problem. *J. Combinatorial Theory 7* (1969), 342–352.

[36] Robertson, N., Sanders, D. P., Seymour, P., and Thomas, R. A new proof of the four-colour theorem. *Electron. Res. Announc. Amer. Math. Soc. 2*, 1 (1996), 17–25.

[37] Sheehan, J. The multiplicity of Hamiltonian circuits in a graph. 477–480.

[38] Tait, P. G. Listing's topologie. *Philosophical Magazine, 5th Series 17* (1884), 30–46.

[39] Thomason, A. G. Hamiltonian cycles and uniquely edge colourable graphs. *Ann. Discrete Math. 3* (1978), 259–268. Advances in graph theory (Cambridge Combinatorial Conf., Trinity College, Cambridge, 1977).

[40] Turán, P. Eine Extremalaufgabe aus der Graphentheorie. *Mat. Fiz. Lapok 48* (1941), 436–452.

[41] Tutte, W. T. On Hamiltonian circuits. *J. London Math. Soc. 21* (1946), 98–101.

[42] Tutte, W. T. The factors of graphs. *Canadian J. Math. 4* (1952), 314–328.

[43] van Lint, J. H. *Introduction to coding theory*, third ed., vol. 86 of *Graduate Texts in Mathematics*. Springer-Verlag, Berlin, 1999.