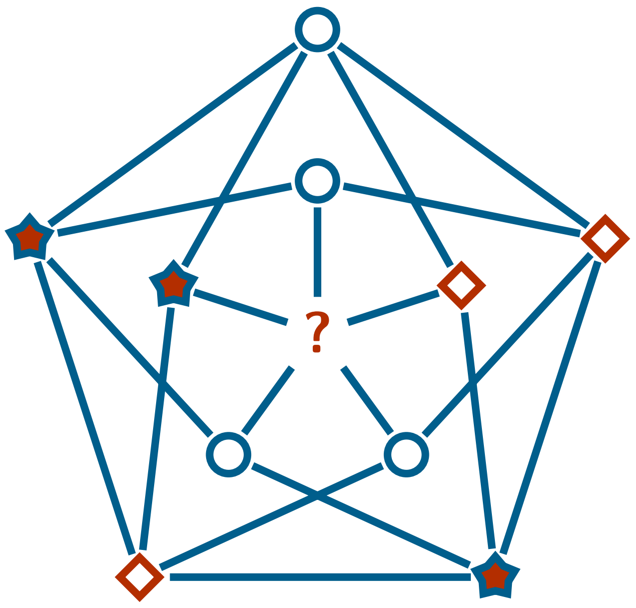
Mikhail Lavrov

Start Doing Graph Theory

Part V: Hard Problems



available online at https://vertex.degree/

Contents

About this document

Where it's from

If you downloaded this PDF file yourself from `https://vertex.degree/contents`, presumably you know what you were doing. But in case you're confused or got the PDF file somewhere else, let me explain!

This is not an entire graph theory textbook. It is one part of the textbook *Start Doing Graph Theory* by Mikhail Lavrov, in case it's convenient for you to download a few smaller files instead of one large file. You can find the entire book at `https://vertex.degree/`; all of it can be downloaded for free.

The complete textbook has some features that the individual parts couldn't possibly have. In this PDF, if you click on a chapter reference from a different part of the book, then you will just be taken to this page, because I can't take you to a page that's not in this PDF file. The hyperlinks in the complete book are fully functional; there is also a preface and an index.

What's inside

In this part of the book, Chapter 17 covers Hamiltonian graphs and Hamilton cycles, Chapter 18 covers cliques and independent sets, and Chapter 19 covers graph coloring. These topics are not entirely related, but united by our approach to them: since solving these problems is computationally hard, we are intested in partial results: necessary conditions, sufficient conditions, and lower and upper bounds.

Chapter 20 introduces line graphs, in which many of these problems become easy, and which let us draw connections between the topics in this part of the book and earlier topics.

The cover

The cover of this PDF shows $\frac{3}{4}$ of a decomposition of the hypercube graph Q_4 as a union of two Hamilton cycles.

The license

17 Hamilton cycles

The purpose of this chapter

This chapter begins with many examples of analyzing concrete graphs to find Hamilton cycles, or prove that none exist. The second half of this chapter is more theoretical. If you are reading this book on your own, you should feel free to skip to the section on tough graphs as soon as you feel like you've had enough of the discussions that come before it. If you are teaching from this book, then you can give more of the concrete examples for a gentle introduction, or leave your students to read them on your own if your goal is to set a faster pace.

The problem of identifying Hamiltonian graphs is the first of the "hard problems" we will consider in this book. What do I mean by this? Well, most precisely put, it is an issue of complexity theory: all the "hard problems" covered in this part of the textbook are so-called NP-complete problems. You may have heard of the "P versus NP problem", an open problem in computer science whose solution is worth \$1 000 000. Finding an efficient algorithm to solve an NP-complete problem, or proving that such an algorithm does not exist, would earn you that million-dollar prize!

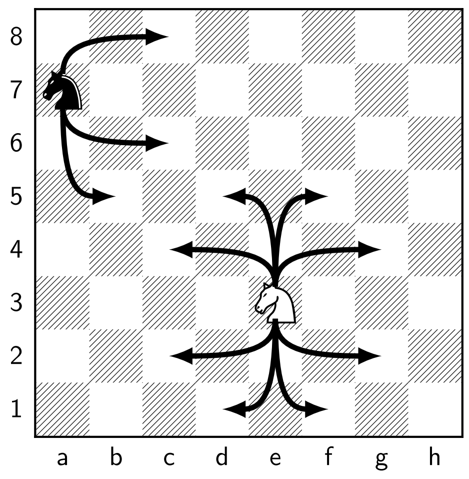
This is not a textbook on complexity theory, however, so I will not go into the details. For us, the consequence of tackling hard problems is that we will not be able to solve them efficiently, or prove theorems with simple if-and-only-if conditions describing when they have a solution. Instead, we will prove partial guarantees, lower and upper bounds, and discuss algorithms that only sometimes work.

17.1 Knight's tours

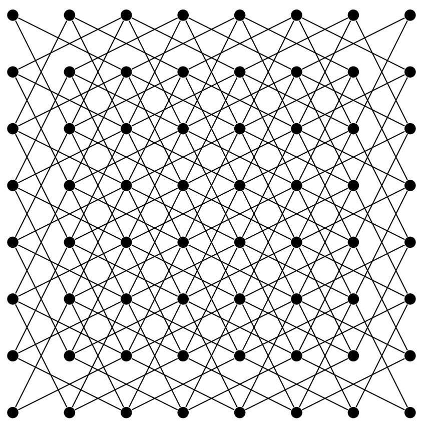
A knight is a chess piece that moves either by taking one step vertically and two steps horizontally, or by taking two steps vertically and one step horizontally, as shown in Figure 17.1a. A knight in the middle of the board can move to a total of 8 different squares, though this number is reduced if the knight is near the edges of the board.

For the purpose of graph theory, Figure 17.1b is the correct way to represent how a knight moves in chess. This is the 64-vertex graph with a vertex for each square of the chessboard, where two vertices are adjacent if a knight can move from one square to the other. In this chapter, we will just call this graph the 8×8 *knight graph*.

A knight's tour is a sequence of knight moves that visits every square of the chessboard exactly once. Finding a knight's tour is a famous mathematical chess puzzle. A slightly harder variant is the problem of finding a closed knight's tour: here, after visiting every square, the knight must make one final move and end at the square where it started. Both problems have been studied extensively long before graph theory, both with generic methods and ones specialized to



(a) How a knight moves in chess



(b) The graph of knight moves

Figure 17.1: Chess, knights, and graph theory

the chessboard [1]. For example, in 1832, H. C. von Warnsdorff proposed the short rule, "Start anywhere, and always move to the space from which the number of moves to unvisited squares is smallest." This is quite likely (though not certain) to produce a knight's tour; it is also a reasonable heuristic for the general problem we are about to study.

Graph-theoretically, the knight's tour problem is the problem of finding a path or cycle in the 8×8 knight graph that contains every vertex of the graph. In general, we say:

Definition 17.1. *A* ***Hamilton path*** *(or spanning path) in a graph is a path that passes through every vertex of the graph. A* ***Hamilton cycle*** *or (or spanning cycle) in a graph is a cycle that passes through every vertex of the graph.*

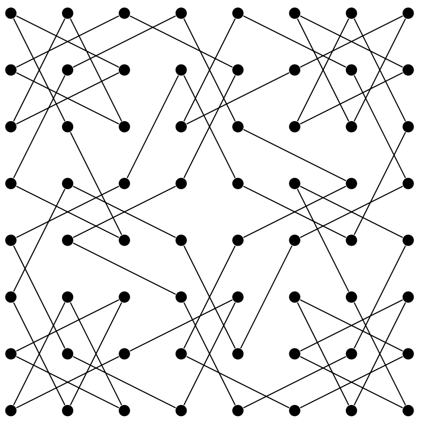
Usually, Hamilton cycles are considered to be the more fundamental object in graph theory, because they have a symmetry that paths lack: every vertex plays an identical role. As a result, graphs with Hamilton cycles are the ones we give a special name.

Definition 17.2. *A graph is called* ***Hamiltonian*** *if it has a Hamilton cycle.*

(Actually, there is also a term for graphs that have a Hamilton path: they are called traceable. This term is much less commonly used.)

Figure 17.2 shows two interesting subgraphs in the 8×8 knight graph. Both of these were found by Dan Thomasson [25], whose website features an impressive collection of knight's tours.

Question:	However, only one of the diagrams in Figure 17.2 shows a Hamilton cycle. Which one, and what's wrong with the other one?
Answer:	Figure 17.2b shows the Hamilton cycle. In Figure 17.2a, every vertex has degree 2, but the edges don't form a single cycle: there are two connected components!


(a) Is this a Hamilton cycle?

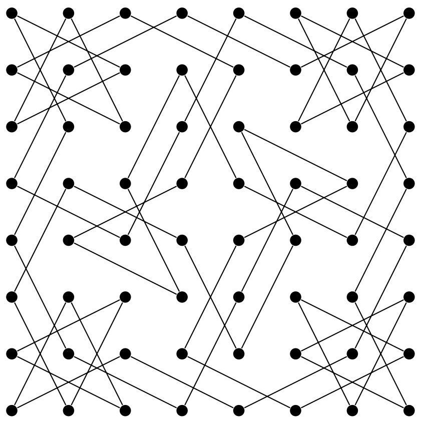
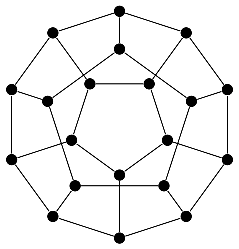

(b) Is this a Hamilton cycle?

Figure 17.2: Two symmetric subgraphs of the 8 × 8 knight graph


(a) The dodecahedron graph

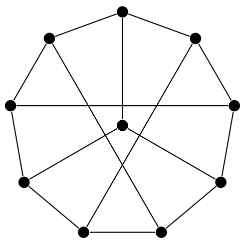

(b) The Petersen graph

Figure 17.3: Are these graphs Hamiltonian?

An object such as the one shown in Figure 17.2a is called a 2-*factor*: a spanning subgraph (in this case, of the 8×8 knight graph) in which every vertex has degree 2. Every Hamilton cycle is a 2-factor, but not vice versa: a Hamilton cycle must also be connected!

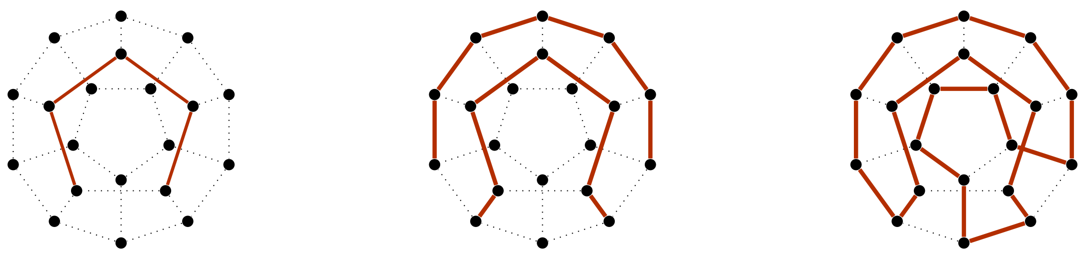
In Chapter 16, we saw the term 1-*factor* used as a synonym for a perfect matching: a spanning subgraph in which every vertex has degree 1. Algorithmically, a 2-factor is not much harder to find than a perfect matching. On the other hand, a Hamilton cycle is much harder to find. There is no known efficient algorithm, and no characterization like Tutte's theorem (Theorem 16.1) of when a Hamilton cycle exists.

Question: Does the 8×8 knight graph have a 1-factor, or perfect matching?

Answer: Yes; one way to be sure without doing any additional work is that we can take every other edge in the knight's tour shown in Figure 17.2b, and get a perfect matching. You might also be able to find some nicely symmetric perfect matchings from scratch.

17.2 Two examples

Figure 17.3 shows two graphs with nice five-fold symmetry in their diagrams. One of these graphs is Hamiltonian; the other is not. Before we develop any general theory, let's consider



(a) Step 1 (b) Step 2 (c) Step 3

Figure 17.4: Solving the “Icosian Game”

these two examples first. (Try them for yourself before you read my explanation, and see what you think.)

The dodecahedron graph in Figure 17.3a has a historical connection to Hamilton cycles. The mathematician William Rowan Hamilton was a big fan of the dodecahedron; in the 1850s, he invented a puzzle called the “Icosian Game”, which was effectively to find a Hamilton cycle in the dodecahedron graph [1]. Although Hamilton’s interest in the puzzle was more related to his study of abstract algebra, the Icosian Game is the reason why we refer to spanning cycles as Hamilton cycles.

There is no obvious starting point to the puzzle, so I will give you an “in” with some mathematical trickery.

Question: As a 3-dimensional shape, the dodecahedron has 12 faces, each of which is a regular pentagon. A Hamilton cycle in the dodecahedron graph uses several edges from each pentagon. What is the average number of edges used per pentagon?

Answer: Since the graph has 20 vertices, a Hamilton cycle has 20 edges. Each edge is also an edge geometrically: an edge between two faces. So if we add up the number of edges of the cycle on each pentagon, we will get 40: each edge will be counted twice. There are 12 pentagons, so the average number of edges per pentagon is $\frac{40}{12} = \frac{10}{3}$.

Question: Is it possible that there is no pentagon from which we use 4 edges?

Answer: No. We can’t use all 5 edges from a pentagon—then our cycle would end early. But we also can’t use only 3 or fewer edges from all pentagons: then every pentagon would be below average, which is impossible.

The pentagons are all alike. (Formally, as in Chapter 2, we should say that there is an automorphism of the dodecahedron graph that can turn any dodecahedron into any other.) So we can arbitrarily decide that the “front” pentagon will be the one in which 4 edges are used. This leads us to the partial cycle in Figure 17.4a.

From here, we can make two other deductions.

Question: Our partial cycle is really a path with two ends. How should we continue from those ends?

Answer: We can't use the edge connecting them, because then our cycle would end early. So we have to take the only remaining edges: the ones going outward.

Question: Which edges incident to the topmost vertex should be part of the cycle?

Answer: There is no choice here: the vertex below the topmost vertex already has degree 2, so it can't accept any more edges. We must use the two edges going left and right.

Applying the same logic to several other vertices on the perimeter, we arrive at a bigger partial solution: the one shown in Figure 17.4b.

At this point, one more "symmetry breaking" step is necessary. From the bottom vertex in the diagram, we must use either the edge going left or the edge going right. Which one? Well, it can't possibly matter: both the diagram, and our partial solution so far, have left-right symmetry! So we can arbitrarily pick the edge going right to get one possible solution; its mirror image will contain the edge going left, instead.

After this choice, there are several more forced decisions like ones we already made: once we use two edges out of a vertex x, if xy is the third edge, then it cannot be part of the Hamilton cycle, which forces our hand at vertex y. Repeating these steps is all that's necessary to arrive at Figure 17.4c, the complete solution.

As you see from this example, finding a Hamilton cycle really is more like solving a puzzle than solving a math problem. Although deductions like these can often handle small graphs with enough low-degree vertices, they are not reliable. We might have to make guesses and backtrack if we fail, which is a slow and tedious process.

Rather than attempt such a backtracking solution for the Petersen graph, I will show you a different style of argument.

Proposition 17.1. *The Petersen graph is not Hamiltonian.*

Proof. This proof will rely on just two properties of the Petersen graph. First, it is 3-regular, which can be checked quickly by looking at Figure 17.3b. Second, as we checked in Lemma 5.3, it contains no cycles of length 3 or 4.

Now, we proceed in an unusual way. Rather than starting with the Petersen graph and looking for a Hamilton cycle, let's start with the cycle and look for the Petersen graph!

What do I mean? Well, if there were any Hamilton cycle in the Petersen graph, then we would be able to arrange the vertices around a regular decagon whose edges are the edges of that cycle. This would look like the diagram in Figure 17.5a—with five more edges we haven't placed yet completing the diagram of the 15-edge Petersen graph. For example, from the topmost vertex, exactly one of the three dashed edges must exist, we just don't know which one. (We need a

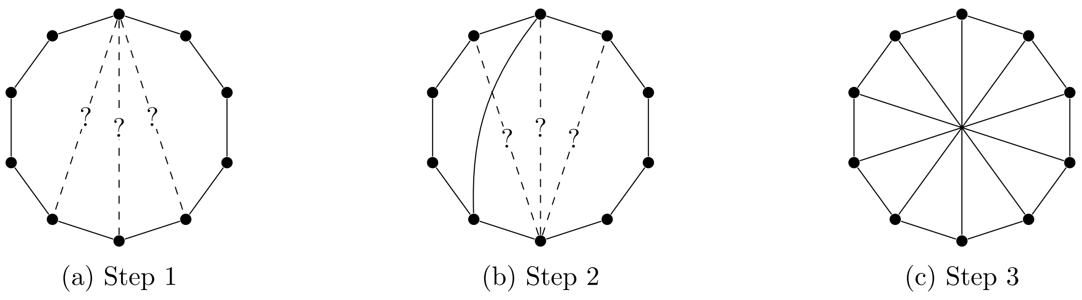


Figure 17.5: Diagrams for the proof of Proposition 17.1

third edge from that vertex to give it degree 3, but any edge other than the three dashed edges would create a short cycle.)

Suppose we pick the left dashed edge. Then we arrive at Figure 17.5b, and we can ask the same question for the bottom-most vertex…

Question: Which of the three dashed edges in Figure 17.5b can we use, if we want to eventually complete the picture to a diagram of the Petersen graph?

Answer: None of them! They all create a 3-cycle or 4-cycle.

So we cannot pick the left dashed edge (or, by the same argument, the right one). We must pick the edge going directly across. Since there's nothing special about the top vertex, the same argument applies to every vertex: the only way we can hope to obtain the Petersen graph is by joining each vertex to the vertex directly opposite. This gives us the graph in Figure 17.5c.

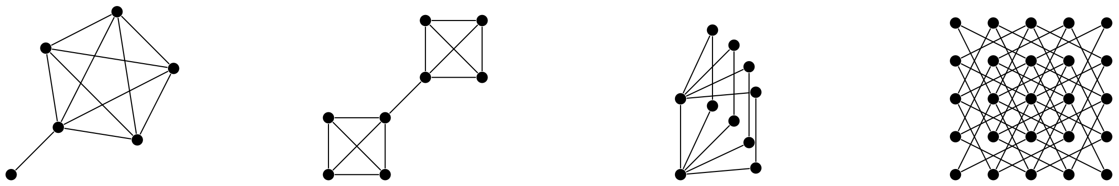
Question: Why is the graph in Figure 17.5c not the Petersen graph?

Answer: It, too, has cycles of length 4: starting from any vertex, take a step across, a step clockwise, another step across, and a step counterclockwise.

Since all possible attempts to expand a 10-vertex cycle to get a Petersen graph fail, we conclude that the Petersen graph has no 10-vertex cycle as a subgraph: it is not Hamiltonian. □

17.3 Tough graphs

The proof of Proposition 17.1 is rather long. I can't honestly tell you that the Petersen graph is an unusually bad example; for many graphs, finding a Hamilton cycle or proving that one does not exist is even harder. However, there are also many graphs for which we can give a quick argument to rule out a Hamilton cycle. Four of them are shown in Figure 17.6.



(a) Lollipop graph $L_{5,1}$ (b) 4-barbell graph (c) Book graph B_4 (d) 5×5 knight graph

Figure 17.6: Four graphs which are not Hamiltonian

Question: Why is the lollipop graph in Figure 17.6a not Hamiltonian?

Answer: It has a leaf: a Hamilton cycle could not possibly enter that vertex and leave it.

Question: Why is the barbell graph in Figure 17.6b not Hamiltonian?

Answer: The edge joining the two copies of K_4 is a bridge; by Lemma 9.2, it is not part of any cycles. But without using it, a cycle can't contain vertices from both copies of K_4.

Question: Why is the book graph in Figure 17.6c not Hamiltonian?

Answer: Between visits to every "page" of the book (the vertices of degree 2), a cycle must pass through at least one vertex of the "spine" (the vertices of degree 5). But there are only 2 vertices in the spine, so a cycle cannot visit all 4 pages.

Question: Why is the 5×5 knight graph in Figure 17.6d not Hamiltonian?

Answer: Remember the black-and-white covering of a chessboard? On a 5×5 chessboard, there would be 13 squares of one color, and 12 on the other. A knight always moves to squares of a different color (put differently, the knight graph is bipartite), so a cycle will visit an equal number of squares of each color: it can't visit all 25 vertices.

Surprisingly, all four arguments are special cases of one result!

We say that a graph G is *tough* if, for any $k \geq 1$, deleting k vertices from G results in at most k connected components. You might think, from this definition, that proving that a graph is tough is a headache involving lots of casework: there are too many ways to delete some number of vertices from a graph! In general, you'd be right.

In some cases, it's not so bad. Here is a lemma that proves that the cycle graph C_n is tough without any casework at all—and, as a bonus, it comes with a mini-review of Chapter 10.

Lemma 17.2. *For all $n \geq 3$, the cycle graph C_n is tough.*

Proof. The cycle graph C_n has only one cycle: the entire graph. So as soon as we delete any vertices from it, we get a graph with no cycles, which we know as a forest. Specifically, if we delete k vertices from C_n, we get an $(n-k)$-vertex forest. Each deleted vertex has degree 2 in C_n, so it costs us at most two edges. Therefore the forest we are left with has at least $n-2k$ edges.

Question:	Why do I say "at most" and "at least"; shouldn't every vertex we delete cost us both edges incident to that vertex?
Answer:	We might delete both endpoints of an edge, in which case we shouldn't count the edge as removed twice.

By a result from much earlier in this book, Proposition 10.3, the number of connected components in a forest is the number of vertices minus the number of edges. In this case, this is at most $(n-k)-(n-2k)=k$. Therefore deleting k vertices results in a graph with at most k components, proving the lemma. □

Starting at C_n is useful, because it gives us a connection to Hamiltonian graphs. This is a result of Václav Chvátal, who defined tough graphs in 1973 for this very purpose [6].

Corollary 17.3. *All Hamiltonian graphs are tough.*

Proof. Every n-vertex Hamiltonian graph G contains a copy of C_n as a subgraph: that's what a Hamilton cycle is! We can think of G as this Hamilton cycle, together with some extra edges it didn't use.

By Lemma 17.2, if we delete k vertices from the Hamilton cycle, the result will have at most k connected components. What happens if we delete those same vertices from G? Those same connected components will still be connected, but it's possible that the extra edges will join some of them together into larger components. This can only reduce the number of connected components, so there are still at most k; therefore G is tough. □

The most profitable way to use Corollary 17.3 is in the form of its contrapositive: if a graph is not tough, then it is not Hamiltonian. If a graph is not tough, then there is a short proof of that fact: simply say which k vertices must be deleted to leave $k+1$ or more connected components. (It is a short proof, not an easy proof, because finding those k vertices may be hard.)

For example, none of the graphs in Figure 17.6 are tough—and the sets of vertices we must delete to prove this are closely related to our earlier arguments proving that none of these graphs are Hamiltonian. This unifies our previous arguments into a single idea.

Corollary 17.3 is a necessary condition: to be Hamiltonian, a graph must be tough. However, it is not sufficient, so we cannot always rely on it. The Petersen graph—a thorn in the side of every graph theorist—is a counterexample.

Proposition 17.4. *The Petersen graph is tough.*

Proof. As we saw in Chapter 5, the Petersen graph has many automorphisms, and in particular, it has an automorphism taking any vertex to any other vertex. Due to this symmetry, it's enough to see what happens when the vertices we delete from the Petersen graph include the vertex at the center of Figure 17.3b, which we will call x.

If we delete x alone, the remaining graph is connected, so the definition of a tough graph holds so far. What's more, we can see from the diagram in Figure 17.3b that the remaining graph is Hamiltonian: we get a Hamilton cycle by going around the perimeter of the diagram. By Corollary 17.3, this graph is tough. So if we delete x and k more vertices from the Petersen graph, we're left with at most k connected components. Since we've deleted $k+1$ vertices, that's even slightly stronger than what we need to conclude that the Petersen graph is tough. □

17.4 Sufficient conditions

It is also possible to find conditions for a graph to be Hamiltonian that are sufficient, but not necessary. That is, if a graph G satisfies these conditions, we can be sure that it has a Hamilton cycle; however, if G does not satisfy these conditions, we learn nothing.

What's the point, then? Well, we can look for conditions that are easy to check. When faced with a concrete problem, we can begin by checking a few such conditions, and if we're lucky, this will tell us that G is Hamiltonian right away. Otherwise, we may have no choice but to embark on a tedious journey full of backtracking.

A promising place to start looking for such conditions is the degree sequence of G. Computing the degree of every vertex may take some time, if the graph is large, but it is not a computationally challenging problem: we can even do it by hand, just by looking at a diagram. At the same time, the degrees of the vertices of G tell us a lot about the structure of G, so we can hope to learn something about Hamilton cycles in G, as well.

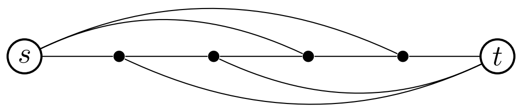
Historically, the first such result was a 1952 theorem by Gabriel Dirac [8] (the son of Paul Dirac, the quantum physicist). More results followed, and eventually, the essence of the proof was distilled into a rather peculiar statement, proven by John Bondy and Václav Chvátal in 1976 [2].

Theorem 17.5. *In a graph G with n vertices, let s and t be two non-adjacent vertices with $\deg_G(s) + \deg_G(t) \geq n$. If $G + st$ is Hamiltonian, then so is G.*

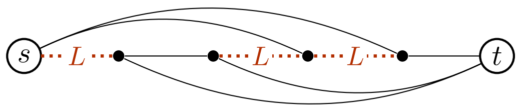
Question: Is it possible that G is Hamiltonian, but $G + st$ is not?

Answer: No: adding an edge can never destroy a Hamilton cycle. We could have stated Theorem 17.5 as an if-and-only-if condition: $G + st$ is Hamiltonian if and only if G is Hamiltonian.

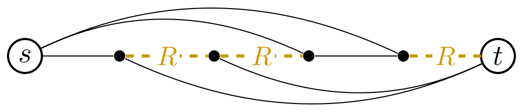
What is going on here? Well, there are many works of fiction in which the hero is granted special powers by a magic item. Later in the story, the magic item is lost, and the hero is distressed—only to discover that the item wasn't necessary to have the special powers. The magic was really inside the hero the whole time! The statement of Theorem 17.5 is similar: the graph G is granted the special power of being Hamiltonian by the magic edge st. However



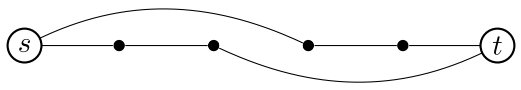
(a) P together with all edges incident to s or t



(b) Edges in the set L



(c) Edges in the set R



(d) The cycle in G

Figure 17.7: Turning the $s - t$ path in the proof of Theorem 17.5 into a cycle

(provided that edge st satisfies the degree condition) the graph G didn't need that edge: the Hamilton cycle was really inside G the whole time.

Theorem 17.5 tries to help us by giving us a different problem to solve: instead of checking whether G is Hamiltonian, we can check whether $G + st$ is Hamiltonian.

Question: How can this possibly help us?

Answer: Well, $G + st$ has one more edge, so finding a Hamilton cycle in $G + st$ might be easier. A single edge might not make a difference, but if we apply Theorem 17.5 repeatedly, we can hope to turn a graph with a well-hidden Hamilton cycle into a graph with many blindingly obvious Hamilton cycles.

Let's prove the theorem first, though.

Proof of Theorem 17.5. Take a Hamilton cycle in $G + st$. If edge st is not part of that cycle, we're done: it also exists in G. If edge st is part of that cycle, then deleting that edge leaves an $s - t$ Hamilton path in G. Let this path be represented by the walk $(x_1, x_2, \dots, x_n)$ where $x_1 = s$ and $x_n = t$.

Our goal is to find an alternate way to complete the path P to a Hamilton cycle in G, without using edge st. All we really know about G is that, because $\deg_G(s) + \deg_G(t) \geq n$, there must be many other edges incident to s or t. (There might be other edges, too, but we cannot count on them.) In short, the subgraph of G we have available to work with might look something like the diagram in Figure 17.7a.

For every edge incident to either s or t, something special happens: we obtain a different Hamilton path, which uses that edge, but leaves out one edge of P. Here are the two ways this can happen:

- If $sx_i \in E(G)$ for some i, then add edge sx_i to P and delete edge $x_{i-1}x_i$. The resulting graph is an $x_{i-1} - t$ Hamilton path.

  We define the set L (for "left") to be the set of all edges of P that can be left out in this way: $L = \{x_{i-1}x_i : sx_i \in E(G)\}$. These are highlighted in Figure 17.7b.

- If $x_{i-1}t \in E(G)$ for some i, then add edge $x_{i-1}t$ to P and delete edge $x_{i-1}x_i$. The resulting graph is an $s - x_i$ Hamilton path.

  We define the set R (for "right") to be the set of all edges of P that can be left out in this way: $R = \{x_{i-1}x_i : x_{i-1}t \in E(G)\}$. These are highlighted in Figure 17.7c.

Question: Can we say anything about the sizes of the sets L and R?

Answer: We have $|L| = \deg_G(s)$ and $|R| = \deg_G(t)$, because for every edge incident to s, we get an element of L, and for every edge incident to t, we get an element of R.

By themselves, edges in L or in R are not anything special, since we aren't looking for more Hamilton paths. The magic happens if we find $x_{i-1}x_i$ in both sets!

Suppose $x_{i-1}x_i \in L \cap R$, so that edges sx_i and $x_{i-1}t$ are both present. Let C be the graph obtained from P by deleting edge $x_{i-1}x_i$ and adding both sx_i and $x_{i-1}t$, as shown in Figure 17.7d. This graph C is a spanning subgraph of G. It is connected: deleting edge $x_{i-1}x_i$ separated P into two connected components, but both sx_i and $x_{i-1}t$ join two vertices in different components, reconnecting the graph. It is 2-regular: compared to P, the degrees of s and t go up by 1, while all other degrees remain the same. Therefore C is a Hamilton cycle!

Question: The situation is a bit unusual if $x_1x_2 \in L \cap R$. What happens?

Answer: This would imply that $x_1x_n = st$ is an edge of G. In principle, this makes a cycle in an even simpler way: without deleting anything, we just add edge st to P. However, Theorem 17.5 assumes that this case does not occur: s and t are not adjacent.

Finding an edge in $L \cap R$ is why the degree condition on s and t is needed. Since $|L| = \deg_G(s)$ and $|R| = \deg_G(t)$, we know that $|L| + |R| \geq n$ from the hypotheses of Theorem 17.5. However, L and R are both subsets of $E(P)$, and an n-vertex path has only $n-1$ edges, so $|L \cup R| \leq n-1$. Since $|L \cup R| < |L| + |R|$, there must be some overlap between L and R; as we've already seen, a single edge in $L \cap R$ proves that G is Hamiltonian, which proves the theorem. □

As an example of applying Theorem 17.5, we will use it to derive Dirac's 1952 theorem on Hamilton cycles, with a sufficient condition based on the minimum degree $\delta(G)$.

Corollary 17.6. *If G has $n \geq 3$ vertices and $\delta(G) \geq \frac{1}{2}n$, then G is Hamiltonian.*

Proof. The minimum degree condition on G guarantees that for every two vertices s and t which are not adjacent, we have

$$\deg_G(s) + \deg_G(t) \geq \frac{1}{2}n + \frac{1}{2}n = n,$$

so the hypotheses of Theorem 17.5 hold. Therefore we can add any edges we like to G without changing the problem: if we get a Hamiltonian graph, then G must have been Hamiltonian all along.

The edges we will choose to add to G are all the edges missing from G (one at a time). Once we're done, we get a graph isomorphic to the complete graph K_n. But K_n is definitely a Hamiltonian graph: you can visit the vertices in any permutation you like, then go back to the start, and this will be a Hamilton cycle. Therefore the graph we started with, G, must also have been Hamiltonian. $\square$

How good is Corollary 17.6? Well, we can compare it to Proposition 4.3, which gives $\delta(G) \geq \frac{n-1}{2}$ as a sufficient condition for G to be connected. As we saw in Chapter 4, that condition is the best possible: if the minimum degree of G were any lower than $\frac{n-1}{2}$, then it would not necessarily be connected.

When $\delta(G) = \frac{n-1}{2}$ exactly, we are in the narrow gap between these two results, where the graph is guaranteed to be connected by Proposition 4.3, but not yet guaranteed to be Hamiltonian by Corollary 17.6.

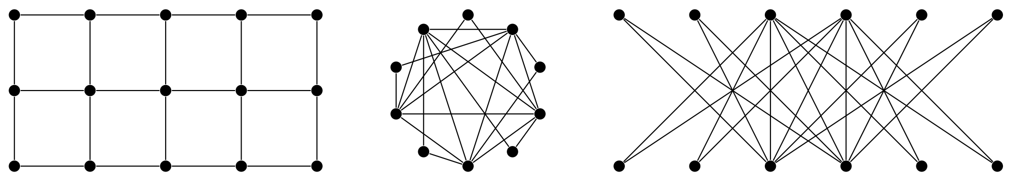
Question: Are there any n-vertex graphs G with $\delta(G) = \frac{n-1}{2}$ which are not Hamiltonian?

Answer: Yes: one example is the slightly unbalanced complete bipartite graph $K_{(n-1)/2,(n+1)/2}$. This graph is not Hamiltonian for the same reason that the 5×5 knight graph in Figure 17.6d was not Hamiltonian.

This tells us that the minimum degree condition of Corollary 17.6 is also the best possible: we cannot reduce it any further and still have a true statement.

17.5 Practice problems

1. For each of the graphs below, find a Hamilton cycle, or explain why a Hamilton cycle cannot exist.



2. a) Show that for all $n \geq 5$, the graph $\overline{C_n}$ (the complement of the n-vertex cycle graph) is Hamiltonian.

b) Find a 5-vertex graph G such that neither G nor $\overline{G}$ is Hamiltonian.

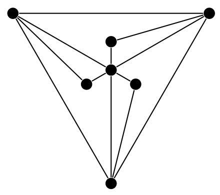
3. a) Find a Hamilton cycle in the cube graph Q_3.

   b) Prove that the hypercube graph Q_n has a Hamilton cycle for all $n \geq 3$, by induction on n.

   c) A Gray code is an alternate binary encoding of numbers with a useful property: to go from the encoding of k to the encoding of $k+1$, only one bit needs to be changed.

   Explain how to obtain a Gray code for the numbers $0, 1, \ldots, 2^n - 1$ using a Hamilton cycle in Q_n.

4. In Chvátal's original paper defining tough graphs [6], the following example is given:



   a) Prove that this graph is not Hamiltonian.

   b) Prove that this graph is tough.

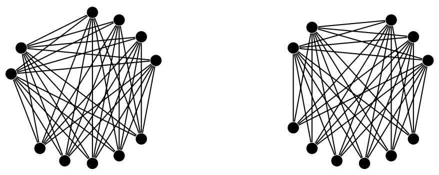
5. Let G be a bipartite graph with bipartition (A, B). Prove that if there is a subset $S \subseteq A$ with $|N(S)| < |S|$, then G cannot be Hamiltonian. (This statement is also a consequence of the statement of the next problem, but it can be proved directly.)

6. Prove that every tough graph with an even number of vertices (not necessarily bipartite) has a perfect matching.

7. Let G be an arbitrary graph, and let H be the graph obtained by adding a new vertex to G, adjacent to all vertices which already existed in G. Prove that G is traceable (G has a Hamilton path) if and only if H is Hamiltonian (H has a Hamilton cycle).

   (This argument is the reason why we do not spend much time thinking about traceable graphs and Hamilton paths: we can use it to take almost any fact about Hamilton cycles and turn it into a fact about Hamilton paths, instead.)

8. A complete tripartite graph is formed by taking three groups of vertices A, B, and C, then adding an edge between every pair of vertices in different groups: this is a generalization of complete graphs and complete bipartite graphs. We write $K_{a,b,c}$ for the complete tripartite graph with $|A| = a$, $|B| = b$, and $|C| = c$. For example, below are diagrams of $K_{2,4,5}$ (left) and $K_{2,3,6}$ (right).



   a) One of these graphs is Hamiltonian. Find a Hamilton cycle in that graph.

b) The other graph is not Hamiltonian. Give a reason why it does not have a Hamilton cycle.

c) Generalize: find and prove a rule that tells you when $K_{a,b,c}$ is Hamiltonian. You may assume $a \leq b \leq c$.

9. a) Prove that all graphs with the degree sequence $7, 7, 7, 7, 7, 7, 5, 5, 2$ are Hamiltonian.

   b) Find a general condition on degree sequences $d_1, d_2, \ldots, d_n$ which guarantees that all graphs with that degree sequence are Hamiltonian, and which generalizes your argument from part (a).

10. (BMO 2011) Consider the numbers $1, 2, \ldots, n$. Find, in terms of n, the largest integer t such that these numbers can be arranged in a row so that all consecutive terms differ by at least t.

18 Cliques and independent sets

The purpose of this chapter

The focus of this chapter betrays my interests: more than half of it has ended up related to Ramsey numbers and Ramsey's theorem. At the same time, I am laying the groundwork for several topics in the next chapter: interval graphs, greedy algorithms, and even random graphs with no large cliques or independent sets will all help us understand vertex coloring.

I briefly considered whether this book has too many chess puzzles in it, but the contrast between the queen placement problem in this chapter and the rook placement problem in Chapter 13 will be very useful for us as an example later on in Chapter 20. (I mention some of the connections in this chapter as well, so be sure you remember what the matching number $\alpha'(G)$ and vertex cover number $\beta(G)$ mean.)

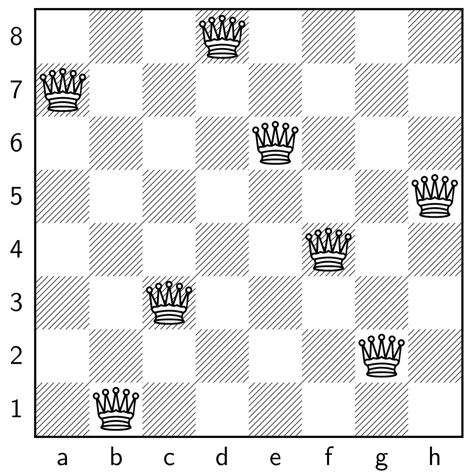
There is some redundancy built into my presentation of several of these topics, which explains how long this chapter has gotten. Proposition 18.2 and Theorem 18.4 arrive at almost the same destination, one using algorithms and the other by induction. Before the proof of Theorem 18.5, I present the same argument with concrete numbers. I do this so that you (as a reader or as a teacher) can choose which of these work best for you.

18.1 Two problems about queens

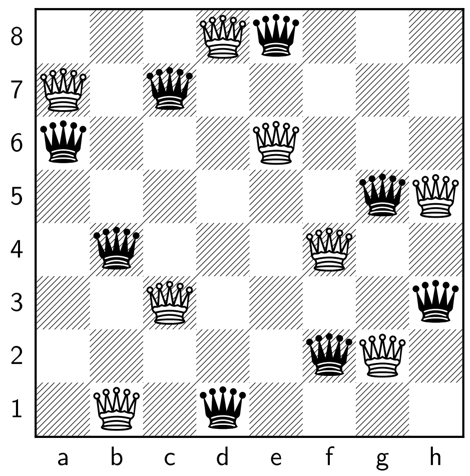
We have already looked at several puzzles about chess pieces, with rooks appearing in Chapter 13 and knights appearing in Chapter 17. Now it is time to look at the most powerful chess piece of all: the queen. The queen is a chess piece that can move any number of squares horizontally, vertically, or diagonally.

The classic 8 queens puzzle is the following: can you place 8 queens on a standard 8×8 chessboard so that none of the queens attack each other—that is, none of them can move to a square occupied by another? This is a strictly harder version of the problem posed in Chapter 13, where we placed rooks under the same condition, because a queen has all the movement potential of a rook, and more. It is still true that we cannot place two queens in the same rank or the same file, and so 8 queens is the absolute limit. However, putting all the queens in different ranks and files is not enough, due to the diagonal moves.

An early summary of the problem was given in *Mathematical Recreations and Essays* by W. W. Rouse Ball in 1892, republished many times since [1]. A total of 92 solutions are possible, but if we consider two solutions to be the same if they differ only by a rotation or reflection, then there are only 12 different possible solutions to consider. One of them is shown in Figure 18.1a; can you find a different solution yourself?



(a) 8 non-attacking queens



(b) 8 non-attacking queens, twice

Figure 18.1: Solutions to two puzzles about queen placements

We can keep going. Figure 18.1b shows two simultaneous solutions to the 8 queens puzzle that do not share any squares: one with 8 white queens, and one with 8 black queens. An even more impressive solution was found by Rob Pratt [21], who placed six solutions to the 8 queens puzzle on a chessboard simultaneously, and verified that more is impossible.

Question:	In fact, without seeing either of the diagrams in Figure 18.1, you can argue that if a solution to the 8 queens puzzle exists, then we can solve the version with 16 queens as well. How?
Answer:	Combine a solution to the 8 queens puzzle with its mirror image! The mirror image of a queen's location cannot be occupied by another queen, because then those two queens would attack each other.

Both of these problems can be modeled using graph theory. For the 8 queens puzzle itself, we define the 8×8 *queen graph* to have 64 vertices, one for each square of the chessboard, with an edge for between every two vertices that share a rank, file, or diagonal. Then our goal is to select 8 vertices from this graph with no edges between them.

For the problem of simultaneous solutions in multiple colors, define a graph whose 92 vertices are the 92 different possible solutions to the 8 queens puzzle, Make two vertices adjacent if the solutions overlap, so that they cannot be placed on the chessboard simultaneously. Then our goal is to select as many vertices as possible from this graph with no edges between them—another instance of the same problem!

In general, we make the following definitions:

Definition 18.1. *An **independent set** in a graph G is a set of vertices $I \subseteq V(G)$ such that no two vertices in I are adjacent. The **independence number** of G, denoted $\alpha(G)$, is the number of vertices in a maximum independent set in G.*

We defined the graphs in our previous problems such that edges represent conflicts, and so the set that we want to choose is a set of vertices with no edges between them. Instead, we could have defined the graphs so that edges represent compatibility. That is, two squares on the chessboard would be adjacent if two queens on those squares do not attack each other, and two solutions to the 8 queens puzzle would be adjacent if they do not overlap. This graph would be the complement of the 8×8 queen graph.

If we had done this, we would be looking for a different object, which has its own definition.

Definition 18.2. *A **clique** in a graph G is a set of vertices $Q \subseteq V(G)$ such that every pair of vertices in Q is adjacent. The **clique number** of G, denoted $\omega(G)$, is the number of vertices in a maximum clique in G.*

It is hard to say why α (alpha) and ω (omega) were chosen to represent these quantities, but at the very least there is some relationship there: α is the first letter in the Greek alphabet, and ω is the last.

Previously, we've referred to complete graphs as cliques, and this is not a coincidence: a set of vertices Q is a clique precisely when the induced subgraph $G[Q]$ is a copy of the complete graph $K_{|Q|}$. In principle, we could have defined cliques to be complete subgraphs of G, but this is not as useful. The edges in the subgraph don't tell us anything: they are simply all present.

Question: What kind of subgraph is induced by a m-vertex independent set?

Answer: A copy of $\overline{K_m}$, also known as the empty graph.

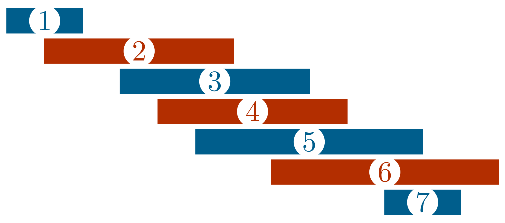
The queen graph is a bit asymmetric: the corner and edge vertices have fewer neighbors. A more mathematically elegant version of the problem makes all vertices equal, by allowing queen moves that wrap around the board. For example, the diagonal through the squares `f1`, `g2`, `h3` would continue to `a4`, `b5`, and so on, as though we had glued opposite sides of the board to make a cylinder.

More generally, we define the periodic $n \times n$ queen graph to have vertices (x, y) where $1 \le x \le n$ and $1 \le y \le n$; here, x and y represent the horizontal and vertical coordinates of a square. Two vertices (x_1, y_1) and (x_2, y_2) are adjacent when $x_1 = x_2$ or $y_1 = y_2$ or $x_1 + y_1 \equiv x_2 + y_2 \pmod{n}$ or $x_1 - y_1 \equiv x_2 - y_2 \pmod{n}$. The "mod n" is what makes the board wrap around. With this modification, the side and corner squares are now just like every other square of the board.

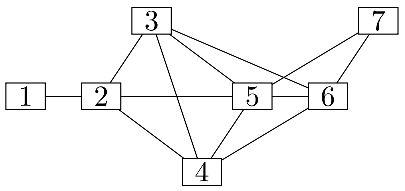
Unfortunately, there is no 8-vertex independent set in the periodic 8×8 queen graph: no solution to the 8 queens problem on a board that wraps around. George Pólya proved this in 1921 [20], along with a general statement: the periodic $n \times n$ queen graph has an n-vertex independent set exactly when n is not divisible by 2 or 3.

Remarkably, in such cases, we can even place n disjoint solutions on the $n \times n$ board, occupying every square! This object is known as a Knut Vik design, and has applications in minimizing the variability of statistical experiments. A Knut Vik design for all $n \times n$ boards where n is not divisible by 2 or 3 was first constructed by Samad Hedayat and Walter Federer in 1975 [16].

(In general, a partition of a graph into independent sets is called a *proper coloring*, and will be discussed in the next chapter.)



(a) 7 overlapping intervals



(b) The corresponding interval graph

Figure 18.2: Constructing an interval graph

18.2 Redundancies and relationships

Applications such as the 8 queens puzzle could be modeled either by cliques or by independent sets. Why, then, do we define both invariants to begin with? We could simply adopt the convention that we always use the "conflict" definition, where two vertices are adjacent if they're incompatible. Then, we'd always be looking for independent sets, and we'd never need to know the definition of a clique.

Question: We can always reduce the problem of finding cliques in a graph G to finding independent sets, even if it did not arise from such an application. How?

Answer: Take the complement graph $\overline{G}$, which has exactly the edges not present in G. Then a clique in G is precisely an independent set in $\overline{G}$.

One of the reasons we don't do this is that we might prefer one version of the graph for other reasons. This is true of the 8×8 queen graph where two vertices are adjacent if a queen on one square attacks the other, for example. In this graph, walks correspond to possible sequences of queen moves, which could have other applications. So if we want to keep the same definition of the graph for multiple puzzles, this forces our hand.

Sometimes it is even interesting to study both cliques and independent sets in the same graph. For example, this occurs in some scheduling problems. If we are scheduling a number of events (such as presentations at a conference or a convention) then we might want to think about which events are happening at the same time: we could make a graph where every event is a vertex, and two vertices are adjacent if the event times overlap.

Slightly more generally, suppose we have a set of intervals

$$[a_1, b_1], [a_2, b_2], \dots, [a_n, b_n]$$

in the real line. (In the scheduling problem, a_i and b_i are the starting and ending time of the i^{th} event.) Then we can define a graph whose vertices are these intervals, with an edge whenever two intervals overlap. Such a graph is called an **interval graph**.[1]

Figure 18.2 shows a collection of intervals, together with the corresponding interval graph.

[1] Often, we also say that a graph isomorphic to a graph obtained in this way is also an interval graph. This way, being an interval graph becomes a true graph invariant, and we can discuss the structural question of which graphs can be represented as interval graphs.

Question: In an interval graph representing a schedule of events, what would a clique represent?

Answer: A clique corresponds to a set of intervals that all overlap: a set of events all happening at the same time. (You might care, for example, because you want to put them all in different rooms.)

Question: What would an independent set represent?

Answer: An independent set corresponds to a set of disjoint intervals: a set of events that don't conflict. (You might care, for example, because you want to attend all of them.)

Returning to the original topic, another reason to study both cliques and independent sets is that we might study the relationship between these objects and other properties of the graph. For example, we might ask: how does the maximum degree of a graph affect the clique number, and how does it affect the independence number? Both are valid questions, with very different answers.

In fact, there are already such connections to be explored. The independence number $\alpha(G)$ is closely related to both invariants introduced in Chapter 14: the matching number $\alpha'(G)$ and the vertex cover number $\beta(G)$.

As far as the matching number goes, you might be a bit suspicious due to the similar notation. It is not an accident: the $'$ symbol indicates that $\alpha'(G)$ is the "edge version" of $\alpha(G)$. Where $\alpha(G)$ is the maximum number of vertices that do not share any edge, $\alpha'(G)$ is the maximum number of edges that do not share any vertex. We will explore this connection and others like it in more detail in Chapter 20.

It is the connection to $\beta(G)$ that you might be surprised by.

Question: Suppose U is a particularly small vertex cover in a graph G: a set of vertices including at least one endpoint of every edge. How can we use it to find a particularly large independent set in G?

Answer: The set $I = V(G) - U$ consisting of all vertices not in U is an independent set![2] If two vertices $x, y \in I$ were adjacent, then U would not contain any endpoint of the edge xy, contradicting its status as a vertex cover.

Numerically: $\alpha(G) = |V(G)| - \beta(G)$. Once again, the question arises: why do we have two invariants $\alpha(G)$ and $\beta(G)$ when one of them can easily be used to obtain the other?

A good answer in this particular case is that we might be interested in ratios between the number of vertices in two independent sets, or two vertex covers. For example, if we have an algorithm that finds a vertex cover of at most $2\beta(G)$ vertices, or an independent set of at least

[2]Together, U and I make V, just as "you" and "I" make "we".

$\frac{1}{2}\alpha(G)$ vertices, we might say: that's within a factor of 2 of the true answer, so the guarantee is good enough.

In fact, such an approximation algorithm for vertex covers does exist—but such an approximation algorithm for independent sets does not, as far as we know! That's because the relationship $I = V(G) - U$ does not play well with ratios: doubling the size of the set U is not at all the same thing as halving the size of the set I.

18.3 Greedy algorithms

In general, algorithms that find maximum cliques or maximum independent sets must do so by backtracking: pick a vertex, try including it, and if it doesn't work out, try leaving it out. This is better than trying all $2^{|V(G)|}$ sets of vertices one by one: if we pick a vertex, then we can make some deductions about which other vertices cannot be picked. However, even the best backtracking algorithms take an exponentially long time to finish.

If we want an answer quickly, we must give up on insisting that we get the right answer. The most basic technique is to use a greedy algorithm: to pick vertices to add to our set without considering the consequences for future decisions, and to see where that gets us.

Here is one possible greedy algorithm for finding an independent set I in a graph G. We begin by setting $I = \varnothing$; we also keep track of a set A which contains all the vertices that are still available for use. Initially, $A = V(G)$. A single iteration of the greedy algorithm consists of the following steps:

1. Let x be any vertex in A; add it to the independent set, replacing I by $I \cup \{x\}$.
2. Replace A by $A - (\{x\} \cup N(\{x\}))$: remove both x and $N(\{x\})$ (the set of vertices adjacent to x) from A.
3. If $A = \varnothing$, stop.

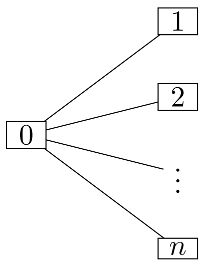
It is impossible for the set I to ever contain two adjacent vertices x and y: if we ever add x to I, we remove y from A, and if we ever add y to I, we remove x from A. Therefore the output of this algorithm is guaranteed to be an independent set.

Question: If we want a greedy algorithm for finding a clique, how should we modify this algorithm?

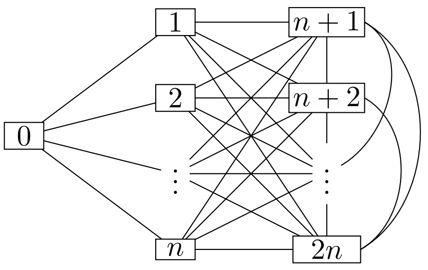
Answer: At each iteration, we should replace A by $A \cap N(\{x\})$ instead, so that all vertices we add in future iterations are adjacent to x.

What's more, the output I is guaranteed to be a *maximal* independent set (following the maximal/maximum distinction introduced in Chapter 13). The set I is not necessarily the largest independent set, but there is no bigger independent set I' that contains all of I. This is true because every vertex $y \in V(G)$ is removed from A in some iteration, for one of two reasons:

1. y is equal to x, the vertex added to I in that iteration.



(a) A star graph



(b) Extending the star graph

Figure 18.3: Graphs which confuse the greedy independent set algorithm

2. y is adjacent to x.

In both cases, $I \cup \{y\}$ is not a bigger independent set. In the first case, that's because $I \cup \{y\}$ is the same set as I. In the second case, that's because $I \cup \{y\}$ is not an independent set; it contains the edge xy.

This proves that I is a maximal independent set. However, I is not guaranteed to be a maximum independent set: there may well be other independent sets which are larger.

Question: Can you find a graph in which the set I we find is much smaller than another independent set?

Answer: One of many examples is the graph shown in Figure 18.3a: a copy of the star graph S_{n+1}. If the algorithm adds vertex 0 to I, it will immediately get $A = \varnothing$ and stop; however, $I = \{0\}$ is much smaller than the independent set $\{1, 2, 3, \dots, n\}$.

We might say: okay, the problem is clear. We chose a vertex with very high degree and removed all its neighbors from A, causing the algorithm to stop very quickly. What if we try to be smarter about step 1 of the algorithm, so that at each iteration we choose the vertex x with the fewest neighbors in A?

This might sometimes be a useful heuristic, but it is not guaranteed to work. Consider instead the graph in Figure 18.3b; here, we've added vertices $n+1, n+2, \dots, 2n$, adjacent to vertices $1, 2, \dots, n$ and to each other.

Question: What will the greedy algorithm do, given the graph in Figure 18.3b, if it chooses the vertex with the fewest neighbors in A at each step?

Answer: The first vertex chosen will still be vertex 0, because it has degree n and all other vertices have degree at least $n+1$. Then, vertices $1, \dots, n$ will be deleted, and the second vertex will be one of $n+1, \dots, 2n$. Then, all remaining vertices will be deleted, and the algorithm will find a 2-vertex independent set.

Question: What is the largest independent set in the graph in Figure 18.3b?

Answer: The set $\{1, 2, \dots, n\}$, which is $\frac{n}{2}$ times bigger than the set found by the greedy algorithm.

There is no rule for choosing vertices in the greedy algorithm that's always guaranteed to do well. However, at the very least, we can use the greedy algorithm to prove worst-case bounds.

Proposition 18.1. *If G is an n-vertex graph with maximum degree $\Delta(G)$, then $\alpha(G) \geq \frac{n}{\Delta(G)+1}$.*

Proof. To prove the lower bound, we find an independent set in G using the greedy algorithm, taking no special care in choosing the vertex x at each iteration.

Nevertheless, what we can guarantee is that at each iteration, at most $\Delta(G) + 1$ vertices are removed from A. We remove x and all neighbors of the chosen vertex x in A. There are at most $\Delta(G)$ such neighbors; there could be fewer if $\deg_G(x) < \Delta(G)$, or if not all neighbors of x are still in A, but there cannot be more.

The algorithm does not stop until A is empty, but since each iteration removes at most $\Delta(G)+1$ out of n vertices from A, there must be at least $\frac{n}{\Delta(G)+1}$ iterations. At each iteration, one vertex is added to I; therefore, the output of the algorithm is an independent set I with at least $\frac{n}{\Delta(G)+1}$ vertices. Therefore $\alpha(G)$ is also at least $\frac{n}{\Delta(G)+1}$. □

Question: What lower bound can we get for the clique number $\omega(G)$?

Answer: Since $\omega(G) = \alpha(\overline{G})$, we have

$$\omega(G) \geq \frac{n}{\Delta(\overline{G}) + 1} = \frac{n}{n - \delta(G)},$$

where $\delta(G)$ is the minimum degree of G.

Here, the identity $\Delta(\overline{G}) = n - 1 - \delta(G)$ holds because a maximum-degree vertex in $\overline{G}$ is a minimum-degree vertex in G, and because $\deg_{\overline{G}}(x) = n - 1 - \deg_G(x)$ for any vertex x.

18.4 An indecisive algorithm

In preparation for our next topic, consider a variant of the greedy algorithm that hasn't made up its mind yet about whether it wants to find a clique or an independent set. It will still keep track of a set A of available vertices, initially equal to $V(G)$. It will also build up a set we'll simply call S, initially equal to $\varnothing$; at each iteration, it will move a vertex x from A to S, then update A.

The upside to being indecisive is that at each iteration, the algorithm has a choice:

- It could remove x and all of its neighbors from A.
- It could remove x and all vertices which are not its neighbors from A.

If the algorithm always chooses the option that keeps A as large as possible, then it's guaranteed to keep at least $\frac{|A|-1}{2}$ vertices for the next iteration (rounded up). It must always remove x, but it can keep at least half of the remaining vertices.

The downside to being indecisive is that the final set S will not be either a clique or an independent set. Suppose that after k iterations, we obtain $S = \{x_1, x_2, \dots, x_k\}$, where x_i is the vertex added to S at the i^{th} iteration.

Question: Is there any useful property at all that this set S has?

Answer: We can at least guarantee that a vertex x_i is either adjacent to all of the vertices $x_{i+1}, x_{i+2}, \dots, x_k$, or to none of them. The first case occurs when we removed all the non-neighbors of x_i from A in the i^{th} iteration; the second case occurs when we removed all the neighbors of x_i from A in the i^{th} iteration.

This property seems hard to make use of, but let's try. Suppose we call a vertex x_i a "Q-type" vertex (Q for clique) if it is adjacent to all of $x_{i+1}, x_{i+2}, \dots, x_k$, and a "I-type" vertex (I for independent set) if it is adjacent to none of these vertices. We can write S as the union $Q \cup I$, where Q is the set of all Q-type vertices and I is the set of all I-type vertices.

Question: What can we say about Q and I?

Answer: Q is a clique and I is an independent set.

Question: What is the type of vertex x_k?

Answer: It is both Q-type and I-type, since there are no vertices that come after it, so both conditions are vacuously true.

Except for x_k, every vertex is only one of Q-type or I-type, so $|Q| + |I| = |S| + 1$. Therefore, if the indecisive algorithm only picks at the last possible moment whether to return the clique Q or the independent set I, it is guaranteed to find a set of $\frac{|S|+1}{2}$ vertices.

The indecisive algorithm is not usually useful in practice: for most problems, we have already decided whether we are looking for a clique or an independent set. However, we can use it to prove a relationship between these quantities.

Proposition 18.2. *If G is a graph with at least 2^{n-2} vertices, then $\alpha(G) + \omega(G) \ge n$. In other words, $\alpha(G) + \omega(G) \ge 2 + \log_2 |V(G)|$.*

Proof. Apply the indecisive algorithm to G. At the beginning, $|A| \ge 2^{n-2}$. After one iteration, we replace A by a set with at least $\frac{|A|-1}{2}$ elements in it, so we can say that $|A| \ge \frac{2^{n-2}-1}{2}$; rounding up, $|A| \ge 2^{n-3}$. Similarly, after a second iteration, we can still guarantee $|A| \ge 2^{n-4}$. This argument can be repeated for as long as our lower bound on $|A|$ is even, ensuring that $\frac{|A|-1}{2}$ can be rounded up to $\frac{|A|}{2}$.

After the $(n-2)^{\text{th}}$ iteration, we can guarantee that $|A| \geq 2^{n-(n-2)-2}$, or $|A| \geq 1$. This is still enough to know that there will be an $(n-1)^{\text{th}}$ iteration. However, it's possible that the $(n-1)^{\text{th}}$ iteration is the last, because it removes the only remaining element in A.

At the end, the indecisive algorithm chooses between returning a clique Q or an independent set I, which satisfy $|Q| + |I| = |S| + 1$. Since $|S| \geq n-1$, we conclude that $|Q| + |I| \geq n$. This completes the proof, since $\alpha(G)$ is at least $|I|$ and $\omega(G)$ is at least $|Q|$: the largest clique and independent set are at least as large as the ones we found. □

Intuitively, we can make the independence number $\alpha(G)$ small by giving G lots of edges, and we can make the clique number $\omega(G)$ small by giving G few edges. However, these strategies directly oppose each other, so it makes sense that we wouldn't be able to keep both $\alpha(G)$ and $\omega(G)$ from growing. Proposition 18.2 is the first result we will prove that quantifies this relationship.

18.5 Ramsey numbers

We can improve on Proposition 18.2 by noticing a flaw in how the indecisive algorithm chooses which vertices to keep in A. All it does is pick the option that keeps A as large as possible, but if you want the final clique or independent set to be as large as possible, that's not always correct.

Question: Why would you ever decide to keep a smaller set to be set A in the next iteration?

Answer: Suppose that the current set S contains many Q-type vertices and few I-type vertices. Then you might keep all the neighbors of x in A, making x another Q-type vertex, even if the algorithm stops a bit sooner as a result.

To be strategic about it, we need to quantify the exact trade-off: how many fewer vertices in A are an acceptable price to pay in exchange for guaranteeing a vertex of the right type? This depends on our goals, of course. If we'd be happy with a clique of size 100 and we already have 98 Q-type vertices, we should make x another Q-type vertex at almost any cost—even if x only has one or two neighbors in A. If we're further from our goal, we should be more conservative.

That's still vague; to make it more concrete, we need to know exactly how many vertices in A are necessary to reach any particular goal we might have. So for integers $k \geq 1$ and $l \geq 1$, we define the *Ramsey number* $R(k, l)$ to be the least integer n such that any n-vertex graph G satisfies either $\alpha(G) \geq k$ or $\omega(G) \geq l$.

Question: Can we be certain that any such integer n exists?

Answer: Yes, by Proposition 18.2. Take a graph G with $n = 2^{k+l-3}$ vertices; then $\alpha(G)+\omega(G) \geq k+l-1$, which cannot be true if both $\alpha(G) \leq k-1$ and $\omega(G) \leq l-1$. Therefore 2^{k+l-3} vertices are enough, and $R(k, l)$ is at most this big.

Ramsey numbers are named after Frank Ramsey, who was the first to prove a result analogous to Proposition 18.2 in 1930 [23], though he was not interested in concrete bounds on the necessary number of vertices.

Let me describe a concrete situation to motivate our need for Ramsey numbers. Suppose we are looking for a clique or an independent set of size 5. Currently, we have 3 vertices in S: two Q-type vertices (adjacent to each other and all vertices in A) and one I-type vertex (with no neighbors in A). We've chosen a vertex x in the fourth iteration, and we're considering what to do with the neighbors of x to prepare for the next iteration.

1. If we keep all the neighbors of x in A, then x will be another Q-type vertex. We will need to find either two more Q-type vertices or four more I-type vertices in the future.
2. If we keep all the non-neighbors of x in A, then x will be another I-type vertex. We will need to find either three more Q-type vertices or three more I-type vertices in the future.

Question:	How should we choose between these two options?
Answer:	If x has at least $R(4,2)$ neighbors in A, then the first option is guaranteed to succeed. If x has at least $R(3,3)$ non-neighbors in A, then the second option is guaranteed to succeed. If neither is true, then both options are risky, and we might need to guess.

Some people are very comfortable with algorithms; some people prefer more mathematical arguments. In case you are the second type of person, you will be happier with the following lemma, whose proof is inspired by the indecisive algorithm but doesn't use it directly.

Lemma 18.3. *For all integers $k \geq 2$ and $l \geq 2$, the Ramsey number $R(k,l)$ is at most the sum $R(k,l-1) + R(k-1,l)$.*

Proof. Let G be an arbitrary graph with $R(k-1,l) + R(k,l-1)$ vertices. Our goal is to prove that this is enough to guarantee that either $\alpha(G) \geq k$ or $\omega(G) \geq l$.

Choose any vertex $x \in V(G)$, and divide $V(G) - \{x\}$ into two sets: A_1, the vertices adjacent to x, and A_2, the vertices not adjacent to x.

It is impossible that both $|A_1| < R(k,l-1)$ and that $|A_2| < R(k-1,l)$. In that case, $|V(G)| = |A_1|+|A_2|+1$ would be at most $(R(k,l-1)-1)+(R(k-1,l)-1)+1$ or $R(k-1,l)+R(k,l-1)-1$: one less than the number of vertices we defined G to have.

If $|A_1| \geq R(k,l-1)$, then the induced subgraph $G[A_1]$ is big enough to either contain a k-vertex independent set I or an $(l-1)$-vertex clique Q, by the definition of the Ramsey number $R(k,l-1)$. Therefore G contains either a k-vertex independent set I or an l-vertex clique $Q \cup \{x\}$ (since x is adjacent to all vertices in A_1, and in particular all vertices in Q).

If instead $|A_2| \geq R(k-1,l)$, then $G[A_2]$ is big enough to either contain a $(k-1)$-vertex independent set I or an l-vertex clique Q. Therefore G contains either a k-vertex independent set $I \cup \{x\}$ or an l-vertex clique Q.

In all cases, we either find a k-vertex independent set or and l-vertex clique in G. Since G was chosen to be an arbitrary graph with $R(k-1,l) + R(k,l-1)$ vertices, this proves that $R(k,l) \leq R(k-1,l) + R(k,l-1)$. □

Lemma 18.3 is the foundation of a recurrence that can be used to get upper bounds on the Ramsey numbers. To use the recurrence, we also need base cases.

Question:	Lemma 18.3 starts at $k \geq 2$ and $l \geq 2$. What is $R(k, l)$ when $k = 1$ or when $l = 1$?
Answer:	In all these cases, it is just 1; even something like $R(1, 10000)$ is 1. There is only one possible 1-vertex graph, and it has both a 1-vertex clique and a 1-vertex independent set.

Question:	$R(k, 2)$ and $R(2, l)$ can also be computed exactly; what are they?
Answer:	$R(k, 2) = k$ and $R(2, l) = l$. For the first of these, suppose we have a k-vertex graph. Either it has no edges (and there is a k-vertex independent set) or it has an edge xy (and $\{x, y\}$ is a 2-vertex clique). The argument for $R(2, l)$ is similar.

The general upper bound on $R(k, l)$ using this recurrence is sometimes also called Ramsey's theorem, but actually, it was only proven in 1955 by Robert Greenwood and Andrew Gleason [11] (who also proved Lemma 18.3 in the process).

Theorem 18.4. *For all integers $k \geq 1$ and $l \geq 1$, $R(k, l) \leq \binom{k+l-2}{k-1}$.*

Proof. We induct on $k + l$. (That is, if $P(n)$ is the statement that the theorem holds for all integers $k \geq 1$ and $l \geq 1$ with $k + l = n$, then we prove $P(n)$ for all $n \geq 1$ by induction on n.)

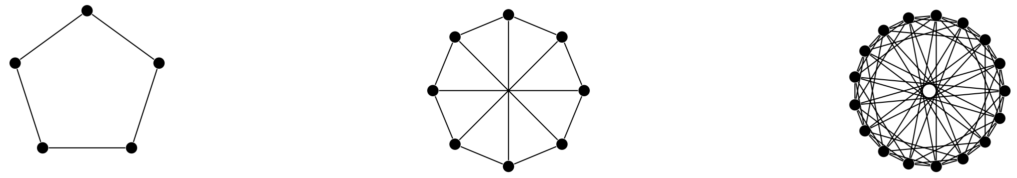
As our base case, we can take any of $k + l = 1$ (where there is nothing to prove), or $k + l = 2$ (where $R(1, 1) = 1 = \binom{0}{0}$), or $k + l = 3$ (where $R(2, 1) = 1 = \binom{1}{0}$ and $R(1, 2) = 1 = \binom{1}{1}$), but eventually we will have to induct.

Consider some $k \geq 1$ and $l \geq 1$ with $k + l \geq 3$, and assume the following induction hypothesis: for all $k' \geq 1$ and $l' \geq 1$ with $k' + l' = k + l - 1$, we have the upper bound $R(k', l') \leq \binom{k'+l'-2}{k'-1}$. Actually, we only need it for two values of k' and l': the ones that appear in the right-hand side of Lemma 18.3.

Question:	If $k = 1$ or $l = 1$, we cannot apply Lemma 18.3; what do we do?
Answer:	If $k = 1$ or $l = 1$, we know that $R(k, l) = 1$, and in both cases the upper bound $\binom{k+l-2}{k-1}$ simplifies to $\binom{l-1}{0} = 1$ or $\binom{k-1}{k-1} = 1$, so we do not even need the induction hypothesis.

Otherwise, $k \geq 2$ and $l \geq 2$. By Lemma 18.3, $R(k, l) \leq R(k, l - 1) + R(k - 1, l)$, and both of the Ramsey numbers on the right-hand side are suitable for the induction hypothesis. We can conclude that

$$R(k, l) \leq \binom{k + (l-1) - 2}{k-1} + \binom{(k-1) + l - 2}{(k-1) - 1} = \binom{k + l - 3}{k-1} + \binom{k + l - 3}{k-2}.$$



(a) The cycle graph C_5 (b) $\mathrm{Ci}_8(1,4)$ (c) $\mathrm{Ci}_{17}(1,2,4,8)$

Figure 18.4: Lower bounds for a few small Ramsey numbers

The right-hand side of this inequality simplifies directly to $\binom{k+l-2}{k-1}$. This is called Pascal's identity, and if you haven't seen it before, there is a short combinatorial argument. By its combinatorial definition, $\binom{k+l-2}{k-1}$ counts the $(k-1)$-element subsets $S \subseteq \{1, 2, \dots, k+l-2\}$. These come in two types. If $k+l-2 \notin S$, then S is a subset of $\{1, 2, \dots, k+l-3\}$, and so there are $\binom{k+l-3}{k-1}$ cases of this type. If $k+l-2 \notin S$, then $S - \{k+l-2\}$ is a $(k-2)$-element subset of $\{1, 2, \dots, k+l-3\}$, and so there are $\binom{k+l-3}{k-2}$ cases of this type.

Altogether, there must be $\binom{k+l-3}{k-1} + \binom{k+l-3}{k-2}$ subsets. We already know there are $\binom{k+l-2}{k-1}$ subsets, so these two expressions must be equal. Returning to our upper bound on $R(k,l)$, we can replace one expression by the other and get $R(k,l) \le \binom{k+l-2}{k-1}$. This proves the induction step, completing the proof of the theorem. □

Both Ramsey and later Greenwood and Gleason also considered a generalized problem. Here, we define the Ramsey number $R(k_1, k_2, \dots, k_m)$ to be the least n with the following property: whenever we write the complete graph K_n as the union of n-vertex graphs $G_1 \cup G_2 \cup \dots \cup G_m$, there will be some G_i such that $\omega(G_i) \ge k_i$. Sometimes, this is phrased as coloring the edges of K_n using m different colors, but I will avoid this so that you don't confuse it with all the other ways we will color a graph.

18.6 Lower bounds

Once we have defined Ramsey numbers, it is tempting to try to figure out an exact formula for them, or at least an approximate one. Our first upper bound on $R(k,l)$ was 2^{k+l-3}, and our second bound of $\binom{k+l-2}{k-1}$ has a similar rate of growth, at least when $k = l$. Is this anywhere close to the truth? How would we know?

In the definition of the Ramsey number $R(k,l)$, we ask for the least n such that every n-vertex graph satisfies a condition. To prove that $R(k,l) \ge n$, it is enough to find a single $(n-1)$-vertex graph which does not yet satisfy the condition. (Such graphs are sometimes called Ramsey graphs, because of their use in proving lower bounds for Ramsey numbers.)

Figure 18.4 shows a few of the graphs we need.

- The cycle graph C_5, shown in Figure 18.4a, has no 3-vertex clique or 3-vertex independent set. This proves that $R(3,3) \ge 6$: we need at least 6 vertices to guarantee one of these objects, because 5 are not enough.

  In fact, Theorem 18.4 proves an upper bound of $\binom{3+3-2}{3-1} = \binom{4}{2} = 6$ on $R(3,3)$, so we know that $R(3,3) = 6$.

- The circulant graph $\mathrm{Ci}_8(1,4)$ shown in Figure 18.4b has no 3-vertex clique or 4-vertex independent set, and it has 8 vertices, so it proves that $R(4,3) \ge 9$. Theorem 18.4 only proves that $R(4,3) \le \binom{4+3-2}{4-1} = \binom{5}{3} = 10$, but in one of the practice problems, I will ask you to improve this upper bound: in fact, $R(4,3) = 9$.

Question: What about $R(3,4)$?

Answer: It is also equal to 9; in fact, $R(k,l)$ and $R(l,k)$ are always equal. The lower bound that proves $R(3,4) \ge 9$ is the complement of $\mathrm{Ci}_8(1,4)$: the circulant graph $\mathrm{Ci}_8(2,3)$. A similar trick of complements can be used to show that every n-vertex graph contains a k-vertex independent set or an l-vertex clique, then it is also true that every n-vertex graph contains an l-vertex independent set or a k-vertex clique.

- Taking $R(3,4) = R(4,3) = 9$ for granted, it follows from Lemma 18.3 that $R(4,4) \le R(4,3) + R(3,4) = 18$. In fact, $R(4,4) = 18$; to prove this, we need a 17-vertex graph.

  The graph in Figure 18.4c is one such graph: the circulant graph $\mathrm{Ci}_{17}(1,2,4,8)$. This graph has no 4-vertex clique or 4-vertex independent set, so it proves the lower bound $R(4,4) \ge 18$.

Of course, we cannot go on forever like this; to give a general lower bound, some general scheme for constructing these graphs is needed. Unfortunately, it has been very difficult to construct these graphs. We run into two kinds of problems. First, most simple rules defining an arbitrarily large graph will give it a large clique or a large independent set. Second, when we eventually do find a general construction that looks promising, we often find out that proving something about its cliques or independent sets turns into an unsolved problem in number theory.

(Why number theory? Well, the offsets of the circulant graph $\mathrm{Ci}_{17}(1,2,4,8)$ in Figure 18.4c are not arbitrary: they are the quadratic residues modulo 17. That is, every perfect square is congruent modulo 17 to one of ±1, ±2, ±4, or ±8. Graphs constructed in this way are called Paley graphs, and they are often useful for proving lower bounds on Ramsey numbers; however, we don't understand quadratic residues well enough to say what the lower bounds are in general.)

The best we've been able to do is to give up and pick a graph at random, showing that the result is very unlikely to have a large clique or a large independent set. Let me first illustrate this with concrete numbers. Suppose we define a graph G on 1000000 vertices by flipping a coin for every possible edge to see if it's present or absent. What are the odds that this random graph G will have a 40-vertex clique?

There are $\binom{1000000}{40} \approx 1.22 \times 10^{192}$ ways to choose 40 of the vertices of G. Each one of those 40-vertex sets could in theory be a 40-vertex clique… but that's very unlikely. There are $\binom{40}{2} = 780$ edges between those 40 vertices, so we flip 780 coins to decide which edges between the vertices are present. In order for us to get a clique, all the coin flips have to go one way, which has a probability of $2^{-780} \approx 1.57 \times 10^{-235}$.

If you bought $\binom{1000000}{40}$ tickets for a lottery that chose one winning ticket out of 2^{780}, then to find your total chances of winning, it would be enough to multiply these two numbers together. This would give us $(1.22 \times 10^{192}) \cdot (1.57 \times 10^{-235})$ or about 1.93×10^{-43}. The clique problem is actually even worse, due to overlaps. After we consider the first clique, the second doesn't contribute its whole 2^{-780} probability, because some of those cases were already counted: the two cliques appear simultaneously!

Rather than consider what the situation with overlaps is, we can simply say that the probability of having a 40-vertex clique is at most 1.93×10^{-43}. We get the same probability for a 40-vertex independent set: once again, all the coin flips have to go one way for all 780 edges between vertices of a set. Since both probabilities are tiny, we conclude that this random 1000000-vertex graph is almost certain not to have a 40-vertex clique or 40-vertex independent set, proving that $R(40, 40) > 1000000$.

In 1947, Pál Erdős proved [10] that this gives an exponential lower bound on the Ramsey numbers:

Theorem 18.5. *For all $k \geq 3$, $R(k, k) > 2^{k/2}$.*

Proof. Pick a random graph on $n = 2^{k/2}$ vertices by flipping a coin for every edge. There are $\binom{n}{k}$ possible k-vertex sets; each has a $2^{-\binom{k}{2}}$ chance of being a clique, and another $2^{-\binom{k}{2}}$ chance of being an independent set, for a total probability of $2^{1-\binom{k}{2}}$ of doing anything of interest.

As before, we get an upper bound on the probability that any of these cliques or independent sets are created by multiplying these quantities together: $\binom{n}{k} \cdot 2^{1-\binom{k}{2}}$. It is always true that $\binom{n}{k} \leq \frac{n^k}{k!}$, and for $k \geq 3$, it is true that $k! > 2^k$. Applying both inequalities, our upper bound becomes $n^k \cdot 2^{-k} \cdot 2^{1-k^2/2+k/2}$ or $2^{1-k/2}$, which is less than 1 as soon as $k \geq 3$.

Therefore the probability that the random graph doesn't do what we want is less than 1: some of the n-vertex graphs we could get have neither k-vertex cliques nor k-vertex independent sets. This shows that $R(k, k) > n$. □

The bounds $R(k, k) > 2^{k/2}$ and $R(k, k) \leq \binom{2k-2}{k-1}$ (which grows roughly as 2^{2k}) are fairly far apart, though they are both exponential; for example, they tell us that $32 < R(10, 10) \leq 48620$. It has been at least 70 years since any of the results mentioned in this chapter so far; has there been improvement since then?

There has been and there still is a lot of work done on specific small Ramsey numbers. A dynamic survey of the best-known bounds is maintained by Stanisław Radziszowski [22]. For reference, at the time of writing, it gives the bounds $798 \leq R(10, 10) \leq 16064$: much better than 32 and 48620.

Improvements to general bounds on $R(k, k)$ have also been made. When I first taught graph theory, I had to conclude this topic by saying that the bases of the exponents in the bounds ($\sqrt{2}$ for the lower bound and 4 in the upper bound) are still untouched, with improvements only in polynomial factors. As of 2023, that is no longer true: a paper by Campos, Griffiths, Morris, and Sahasrabudhe [4] followed by another from Gupta, Ndiaye, Norin, and Wei [14] have brought down the upper bound to roughly $R(k, k) \leq 3.8^k$.

18.7 Practice problems

1. Consider the graph whose vertices are the ordered pairs (x, y) where $1 \le x \le a$ and $1 \le y \le b$, and where two vertices (x, y) and (x', y') are adjacent whenever $x \ne x'$. (This is a complete a-partite graph where each part has size b.)

   a) What is the clique number of this graph? Describe what the largest cliques look like.

   b) What is the independence number of this graph? Describe what the largest independent sets look like.

2. Let G be the graph with vertex set $\{2, 3, \dots, 100\}$ in which two vertices are adjacent if one is divisible by the other.

   a) What is the independent set in G found by a greedy algorithm which looks at all the vertices of G in ascending order? (And why is this a uniquely bad order in which to look at the vertices?)

   b) Find the independence number of G.

   c) Find the clique number of G.

3. Take an $n \times n$ grid and number the rows and columns 1 through n. Then fill the grid as follows: in the entry in row i and column j, write the remainder when $i + 2j$ is divided by n: a number between 0 and $n - 1$. (This is the construction found by Hedayat and Federer in [16].)

   Prove that that when n is not divisible by 2 or 3, these labels form a Knut Vik design: if we pick any label between 0 and $n - 1$ and place a queen on all entries with that label, then this is a set of n non-attacking queens, even if diagonals wrap around.

4. a) Verify that the graphs in Figure 18.4 really do prove the lower bounds $R(3, 3) \ge 6$, $R(4, 3) \ge 9$, and $R(4, 4) \ge 18$, by checking that they have no cliques or independent sets of the relevant sizes.

   b) Prove that $R(4, 3) \le 9$ by contradiction, supposing there is a 9-vertex graph with no vertex x that has $R(4, 2)$ neighbors or $R(3, 3)$ non-neighbors. What would the degree sequence of this graph be?

      If you prefer, prove the generalization: when $R(k, l - 1)$ and $R(k - 1, l)$ are both even, then Lemma 18.3 can be improved to $R(k, l) \le R(k, l - 1) + R(k - 1, l) - 1$.

5. In terms of n, find the minimum and maximum number of edges in a $3n$-vertex graph with independence number $2n$.

6. In this problem, you will see one possible deterministic alternative to Theorem 18.5, and see how it compares to the random construction.

   Let G_1 be the 5-cycle C_5. Then, for each $k > 1$, construct G_k by starting with G_{k-1}, and then replacing

   - Each vertex x of G_{k-1} by five vertices x_1, x_2, x_3, x_4, x_5 with a 5-cycle through them;
   - Each edge xy of G_{k-1} by 25 edges $x_i y_j$ for $1 \le i, j \le 5$.

In other words, we replace the vertices of G_{k-1} by copies of C_5.

a) Prove that $\alpha(G_2) = \omega(G_2) = 4$.

b) Prove that $\alpha(G_k) = \omega(G_k) = 2^k$. (Induct on k.)

c) We get a lower bound $R(17, 17) > n$ by finding an n-vertex graph G with $\alpha(G) \leq 16$ and $\omega(G) \leq 16$. What lower bound does the construction in this problem give for $R(17, 17)$, and how does it compare to the lower bound from Theorem 18.5?

   What if we try to bound $R(33, 33)$ instead?

7. An open problem called Conway's 99-graph problem, already mentioned in Chapter 4, is to determine whether there is a 99-vertex graph with the following properties:

   - Every two adjacent vertices have exactly one common neighbor;
   - Every two non-adjacent vertices have exactly two common neighbors.

   If such a graph exists, it is known (spoilers for the previous practice problem) that it is 14-regular.

   Suppose G is a 99-vertex graph satisfying Conway's conditions. What is the best upper bound you can prove on $\alpha(G)$?

   (When I tried solving this problem, I found three arguments, giving upper bounds of 43, 33, and 22. See how well you can do!)

8. (APMO 2003) Given two positive integers m and n, find the smallest positive integer k such that among any k people, either there are $2m$ of them who form m pairs of mutually acquainted people or there are $2n$ of them forming n pairs of mutually unacquainted people.

   In other words, find the smallest k such that for any k-vertex graph G, either $\alpha'(G) \geq m$ or $\alpha'(\overline{G}) \geq n$, where $\overline{G}$ is the complement of G.

9. This question is more of a puzzle than a mathematical problem. If 8 queens are placed on the chessboard as in Figure 18.1a, there is a closed knight's tour of just the 56 unoccupied squares. Can you find it?

   (Up to symmetry, this is the only solution to the 8 queens puzzle for which such a tour exists. Can you guess what goes wrong with the other solutions?)

19 Graph coloring

The purpose of this chapter

Graph coloring is one of the iconic problems in graph theory, though usually people think of it in the context of coloring maps. This application was mentioned in Chapter 1, and we will see it in detail later, in Chapter 24. The applications you will see in this chapter are more metaphorical versions of coloring—but that's par for the course in graph theory, which is all about representing the world with metaphorical points and metaphorical lines.

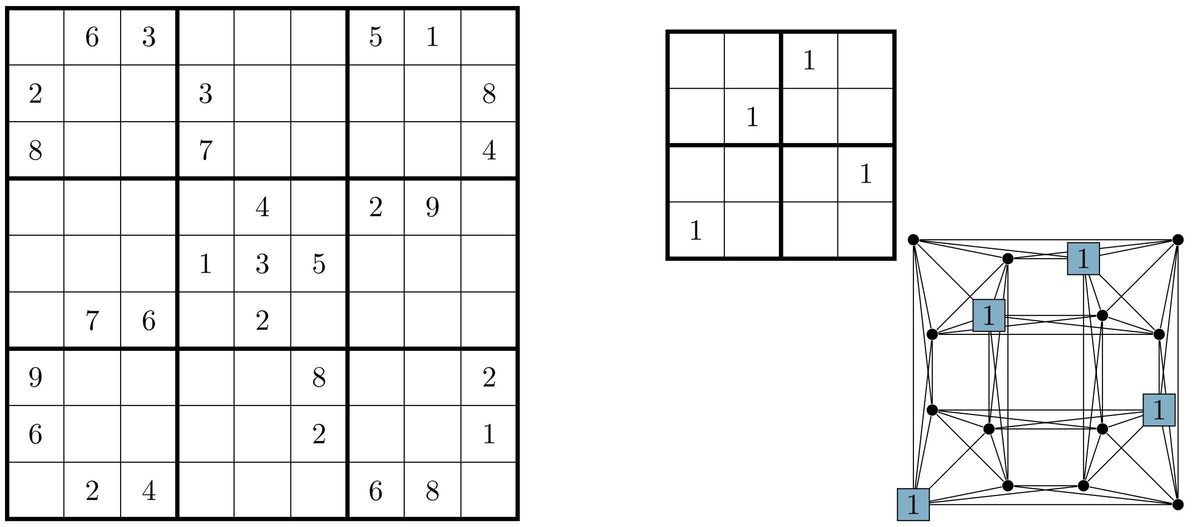
I begin this chapter with one upper bound and two lower bounds on chromatic number, and I think these are the basics you should know of the theory of graph coloring. Meanwhile, Theorem 19.4 and Theorem 19.7 demonstrate how these basics can be used. The motivation for these specific demonstrations is to show you an example where the clique bound of Proposition 19.2 is very useful, and an example where it's not at all useful; in either case, I hope that it gives you a sense of why the clique number does not necessarily tell us the chromatic number.

Brooks' theorem (which describes the conditions under which $\chi(G) \le \Delta(G)$) is a common result to cover in an introductory graph theory class, but I've left it out. I have done this because I think seeing it for the first time at this point is extraordinarily uncompelling—and I say this as someone who has both used the theorem in research and taught a week-long class about variants of the theorem. When you just start out learning about graph coloring, it is unsatisfying to go to great effort to improve Proposition 19.1 by a tiny amount. I recommend seeing Brooks' theorem for the first time as a more experienced graph theorist, for example by reading "Brooks' Theorem and Beyond" by Cranston and Rabern [7], which presents a variety of proofs of the theorem and its extensions.

19.1 Graph coloring and Sudoku

In case you have never encountered a Sudoku puzzle, let me explain how they work. Figure 19.1a shows an example of a Sudoku puzzle. It is a 9×9 grid, with its 81 cells grouped together into 9 boxes. Some of the cells already have numbers in them. The objective of the puzzle is to figure out which numbers should go in the empty cells! The rule is that in each row, each column, and each 3×3 box, the nine cells should contain the numbers 1 through 9 in some order.

For example, the top left corner of Figure 19.1a could not contain the numbers $2, 3, 6, 8$, because they are already present in its 3×3 box; it could also not contain the numbers $1, 5$ (already present in its row) or 9 (already present in its column). The numbers 4 or 7 have not been ruled out, though only one of them is correct; the other would eventually lead to a contradiction if you wrote it in, even though there's no obvious problem yet.



	6	3				5	1	
2			3					8
8			7					4
				4		2	9	
			1	3	5			
	7	6		2				
9					8			2
6					2			1
	2	4				6	8	

(a) A Sudoku puzzle

		1	
	1		
			1
1			

(b) The 4 × 4 Sudoku graph

Figure 19.1: Sudoku puzzles and graph coloring

As usual, we would like to turn this logic puzzle into a problem of graph theory. The first task is to choose a graph to model it with. Although other options exist, for us it will be most convenient to make the 81 cells in the grid be the vertices of the Sudoku graph. Make two cells adjacent in this graph if they are in the same row, the same column, or the same 3×3 box.

Question: How can the rules of Sudoku be expressed in terms of the edges of this graph?

Answer: The goal of Sudoku is now to assign one of the numbers 1 through 9 to each vertex of the graph, so that adjacent vertices are assigned different numbers.

Since a diagram of this Sudoku graph would look like utter chaos and not give you any intuition at all, I have drawn a diagram of the 4×4 Sudoku graph in Figure 19.1b. In 4×4 Sudoku, the rules are the same, except that each row, column, and 2×2 box has four numbers in it, and the cells are allowed to contain the numbers 1 through 4.

Question: In Figure 19.1b, the 1's are all entered into the grid (and shown in the graph). In general, what can you say about the set of all vertices assigned the number 1 in a Sudoku puzzle?

Answer: Two adjacent vertices cannot both be assigned the number 1, so the set of vertices with a 1 must be an independent set.

The graph-theoretic rules of Sudoku happen to match a well-studied problem in graph theory, which is why I used Sudoku as an example to begin this chapter. For historical reasons, rather than assigning every vertex a number, we assign every vertex a color. For practical reasons, graph theorists are very flexible about what "colors" are. It's fine to say, "We will use the set of colors $\{1, 2, \ldots, 9\}$," for example, in which case the numbers in a Sudoku grid are our colors.

Definition 19.1. *A* ***proper coloring*** *(or just* ***coloring****) of a graph G is a function $f\colon V(G) \to C$ such that for all edges $xy \in E(G)$, $f(x) \neq f(y)$. The elements of C are called colors, but can be any set we like.*

Allow me to justify my terminology. The term "proper coloring" is the formal name for a coloring f with the condition that $f(x) \neq f(y)$ for all edges xy, so that's what we should be using. However, almost every single time we color something in this book, we will want a proper coloring. To omit needless words, I will simply say "coloring" unless emphasis is needed: for example, when we want to check that a coloring f really is proper. On the rare occasion where we want a coloring that might give the same color to both endpoints of an edge, I will call it an *improper coloring* to emphasize this.

Back to graph theory! Sometimes it is convenient to think about coloring a graph in a different way: not as a function, but as a partition. We say that a **color class** of a coloring is a set of of all vertices of some color. Every vertex is in exactly one color class, so the color classes are a partition of $V(G)$; a coloring is proper if and only if the endpoints of every edge are in different color classes.

Question: What are the color classes in the solution to a Sudoku puzzle?

Answer: There are 9 of them: for each number from 1 to 9, the set of all cells containing that number is a color class.

The study of graph coloring originated with a very practical coloring problem: given a map divided into several regions, how can we color the regions so that neighboring regions receive different colors? This is a coloring of the *graph of adjacencies* of the map: its vertices are the regions, with an edge between every pair of regions that share a border. We will not investigate this problem in detail in this chapter; we will return to it in Chapter 24, once we know more properties of graphs we can obtain from a map.

One thing is still missing from our definition, however. It is too easy to color a graph: just give every vertex its own color! We are not very happy with this solution, for the same reason that Sudoku puzzles would not be very fun if you could solve them by writing the numbers $1, 2, \ldots, 81$ in the squares in any order. We are more interested in a coloring of a graph if it uses fewer colors; here are some definitions to express this.

Definition 19.2. *A k-coloring of G is a coloring of G in which the set of colors has size k. A graph which has a (proper) k-coloring is called* ***k-colorable****.*

The ***chromatic number*** *$\chi(G)$ is the least value of k for which G is k-colorable.*

Question: The Sudoku graph is 9-colorable; is it 8-colorable?

Answer: No: all 9 vertices in a row (or column, or box) must use different colors, so at least 9 vertices are necessary. This is because the vertices in a row, column, or box form a clique.

Finding the chromatic number $\chi(G)$ is an optimization problem, like many we've seen before, and there are several observations we can make which are common to all optimization problems. (See Appendix A for more details.) Determining that $\chi(G) = k$ always comes down to two steps: we must show that $\chi(G) \leq k$, and that $\chi(G) \geq k$. These look like similar inequalities, but they're very different in practice.

- Every coloring of a graph G gives us an upper bound on $\chi(G)$: a k-coloring of G tells us that $\chi(G) \leq k$. A single example is a proof; it might be hard to find, but it is short.
- To prove a lower bound $\chi(G) \leq k$, we must show that no $(k-1)$-colorings of G are possible. Unless we think of something clever, we must resort to an exhaustive search.

We will see arguments for both lower and upper bounds in this chapter; however, the graph coloring problem is a hard one in general, so we will not always be able to make the bounds meet.

Question: For the clique number $\omega(G)$ and the independence number $\alpha(G)$, this is reversed: lower bounds are proved by example, and upper bounds need an exhaustive argument. Why?

Answer: Both $\omega(G)$ and $\alpha(G)$ are maximization problems: we are trying to find the largest clique or independent set. When coloring a graph, we are trying to use the smallest set of colors: $\chi(G)$ is a minimization problem.

That being said, the special case of 2-coloring is much easier than the general case, and we have already encountered it in this book.

Question: What name do we already have for 2-colorable graphs?

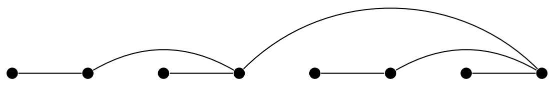
Answer: They are precisely the bipartite graphs! The two color classes in a 2-coloring are precisely the two sides of a bipartition.

Intuitively, finding a 2-coloring is easier because, as soon as you give a vertex a color, you eliminate that color as an option for its neighbors, leaving only one option for them. This breaks down if a third color is available: if you give a vertex a color, its neighbors still have two options left.

19.2 Greedy coloring

Just now, I told you that a single example of a k-coloring is enough to prove that a graph G is k-colorable. While that's true, it's often not good enough from a theoretical point of view, because we want to prove general results. To deal with a whole family of graphs and not just one graph, we need a strategy for coloring them. From a practical point of view, of course, a strategy for coloring graphs is also very important.

The simplest graph coloring algorithm is the greedy coloring algorithm. It could be briefly summarized as, "Go through the vertices and give each vertex the first color not used on its


(a) A bad vertex ordering

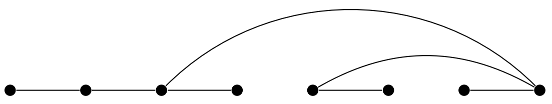

(b) A good vertex ordering

Figure 19.2: Two ways to order the vertices of a tree before coloring it greedily

neighbors." To make the algorithm well-specified, though, so that we know what it does in every situation and can use it in proofs, we need to be explicit about the choices we're making.

First of all, we should give the colors an order, so that the phrase "the first color not used" makes sense. It does not matter how we do this, because the set of colors does not matter beyond its size. We can say that the set of colors is $\{1, 2, \ldots, k\}$ for some k, in which case the first unused color is just the least unused number. Often, we don't know how many colors we'll need when we start color, so we can begin by allowing all the positive integers, and then narrow down the set of colors to the ones we ended up using.

More importantly, we should give the vertices an order: the order in which we color them. This choice is vitally important to the outcome. Consider the two diagrams in Figure 19.2, which show the same tree (or isomorphic trees). Of course, we know that the tree is 2-colorable, because all trees are bipartite—but the greedy algorithm doesn't know!

Question: What happens if you greedily color the vertices of the tree by going left to right in Figure 19.2a?

Answer: The colors used will be $1, 2, 1, 3, 1, 2, 1, 4$ in that order; we use 4 colors.

Question: What happens if you greedily color the vertices of the tree by going left to right in Figure 19.2b?

Answer: The colors used will be $1, 2, 1, 2, 1, 2, 1, 2$ in that order; we use 2 colors.

In the practice problems at the end of this chapter, I will ask a few more questions about these examples. To summarize, if the vertices are ordered badly, then the greedy algorithm can color even a tree with arbitrarily many colors. However, if you know an optimal way to color a graph ahead of time, you can always convince the greedy algorithm to do just as well. Finding a good way to order the vertices without knowing the answer in advance is the hard part.

Even if we don't have a good idea for how to order the vertices, though, there are some worst-case guarantees.

Proposition 19.1. *Any graph G with maximum degree $\Delta(G)$ is $(\Delta(G) + 1)$-colorable.*

Proof. Give the vertex set $V(G)$ an arbitrary order $x_1, x_2, \ldots, x_n$, and color G using the greedy algorithm, using the positive integers as the set of colors.

For each i, when the greedy algorithm gets to vertex x_i, that vertex has at most $\Delta(G)$ neighbors among $x_1, x_2, \dots, x_{i-1}$. At most $\Delta(G)$ different colors can be represented on those neighbors. Therefore, at least one of the colors $1, 2, \dots, \Delta(G)+1$ does not appear on any of x_i's neighbors.

The greedy algorithm will never use a number higher than $\Delta(G)+1$ if a lower one is available, so it will give x_i some color from the set $\{1, 2, \dots, \Delta(G)+1\}$. This is true for each i, so at the end, the greedy algorithm has found a coloring using at most $\Delta(G)+1$ colors. □

The bound of Proposition 19.1 is better than having no bound at all, but it's not very good. However, that does not mean that the greedy algorithm is no good! In this chapter and later in the book, we will see examples where we can deduce better upper bounds from the greedy algorithm for specific families of graphs. To do this, we will use the structure of those graphs to come up with a better ordering of the vertices.

(Of course, there are also proofs based on more sophisticated algorithms.)

19.3 Two lower bounds

If we want to know the chromatic number of any graph for certain, it's not enough to find a coloring: we must also prove that we used the least number of colors possible. To do this, we need lower bounds.

Unfortunately, a typical answer to why a graph is not 7-colorable (for example) might be, "We tried all possible ways to color the vertices, by making guesses and backtracking and trying something else when they were wrong, and never arrived at a 7-coloring of the entire graph." State-of-the-art techniques might combine this with a sequence of partial deductions of a standard type, but these are outside the scope of this textbook and can still be exponentially long in some cases.

In order to make something resembling progress, we will need to lower our standards considerably. First, we will accept lower bounds that are often far from the truth, in the hope that sometimes they will still help us. Second, we will accept lower bounds that are themselves difficult to find, as long as they provide us with a short proof of a lower bound. This is more useful in theoretical problems, where we can generalize that proof to deal with a family of graphs at once.

To arrive at the first such lower bound, we begin by asking: what's the most straightforward graph that requires k colors? It is the complete graph K_k: its k vertices are all adjacent, so all of them need different colors.

Moreover, if a large graph G contains a copy of K_k inside it, then we know that $\chi(G) \geq k$ as well. Even the copy of K_k inside G needs k colors: coloring the other vertices can only make things worse. (In general, if H is a subgraph of G, then $\chi(G) \geq \chi(H)$, because to color G, we must first color H.)

Copies of K_k inside G correspond to k-vertex cliques. The size of the largest clique in G is the clique number $\omega(G)$. We conclude:

Proposition 19.2. *For any graph G, $\chi(G) \geq \omega(G)$.*

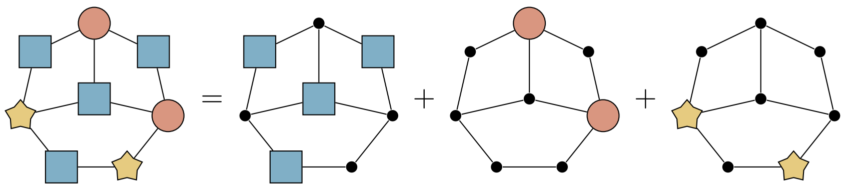


Figure 19.3: A 3-coloring of a graph viewed as a partition into color classes

To be honest, I suspect that I don't need to work hard to prove to you that Proposition 19.2 is true. The difficult thing for most people to accept—and we will see later in this book that this has been difficult for serious mathematicians as well as beginners in graph theory—is that Proposition 19.2 does not tell us everything: that $\chi(G)$ is not always equal to $\omega(G)$.

To see why this is true, we can start by understanding when $\chi(G) \geq 3$, because another way to say "$\chi(G) \geq 3$" is "G is not bipartite", and we already know a lot about bipartite graphs.

Question: What does Proposition 19.2 say about when a graph is not bipartite?

Answer: It says that $\chi(G) \geq 3$ whenever $\omega(G) \geq 3$: that G is not bipartite whenever G contains a copy of K_3.

Question: What do we already know about subgraphs that mean a graph is not bipartite?

Answer: By Theorem 13.1, such subgraphs are copies of $C_3, C_5, C_7, C_9, \dots$: cycles of odd length.

Of these graphs, C_3 is isomorphic to K_3: in both cases, we have three vertices, and any two are adjacent. So we might say that Proposition 19.2 identifies the smallest fundamental non-bipartite graph, but misses all the rest.

For a higher number of colors, the reality is even more complicated, and Proposition 19.2 is even more of an oversimplification. Still, it is a useful lower bound that is often correct in practice.

What else can we use?

The independence number $\alpha(G)$ is the exact opposite of the clique number $\omega(G)$: it is the size of the largest set of vertices with no edges between them. So it may seem surprising that the independence number can also help us put lower bounds on the chromatic number.

To see the connection, we switch to thinking of a coloring as a partition into color classes; Figure 19.3 shows an example of such a partition. We started with the definition that a coloring is proper when the endpoints of any edge are in different color classes; however, another way to say this is that no two vertices in the same color class are adjacent. In other words, a coloring is proper if and only if every color class is an independent set.

The independence number $\alpha(G)$ tells us how big such color classes can get, and we can use this to deduce another lower bound on the chromatic number of G.

Proposition 19.3. *For any n-vertex graph G, $\chi(G) \geq \frac{n}{\alpha(G)}$.*

Proof. The independence number $\alpha(G)$ is the largest number of vertices in any independent set, so in particular every color class in a k-coloring has at most $\alpha(G)$ vertices. Together, the k color classes contain at most $k \cdot \alpha(G)$ vertices. Every vertex must be in one of the color classes, so if there are n vertices, then $k \cdot \alpha(G) \geq n$. Rearranging, we get $k \geq \frac{n}{\alpha(G)}$. □

If Proposition 19.3 and not Proposition 19.2 is the reason why the chromatic number of a graph is large, then this could be a very subtle reason. We might not notice, when coloring a small portion of the graph, that there is a problem; the effect of having no large independent sets is only noticeable when we look at the graph as a whole.

But is this scenario actually possible? Are there graphs G where the independence number, and not the clique number, controls $\chi(G)$?

In fact, not only do such graphs exist, but they are very common. In Chapter 18, we explored what happens when a graph on 1000000 vertices is constructed by flipping coins to for each possible edge to decide if we include it. We discovered that it is extremely unlikely for this graph to have cliques or independent sets with more than 40 vertices.

Question: What would Proposition 19.2 tell us about such a graph?

Answer: If there are no cliques with more than 40 vertices, the best we could hope is a lower bound of $\chi(G) \geq 40$, and we might not even be able to get that.

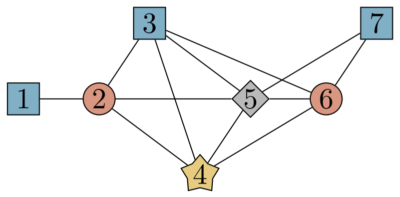
Question: What would Proposition 19.2 tell us about such a graph?

Answer: If there are 1000000 vertices, but there are no independent sets of more than 40 vertices, then at least $\frac{1000000}{40} = 25000$ colors are needed in a coloring.

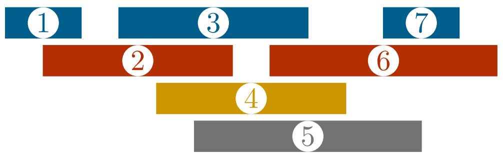
By flipping a coin for every edge, we make every 1000000-vertex graph equally likely. This means that, if something is true with a very high probability for this random graph, it is true for almost all 1000000-vertex graphs. In other words, $\chi(G)$ is much larger than $\omega(G)$ for almost all of these graphs!

This does not mean, however, that Proposition 19.2 is useless. While random graphs are a good source of examples, they do not actually reflect most graphs we study: if a graph is interesting to us, there is a good chance that it is atypical somehow. This example does not tell the whole story.

In the rest of this chapter, we will see some alternative scenarios: a class of graphs in which the lower bound of Proposition 19.2 always gives the chromatic number exactly, and class of graphs in which the chromatic number is much larger than both of our lower bounds.



(a) A coloring of an interval graph



(b) Solving the scheduling problem

Figure 19.4: Coloring an interval graph

19.4 Coloring interval graphs

In Chapter 18, we introduced interval graphs: graphs obtained from a collection of intervals, with an edge between vertices whenever the intervals overlap. These are a nice application of graph coloring for two reasons: first, coloring interval graphs has useful practical applications, and second, it is easier than the general graph coloring problem.

First, on the applications. Suppose that the intervals we use to build an interval graph are the start and end times of events: this is the setting we looked at in Chapter 18. We previously said that both cliques and independent sets have meaning in this setting: cliques are sets of events all happening at the same time, and independent sets are sets of events that don't conflict with each other.

Question:	Actually, there's a subtlety: by definition, a clique is a set of events where every pair has an overlap, but does this mean that there's a single point in time when all events in the clique overlap?
Answer:	Yes; to see this, take a clique of events, let a be the first of these events to end, and let z be the of these events to start. Events a and z must overlap, because they're part of a clique together. At any time in that overlap, all events in the clique have started (because even z has started) but none have ended (because even a hasn't ended).

The chromatic number of an interval graph is also a useful quantity. Suppose we want to pick locations for the events. Two events happening at the same time should be in different locations. At the same time, we would like to use as few locations as possible: this makes it easier to go from one event to another, and we might also have a limited number of locations available. Choosing locations for the events is exactly a graph coloring problem, where the locations are the colors; Figure 19.4 shows a coloring of an interval graph, and the corresponding solution to the scheduling problem for the intervals.

The same model can be useful in situations where the scheduling is more metaphorical. For example, graph coloring can be used in the register allocation problem in computer science [5]. Here, we want to assign the variables used by a program to a limited supply of memory locations called registers (if possible); however, two variables cannot be assigned to the same register if they are in use at the same time. In simple settings, the graph that represent conflicts between the variables becomes an interval graph, and the assignment to registers is a coloring of that interval graph.[3]

The interesting thing about coloring interval graphs is that the complexity I alluded to in previous sections is almost not present. The greedy algorithm is enough to color them optimally, and the lower bound of Proposition 19.2 gives the exact chromatic number, with no funny business.

Theorem 19.4. *If G is an interval graph, then $\chi(G) = \omega(G)$. Moreover, if the greedy algorithm is given the vertices of G in order of the starting points of the intervals, then it will use only $\omega(G)$ colors.*

Proof. The second half of the statement of the theorem is actually all we need to prove. If the greedy algorithm uses only $\omega(G)$ colors, then $\chi(G) \le \omega(G)$; however, by Proposition 19.2, $\chi(G) \ge \omega(G)$, so the two must be equal.

Let's consider an iteration of the greedy algorithm in which it must color a vertex x associated to interval $[a, b]$. To see which colors the algorithm cannot use, we look at the neighbors of x that already have a color. What are these neighbors? They correspond to intervals that have an overlap with $[a, b]$, but have starting points less than a.

In order to overlap $[a, b]$, these intervals must contain the point a. This means that they all overlap each other, too! In fact, x and the neighbors of x already given a color must form a clique: call this clique Q.

There are $|Q| - 1$ neighbors of x that have already been colored, because the last element of Q is x itself. Therefore at most $|Q| - 1$ colors are ruled out for x, and the greedy algorithm will be able to give x one of the first $|Q|$ colors on its list.

The clique Q can be bigger or smaller depending on x, but in all cases, $|Q| \le \omega(G)$ by definition of the clique number. Therefore the greedy algorithm always uses one of the first $\omega(G)$ colors on the list, completing the proof. □

19.5 Mycielski graphs

We have already seen that in a randomly chosen graph, the chromatic number can be much higher than the clique number. But the truth is even worse than this: it is possible to increase the chromatic number arbitrarily high while the clique number does not grow at all, and find graphs where both Proposition 19.2 and Proposition 19.3 drastically underestimate the chromatic number!

One method for doing this is a construction found in 1955 by Jan Mycielski [19]. We will first see it as an operation on graphs: a graph that transforms a graph G into its Mycielskian $M(G)$, increasing its chromatic number but not its clique number. Then, the sequence of Mycielski graphs is obtained by applying this operation over and over again.

The *Mycielskian* $M(G)$ of a graph G is constructed in three steps:

[3]More sophisticated models can produce graphs with fewer edges than the interval graph, with a smaller chromatic number that's harder to compute, but I will not get into those details.

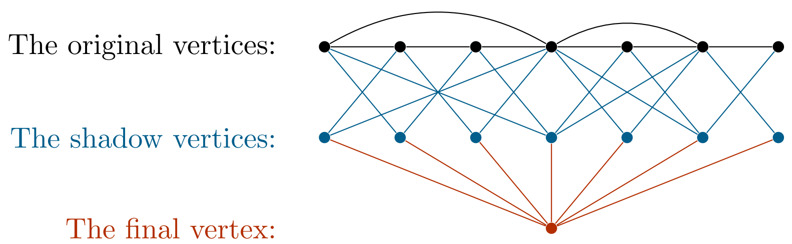


Figure 19.5: The Mycielskian of a graph

1. Start with a copy of G. (I will call the vertices in this copy of G the *original vertices*, for ease of reference.)
2. For each original vertex x, add a new vertex x' adjacent to the same original vertices as x. (I will call x' a *shadow vertex*: the shadow of x.) No edges are added between the shadow vertices.
3. Finally, add one more vertex (which I will call the *final vertex*) adjacent to all the shadow vertices.

Figure 19.5 illustrates this construction in an example. I should note that the terms "original vertices", "shadow vertices", and "final vertex" are made up by me; there's no standard terminology.

The first objective of this definition is to avoid increasing the clique number: we want $\omega(M(G))$ to stay equal to $\omega(G)$. Technically, this is violated when G has no edges: then $\omega(G) = 1$, but $M(G)$ will gain some edges when the final vertex is added, so $\omega(M(G)) = 2$. However, it will be true in all other cases.

Lemma 19.5. *For all graphs G with $\omega(G) \geq 2$, $\omega(M(G)) = \omega(G)$.*

Proof. We do not know anything about cliques formed by the original vertices: these are cliques that already existed in G. In particular, because $M(G)$ contains a copy of G, $\omega(M(G))$ is at least $\omega(G)$: every clique in G can also be found in $M(G)$. It will also be possible that $M(G)$ has new large cliques—but we will show that none of these are bigger than cliques G already had.

To see this, suppose that Q is a clique in $M(G)$. If $|Q| \leq 2$, then we know that $|Q| \leq \omega(G)$ because we assumed $\omega(G) \geq 2$, so we may assume that Q contains at least 3 vertices.

Question: Can Q contain the final vertex?

Answer: No, because none of the neighbors of the final vertex are adjacent to each other.

Question: Can Q contain any shadow vertices?

Answer: Yes, but at most one: no two shadow vertices are adjacent, so C cannot contain two or more shadow vertices.

If Q contains no shadow vertices at all, we are not interested: it's contained in a copy of G, therefore $|Q| \leq \omega(G)$. So suppose Q contains the original vertices $x_1, x_2, \dots, x_{k-1}$ and the shadow vertex x'_k: the shadow of x_k. Since the original vertex x_k is not adjacent to its shadow x'_k, we know that $x_k \notin Q$.

By the construction of x'_k, among the original vertices, it has only the neighbors that x_k does: since x'_k is adjacent to $x_1, x_2, \dots, x_{k-1}$, we know x_k must be adjacent to them, as well. Therefore $\{x_1, x_2, \dots, x_k\}$ is also a clique, and it has the same number of vertices as Q.

However, $\{x_1, x_2, \dots, x_k\}$ consists solely of original vertices, so it can have at most $\omega(G)$ vertices! This means that $|Q| \leq \omega(G)$ as well.

We've now established the inequality $|Q| \leq \omega(G)$ for all cliques Q in $M(G)$, so we conclude that $\omega(M(G)) \leq \omega(G)$. This concludes the proof. □

The most interesting case of Lemma 19.5 is when $\omega(G) = 2$. (Such graphs are sometimes called triangle-free graphs.) In this case, the graphs $M(G)$, $M(M(G))$, and so on all have clique number 2 as well: the least clique number that a graph with edges can have.

Meanwhile, the chromatic number of G ticks up!

Lemma 19.6. *For all graphs G, $\chi(M(G)) = \chi(G) + 1$.*

Proof. Here, we are expressing the solution to one optimization problem (the chromatic number of $M(G)$) in terms of the solution to another optimization problem (the chromatic number of G), so it's worth thinking carefully about what we need to prove.

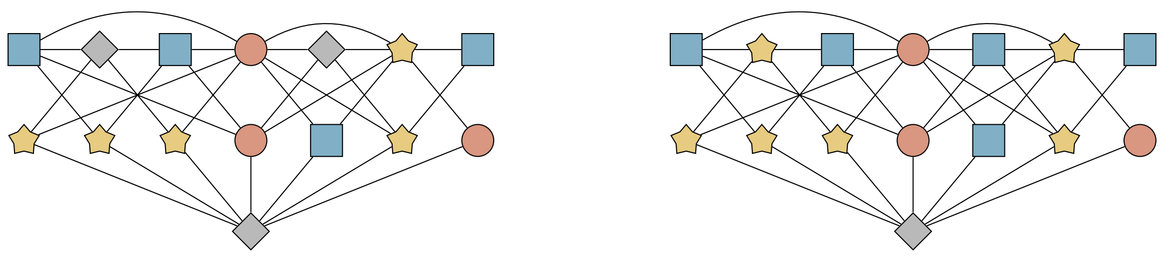
To show that $\chi(M(G)) \leq \chi(G) + 1$, we need to show that $M(G)$ has a coloring with $\chi(G) + 1$ colors. But we don't need to come up with the coloring of $M(G)$ from scratch: by definition of the chromatic number, we know that G has a $\chi(G)$-coloring, and we can use such a coloring to help us.

We might imagine that to show that $\chi(M(G)) \geq \chi(G) + 1$, we need to show that $M(G)$ cannot be colored with fewer colors, but proofs of impossibility are tricky: the paradigm of turning one coloring into another is much friendlier. So we rewrite this inequality as $\chi(G) \leq \chi(M(G)) - 1$, and now we're once again proving an upper bound on a chromatic number. We need to show that G has a coloring with $\chi(M(G)) - 1$ colors, and in the process, we may use a $\chi(M(G))$-coloring of $M(G)$.

With the strategy talk out of the way, we can go on to the proof.

For the first step... I don't like phrases like, "The obvious thing will work," because what's obvious and what isn't depends on your experience. But the obvious thing will work. It's okay if it doesn't feel obvious yet; use it to train your intuition so that future arguments like it will be be more intuitive.

We start with a coloring of G using $\chi(G)$ colors. We would like to color $M(G)$ using only one more color than that. Since $M(G)$ starts with a copy of G inside it, the natural thing to do is to give every original vertex the same color as in the $\chi(G)$-coloring of G. Next, for each original vertex x, give its shadow x' the same color as x: this is guaranteed to work, because the vertices adjacent to x' are all original vertices adjacent to x, so none of them can have the same color as x. For the final vertex, use a new color we haven't used yet.



(a) A proper coloring of $M(G)$ (b) An improper coloring of $M(G)$

Figure 19.6: Going from a k-coloring of $M(G)$ to a $(k-1)$-coloring of G

The second step is harder. Let $k = \chi(M(G))$, so that there is a k-coloring of $M(G)$. We would like a coloring of G using only $k-1$ colors. Inside our coloring of $M(G)$, there is a coloring of G... but it may well use all k colors. Figure 19.6a shows an example, using the same $M(G)$ as in Figure 19.5.

We want to modify this coloring so that it only uses $k-1$ colors on the original vertices. Motivated by our first step, we would like the color used on the final vertex (let c be this color) to be the color we get rid of. It's a problem if any of the original vertices have color c; to "fix" it, whenever an original vertex x is assigned color c, we give x the color of its shadow x'.

Question: What if the shadow x' is also assigned color c?

Answer: That's impossible: x' is adjacent to the final vertex, which has color c, so x' must have some other color.

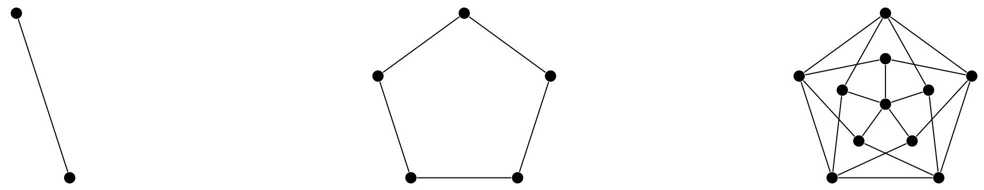
An example of the modified coloring we get is shown in Figure 19.6, and you can see from this example that it might be an improper coloring. In Figure 19.6, an original vertex recolored to ★ is adjacent to two shadow vertices that also have color ★. In general, when we give x the color of x', there is no guarantee that no neighbor of x has this color.

If we've gotten an improper coloring, surely our attempt was a failure? Not quite: remember, we only want a coloring of G, not all of $M(G)$. It turns out that if we only look at the original vertices, the coloring we have is a proper coloring!

To see this, let x and y be two adjacent original vertices. If neither x nor y had color c, then the colors of x and y are unchanged from the k-coloring we started with, so the colors of x and y are still different. It's also not possible that both x and y both had color c; because we started with a proper k-coloring.

The last case is that only one of x and y (without loss of generality, x) had color c. In this case, the color of x is changed to the color of x' in the k-coloring. By the construction of $M(G)$, x' and y are also adjacent; therefore the color of x' is not the same as the color of y. It follows that x and y still have different colors after the change.

As a result, if we keep only the original vertices, then we do get a proper $(k-1)$-coloring of G. Therefore $\chi(G) \leq k-1$; since $k = \chi(M(G))$, this completes the second step and concludes the proof of the theorem. □



(a) M_2: the base case (b) M_3: the 5-cycle (c) M_4: the Grötzsch graph

Figure 19.7: The first few graphs in the sequence of Mycielski graphs

Lemma 19.5 and Lemma 19.6 are the two properties of the Mycielskian we needed to prove the following theorem.

Theorem 19.7. *For all $k \geq 1$, there exists a M_k with $\omega(M_k) \leq 2$ and $\chi(M_k) = k$.*

Proof. We induct on k. When $k = 1$ and when $k = 2$, a complete graph (K_1 and K_2, respectively) will do, so that's what we use for M_1 and M_2.

For the induction step, assume that we've already constructed a graph M_{k-1} with $\omega(M_{k-1}) \leq 2$ and $\chi(M_{k-1}) = k - 1$. Let $M_k = M(M_{k-1})$: the Mycielskian of M_{k-1}. By Lemma 19.5, $\omega(M_k) \leq 2$; by Lemma 19.6, $\chi(M_k) = \chi(M_{k-1}) + 1 = k$. These are exactly the two facts we needed to complete the induction step.

By induction, M_k exists for all $k \geq 1$. □

The graphs in the sequence $M_2, M_3, M_4, \dots$ are called the *Mycielski graphs*. (We rarely include M_1, because it is not truly part of the recurrence: M_2 is not the Mycielskian of M_1, but a second base case.) Figure 19.7 shows the first few graphs in the sequence.

The graph M_3 is isomorphic to the cycle graph C_5, though it takes some "untwisting" to obtain a symmetric diagram from a 3-layer diagram in the style of Figure 19.5.

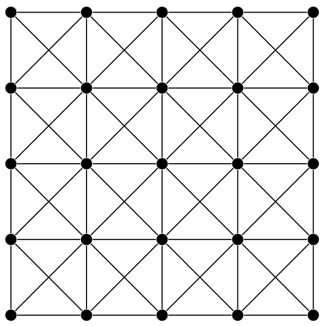
Finally, let help you make sense of how we get from C_5 to the diagram in Figure 19.7c. The original vertices have stayed in their place, with their shadows placed halfway to the center of the diagram; the final vertex placed at the center. The resulting graph, M_4, is known as the Grötzsch graph. It is named after Herbert Grötzsch, who used it in 1959 for the same purpose as Mycielski: as an example of a triangle-free graph with chromatic number 4 [13].

Question: Since the Mycielski graphs have clique number 2, Proposition 19.2 does not give a good lower bound on their chromatic number. What about Proposition 19.3?

Answer: It is almost no better: the shadow vertices in M_k form an independent set containing almost half of all the vertices. Therefore Proposition 19.3 only proves that $\chi(M_k) \geq 3$.

19.6 Practice problems

1. Let G be the 25-vertex graph shown below:



   a) What is the clique number $\omega(G)$? What lower bound on $\chi(G)$ does it give?

   b) What is the independence number $\alpha(G)$? What lower bound on $\chi(G)$ does it give?

   c) What is the maximum degree $\Delta(G)$? What upper bound on $\chi(G)$ does it give?

   d) What is the actual chromatic number of G?

2. Find a greedy coloring of the circulant graph $\mathrm{Ci}_9(1,3)$, going through its verticesin numerical order. Is there a coloring of $\mathrm{Ci}_9(1,3)$ that uses fewer colors than the greedy coloring you found?

3. Let G_n be the graph with $V(G_n) = \{1, 2, \dots, 3n\}$ in which vertices x and y are adjacent if $|x - y|$ is odd and not equal to 1.

   a) If G_n is colored greedily by going through the vertices in numerical order, how many colors are used? Find the answer in terms of n.

   b) What is the chromatic number of G_n, and why?

4. Alice and Bob take turns coloring the cube graph Q_3, one vertex at a time. They have a fixed set of colors $C = \{\text{red}, \text{blue}, \text{yellow}\}$, and neither player is allowed to give a vertex a color that has already been used on a neighbor of that vertex. However, they have different goals: Alice (who goes first) wants to get a coloring of Q_3, while Bob wants to arrive at a partial coloring which cannot be finished.

   If both players play as well as possible, which player will win?

5. Look back at Figure 19.2a, then:

   a) Find a tree and a vertex ordering of that tree for which the greedy algorithm will use 5 different colors.

   b) Prove that for all positive integers k, there is a tree and a vertex ordering of that tree for which the greedy algorithm will use k different colors.

6. The lowest-degree-last vertex ordering of an n-vertex graph G is constructed as follows, recursively. We choose the last vertex, x_n, to be any vertex whose degree is the lowest degree in G. To find the ordering $x_1, x_2, \dots, x_{n-1}$ of the other vertices, we find the lowest-degree-last ordering of the $(n-1)$-vertex graph $G - x_n$.

   Prove that if G is a tree, then the greedy coloring algorithm, using the lowest-degree-last ordering, will never use more than 2 colors.

7. Let G be a graph with chromatic number k, and let $f \colon V(G) \to \{1, 2, \dots, k\}$ be a k-coloring of G.

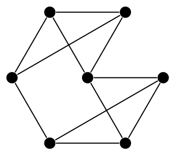
   Prove that if we sort the vertices of G by their f-values (putting all vertices given color 1 first and all the vertices given color k last), then the greedy algorithm will also find a k-coloring of G.

   (Be careful: if your proof shows that the greedy algorithm will find the coloring f we started with, then it's wrong, because that claim is false in general.)

8. a) Suppose that the greedy algorithm uses k colors on a graph G. Prove that G must have at least $1 + 2 + \cdots + (k-1) = \binom{k}{2}$ edges.

   b) Prove that if G has m edges, then $\chi(G) \le 1 + \sqrt{2m}$.

9. The graph below is called the Moser spindle. It is a unit-distance graph: its vertices can be placed in the plane in such a way that two vertices are adjacent exactly when they are at distance 1 from each other.



   a) Find a unit-distance drawing of the Moser spindle.

   b) Prove that the Moser spindle has chromatic number 4.

   The chromatic number of the plane is the least number of colors needed to color every point of the plane so that no two points at distance from each other are the same color. Leo and William Moser constructed the Moser spindle in 1961 [18], proving that the chromatic number of the plane is at least 4.

   This was the best known lower bound until 2018, when Aubrey de Grey constructed a 1581-vertex unit distance graph with chromatic number 5 [12].

10. The wheel graph with n spokes is the graph obtained from the cycle graph C_n by adding a new vertex $n+1$ adjacent to all n vertices of the cycle.

    a) Draw a diagram of a wheel graph with 5 spokes.

    b) Show that this graph has clique number 3 but chromatic number 4.

    c) Generalize this construction to find a graph G in which $\omega(G) = k$ but $\chi(G) = k+1$, for every $k \ge 3$.

11. It is interesting to think about coloring the co-bipartite graphs: graphs whose complement is bipartite. A graph G is a co-bipartite graph if and only if we can write $V(G)$ as the disjoint union of two sets A and B which are both cliques in G (and, additionally, there may be some edges between A and B).

    a) Show that in any coloring of a co-bipartite graph, each color class contains at most 2 vertices.

    b) Suppose that a co-bipartite graph G has a coloring where k of the color classes have 2 vertices.

       How else can you describe those k color classes, and their role in the complement graph $\overline{G}$ (a bipartite graph)?

    c) What is the connection between a clique in a co-bipartite graph G and a vertex cover in its complement $\overline{G}$?

    d) Use parts (a)–(c) to prove that if G is a co-bipartite graph, then $\chi(G) = \omega(G)$.

12. The following inequality is true whenever $n_1, n_2, \ldots, n_k$ are positive integers with $n_1 + n_2 + \cdots + n_k = n$:

$$\sum_{i=1}^{k} \binom{n_i}{2} \geq k \binom{n/k}{2} = \frac{n(n-k)}{2k}.$$

    Use this inequality to find the maximum number of edges in an n-vertex graph with chromatic number k.

20 Line graphs

The purpose of this chapter

It is a little strange of me to put the chapter on line graphs in the part of the textbook devoted to hard problems, because there is not really a hard problem associated to line graphs. The opposite is true: the hard problems in this section are often easier to solve when we are solving them in a line graph. And that's exactly why I think these chapters belong together!

By looking at line graphs, we will also see many connections between problems that seemed very different before. There is an occasionally useful relationship between Euler tours (Chapter 8) in a graph G and Hamilton cycles (Chapter 17) in its line graph $L(G)$. Matchings (Chapter 13 through Chapter 16) are exactly the same thing as independent sets (Chapter 18) in the line graph. And by extending graph coloring to line graphs, we will arrive at edge coloring, a new problem with its own applications.

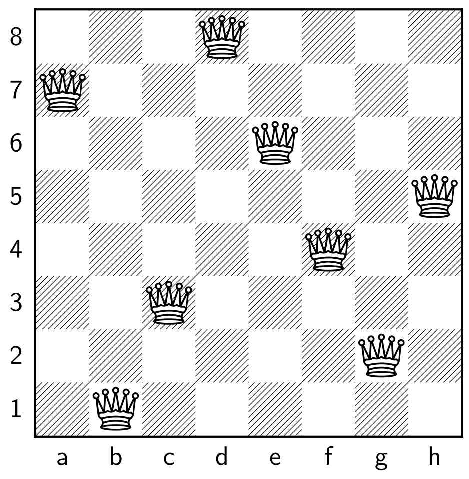
You can see from this list that to understand all the material in this chapter, you need to be familiar with many concepts in previous parts of the textbook. Apart from the chapters mentioned above, you should be familiar with Chapter 7, because both directed graphs and multigraphs will make an appearance.

It is not necessarily the case that you want to read the entire chapter if you're learning graph theory for the first time (or to teach the entire chapter in an introductory course). The application to de Bruijn sequences is interesting, but self-contained. Vizing's theorem is a natural observation to make after observing enough examples, but difficult to prove, though I have tried to pick out a less complicated proof.

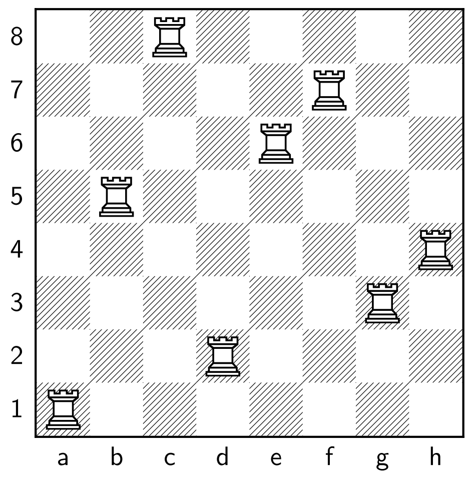
20.1 Line graphs

In Chapter 13 and Chapter 18, I used two very similar problems as an example: the problem of placing eight rooks on a chessboard, and the problem of placing eight queens on a chessboard, so that they do not attack each other. The solutions are shown once again in Figure 20.1, for your reference. (Of course, the eight queens problem is strictly harder than the eight rooks problem, because a queen has all the movement power of a rook and more, so instead of Figure 20.1b, we could have just taken Figure 20.1a and changed all the queens to rooks.)

However, there's something strange about how we modeled these problems. The eight queens problem was our first example of finding independent sets in a graph: a hard problem we have no good algorithms for. It would have been equally natural to try solving the eight rooks problem in the same way. Define the 8×8 *rook graph* to be the graph whose 64 vertices are the squares of a chessboard, with an edge between two squares which are in the same rank or file. Then a solution to the eight rooks problem is an independent set in the 8×8 rook graph.



(a) Eight queens: an independent set



(b) Eight rooks: a… matching?

Figure 20.1: Rook and queen placements

We did something different in Chapter 13—and a good thing, too! Instead of finding an independent set, we were able to solve the problem by finding a matching, which is much easier. But why is it that the eight rooks problem has these two different models, and what is the relationship between them?

The answer is: the 8×8 rook graph happens to be a line graph, whose special structure makes many problems easier.

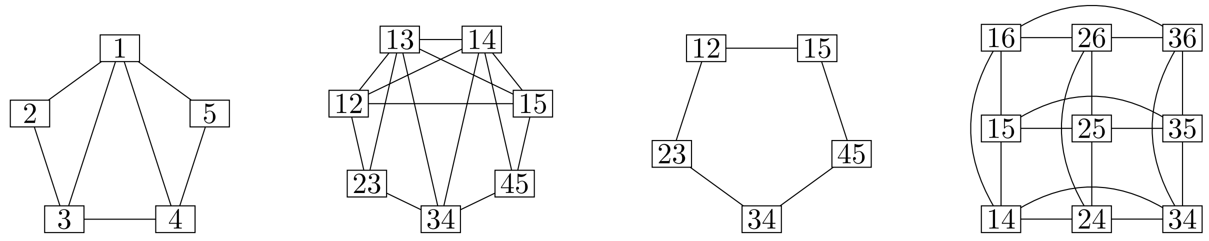
Definition 20.1. *The* ***line graph*** $L(G)$ *of a graph* G *is the graph whose vertices are the edges of* G*; two vertices* e, e' *of* $L(G)$ *are adjacent if and only if edges* e *and* e' *share an endpoint in* G*.*

Why is this the "line" graph? The name was introduced by Frank Harary and Robert Norman in 1960 [15], who referred to the vertices and edges of graphs as "points" and "lines". As a result, Harary and Norman called $L(G)$ the line graph because its vertex set (or "point set") was the set of "lines" of G. The name stuck.

Let's look at some examples. Figure 20.2a and Figure 20.2b show a graph G with nothing particularly special about it, and its line graph $L(G)$. The line graph has 7 vertices, which correspond to the 7 edges of G. Observe what happens to the four edges 12, 13, 14, and 15 incident to vertex 5: they turn into a 4-vertex clique in $L(G)$.

Most graphs have line graphs that are bigger than they are in every way, but not all. For example, Figure 20.2c shows the line graph of C_5: we see that $L(C_5)$ is isomorphic to C_5 itself, and the same will happen for every cycle graph.

Question:	What is the line graph of a path graph P_n isomorphic to?
Answer:	$L(P_n)$ is isomorphic to P_{n-1}: an n-vertex path has $n-1$ edges, each sharing an endpoint with the previous edge and the next edge along the path.



(a) G (b) $L(G)$ (c) $L(C_5)$ (d) $L(K_{3,3})$

Figure 20.2: Several examples of line graphs

Finally, Figure 20.2d shows the line graph of the complete bipartite graph $K_{3,3}$ (with vertices $\{1,2,3\}$ on one side and $\{4,5,6\}$ on the other side of the bipartition). It is also isomorphic to the 3×3 rook graph, showing the possible moves a rook could make on a 3×3 chessboard. With a little imagination, this should let you see how the 8×8 rook graph is isomorphic to the line graph $L(K_{8,8})$.

It is also possible to define the line digraph of a directed graph; the idea is similar, but two arcs that share an endpoint will only correspond to an arc in the line digraph when their orientations are compatible.

Definition 20.2. *The **line digraph** $L(D)$ of a directed graph D is the directed graph whose vertices are the arcs of D. For every pair of arcs (x,y) and (y,z) in D such that the first arc ends where the second arc starts, there is an arc $\big((x,y),(y,z)\big)$ in $L(D)$.*

It often turns out that one problem in graph theory is equivalent to another problem in disguise: solving one problem in the graph G is equivalent to solving the other in $L(G)$. Even when these are not exactly equivalent, the relationship can help us understand both problems better.

For example, as we see in the case of the eight rooks problem, finding a matching in a graph G is equivalent to finding an independent set in $L(G)$. A matching in G is a set of edges such that no two edges share an endpoint. A set of edges in G is precisely a set of vertices in $L(G)$, and sharing an endpoint is precisely what defines adjacency in $L(G)$. Therefore a set $M \subseteq E(G)$ is a matching in G if and only if it is an independent set in G.

The notation that we've used for the two problems is suggestive of this relationship: we used $\alpha(G)$ to denote the independence number of G, and $\alpha'(G)$ to denote the matching number. We could already have said the vague phrase, "$\alpha'(G)$ is the edge version of $\alpha(G)$". Now we can make this precise: $\alpha'(G) = \alpha(L(G))$.

Beyond this example, it is often the case that a numerical graph invariant $f(G)$ will be given an edge version $f'(G)$, defined as $f(L(G))$. Unfortunately, sometimes, an invariant will also be written $f'(G)$ if it gives off vague vibes of being the edge version of $f(G)$, even if it is not always equal to $f(L(G))$. You should be careful and check if $f'(G) = f(L(G))$ holds when faced with new notation; don't just assume it.

Question: What do cliques in $L(G)$ correspond to?

Answer: With one exception, a set of vertices in $L(G)$ is a clique when those vertices are edges in G that all share a common endpoint. If it weren't for that pesky exception, we could think of the maximum degree $\Delta(G)$ as being an "edge clique number" $\omega'(G)$...

Question: What's the pesky exception?

Answer: There's another way for vertices in $L(G)$ to form a clique: if they correspond to the three edges of a cycle of length 3. Thus, if G is a graph with maximum degree 2, but one connected component of G is a cycle of length 3, then $\omega(L(G))$ will be 3, not 2.

The benefit of identifying the matching number $\alpha'(G)$ as $\alpha(L(G))$ is that it makes the problem of finding an independent set easier in a line graph. If we identify a graph H as being isomorphic to $L(G)$ for some G, then we can reduce the hard problem of finding $\alpha(H)$ to the easier problem of finding $\alpha'(G)$.

20.2 Euler tours and Hamilton cycles

An Euler tour in a graph is a closed walk that uses each edge exactly once. A Hamilton cycle in a graph is a cycle that uses each vertex exactly once. When the descriptions of two problems match so closely, but one uses "vertex" where the other uses "edge", it is natural to suspect that line graphs could be involved.

In fact, half of a relationship is present. We can prove the following result:

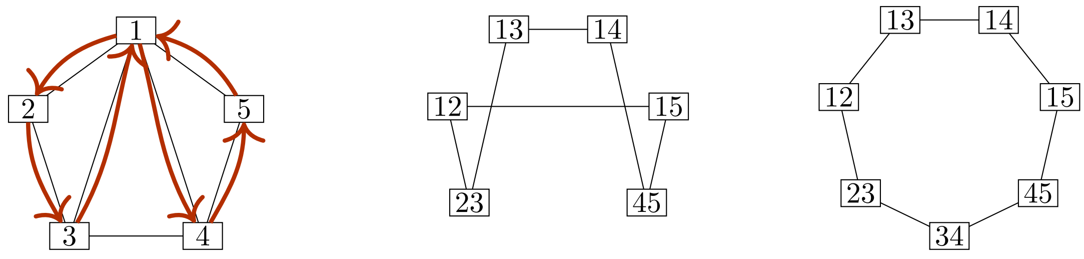
Proposition 20.1. *Every Euler tour in a graph G corresponds to a Hamilton cycle in $L(G)$.*

Proof. Suppose that the closed walk $(x_0, x_1, x_2, \dots, x_m)$, with $x_m = x_0$, is an Euler tour in G. Then consider the following sequence of edges of G (or vertices of $L(G)$):

$$(x_0x_1, x_1x_2, x_2x_3, \dots, x_{m-1}x_m, x_mx_0, x_0x_1).$$

These are, in fact, edges of G, because they are pairs of consecutive vertices of the closed walk. In fact, by the definition of an Euler tour, if we leave out the last element of this sequence (which is the same as the first element), then every edge of G is included exactly once. Finally, two consecutive edges in this sequence have the form xy, yz, and share an endpoint: so they are adjacent in $L(G)$. These are exactly the conditions needed to be certain that this sequence is a closed walk representing a Hamilton cycle in $L(G)$. □

Figure 20.3 shows a partial illustration of this principle, using as an example the graph G from Figure 20.2a, whose line graph was shown in Figure 20.2b. The closed walk in Figure 20.3a is



(a) A walk with no repeated edges (b) The corresponding 6-cycle (c) A Hamilton cycle in $L(G)$

Figure 20.3: Euler tours in G and Hamilton cycles in $L(G)$

not an Euler tour, but it is a closed walk with no repeated edges, so it shares most of the same properties. We can apply the same transformation

$$(1, 2, 3, 1, 4, 5, 1) \rightsquigarrow (12, 23, 13, 14, 45, 15, 12)$$

as we did in the proof of Proposition 20.1. (I have adjusted the order in which an edge is written, in some cases, so that it matches how Figure 20.3b is labeled.)

Question: What can the argument of Proposition 20.1 tell us if we apply it to closed walks like this one, which use each edge at most once?

Answer: The resulting closed walk in $L(G)$ represents a cycle, but not necessarily a Hamilton cycle. Here, the original walk did not use the edge 34, and so vertex 34 is not part of the cycle.

Figure 20.3c, on the other hand, shows a Hamilton cycle in $L(G)$. We should be suspicious of this, because the original graph G is not Eulerian: vertices 3 and 4 have odd degree! And, in fact, the converse to Proposition 20.1 is false: Hamilton cycles in $L(G)$ do not always correspond to Euler tours in G.

Question: What went wrong?

Answer: When visiting vertices $15, 14, 13, 12$ in $L(G)$, though they all share an endpoint, they all share the same endpoint. To continue a walk in G, it is not enough to know which edge was the last edge used; we need to know which of its endpoints was the last vertex used, to continue from that vertex.

This problem goes away if we consider directed graphs and their line digraphs. If a, a' are two arcs in a directed graph D, then the line digraph $L(D)$ can afford to be pickier about when it has an arc (a, a'). The arc (a, a') is not just included when a shares an endpoint with a', but when they share endpoints compatibly: when a ends where a' starts. This lets us prove the following:

Proposition 20.2. *If D is a directed graph, then Euler tours in D correspond to Hamilton cycles in $L(D)$, and vice versa.*

Proof. I will omit the proof that an Euler tour in D corresponds to a Hamilton cycle in $L(D)$, because the argument is the same as in Proposition 20.1.

Suppose that in $L(D)$, we have a Hamilton cycle represented by the walk

$$(a_0, a_1, \dots, a_{m-1}, a_0).$$

For each $i = 0, 1, \dots, m-1$, let $a_i = (x_i, y_i)$, and consider the sequence

$$T = (x_0, x_1, \dots, x_{m-1}, x_0)$$

of vertices in D.

For each $i = 0, 1, \dots, m-1$, there must be an arc (a_i, a_{i+1}) in $L(D)$, meaning that $y_i = x_{i+1}$. Therefore there is an arc (x_i, x_{i+1}) in D: this is another way to write the arc a_i. Similarly, there is an arc (a_{m-1}, a_0) in $L(D)$, so $y_{m-1} = x_0$, and therefore D has the arc (x_{m-1}, x_0). This means that T is a walk in D.

The arcs used by T are precisely the arcs $a_0, a_1, \dots, a_{m-1}$, in that order. Since we started with a Hamilton cycle in $L(D)$, these arcs include each vertex of $L(D)$ exactly once, so they include each arc of D exactly once. Therefore T is an Euler tour. □

We solved the problem of finding Euler tours completely in Chapter 8, while the problem of finding Hamilton cycles is difficult and remains difficult in directed graphs. This means that if we spot that a certain directed graph happens to be a line digraph, we can make use of this to turn a difficult problem into an easy problem.

20.3 De Bruijn sequences

This happens, for example, in the study of de Bruijn sequences. To introduce these, let me give an example: the sequence

aabaacabbabcacbaccbbbcbccca,

which you should think of as being cyclic: when you get to the end, keep reading again from the start.[4] A 3-letter *substring* of this sequence is what you get if you start anywhere and read the next 3 letters: for example, you could start at the beginning, you will get aab, or start at the first c and get cab. There are 27 letters in the sequence, so there are 27 possible substrings; the magical property of this sequence is that every possible substring occurs exactly once.

Question:	Where does the substring aaa occur?
Answer:	You must start at the last character, reading an a and then reading aa from the beginning.

[4]Don't get stuck in an infinite loop, though. Eventually, stop reading the sequence and go back to reading about graph theory.

In general, given an alphabet A (any set of symbols) and a positive integer n, we can define the *de Bruijn sequence of order n* over A to be a cyclic sequence with the same property: its length-n substrings contain every possible substring exactly once. Of course, we can define anything we like, but that's no guarantee that such a thing exists!

These sequences are named after Nicolaas Govert de Bruijn, who proved in 1946 [3] that they exist: for all n and all alphabets A. To do this, de Bruijn used graph theory; but what is the right graph model for this problem?

One extreme option is to use A as the set of vertices, putting every possible edge between them, so that a de Bruijn sequence is a particular kind of closed walk in the graph. Such an approach is almost never the right choice, and it is not the right choice here. The problem is that the graph we get doesn't manage to encode the rules of the problem in any way: we don't benefit from looking at the problem with the graph in mind. Ideally, once we build the graph that describes the problem, the solution should be some very standard object inside that graph.

Another option is to take our vertices to be the set of all $|A|^n$ of the n-symbol strings that we want to see inside the de Brujin sequence. It is still tempting to have the de Bruijn sequence be a closed walk in the graph we define. To make this happen, we need the edges to answer the question: if string x is the n-symbol string at position i in the sequence, which strings could start at position $i+1$?

Question: Suppose that reading from position i, we see the symbol $x_0x_1\dots x_{n-1}$. What strings could we see, reading from position $i+1$?

Answer: We could see any of the sequences of the form $x_1x_2\dots x_n$, where $x_n \in A$ could be any symbol.

This is an asymmetric relationship, so we define a directed graph with all vertices of the form $x_0x_1\dots x_{n-1}$ and all arcs of the form $(x_0x_1\dots x_{n-1}, x_1x_2\dots x_n)$.

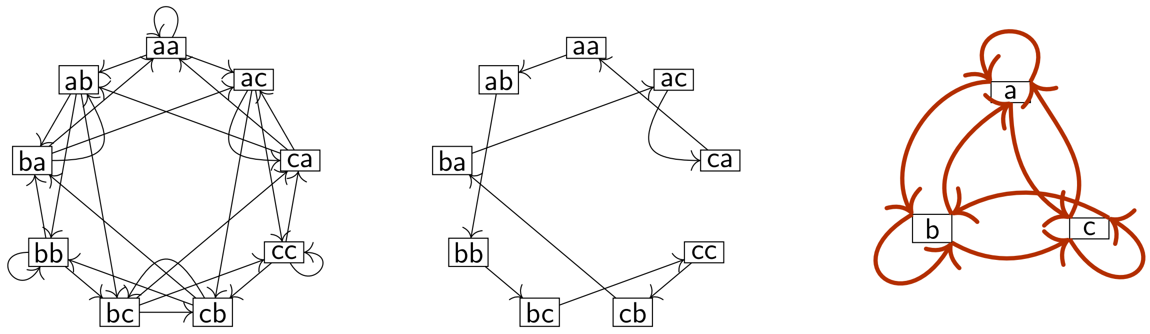
We call this directed graph the *de Brujin digraph*; the notation $B(k,n)$ is sometimes used for the de Bruijn digraph for strings of order n with a k-symbol alphabet. (Up to isomorphism of the graph, it does not matter what the alphabet is.) An example is shown in Figure 20.4a. I have kept $A = \{\mathsf{a},\mathsf{b},\mathsf{c}\}$, but to avoid making this graph too large, I have reduced n to 2.

Unfortunately, there's a problem. If we want to find a de Bruijn sequence of order n in the graph $B(k,n)$, we must find a Hamilton cycle in the graph, because we want to visit every vertex. Figure 20.4b shows such a cycle in $B(3,2)$. This gives us the de Bruijn sequence aabbccbac… but at what cost? Finding Hamilton cycles is very difficult, so we might as well have brute-forced the problem.

The trick is to realize the following incredibly convenient coincidence:

Proposition 20.3. *For all $k \geq 1$ and $n \geq 2$, the de Bruijn digraph $B(k,n)$ is isomorphic to the line digraph of $B(k,n-1)$.*

Proof. To find an isomorphism between $L(B(k,n-1))$ and $B(k,n)$, we need a function φ from the vertices of $L(B(k,n-1))$ (or the arcs of $B(k,n-1)$) to the vertices of $B(k,n)$.



(a) A diagram of $B(3,2)$ (b) A Hamilton cycle in $B(3,2)$ (c) An Euler tour in $B(3,1)$

Figure 20.4: Approaches to solving the de Bruijn sequence problems

An arc in $B(k, n-1)$ is a pair $(x_0x_1 \dots x_{n-2}, x_1x_2 \dots x_{n-1})$, where $x_0, x_1, \dots, x_{n-1} \in A$. Since the vertices of $B(k, n)$ are also defined by n symbols from A, the natural thing to try is to define

$$\varphi\Big((x_0x_1 \dots x_{n-2}, x_1x_2 \dots x_{n-1})\Big) = x_0x_1 \dots x_{n-1}$$

and see if this works.

Question: What does it mean for φ to work: what will make it an isomorphism?

Answer: We need to check if φ preserves arcs: we want (x, y) to be an arc in $L(B(k, n-1))$ if and only if $(\varphi(x), \varphi(y))$ is an arc in $B(k, n)$.

Take two vertices in $L(B(k, n-1))$: let one vertex be the pair $(x_0x_1 \dots x_{n-2}, x_1x_2 \dots x_{n-1})$, and let the other be the pair $(y_0y_1 \dots y_{n-2}, y_1y_2 \dots y_{n-1})$. There is an arc from the first two the second exactly when $x_1x_2 \dots x_{n-1} = y_0y_1 \dots y_{n-2}$, by the definition of a line graph.

Now apply φ to both vertices: we get $x_0x_1 \dots x_{n-1}$ and $y_0y_1 \dots y_{n-1}$. By the definition of $B(n, k)$, there is an arc from the first to the second exactly when $x_1x_2 \dots x_{n-1} = y_0y_1 \dots y_{n-2}$: the same condition!

Therefore φ really is an isomorphism, completing the proof. □

Question: How does Proposition 20.3 help us?

Answer: Instead of a Hamilton cycle in $B(k, n)$, we can look for an Euler tour in $B(k, n-1)$, which is much easier.

For example, Figure 20.4c shows the same de Bruijn sequence aabbccbac, but this time it is obtained from an Euler tour in $B(3, 1)$. (It's very hard to show Euler tours in a diagram; here, different visits to the same vertex touch it at different points. To obtain aabbccbac, start at the northeast corner of vertex a and follow the arrows.)

Theorem 20.4. *For any set of symbols A and any integer $n \geq 1$, there is a de Bruijn sequence of order n over A.*

Proof. For $n = 1$, just write all the symbols in A, one after the other.

For $n \geq 2$, we must prove that the digraph $B(k, n-1)$ is Eulerian, where $|A| = k$. By Corollary 8.6, we must check two properties of $B(k, n-1)$. (Both properties were defined in Chapter 8.)

First, we check that $B(k, n-1)$ is *weakly connected.* We can prove this by giving an $x - y$ walk for any two vertices $x = x_0x_1 \dots x_{n-2}$ and $y = y_0y_1 \dots y_{n-2}$. One possible walk passes through all vertices of the form

$$\underbrace{x_ix_{i+1} \dots x_{n-1}x_{n-2}}_{n-2-i \text{ symbols from } x}\underbrace{y_0y_1 \dots y_{i-2}y_{i-1}}_{i \text{ symbols from } y}$$

for $i = 0, 1, \dots, n-2$, in that order.

Second, we check that all vertices of $B(k, n-1)$ are *balanced*, with indegree equal to outdegree. In fact, each vertex $x_0x_1 \dots x_{n-2}$ has both indegree and outdegree k: it receives an arc from every vertex of the form $xx_0x_1 \dots x_{n-3}$, where $x \in A$, and sends an arc to every vertex of the form $x_1x_2 \dots x_{n-2}x$, where $x \in A$.

Question: Whenever you see that $n - 3$, you should be worried that $n - 3$ is negative. What happens then?

Answer: Then, interpret $x_0x_1 \dots x_{n-3}$, and for that matter $x_1x_2 \dots x_{n-2}$, as being empty. When $n = 2$, each vertex receives an arc from every vertex of the form x, where $x \in A$; that is, from every vertex.

By applying Corollary 8.6, we obtain an Euler tour in $B(k, n-1)$, which corresponds to a Hamilton cycle in $B(k, n)$, which gives us a de Bruijn sequence of order n. □

20.4 Edge coloring

So far, we've used line graphs to draw connections between ideas we've already seen; now, let's use them to define something new: the edge chromatic number. This invariant is also sometimes called the "chromatic index"; I don't like this terminology, because there's nothing about the words "number" and "index" suggesting that one is about vertices and the other is about edges.

If G is a graph, we define the *edge chromatic number* $\chi'(G)$ to be $\chi(L(G))$: the chromatic number of the line graph of G. We can also avoid invoking $L(G)$, and instead define $\chi'(G)$ to be the least number of colors needed for an *edge coloring* of G. This is a function $f \colon E(G) \to C$, where C is a set of colors, such that each vertex is incident to at most one edge of each color.[5]

Question: What are the color classes of an edge coloring?

Answer: The color classes of a vertex coloring of G are independent sets in G, so the color classes of an edge coloring of G are independent sets in $L(G)$: these correspond to matchings in G.

[5] As in Chapter 19, this is really the definition of a *proper edge coloring*, but we will not consider any other kind.

The upper and lower bounds from the previous chapter can also be used here: we have $\omega(G) \le \chi(G) \le \Delta(G)+1$ for all graphs G, so in particular, $\omega(L(G)) \le \chi(L(G)) \le \Delta(L(G))+1$.

Question: What does $\omega(L(G)) \le \chi(L(G))$ say, when translated from $L(G)$ back to G?

Answer: We've already seen that $\omega(L(G))$ is $\Delta(G)$, with one exception, and $\omega(L(G)) \ge \Delta(G)$ even then. Therefore $\chi'(G) \ge \Delta(G)$.

Question: What about the upper bound $\chi(L(G)) \le \Delta(L(G))+1$?

Answer: An edge xy shares an endpoint with at most $\Delta(G)-1$ other edges at x, and $\Delta(G)-1$ more at y; therefore xy has degree at most $2\Delta(G)-2$ in $L(G)$. We can conclude that $\chi'(G) \le 2\Delta(G)-1$.

The inequalities $\Delta(G) \le \chi'(G) \le 2\Delta(G)-1$ are intriguing: the lower and upper bound are both in terms of the maximum degree. However, much more will turn out to be true!

Although the definition of $\chi'(G)$ is new, we have already seen one very similar problem already. In Chapter 16, we looked at 1-factorizations, which are decompositions of a regular graph into perfect matchings.

Question: If a k-regular graph G has a 1-factorization, what does that say about $\chi'(G)$?

Answer: We can use the 1-factorization to get an edge coloring of G, by making each perfect matchings one of the color classes. There are k perfect matchings in the 1-factorization, so $\chi'(G) = k = \Delta(G)$: we already know this is the minimum possible.

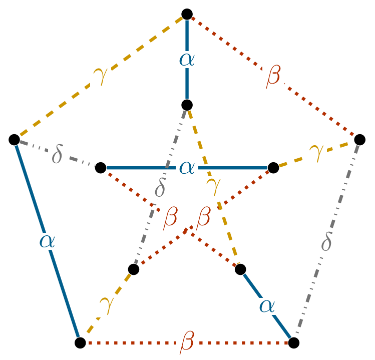
In Chapter 16, we proved two results about 1-factorizations. Theorem 16.3 says that for all even n, K_n has a 1-factorization; this means $\chi'(K_n) = n-1$ when n is even. Also, Theorem 16.5 says that every k-regular bipartite graph G has a 1-factorization; this means $\chi'(G) = k$.

With just a little bit of trickery, we can make the second result more powerful. Let G be any bipartite graph with maximum degree $\Delta(G)$. There are many ways to find a bipartite graph H which contains G as a subgraph, and is $\Delta(G)$-regular. (I will leave it to you to discover how to do this, in a practice problem at the end of this chapter.) Then $\chi'(H) = \Delta(G)$, and so in particular $\chi'(G) = \Delta(G)$: inside every edge coloring of H, there is an edge coloring of G. We conclude:

Corollary 20.5. *If G is any bipartite graph, then $\chi'(G) = \Delta(G)$.*

(This result, in somewhat different terminology, was proven by Kőnig in 1916 together with Theorem 16.5 [17].)

Thus far, we've seen many instances where $\chi'(G) = \Delta(G)$. Is this universal? No: for example, this will not happen for K_n when n is odd. In this case, the largest matching has only $\frac{n-1}{2}$



(a) An edge 4-coloring of the Petersen graph

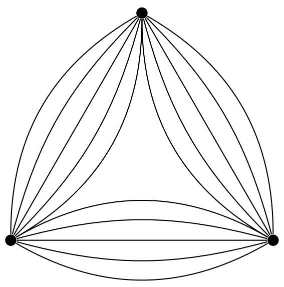

(b) All edges must have different colors

Figure 20.5: Two examples of lower bounds on edge coloring

edges, so it takes at least n matchings to cover all $\frac{n(n-1)}{2}$ edges of K_n. However, Corollary 16.4 says that when n is odd, K_n has a decomposition into n nearly perfect matchings; this means $\chi'(K_n) = n$ when n is odd.

Another interesting example is the Petersen graph. It is 3-regular, and has many perfect matchings, so you might be tempted to hope that it has edge chromatic number 3. However, this is not true. Suppose for contradiction that the Petersen graph has an edge coloring with 3 colors $\{\alpha, \beta, \gamma\}$. (It is traditional to use Greek letters for the colors of edges; this has nothing to do with the graph invariants $\alpha(G)$ or $\beta(G)$.)

To color all 15 edges, each color class must contain 5 edges and be a perfect matching. If we take the subgraph formed by the edges with colors α and β, it is 2-regular: each connected component is a cycle. Moreover, the edges around a cycle must alternate between the colors α and β, so each component has even length.

There are two ways to have a 10-vertex graph whose connected components are cycles of even length: it can be a copy of C_{10}, or the disjoint union of copies of C_4 and C_6. In the Petersen graph, neither is possible: the first case contradicts Proposition 17.1, where we proved that the Petersen graph does not have a Hamilton cycle, and the second case contradicts Lemma 5.3, where we proved that it has no cycles of length 3 or 4. So the Petersen graph cannot have an edge coloring with only 3 colors.

The Petersen graph does have an edge coloring with 4 colors, as shown in Figure 20.5a. It is another example of a graph G with $\chi'(G) = \Delta(G) + 1$.

Is this the limit? Not if we consider multigraphs. Figure 20.5b shows an example given by Claude Shannon [24]; in this multigraph, any two edges share an endpoint, so they must all be given different colors. If there are k copies of each edge of the triangle, then the graph has maximum degree $2k$, but edge chromatic number $3k$ (equal to the number of edges). Shannon also proved that this is the worst case: $\chi'(G) \le \frac{3}{2}\Delta(G)$, even if G is a multigraph.

For simple graphs, however, we have only seen examples where $\chi'(G)$ is either $\Delta(G)$ or $\Delta(G)+1$. In 1964, Vadim Vizing proved that this is true for all simple graphs G [26]. Since there are only two possible values of $\chi'(G)$, sometimes a graph G will be referred to as "class one" if $\chi'(G) = \Delta(G)$ and "class two" if $\chi'(G) = \Delta(G) + 1$.

I will conclude this chapter with a proof of Vizing's theorem:

Theorem 20.6 (Vizing's theorem)**.** *For all graphs G, $\chi'(G) \le \Delta(G) + 1$.*

20.5 Vizing's theorem

All proofs of Vizing's theorem that I am aware of rely on a variant of the alternating paths we used in Chapter 14 or Chapter 16 when working with matchings. This makes sense, since a color class in an edge coloring of G is an independent set in $L(G)$: a matching in G. However, instead of trying to improve a single matching, we will be trying to rearrange the existing matchings to make room for one more edge.

Suppose that α and β are two of the colors used in an edge coloring of a graph. The subgraph formed by edges of colors α and β is the union of two matchings, so its connected components consist of paths and cycles. A path component of this subgraph is called an *α/β-path.*

The colors of the edges of an α/β-path alternate between α and β. If x is the start or end of an α/β-path, then only one of the two colors is used on edges incident to x, and it is the edge beginning the α/β-path. Conversely, if only one of α or β is used on edges incident to some vertex x, then we can find an α/β-path starting at x, by greedily following edges of color α or β until we cannot continue.

The use of an α/β-path is that we can swap the colors α and β on every edge of the path and obtain a different edge coloring. We can hope to use such a swap to make color α or β available for an edge we have not yet colored.

The proof that follows is based on a proof of Ehrenfeucht, Faber, and Kierstead [9], simplified from their more general statement.

Proof of Theorem 20.6. For a fixed number of colors q, we induct on the number of vertices in G, proving that if $\Delta(G) \le q - 1$, then G has an edge q-coloring. We can take a 1-vertex graph as our base case, in which case no edges need to be colored at all. To induct, we take a graph G with $\Delta(G) \le q - 1$ and an arbitrary vertex x, and assume that $G - x$ has an edge q-coloring; we must find a way to turn it into an edge q-coloring of G.

As we color the edges incident to x one at a time, we keep a partition of these edges into three sets $S \cup T \cup U$, each with their own meaning:

- Edges in S are *settled*, and their color will not change.
- There is at most one edge in T, which is *tentatively* colored.
- Edges in U are *uncolored.*

Our algorithm will maintain the following invariant at all times: the colors on $E(G - x) \cup S \cup T$ must always be an edge coloring of $G - U$.

When we begin, we put all edges incident to x in U. These edges have a valuable flexibility: in $L(G)$, they have at most $\Delta(G) - 1$, or $q - 2$, neighbors that have been given a color, so they have at least 2 colors available to them. To preserve this flexibility, we select two *candidate colors* for each edge $xy \in U$, satisfying a second invariant. For as long as edge xy remains in U, it will have two candidate colors, which must not appear on edges incident to y, nor on edges in S; we allow them to appear on an edge in T, since it is colored only tentatively.

Given this setup, there are several cases where we can make quick progress:

- First, if $T = \varnothing$, choose an arbitrary edge in U; give it one of its candidate colors, and move it to T. Otherwise, we will assume $T = \{xy\}$ and let α be the color of edge xy.
- Second, if α is not a candidate color of any edge in U, move xy from T to S. If α is a candidate color of only one edge $xz \in U$, we still move xy from T to S, but also move xz from U to T, giving xz its other candidate color.
- Third, if some edge $xz \in U$ has a color $\beta \neq \alpha$, and no other edge in U has β as one of its candidate colors, give edge xz color β and move it from U to S directly.

If none of these apply, then every color that appears as a candidate color in U (which includes α) must appear at least twice. Every edge in U only has two candidate colors, so at most $|U|$ colors appear as candidate colors at all. At most $|S| + |U|$ colors appear on edges incident to x in any fashion: as a color on any edge, or as a candidate color. But $|S| + |U| \leq \Delta(G) - 1$, and $q = \Delta(G) + 1$, so we can pick a color γ different from all of these: not used on any edge in $S \cup T$, nor as a candidate color of any edge in U.

Let P be the α/γ-path starting at x (with xy as its first edge), and swap the colors α and γ along P.

Question: Can this swap interfere with the colors of edges in S, or with the candidate colors of edges in U?

Answer: Edges in S cannot have color α or γ, so they are untouched by the swap. An edge $xz \in U$ might have α as one of its candidate colors. If so, P might visit vertex z, but it will have to stop there, because z is not incident to any edges of color α.

Suppose this last possibility occurs: the path P ends at a vertex z such that $xz \in U$, and α is a candidate color of z. (The last edge of P must have had color γ, swapped to α.) Then we make one further modification: we replace α by γ as a candidate color of xz. After all, α is no longer available for z, but γ is.

Whether or not this happens, γ (the new color of xy) appears at most once as a candidate color, so we return to one of the cases where we can make quick progress. This lets us keep going until all edges are in S and the edge coloring of G is complete.

Question: How do we know that we eventually reach this point?

Answer: In each of the cases where we make quick progress, we move an edge "earlier in the alphabet": from T to S, or from U to T, or from U directly to S. This can only stop when all edges are in S.

This completes the induction step, showing that if $\chi'(G - x) \leq q$, then $\chi'(G) \leq q$ as well. By induction, $\chi'(G) \leq q$ for all graphs G with $\Delta(G) \leq q - 1$, proving Vizing's theorem. □

20.6 Practice problems

1. A graph G is called claw-free if it does not have a copy of the star graph S_4 as an induced subgraph. In other words, no vertex $x \in V(G)$ can have three neighbors y_1, y_2, y_3 with no edges between them.

   Prove that all line graphs are claw-free graphs.

2. Use an Euler tour to find a de Bruijn sequence of order 4 over the alphabet $A = \{0, 1\}$.

3. Prove that there is a length 99 cyclic sequence of 0's, 1's, and 2's such that among the substrings 00, 01, 02, 10, 11, 12, 20, 21, and 22, there is one that occurs 0 times in the string, one that occurs 1 time, one that occurs 2 times, one that occurs 10 times, one that occurs 11 times, one that occurs 12 times, one that occurs 20 times, one that occurs 21 times, and one that occurs 22 times.

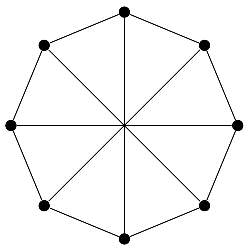
4. The line graph $L(K_{3,3})$ shown in Figure 20.2d has an unusual property: it is isomorphic to its own complement.

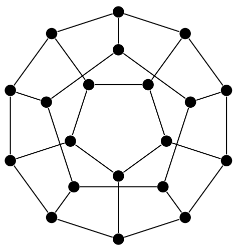
   a) Find one such isomorphism.

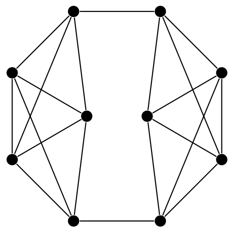
   b) Find an 8-vertex graph with the same property.

   c) Prove that there is no 10-vertex graph with this property.

5. Find the edge chromatic number of the following graphs:

a) 

b) 

c) 

6. To complete the proof of Corollary 20.5, we need to show that every bipartite graph G is a subgraph of a $\Delta(G)$-regular graph H.

   a) Suppose that G has a bipartition (A, B) with $|A| = |B|$. Prove that it is possible to add only edges to G, and no new vertices, to obtain a $\Delta(G)$-regular multigraph H.

      (How can you make use of this result if $|A| \neq |B|$?)

   b) Prove that if $\delta(G) < \Delta(G)$, then it is possible to add edges between two copies of G to obtain a bipartite graph G' with $\Delta(G') = \Delta(G)$ but $\delta(G') = \delta(G) + 1$.

      Conclude that there is a $\Delta(G)$-regular bipartite graph H containing G which has $2^{\Delta(G)-\delta(G)} \cdot |V(G)|$ vertices.

7. More recent proofs of Vizing's theorem, like the one in this chapter, are usually written so that they can also prove stronger claims. Here are two such claims.

   a) Modify the proof to show that if the vertices in G which have degree $\Delta(G)$ form an independent set, then $\chi'(G) = \Delta(G)$.

b) Modify the proof to show that if the subgraph of G induced by the vertices of degree $\Delta(G)$ is a forest, then $\chi'(G) = \Delta(G)$.

8. (BMO 1992) The edges of a connected n-vertex graph G are colored red, blue, and yellow so that each vertex has one incident edge of each color.[6]

   a) Prove that n must be even and that for all even $n > 2$, a graph with such an edge coloring exists.

   b) Suppose that for some subset $S \subseteq V(G)$, there are r red, b blue, and y yellow edges with exactly one endpoint in S. Prove that r, b, y are either all even or all odd.

[6] I have paraphrased; the original problem on the British Mathematical Olympiad had an elaborate and very long statement about dwarfs and orcs.

Bibliography

[1] W. W. Rouse Ball. *Mathematical Recreations and Essays.* Fourth. New York, NY: Macmillan and Company, 1905. URL: https://www.gutenberg.org/ebooks/26839.

[2] J. A. Bondy and V. Chvátal. "A method in graph theory". In: *Discrete Mathematics* 15.2 (1976), pp. 111–135. DOI: 10.1016/0012-365X(76)90078-9.

[3] Nicolaas Govert de Bruijn. "A combinatorial problem". In: *Proceedings of the Koninklijke Nederlandse Akademie van Wetenschappen* 49 (1946), pp. 758–764.

[4] Marcelo Campos et al. "An exponential improvement for diagonal Ramsey". In: *arXiv preprint* (2023). URL: https://arxiv.org/abs/2303.09521.

[5] Gregory Chaitin et al. "Register allocation via coloring". In: *Computer Languages* 6 (1 1981), pp. 47–57. DOI: 10.1016/0096-0551(81)90048-5.

[6] Václav Chvátal. "Tough graphs and Hamiltonian circuits". In: *Discrete Mathematics* 5 (1973), pp. 215–228. DOI: 10.1016/0012-365X(73)90138-6.

[7] Daniel Cranston and Landon Rabern. "Brooks' Theorem and Beyond". In: *Journal of Graph Theory* 80 (3 2015). DOI: 10.1002/jgt.21847.

[8] Gabriel Dirac. "Some theorems on abstract graphs". In: *Proceedings of the London Mathematical Society* 3.1 (1952), pp. 69–81. DOI: 10.1112/plms/s3-2.1.69.

[9] Andrzej Ehrenfeucht, Vance Faber, and Henry Kierstead. "A new method of proving theorems on chromatic index". In: *Discrete Mathematics* 52 (2–3 1984). DOI: 10.1016/0012-365X(84)90078-5.

[10] Pál Erdös. "Some remarks on the theory of graphs". In: *Bulletin of the American Mathematical Society* 53.4 (1947), pp. 292–294. DOI: 10.1090/S0002-9904-1947-08785-1.

[11] Robert Greenwood and Andrew Gleason. "Combinatorial relations and chromatic graphs". In: *Canadian Journal of Mathematics* 7 (1955), pp. 1–7. DOI: 10.4153/CJM-1955-001-4.

[12] Aubrey de Grey. "The chromatic number of the plane is at least 5". In: *arXiv preprint* (2018). URL: https://arxiv.org/abs/1804.02385.

[13] Herbert Grötzsch. "Zur Theorie der diskreten Gebilde, VII: ein Dreifarbenansatz für dreikreisfrei Netze auf der Kugel". In: *Wissenschaftliche Zeitschrift der Martin-Luther-Universität Halle-Wittenberg: Mathematisch-naturwissenschaftliche Reihe* 8 (1959), pp. 109–120.

[14] Parth Gupta et al. "Optimizing the CGMS upper bound on Ramsey numbers". In: *arXiv preprint* (2024). URL: https://arxiv.org/abs/2407.19026.

[15] Frank Harary and Robert Norman. "Some properties of line digraphs". In: *Rendiconti del Circolo Matematico di Palermo* 9 (1960), pp. 161–168. DOI: 10.1007/BF02854581.

[16] A. Hedayat and W. T. Federer. "On the nonexistence of Knut Vik designs for all even orders". In: *The Annals of Statistics* 3.2 (1975), pp. 445–447. DOI: 10.1214/aos/1176343068.

[17] Dénes Kőnig. “Über Graphen und ihre Anwendung auf Determinantentheorie und Mengenlehre”. In: *Mathematische Annalen* 77 (1916), pp. 453–465. DOI: 10.1007/BF01456961.

[18] Leo Moser and William Moser. “Problems for Solution (P10)”. In: *Canadian Mathematical Bulletin* 4 (2 1961), pp. 187–189. DOI: 10.1017/S0008439500025753.

[19] Jan Mycielski. “Sur le coloriage des graphs”. In: *Colloquium Mathematicum* 3 (2 1955), pp. 161–162. DOI: 10.4064/cm-3-2-161-162.

[20] George Pólya. “Uber die “doppelt-periodischen” Lösungen des n-Damen-Problems”. In: *W. Ahrens, Mathematische Unterhaltungen und Spiele* 1 (1921), pp. 364–374.

[21] Rob Pratt. *16 queens puzzle*. 2023. URL: https://puzzling.stackexchange.com/a/122418/47819 (visited on 10/02/2025).

[22] Stanisław Radziszowski. “Small Ramsey numbers”. In: *Electronic Journal of Combinatorics* (DS11 2024). DOI: 10.37236/21.

[23] Frank Ramsey. “On a problem of formal logic”. In: *Proceedings of the London Mathematical Society* 2 (1 1930), pp. 264–286. DOI: 10.1112/plms/s2-30.1.264.

[24] Claude Shannon. “A theorem on coloring the lines of a network”. In: *Journal of Mathematics and Physics* 28 (1–4 1949), pp. 148–152. DOI: 10.1002/sapm1949281148.

[25] Dan Thomasson. *Knight’s Tour Golden Star*. 2002. URL: http://danthomasson.com/kt-golden-star.html (visited on 10/01/2025).

[26] Vadim Vizing. “On an estimate of the chromatic class of a p-graph”. In: *Diskretnyi Analiz* 3 (1964), pp. 25–30.