



Xi'an Jiaotong-Liverpool University

西交利物浦大學

*Discrete Mathematics*  
MTH 229

Aistis Atminas

November 6, 2024

# Contents

<b>1</b>	<b>General framework for problem solving</b>	<b>2</b>
<b>2</b>	<b>Combinatorial methods of proof and counting principles</b>	<b>4</b>
2.1	Cardinalities of sets and the mathematical induction . . . . .	4
2.2	Combinatorial counting techniques . . . . .	11
2.2.1	Sum, Product and Division Rules . . . . .	11
2.2.2	Combinatorial properties of binomial coefficients . . . . .	13
2.3	Algebraic counting techniques . . . . .	16
2.3.1	Binomial Theorem . . . . .	16
2.3.2	Multinomial Theorem . . . . .	19
2.3.3	Principle of Inclusion and Exclusion (PIE) . . . . .	19
2.4	Two more methods of proof/disproof . . . . .	21
2.4.1	Pigeonhole Principle . . . . .	21
2.4.2	Principle of Invariance . . . . .	24
<b>3</b>	<b>Recursion</b>	<b>29</b>
3.1	Recurrence relations and generating functions . . . . .	29
3.1.1	The auxiliary equation method . . . . .	30
3.1.2	Generating functions . . . . .	32
3.1.3	Catalan, Bell and Stirling Numbers . . . . .	34
3.2	Recursive algorithms and complexity . . . . .	38
<b>4</b>	<b>Basic concepts in graph theory</b>	<b>42</b>
4.1	Graphs and their relatives . . . . .	42
4.2	Adjacency, neighbourhood, vertex degree, isomorphism, complement . . . .	45
4.3	Paths, cycles, trees and Kruskal's algorithm . . . . .	47
4.4	Special graphs and graph operations . . . . .	52
4.5	Cliques, independent sets and Ramsey theorem . . . . .	54
<b>5</b>	<b>Graph colouring</b>	<b>59</b>
5.1	Planar graphs . . . . .	59
5.2	Chromatic number and the 4-colour theorem . . . . .	62
5.3	SAT and NP-completeness of Colouring . . . . .	64
5.4	Interval graphs and chromatic polynomial . . . . .	69
<b>6</b>	<b>Introduction to spectral methods</b>	<b>73</b>
6.1	Formal definitions and terminology for matrices of graphs . . . . .	75
6.2	Good Will Hunting problems . . . . .	77

## Chapter 1

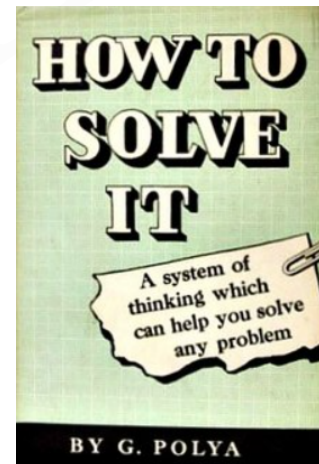
# General framework for problem solving

"An insect tries to escape through the windowpane, tries the same again and again, and does not try the next window which is open and through which it came into the room. A man is able, or at least should be able, to act more intelligently. Human superiority consists in going around an obstacle that cannot be overcome directly..." - an excerpt from "How to solve it" by George Polya.

In 1945, George Pólya published the book 'How To Solve It' which quickly became his most prized publication. The book has been translated into 17 languages, sold over a million copies and has been continuously in print since its first publication. In this book the author identifies four basic principles of problem solving.

1. First, you have to understand the problem.
2. After understanding, make a plan.
3. Carry out the plan.
4. Look back at your work. How could it be improved?

If this technique fails, Pólya advises: "If you can't solve a problem, then there is an easier problem you can solve: find it." Or: "If you cannot solve the proposed problem, try to solve first a related problem. Could you imagine a more accessible related problem?"



### First Principle: Understand the problem

This principle is often neglected as being obvious. Yet, even partial misunderstanding of the problem can hinder any further progress. So it's good to start by asking: "Can I understand all the words in the problem?" Or: "Can I restate the problem in my own words?" Or: "Would a picture/diagram/specific example be helpful to understand the problem?"

## **Second Principle: Devise a plan**

A partial list of strategies are as follows:

- Guess and check.
- Work backward: "Can you start with a goal and work backwards to what you already know?"
- Use induction: "Can you solve your problem by generalizing the observations for small examples?"
- Reduce to a solved problem: "Can you reduce your problem to a problem you already solved?"
- Rephrase: "Can you find a problem analogous to your problem and solve that instead?"
- Find a subproblem: "Can you find a subproblem whose solution would help you to solve your problem?"
- Design an algorithm: "Can you find a way to get a step closer to a solution and reach the solution in finitely many steps?"
- Make an orderly list and eliminate possibilities.
- Consider special cases and look for patterns.
- Draw a picture "Can you draw a picture of your problem?"
- Use symmetry.

## **Third Principle: Carry out the plan**

This step is generally easier than devising the plan. All you need is care and patience, provided you have the necessary skills. You should continue working on the plan persevering any difficulties along the way. If it continues not to work, discard the plan and choose another. Don't be discouraged, it happens even to professionals.

## **Fourth Principle: Review**

Having solved a problem is not the end of the story. Much can be gained by taking time to reflect and look back at what you have done, what worked and what did not. Look at the techniques you used and potential other problems it can be helpful to, and go through the problem again alone or explain it to your friends. This would highly deepen one's understanding, and will enable you to better predict what strategy to use to solve future problems.

## Chapter 2

# Combinatorial methods of proof and counting principles

### 2.1 Cardinalities of sets and the mathematical induction

A set (intuitively speaking) is a collection of objects. These objects are referred to as the elements of the set. Given a set  $A$  and an object  $x$  we will write  $x \in A$  if  $x$  is an element of  $A$ , and  $x \notin A$  otherwise.

Given two sets  $A$  and  $B$ , if every element of  $A$  is an element of  $B$ , we say  $A$  is a subset of  $B$ , or simply  $B$  contains  $A$ , and write  $A \subseteq B$  or  $B \supseteq A$ . If  $A \subseteq B$  and  $B \subseteq A$ , we write  $A = B$ .

The union of two sets  $A$  and  $B$ , is denoted by

$$A \cup B \text{ such that } x \in A \cup B \text{ if and only if } x \in A \text{ or } x \in B.$$

The intersection of two sets  $A$  and  $B$ , is the set denoted by

$$A \cap B, \text{ such that } x \in A \cap B \text{ if and only if } x \in A \text{ and } x \in B.$$

Given two sets  $A$  and  $B$  the difference  $B \setminus A$  is defined to be the set

$$B \setminus A = \{x \in B : x \notin A\}$$

In the special case when  $A \subseteq B$  we define its complement  $A^c$  (relative to  $B$ ) to be the set

$$A^c = B \setminus A = \{x \in B : x \notin A\}$$

A product of  $n$  sets  $A_1, A_2, \dots, A_n$  is the set

$$A_1 \times A_2 \times \dots \times A_n = \{(a_1, a_2, \dots, a_n) : a_i \in A_i \text{ for all } 1 \leq i \leq n\}$$

In other words, the product of  $n$  sets is the set containing ordered  $n$ -tuples, with  $i$ 'th element of each  $n$ -tuple coming from the  $i$ 'th set. When  $A_1 = A_2 = \dots = A_n = A$  we write

$$A^n = \{(a_1, a_2, \dots, a_n) : a_i \in A \text{ for all } 1 \leq i \leq n\}$$

Some sets we will be using in the course:

$\emptyset$  - the empty set (no elements),

$\mathbb{N} = \{1, 2, 3, \dots\}$ ,

$\mathbb{Z} = \{\dots, -2, -1, 0, 1, 2, \dots\}$ ,

$[n] = \{1, 2, \dots, n\}$ .

**Exercise 2.1.1** Which of the following are true:  $\emptyset = \{\emptyset\}$ ,  $\emptyset \in \{\emptyset\}$ ,  $\emptyset \in \{\{\emptyset\}\}$ ,  $\{\emptyset\} \subseteq \{\{\emptyset\}\}$ ?

Consider two sets  $A$  and  $B$  and suppose that with each element  $x$  of  $A$  there is associated some element of  $B$ , which we denote by  $f(x)$ . Then  $f$  is said to be a function from  $A$  to  $B$ . The set  $A$  is called the domain and the set of values of  $f$  is called the range of  $f$ . An example of a function could be an association of each person in the set  $A = \{\text{Alice, Bob, Cedric, Dong}\}$  with their motherland from the set  $B = \{\text{China, US, France, Germany}\}$  defined by  $f(\text{Alice}) = \text{China}$ ,  $f(\text{Bob}) = \text{US}$ ,  $f(\text{Cedric}) = \text{France}$ ,  $f(\text{Dong}) = \text{China}$ . In fact, in discrete mathematics, we will almost always study functions between two finite sets, and would almost never consider functions defined on real numbers as those functions belong to 'continuous' mathematics as opposed to 'discrete'.

Consider a function  $f$  from  $A$  to  $B$ . Given a set  $E \subseteq A$  we define  $f(E)$  to be the set of elements  $f(x)$  for  $x \in E$ . We call  $f(E)$  the image of  $E$  under  $f$ . In this notation,  $f(A)$  is the range of  $f$ . If  $f(A) = B$  we say that  $f$  is surjective. Similarly, if  $E \subseteq B$ ,  $f^{-1}(E)$  denotes the set of all  $x \in A$  such that  $f(x) \in E$ . We call  $f^{-1}(E)$  the inverse image of  $E$  under  $f$ . If  $y \in B$ ,  $f^{-1}(y)$  is the set of all  $x \in A$  such that  $f(x) = y$ . If for each  $y \in B$ ,  $f^{-1}(y)$  consists of at most one element of  $A$   $f$  is said to be one-to-one or injective. Alternatively, an injective function is the function such that whenever  $f(x_1) = f(x_2)$  holds for some  $x_1, x_2 \in A$ , then we have  $x_1 = x_2$ . If the function is both injective and surjective, then we say that it is bijective.

**Exercise 2.1.2** Consider the function  $f$  from the set of people  $A$  to the set of countries  $B$  as defined above. Explain whether the function is injective, surjective, bijective, or neither. Also find  $f(A)$ ,  $f^{-1}(\{\text{China, France}\})$ ,  $f^{-1}(\{\text{Germany}\})$ .

**Definition 2.1.3** (Cardinality of a set) We say that the two sets  $A$  and  $B$  have the same cardinality if there is a bijection  $f$  from  $A$  to  $B$ . In this case, we denote the common cardinality by  $|A| = |B|$ . Moreover, for  $n \geq 1$  we define

$$|\{1, 2, \dots, n\}| = n$$

and define  $|\emptyset| = 0$ . We will say that a set  $A$  is finite if  $|A| = n$  for some  $n \in \mathbb{N} \cup \{0\}$ .

To illustrate this definition, we will prove the following simple lemma.

**Lemma 2.1.4** Let  $A$  and  $B$  be two disjoint finite sets. Then

$$|A \cup B| = |A| + |B|.$$

**Proof.** If  $A$  or  $B$  is empty, the result follows easily (check it!). So assume that we have  $|A| = m$  and  $|B| = n$  for some  $m, n \in \mathbb{N}$ . By the definition, there exist bijections:  $f_A : A \rightarrow \{1, 2, \dots, m\}$  and  $f_B : B \rightarrow \{1, 2, \dots, n\}$ . Define  $f : A \cup B \rightarrow \{1, 2, \dots, n + m\}$  by setting

$$f(x) = \begin{cases} f_A(x) & \text{if } x \in A \\ f_B(x) + m & \text{if } x \in B. \end{cases}$$

Note that since  $A$  and  $B$  are disjoint,  $f$  is well-defined. Now we will prove that  $f$  is a bijection, by showing it is an injection and a surjection.

Let's check that  $f$  is an injection. Suppose  $f(x) = f(x')$  for some  $x, x' \in A \cup B$ . Then two cases arise: either  $f(x) \leq m$  or  $f(x) > m$ . In the first case  $x, x' \in A$  (since for any  $x \in B$  we

have  $f(x) > m$ ) and so we have  $f_A(x) = f_A(x')$ . Since  $f_A$  is bijective, we have  $x = x'$ . In the second case we have  $x, x' \in B$ , so  $f_B(x) + m = f_B(x') + m$  and so  $f_B(x) = f_B(x')$  and since  $f_B$  is bijective, we have  $x = x'$ . Hence  $f$  is injective.

To finish the proof, let's show that  $f$  is surjective. For any  $1 \leq i \leq n + m$  we need to find  $x \in A \cup B$  such that  $f(x) = i$ . If  $1 \leq i \leq m$ , take  $x = f_A^{-1}(i) \in A$ . Then since  $x \in A$ , we have  $f(x) = f_A(x) = i$ . Similarly, if  $m + 1 \leq i \leq n + m$ , we set  $x = f_B^{-1}(i - m) \in B$  and so  $f(x) = f_B(x) + m = i$ . Hence  $f$  is surjective.

Since the function  $f$  is a bijection between  $A \cup B$  and  $\{1, 2, \dots, n + m\}$  we conclude that  $|A \cup B| = n + m = |A| + |B|$  which finishes the proof.

■

**Exercise 2.1.5** With the function  $f$  and the set  $A$  as in the previous exercise, what are the cardinalities  $|A|$  and  $|f(A)|$ ?

Finding cardinalities of the sets is an important part of discrete mathematics. In fact, there is a branch of discrete mathematics, called enumerative combinatorics, devoted to finding the sizes of finite sets. Finding the sizes of the sets is of utmost importance for applications in probability and statistics, and also has many applications in computer science. We will develop a number of tools to find these cardinalities in the following sections, and we start our exploration with the following interesting question regarding sets of even sizes.

How many subsets of  $\{1, 2, 3, \dots, n\}$  have even cardinality?

**Exercise 2.1.6** [Pause to think] How would you approach this problem? Go through the steps of Pólya's methods and suggest how would you approach this problem.

I hope you had time to attempt this problem - highly suggested. Take your time. Don't worry if you have no clue initially, that is what is this course about - to develop research skills that enable you to make progress in the previously unknown territories.

Looking at Pólya's method a number of strategies look possible. For instance, if we somehow knew the goal, maybe we could work backwards to solve it. Also mathematical induction suggests attempting to solve small cases and hopefully generalize. Finally, even looking for symmetry seems to be promising, maybe there is some symmetry between numbers of even/odd sets? In any case, a good place to start is to consider some small examples and see what happens.

Take  $n=1$ :

$$\left\| \begin{array}{l} \text{Even subsets:} \\ \emptyset \end{array} \right\| \left\| \begin{array}{l} \text{Odd subsets:} \\ \{1\} \end{array} \right\|$$

Take  $n=2$ :

$$\left\| \begin{array}{l} \text{Even subsets:} \\ \emptyset \\ \{1, 2\} \end{array} \right\| \left\| \begin{array}{l} \text{Odd subsets:} \\ \{1\}, \{2\} \end{array} \right\|$$

Take  $n=3$ :

$$\left\| \begin{array}{c} \text{Even subsets:} \\ \emptyset \\ \{1, 2\}, \{1, 3\}, \{2, 3\} \end{array} \right\| \left\| \begin{array}{c} \text{Odd subsets:} \\ \{1\}, \{2\}, \{3\} \\ \{1, 2, 3\} \end{array} \right\|$$

Take  $n=4$ :

$$\left\| \begin{array}{c} \text{Even subsets:} \\ \emptyset \\ \{1, 2\}, \{1, 3\}, \{1, 4\}, \{2, 3\}, \{2, 4\}, \{3, 4\} \\ \{1, 2, 3, 4\} \end{array} \right\| \left\| \begin{array}{c} \text{Odd subsets:} \\ \{1\}, \{2\}, \{3\}, \{4\} \\ \{1, 2, 3\}, \{1, 2, 4\}, \{1, 3, 4\}, \{2, 3, 4\} \end{array} \right\|$$

The evidence clearly suggests that as  $n$  increases the number of the even subsets doubles. Since for  $n = 1$  we have 1 even subset (the empty set), we can guess that the answer for general  $n$  is  $2^{n-1}$ . It also looks that the same number of sets are of odd size.

To help proving statements of the form “ $P(n)$  holds for every  $n \in \mathbb{N}$ ” we introduce a very powerful method in mathematics called mathematical induction.

### Principle of Induction

Let  $P(n)$  be a statement about the natural number  $n$ .  
 Suppose  $P(1)$  is true. Suppose also that, if  $P(k)$  is true for  $k \geq 1$ , then  $P(k + 1)$  is also true. Then  $P(n)$  is true for all natural numbers  $n$ .

It might be very hard to prove the the statement  $P(n)$  directly, but by principle of induction, we only need to prove a single step that from  $P(k)$  we can obtain  $P(k + 1)$  (and that base case  $P(1)$  holds - which is often trivial). This is usually easier. To illustrate the mathematical induction, we will provide couple of examples.

**Example 2.1.7** *Prove that*

$$1^2 + 2^2 + 3^2 + \dots + n^2 = \frac{n(n + 1)(2n + 1)}{6}$$

**Proof.** The statement holds for  $n = 1$  (Check it!). Suppose the statement holds for a natural number  $n = k$ , i.e. suppose that

$$1^2 + 2^2 + \dots + k^2 = \frac{k(k + 1)(2k + 1)}{6}.$$

Then for  $n = k + 1$  we have

$$\begin{aligned} 1^2 + 2^2 + \dots + k^2 + (k + 1)^2 &= \frac{k(k + 1)(2k + 1)}{6} + (k + 1)^2 \\ &= (k + 1) \frac{k(2k + 1) + 6(k + 1)}{6} \\ &= (k + 1) \frac{2k^2 + 7k + 6}{6} \\ &= \frac{(k + 1)(k + 2)(2k + 3)}{6} \end{aligned}$$

Hence the statement holds for  $n = k + 1$ . By the principle of induction, the statement holds for all  $n \in \mathbb{N}$ . ■



**Example 2.1.8** A car circuit is going through the towns  $T_1, T_2, \dots, T_n$  in this cyclic order (so from  $T_i$  the route leads to  $T_{i+1}$  and from  $T_n$  back to  $T_1$ ), and a car is to travel around this circuit, starting from one of the towns. At the start of the journey, the tank of the car is empty, but in each town  $T_i$  it can pick up  $f_i$  amount of fuel. Show that if  $\sum_{i=1}^n f_i$  is precisely sufficient to drive round the entire circuit then there is a town such that if the car starts from there then it can complete the entire circuit without running out of fuel.

**Proof.** We will prove the statement by induction on  $n$ . If  $n = 1$ , i.e. there is only one town, then this town has sufficient fuel for a car to travel around the whole circuit.

Now suppose the statement holds for  $n = k$  towns. Consider a circuit with  $n = k + 1$  towns  $T_1, T_2, \dots, T_{k+1}$ . We note that there exists  $i \in \{1, 2, \dots, k + 1\}$ , such that the fuel  $f_i$  placed in the town  $T_i$  is sufficient to reach the town  $T_{i+1}$ . Indeed, if this were not true, we would have that  $\sum_{n=1}^{k+1} f_i$  was not sufficient to travel the whole circuit - a contradiction. Now we delete the town  $T_{i+1}$ , and update the amount of fuel in the town  $T_i$  to  $f_i + f_{i+1}$ . Since we now have  $k$  towns, and enough fuel to travel around the circuit, by the inductive assumption, there exists some  $j$  such that starting from town  $T_j$  we can travel around the whole  $k$ -town circuit. It is not hard to see that starting from  $j$  we can also travel around the whole original  $k + 1$ -town circuit (check this!). By induction, the result follows for all  $n$ .

■

**Exercise 2.1.9** To appreciate the power of induction, try to find alternative direct (non-inductive) proofs for the two examples above.

Let us return to our question on the number of even subsets of  $\{1, 2, \dots, n\}$ . We can now attempt to prove, by induction the statement  $P(n)$  where

$$P(n) = \text{“ The number of even subsets of } \{1, 2, \dots, n\} \text{ is } 2^{n-1} \text{ ”}$$

We know this holds for  $n = 1$ . Suppose it holds for  $n = k \geq 1$ , i.e. suppose there are  $2^{k-1}$  subsets of even size for some  $k \in \mathbb{N}$ . Consider now the even subsets of  $\{1, 2, \dots, k + 1\}$ . The key idea is to split the even subsets of  $\{1, 2, \dots, k + 1\}$  into those that contain the element ‘ $k + 1$ ’ and those who don’t:

$$\left\{ \begin{array}{l} \text{Even subsets of} \\ \{1, 2, \dots, k + 1\} \end{array} \right\} = \left\{ \begin{array}{l} \text{Even subsets of} \\ \{1, 2, \dots, k\} \end{array} \right\} + \left\{ \begin{array}{l} \text{Even subsets of } \{1, 2, \dots, k + 1\} \\ \text{which contain ‘} k + 1 \text{’} \end{array} \right\}$$

The first term in the sum we know by inductive assumption - the number of even subsets of  $\{1, 2, \dots, k\}$  is  $2^{k-1}$ . However, the second term is not that helpful. The number of even subsets of  $\{1, 2, \dots, k + 1\}$  that contain ‘ $k + 1$ ’ is precisely the number of odd subsets of  $\{1, 2, \dots, k\}$ , which we don’t know. It would be helpful, if the induction hypothesis told us the number of odd subsets as well. Putting these observations together we can now prove the result formally, presenting an example of the technique called “strengthening the inductive hypothesis”.

#### Formal inductive proof.

We will prove by induction that  $P'(n)$  holds for all  $n \in \mathbb{N}$ , where

$$P'(n) = \text{“ The set } \{1, 2, \dots, n\} \text{ has } 2^{n-1} \text{ even subsets and } 2^{n-1} \text{ odd subsets ”}$$

Of course,  $P'(1)$  is true, because the set  $\{1\}$  has two subsets -  $\emptyset$  and  $\{1\}$ , one of which has even size and the other has odd size. Suppose now  $P'(k)$  is true, i.e. suppose  $\{1, 2, \dots, k\}$  has  $2^{k-1}$  even and  $2^{k-1}$  odd subsets. Consider the set  $\{1, 2, \dots, k+1\}$ . Then as before,

$$\left\{ \begin{array}{l} \text{Even subsets of} \\ \{1, 2, \dots, k+1\} \end{array} \right\} = \left\{ \begin{array}{l} \text{Even subsets of} \\ \{1, 2, \dots, k\} \end{array} \right\} + \left\{ \begin{array}{l} \text{Even subsets of } \{1, 2, \dots, k+1\} \\ \text{which contain 'k+1'} \end{array} \right\}$$

By induction, the first term in the sum is a family of size  $2^{k-1}$ . The second term is a family, that consist of element ' $k+1$ ' and an odd subset of  $\{1, 2, \dots, k\}$ . Since by induction there are  $2^{k-1}$  odd subsets of  $\{1, 2, \dots, k\}$ , the second term is also a family of size  $2^{k-1}$ . Since these two families of the sets are disjoint, by Lemma 2.1.4 we deduce that there are  $2^{k-1} + 2^{k-1} = 2^k$  even subsets of  $\{1, 2, \dots, k+1\}$ .

Also note that

$$\left\{ \begin{array}{l} \text{Odd subsets of} \\ \{1, 2, \dots, k+1\} \end{array} \right\} = \left\{ \begin{array}{l} \text{Odd subsets of} \\ \{1, 2, \dots, k\} \end{array} \right\} + \left\{ \begin{array}{l} \text{Odd subsets of } \{1, 2, \dots, k+1\} \\ \text{which contain 'k+1'} \end{array} \right\}$$

Similarly as above (interchanging 'odd' with 'even' in the argument above), we can deduce that there are  $2^k$  odd subsets of  $\{1, 2, \dots, k+1\}$ . Thus we conclude that  $P'(k+1)$  holds.

By induction  $P'(n)$  holds for every  $n \in \mathbb{N}$ . This finishes the proof. ■

**Remark 2.1.10** *It is interesting (and counter-intuitive) to note that when we can't prove  $P(n)$ , sometimes it might be possible to find a more general statement  $P'(n)$  which could be easier to prove (here, more general means that  $P'(n) \Rightarrow P(n)$ ). This is what we did in the proof above, and could be described as "strengthening induction hypothesis  $P(n)$  to  $P'(n)$ ".*

**Remark 2.1.11** *There are couple of alternative forms of Principle of Induction. For example, if we know that  $P(17)$  is true, and that if  $P(k)$  holds, then  $P(k+1)$  holds for all  $k \geq 17$ , we can easily conclude that  $P(n)$  holds for  $n \geq 17$ . Indeed, to prove this formally, we can just set  $Q(n)$  to be the statement  $P(n+16)$  and apply the Principle of Induction to  $Q(n)$ .*

*Secondly, if we know that  $P(1)$  and  $P(2)$  are true, and whenever  $P(k)$  and  $P(k+1)$  are true, then  $P(k+2)$  is also true, we can conclude that  $P(n)$  holds for all  $n \in \mathbb{N}$ . Formally, we can set  $Q(n)$  to be the statement ' $P(n)$  and  $P(n+1)$  both hold' and use induction on  $Q(n)$ . Furthermore, by setting  $Q(n) = 'P(m) \text{ holds for all } m < n'$ , it's not hard to see that we can use all the preceding statements  $P(1), P(2), \dots, P(k)$  when proving that  $P(k+1)$  holds. This variant is known as "the strong induction".*

**Exercise 2.1.12** *We have proved if a subset of  $\{1, 2, \dots, n\}$  is chosen uniformly at random, the probability that it is even is exactly  $\frac{1}{2}$ . If 'even' is replaced by 'divisible by 3', would you expect the probability to be exactly  $\frac{1}{3}$ ?*

We end the section with some exercises to practice the principle of induction. Most of these exercises are taken from 'Discrete Mathematics' book by Susanna Epp. The examples illustrate that induction can be used to prove equalities (identities), inequalities, divisibility conditions and even existence of some combinatorial tiling patterns (see the last exercise).

**Exercise 2.1.13** *Prove that*

$$1 + 3 + 5 + \dots + (2n-1) = n^2.$$

**Exercise 2.1.14** Prove that

$$1^3 + 2^3 + \dots + n^3 = \left( \frac{n(n+1)}{2} \right)^2.$$

**Exercise 2.1.15** Prove that

$$\frac{1}{1 \cdot 2} + \frac{1}{2 \cdot 3} + \dots + \frac{1}{n \cdot (n+1)} = \frac{n}{n+1}$$

**Exercise 2.1.16** Prove that for  $n \geq 2$  we have

$$\prod_{i=2}^n \left(1 - \frac{1}{i}\right) = \frac{1}{n}.$$

**Exercise 2.1.17** Prove that for  $n \geq 2$  we have

$$\frac{1}{\sqrt{1}} + \frac{1}{\sqrt{2}} + \dots + \frac{1}{\sqrt{n}} > \sqrt{n}.$$

**Exercise 2.1.18** Show that  $5^n - 1$  is divisible by 4 for each  $n \in \mathbb{N}$ .

**Exercise 2.1.19** Show that  $n^3 - n$  is divisible by 6 for each  $n \in \mathbb{N}$ .

**Exercise 2.1.20** Show that  $2^{3^n} + 1$  is divisible by  $3^{n+1}$ .

**Exercise 2.1.21** Let  $n$  be an integer. Suppose one square is removed from  $2^n \times 2^n$  checkerboard. Show no matter which square has been removed, the remaining squares can be completely covered by the L-shaped trominos - tiles consisting of three squares arranged in an L shape. An example of such tiling is provided in Figure 2.1.

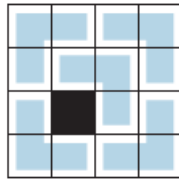


Figure 2.1: A tiling of  $4 \times 4$  checkerboard with L-shaped trominos

**Remark 2.1.22** Consider  $f(n) = n^2 + n + 41$ . It is easy to check that  $f(0) = 41, f(1) = 43, f(2) = 47, f(3) = 53, f(4) = 61, f(5) = 71$  are all prime numbers (only divisible by 1 and itself). Is  $f(n)$  prime for all integers  $n \geq 0$ ? This is an interesting function, where in fact, for all numbers  $n = 0, 1, 2, \dots, 39$ ,  $f(n)$  is always prime. Yet the statement that  $f(n)$  is prime for all  $n \geq 0$  is not true, since  $n = 40$  or  $n = 41$  provides us with easy counterexamples. We note that in mathematics counterexamples can be big, and just because the statement is true for the first 40 values, it doesn't imply that it holds for all values, in fact, the 41'st value might be a counterexample. This further exemplifies the power and the need for rigorous proof by induction.

## 2.2 Combinatorial counting techniques

This chapter is built on the three profound rules, upon which all combinatorial counting arguments rely. These are: sum rule, product rule, and division rule. From the product and the division rule, we also derive the binomial coefficients and explore their combinatorial properties.

### 2.2.1 Sum, Product and Division Rules

**Lemma 2.2.1 (Sum Rule)** *Suppose  $S_1, S_2, \dots, S_m$  are mutually disjoint finite sets and  $|S_i| = n_i$  for  $1 \leq i \leq m$ . Then the number of ways to select one object from any of the sets  $S_1, S_2, \dots, S_m$  is the sum  $n_1 + n_2 + \dots + n_m$ .*

**Proof.** The number of ways to select one object from any of the sets  $S_1, S_2, \dots, S_m$  equal to the cardinality of  $S_1 \cup S_2 \cup \dots \cup S_m$ . Hence, we want to prove that for disjoint subsets  $S_1, S_2, \dots, S_m$  we have

$$|S_1 \cup S_2 \cup \dots \cup S_m| = |S_1| + |S_2| + \dots + |S_m|.$$

The statement clearly holds for  $n = 1$ . Now suppose it holds for  $m = k \geq 1$ . Consider a family of  $k + 1$  mutually disjoint finite subsets  $S_1, S_2, \dots, S_{k+1}$ . Then,  $S_{k+1}$  must be disjoint from  $S_1 \cup S_2 \cup \dots \cup S_k$  (check this!). Hence, by Lemma 2.1.4 we have

$$|S_1 \cup S_2 \cup \dots \cup S_{k+1}| = |S_1 \cup S_2 \cup \dots \cup S_k| + |S_{k+1}|$$

Now, by induction hypothesis we have that  $|S_1 \cup S_2 \cup \dots \cup S_k| = |S_1| + |S_2| + \dots + |S_k|$ , so we conclude that

$$|S_1 \cup S_2 \cup \dots \cup S_{k+1}| = |S_1| + |S_2| + \dots + |S_k| + |S_{k+1}|.$$

Hence, by induction the result holds for all  $m \in \mathbb{N}$ . ■

**Exercise 2.2.2** *Suppose in the classroom there are 2 first-year, 5 second-year and 3 third-year and no fourth-year students. How many students are there in total?*

**Exercise 2.2.3** *Suppose in the classroom 2 students speak English, 5 speak Chinese and 3 speak German, and no-one speaks any other language. How many students are there in total?*

**Lemma 2.2.4 (Product Rule)** *Suppose  $S_1, S_2, \dots, S_m$  are finite sets and  $|S_i| = n_i$  for  $1 \leq i \leq m$ . Then the number of ways to select one object from  $S_1$ , followed by one object from  $S_2$ , and so on, ending with one element from  $S_m$ , is the product  $n_1 n_2 \dots n_m$ , provided that the selections are independent, that is the choice of elements from the sets  $S_1, S_2, \dots, S_{i-1}$  have no influence for selection from  $S_i$ , for every  $i$ .*

**Proof.** The number of ways to select one object from each  $S_i$  with  $1 \leq i \leq m$  is precisely the cardinality of the set  $S_1 \times S_2 \times \dots \times S_m$ . In fact, since  $|S_i| = n_i$ , it is not hard to see that the cardinality of  $S_1 \times S_2 \times \dots \times S_m$  is the same as the cardinality of  $E_1 \times E_2 \times \dots \times E_m$  with  $E_i = \{1, 2, \dots, n_i\}$  for  $1 \leq i \leq m$  (check this by providing a bijection between the two product sets!). Hence, it is enough to prove that for any family of finite sets  $E_1, E_2, \dots, E_m$  with  $E_i = \{1, 2, \dots, n_i\}$  for every  $1 \leq i \leq m$  we have

$$|E_1 \times E_2 \times \dots \times E_m| = |E_1| \times |E_2| \times \dots \times |E_m|.$$

We will prove this by induction on  $m$ . For  $m = 1$  the statement is trivial, so assume the statement holds for some  $m = k$ . Take  $k + 1$  sets  $E_1, E_2, \dots, E_{k+1}$ . The key idea is consider the elements of  $E_1 \times E_2 \times \dots \times E_{k+1}$  according to the last coordinate, by setting

$$X_j = \{(e_1, e_2, \dots, e_k, j) : e_i \in E_i \text{ for } 1 \leq i \leq k\}$$

for each  $j = 1, 2, \dots, n_{k+1}$ . Clearly, the cardinality of each  $X_j$  is  $|E_1 \times E_2 \times \dots \times E_k|$  which by inductive assumption is  $|E_1| \times |E_2| \times \dots \times |E_k|$ . Furthermore, the family  $X_1, X_2, \dots, X_{n_{k+1}}$  is mutually disjoint (the elements from different sets have different last coordinates), so by the Sum Rule we have

$$\begin{aligned} |E_1 \times E_2 \times \dots \times E_{k+1}| &= |X_1 \cup X_2 \cup \dots \cup X_{n_{k+1}}| \\ &= |X_1| + |X_2| + \dots + |X_{n_{k+1}}| \\ &= |E_1| \times |E_2| \times \dots \times |E_k| \times n_{k+1} \\ &= |E_1| \times |E_2| \times \dots \times |E_k| \times |E_{k+1}| \end{aligned}$$

This finishes the proof. ■

**Exercise 2.2.5** Suppose in the classroom there are 2 first-year, 5 second-year and 3 third-year and no fourth-year students. A pair of students not belonging to the same year has to be chosen for a project. How many ways can this be done?

**Exercise 2.2.6** Let  $A$  and  $B$  be two finite sets.

- (a) How many functions  $f : A \rightarrow B$  are there?
- (b) How many injective functions  $f : A \rightarrow B$  are there?

**Exercise 2.2.7** How many national flags can be constructed from three equal vertical strips, using colours red, white blue and green? (It is assumed that colours can be repeated, and that one vertical edge of the flag is distinguished as the ‘flagpole side’.)

**Exercise 2.2.8** A committee of nine people must elect a chairman, a secretary and a treasurer. In how many ways can this be done?

**Exercise 2.2.9** In how many ways can 5 people line-up in a queue?

**Exercise 2.2.10** How many people should there be in a room, so we would be at least 50% certain (more likely than not) that two of them share the same birthday? What should be the sample size to have 95% confidence?

The final core counting rule says that if we have a counting method that counts each object exactly  $k$  times, then the number of objects can be obtained by dividing the result by  $k$ . The formal statement is as follows:

**Lemma 2.2.11 (Division Rule)** If  $f : A \rightarrow B$  is a function such that for each  $b \in B$  there are exactly  $k$  elements  $a \in A$  with  $f(a) = b$ , then  $|B| = \frac{|A|}{k}$ .

**Proof.** Let  $g$  be a bijection from  $B$  to the set  $B' = \{1, 2, \dots, |B|\}$  and consider the function  $h = g \circ f : A \rightarrow B'$ , i.e.  $h(a) = g(f(a))$  for all  $a \in A$ . For any  $i \in B'$  consider  $A_i = h^{-1}(i) \subseteq A$ .

By definition of  $A_i$ , the subsets  $A_1, A_2, \dots, A_{|B|}$  are pairwise disjoint. Hence, by the Sum Rule we have

$$|A_1 \cup \dots \cup A_{|B|}| = \sum_{i=1}^{|B|} |A_i|.$$

Since  $A_1 \cup A_2 \dots \cup A_{|B|} = h^{-1}(B') = A$  and  $|A_i| = |h^{-1}(i)| = |f^{-1}(g^{-1}(i))| = k$  for each  $i = 1, 2, \dots, |B|$ , we conclude that

$$|A| = |B|k.$$

Hence the result follows. ■

**Definition 2.2.12 (Binomial coefficient)** Let  $n \geq k \geq 0$  be two integers. Then we define  $\binom{n}{k}$  to be the number of ways to choose a  $k$ -element subset from an  $n$ -element set. We will pronounce this number  $\binom{n}{k}$  as ‘ $n$  choose  $k$ ’ and occasionally refer to it as a binomial coefficient.

**Lemma 2.2.13**  $\binom{n}{k} = \frac{n!}{k!(n-k)!}$

**Proof.** By the product rule there are  $n \times (n-1) \times \dots \times (n-k+1)$  ways to choose a vector  $(a_1, a_2, \dots, a_k)$  with all the coordinates taken from the given  $n$ -element set being distinct from each other. Let us define a map  $f$  that maps the vector  $(a_1, a_2, \dots, a_k)$  to the subset  $\{a_1, a_2, \dots, a_k\}$ . Clearly each vector obtained by permuting the entries in the vector  $(a_1, a_2, \dots, a_k)$  will be mapped to the same subset  $\{a_1, a_2, \dots, a_k\}$ , so  $f$  maps  $k!$  different vectors to each  $k$ -element subset of the  $n$ -element set. Hence, by the division rule, the number of  $k$ -element subsets of the  $n$ -element set is

$$\frac{n(n-1) \times \dots \times (n-k+1)}{k!} = \frac{n!}{(n-k)!k!},$$

as required. Note that this also proves that  $\binom{n}{k}$  is well-defined. ■

The following are two further exercises that illustrate the use of the division rule.

**Exercise 2.2.14** How many ways are there to arrange 5 people in a dance circle (two circles are considered the same if one can be obtained from the other by rotation).

**Exercise 2.2.15** In how many different ways can  $2n$  students be paired up?

## 2.2.2 Combinatorial properties of binomial coefficients

The remaining of the chapter will present several insights about coefficients  $\binom{n}{k}$  that can help in solving various problems. In the first example, we present by an example how identity could be proven just using the definition of binomial coefficient. This technique might help in the cases, where the algebraic manipulations are too technical to handle.

**Lemma 2.2.16** For any  $n \geq r > 0$  we have

$$\binom{n}{r} = \binom{n-1}{r-1} + \binom{n-1}{r}$$

**Proof.** Let

$\mathcal{A}$  = ‘subsets of  $\{1, 2, \dots, n\}$  of size  $r$ ’.

Consider

$\mathcal{A}_1$  = ‘subsets of  $\{1, 2, \dots, n\}$  of size  $r$  containing element 1’

and

$\mathcal{A}_2$  = ‘subsets of  $\{1, 2, \dots, n\}$  of size  $r$  not containing element 1’

Clearly  $|\mathcal{A}| = \binom{n}{r}$ , while  $|\mathcal{A}_1| = \binom{n-1}{r-1}$  and  $|\mathcal{A}_2| = \binom{n-1}{r}$  (check this!). Since  $\mathcal{A}_1$  and  $\mathcal{A}_2$  are disjoint, by the sum rule the equality follows. ■

**Lemma 2.2.17** *For any  $n \geq k \geq 0$  we have*

$$\binom{n}{k} = \binom{n}{n-k}$$

**Proof.** Let

$\mathcal{A}$  = ‘subsets of  $\{1, 2, \dots, n\}$  of size  $k$ ’.

Set

$\mathcal{B}$  = ‘subsets of  $\{1, 2, \dots, n\}$  of size  $n - k$ ’.

Clearly there is a bijection  $f : \mathcal{A} \rightarrow \mathcal{B}$  sending  $S \in \mathcal{A}$  to its complement  $\{1, 2, \dots, n\} \setminus S \in \mathcal{B}$ . Hence  $|\mathcal{A}| = |\mathcal{B}|$  and therefore the equality follows. ■

**Exercise 2.2.18** *Interpreting each term combinatorially, prove that the following identity holds for all  $n \in \mathbb{N}$ :*

$$\binom{n}{0} + \binom{n}{1} + \dots + \binom{n}{n} = 2^n.$$

**Exercise 2.2.19** *Interpreting each term combinatorially, prove that the following identity holds for all  $n \in \mathbb{N}$ :*

$$\binom{n}{0}^2 + \binom{n}{1}^2 + \dots + \binom{n}{n}^2 = \binom{2n}{n}.$$

Now, let us move to considering arrangements of the letters.

**Example 2.2.20** *How many 0-1 sequences are there of length 5 with exactly two 1's?*

**Proof.** There are 10 such sequences and we can simply list them

$$\{11000, 10100, 10010, 10001, 01100, 01010, 01001, 00110, 00101, 00011\}$$

To check our answer combinatorially, note that we need to pick 2 positions out of 5 where the number 1 is placed and this uniquely determines the sequence. There are  $\binom{5}{2} = 10$  ways to do it, hence our answer is correct. ■

**Example 2.2.21** *How many different words (not necessarily meaningful) can we obtain by rearranging the letters of the word MATHEMATICS?*

**Proof.** The word has 11 letters, which arranged alphabetically are AACEHIMMTTS. There are couple of ways to solve this question. First of all, we can mimic the proof of the previous exercise.

- We need to choose two positions for letter A, which we can do  $\binom{11}{2} = \frac{11 \times 10}{2}$  ways.
- We place letter C into one of the remaining 9 positions .
- We place letter E into one of the remaining 8 positions.
- We place letter H - 7 positions.
- We place letter I - 6 positions.
- Two letters M -  $\binom{5}{2} = \frac{5 \times 4}{2}$  positions.
- Two letters T -  $\binom{3}{2} = \frac{3 \times 2}{2}$  positions.
- Letter S goes into the final 1 position.

By the product rule, we obtain there are  $\frac{11!}{2 \times 2 \times 2}$

Alternatively, we can solve it using the division rule, which might be even more enlightening. In total, there are  $11!$  ways to arrange the indexed letters  $A_1 A_2 C E H I M_1 M_2 T_1 T_2 S$  (treating letters with different indexes as different letters). Now, switching  $A_1$  with  $A_2$  corresponds to the same word with the original unindexed letters, as does switching  $M_1$  and  $M_2$  or  $T_1$  and  $T_2$ . Thus in total  $2 \times 2 \times 2$  indexed words correspond to each unindexed word. By division rule there are  $\frac{11!}{2 \times 2 \times 2}$  words obtained by arranging letters MATHEMATICS. ■

**Exercise 2.2.22** *How many different words (not necessarily meaningful) can be obtained by rearranging the letters of the word LETTERICITY?*

**Exercise 2.2.23** *How many different words (not necessarily meaningful) can be obtained from rearranging the letters of the word which has  $k$  different letters, where the  $i$ 'th letter is appears  $n_i$  times in the word for  $1 \leq i \leq k$ .*

Now we move to the last problem of our chapter:

Suppose the shop has four types of pens: blue, green, red and yellow. We would like to buy 10 pens in total. How many ways can we do this?

We can choose, for example, 3 blue, 2 green, 4 red and 1 yellow pen. At the check-out we place these pens on a tray in the following order: all blue pens first, then all green, then red and finally yellow, and separate different colours by bars '|'. Then our selection looks like:

$BBB|GG|RRRR|Y$

In fact, the bars uniquely determine the letters between them, so instead of the letters, we can just write '0' and we would still be able to recover the collection of pens chosen. In our case the code is:

$000|00|0000|0$



In this way, any selection can be transferred to a code, and vice versa, given any code with 10 zeros and 3 bars, we can uniquely determine the collection chosen. For example, the code

$$0|0000||00000$$

tells us that we chose 1 blue, 4 green, no red and 5 yellow pens, while

$$|0000000000||$$

denotes that we picked 10 green pens and no pens of other colours. We conclude that the number to select pens is the number of 13-symbol codes containing 3 bars, which is

$$\binom{13}{3}.$$

**Exercise 2.2.24** *How many ways are there to buy 5 fruits from a shop that has apples, oranges and pears available?*

**Exercise 2.2.25** *How many solutions in non-negative integers are there to the following equation:*

$$x_1 + x_2 + x_3 = 5$$

**Exercise 2.2.26** *How many solutions in strictly positive integers are there to the following equation:*

$$x_1 + x_2 + x_3 = 8$$

**Exercise 2.2.27** *What is the number of ways to distribute  $k$  one pound coins to  $n$  friends?*

## 2.3 Algebraic counting techniques

In the previous section we learned combinatorial counting arguments. In this section, we will be counting and proving identities with more algebraic methods: using equations and formulas. The two main counting techniques presented will be the binomial/multinomial theorems and the principle of inclusion-exclusion (PIE).

### 2.3.1 Binomial Theorem

**Theorem 2.3.1 (Binomial Theorem)** *For any  $n \in \mathbb{N}$  we have:*

$$(x + y)^n = \sum_{k=0}^n \binom{n}{k} x^k y^{n-k}$$

**Proof 1.** It is clear that  $(x + y)^n$  is a polynomial of degree  $n$  in two variables  $x$  and  $y$ . Now, to obtain a coefficient in front of  $x^k y^{n-k}$  consider the product

$$(x + y)(x + y) \dots (x + y) \quad (n \text{ factors}).$$

The term  $x^k y^{n-k}$  is obtained when from  $x$  is chosen from  $k$  of the factors and  $y$  is chosen from the remaining  $n - k$  factors. Since the number to choose these factors is  $\binom{n}{k}$  the coefficient in front of  $x^k y^{n-k}$  is  $\binom{n}{k}$ . ■

**Proof 2.** We will prove by induction on  $n$ . For  $n = 1$  the result holds trivially (check it!). Assume now the result holds for some  $n = k \geq 1$ . Then we have:

$$\begin{aligned}(x+y)^{k+1} &= (x+y)(x+y)^k \\ &= (x+y)\left(\binom{k}{0}x^k + \binom{k}{1}x^{k-1}y + \dots + \binom{k}{k}y^k\right) \\ &= \binom{k}{0}x^{k+1} + \binom{k}{1}x^k y + \binom{k}{2}x^{k-1}y^2 + \dots + \binom{k}{k}xy^k \\ &\quad + \binom{k}{0}x^k y + \binom{k}{1}x^{k-1}y^2 + \dots + \binom{k}{k-1}xy^k + \binom{k}{k}y^{k+1}\end{aligned}$$

Noting that  $\binom{k}{0} = \binom{k+1}{0}$ ,  $\binom{k}{k} = \binom{k+1}{k+1}$  and that by Lemma 2.2.16 we have that for any  $i = 1, 2, \dots, k$

$$\binom{k}{i} + \binom{k}{i-1} = \binom{k+1}{i},$$

we conclude that

$$(x+y)^{k+1} = \binom{k+1}{0}x^{k+1} + \binom{k+1}{1}x^k y + \binom{k+1}{2}x^{k-1}y^2 + \dots + \binom{k+1}{k+1}y^{k+1}.$$

This proves the case  $n = k + 1$  and hence by induction the result holds for every  $n \in \mathbb{N}$ . ■

The binomial theorem is a useful tool when proving various identities involving binomial coefficients.

**Example 2.3.2** *Prove that:*

$$(a) \quad 2^n = \binom{n}{0} + \dots + \binom{n}{n}$$

$$(b) \quad \binom{n}{0} + \binom{n}{2} + \dots = \binom{n}{1} + \binom{n}{3} + \dots$$

**Proof.**

$$(a) \quad \text{Set } x = y = 1 \text{ in the Binomial Theorem. Then } 2^n = (1+1)^n = \binom{n}{0} + \dots + \binom{n}{n}.$$

$$(b) \quad \text{Set } x = 1, y = -1 \text{ in the Binomial Theorem. Then}$$

$$0^n = (1-1)^n = \binom{n}{0} - \binom{n}{1} + \binom{n}{2} - \binom{n}{3} + \dots + (-1)^n \binom{n}{n}$$

and hence the equality follows.

■

Note that this gives us an easy proof that there are equal numbers of even and odd subsets of  $\{1, 2, \dots, n\}$ . In fact, we can also find the number of subsets whose size is divisible by 3.

**Example 2.3.3** *Prove that:*

$$\sum_{k=0}^{\lfloor n/3 \rfloor} \binom{n}{3k} = \frac{1}{3} \times (2^n + 2\cos(n\pi/3))$$

**Proof.** Consider the number

$$\zeta = e^{i2\pi/3} = \cos(2\pi/3) + i\sin(2\pi/3).$$

Notice that  $\zeta^3 = 1$ , so  $1 - \zeta^3 = (1 - \zeta)(1 + \zeta + \zeta^2) = 0$  and since  $\zeta \neq 1$  we have

$$1 + \zeta + \zeta^2 = 0.$$

By the Binomial Theorem we obtain:

$$\begin{aligned} (1 + 1)^n &= \sum_{k=0}^n \binom{n}{k} \\ (1 + \zeta)^n &= \sum_{k=0}^n \binom{n}{k} \zeta^k \\ (1 + \zeta^2)^n &= \sum_{k=0}^n \binom{n}{k} \zeta^{2k} \end{aligned}$$

Adding the three equations together we obtain:

$$(1 + 1)^n + (1 + \zeta)^n + (1 + \zeta^2)^n = \sum_{k=0}^n \binom{n}{k} (1 + \zeta^k + \zeta^{2k}) \quad (2.1)$$

The key here is that since  $\zeta^3 = 1$  and  $1 + \zeta + \zeta^2 = 0$ , it follows that

$$1 + \zeta^k + \zeta^{2k} = \begin{cases} 3, & \text{if } k \text{ divisible by } 3; \\ 0, & \text{if } k \text{ is not divisible by } 3. \end{cases}$$

Hence the right hand side of the equation (2.1) is

$$3 \times \sum_{k=0}^{\lfloor n/3 \rfloor} \binom{n}{3k}.$$

On the other hand, the left hand side of (2.1) is equal to

$$\begin{aligned} 2^n + (1 + \zeta)^n + (1 + \zeta^2)^n &= 2^n + (-\zeta^2)^n + (-\zeta)^n \\ &= 2^n + (e^{i\pi} \times e^{i4\pi/3})^n + (e^{i\pi} \times e^{i2\pi/3})^n \\ &= 2^n + (e^{i\pi/3})^n + (e^{-i\pi/3})^n \\ &= 2^n + 2\cos(n\pi/3) \end{aligned}$$

Hence the equality follows. ■

Apart from clever substitutions, we can also use factorization of polynomials and tools from calculus such as differentiation.

**Example 2.3.4** Prove that for any  $n \in \mathbb{N}$  we have

$$\binom{n}{0}^2 + \binom{n}{1}^2 + \dots + \binom{n}{n}^2 = \binom{2n}{n}.$$

**Proof.** By the Binomial Theorem we have

$$\begin{aligned}(1+x)^{2n} &= (1+x)^n \times (1+x)^n \\ &= \left( \binom{n}{0} + \binom{n}{1}x + \dots + \binom{n}{n}x^n \right) \times \left( \binom{n}{0} + \binom{n}{1}x + \dots + \binom{n}{n}x^n \right)\end{aligned}$$

Multiplying out the two factors in the expansion above we can see that the coefficient of  $x^n$  is

$$\binom{n}{0}\binom{n}{n} + \binom{n}{1}\binom{n}{n-1} + \dots + \binom{n}{n}\binom{n}{0},$$

and since  $\binom{n}{k} = \binom{n}{n-k}$  for every  $k = 0, 1, \dots, n$ , this coefficient is equal to

$$\binom{n}{0}^2 + \binom{n}{1}^2 + \dots + \binom{n}{n}^2.$$

On the other hand, by the Binomial Theorem the coefficient of  $x^n$  in the expansion of  $(1+x)^{2n}$  is  $\binom{2n}{n}$ . Hence the result follows. ■

**Exercise 2.3.5** Prove that

$$\sum_{k=1}^n k \binom{n}{k} = n \times 2^{n-1}.$$

### 2.3.2 Multinomial Theorem

The Binomial Theorem can be generalized as follows.

**Definition 2.3.6 (Multinomial coefficient)** Let  $n \in \mathbb{N}$  and suppose  $n = n_1 + n_2 + \dots + n_k$  for some  $n_1, n_2, \dots, n_k \in \mathbb{N}$ . We define

$$\binom{n}{n_1, n_2, \dots, n_k} = \frac{n!}{n_1! n_2! \dots n_k!}.$$

**Theorem 2.3.7 (Multinomial Theorem)** For any  $n \in \mathbb{N}$  we have

$$(x_1 + x_2 + \dots + x_k)^n = \sum_{n_1 + n_2 + \dots + n_k = n} \binom{n}{n_1, n_2, \dots, n_k} x_1^{n_1} x_2^{n_2} \dots x_k^{n_k}$$

**Proof.** Follows by similar arguments as the Binomial Theorem, namely by counting the number of times the term  $x_1^{n_1} x_2^{n_2} \dots x_k^{n_k}$  appears in the product of factors at the left hand side of the equation. ■

### 2.3.3 Principle of Inclusion and Exclusion (PIE)

The final algebraic tool of counting is the Principle of Inclusion and Exclusion (PIE). To understand the principle, we first consider a simple example.

**Example 2.3.8** Suppose in the classroom each student speaks at least one of the three languages: chinese, english, spanish. Suppose:

- 20 students speak chinese,
- 15 students speak english,
- 10 students who speak spanish,
- 12 students speak both chinese and english,
- 8 students speak both chinese and spanish,
- 6 students speak both english and spanish
- 5 students speak all three languages

How many students are there?

**Informal proof.** If we added 20 chinese speaking, 15 english speaking and 10 spanish speaking students, we would get  $20 + 15 + 10 = 45$  students. This would be a good answer if nobody spoke two languages, as the three language groups would be disjoint and the result would follow by the Sum Rule. However, the sets are not disjoint as there are people speaking two or even 3 languages. To compensate for this overcounting, we remove all double-counted students: 12 chinese-english, 8 chinese-spanish and 6 english-spanish speakers. So we now have  $45 - (12 + 8 + 6) = 45 - 26 = 19$  students. This number now counts correctly single-language people, and bilingual people, but after removal no longer includes trilingual people (why? draw appropriate Venn diagrams to illustrate this). Thus we need to add trilingual people back, getting the final answer  $19 + 5 = 24$ . ■

This process, of adding and removing sums of intersections of sets is called the Principle of Inclusion and Exclusion.

**Theorem 2.3.9 (Principle of Inclusion and Exclusion)** *Let  $A_1, A_2, \dots, A_n$  be finite sets. Then*

$$\begin{aligned}
 |A_1 \cup A_2 \cup \dots \cup A_n| &= (|A_1| + |A_2| + \dots + |A_n|) \\
 &\quad - (|A_1 \cap A_2| + |A_1 \cap A_3| + \dots + |A_{n-1} \cap A_n|) \\
 &\quad + (|A_1 \cap A_2 \cap A_3| + |A_1 \cap A_2 \cap A_4| + \dots + |A_{n-2} \cap A_{n-1} \cap A_n|) \\
 &\quad \dots \\
 &\quad + (-1)^{(n-1)} |A_1 \cap A_2 \cap \dots \cap A_n|
 \end{aligned}$$

**Proof.** We will prove the statement by induction on  $n$ . If  $n = 1$  or  $n = 2$ , the statement holds (check this!). So assume the statement holds for some  $n = k \geq 2$ . Applying the induction hypothesis for two sets  $A = A_1 \cup \dots \cup A_k$  and  $B = A_{k+1}$  we have

$$\begin{aligned}
 &|A_1 \cup A_2 \cup \dots \cup A_{k+1}| \\
 &= |A_1 \cup A_2 \cup \dots \cup A_k| + |A_{k+1}| - |(A_1 \cup A_2 \cup \dots \cup A_k) \cap A_{k+1}| \\
 &= |A_1 \cup A_2 \cup \dots \cup A_k| + |A_{k+1}| - |(A_1 \cap A_{k+1}) \cup (A_2 \cap A_{k+1}) \cup \dots \cup (A_k \cap A_{k+1})|
 \end{aligned}$$

Since the two terms  $|A_1 \cup A_2 \cup \dots \cup A_k|$  and  $|(A_1 \cap A_{k+1}) \cup (A_2 \cap A_{k+1}) \cup \dots \cup (A_k \cap A_{k+1})|$  are unions of  $k$  sets, we can apply inductive hypothesis to each of them. Writing down the inductive expansions of these two terms the result easily follows (check this!).

■

**Example 2.3.10** How many numbers are there in the set  $\{1, 2, \dots, 100\}$  which are coprime to 100?

**Proof.** Note that  $100 = 2^2 \times 5^2$ . So the numbers that are coprime to 100, are the ones that are not divisible by 2 or 5. Denote  $A_2 = \{2, 4, 6, 8, \dots, 100\}$  and  $A_5 = \{5, 10, 15, 20, \dots, 100\}$ . Then the number of coprime numbers is

$$100 - |A_2 \cup A_5| = 100 - (|A_2| + |A_5| - |A_2 \cap A_5|).$$

Note that  $|A_2| = 50$ ,  $|A_5| = 20$  and  $|A_2 \cap A_5| = 10$ . It follows that the answer is 40. ■

**Exercise 2.3.11** How many numbers are there in the set  $\{1, 2, \dots, 60\}$  which are coprime to 60?

**Remark 2.3.12** One can generalize the previous examples to any natural number  $n \in \mathbb{N}$ . Let  $\phi(n)$  be a number of integers  $m \leq n$  which are coprime to  $n$ . This function, called the Euler  $\phi$  function, or the Euler totient function, is important in number theory. Given that  $n$  is divisible by  $r$  different primes  $p_1, p_2, \dots, p_r$  we denote  $A_i = \{p_i, 2p_i, 3p_i, \dots, n\}$  to be the set of all numbers not greater than  $n$  which are divisible by  $p_i$ . Clearly  $|A_i| = \frac{n}{p_i}$ ,  $|A_i \cap A_j| = \frac{n}{p_i p_j}$ , and so on... Hence

$$\begin{aligned} \phi(n) &= n - \sum_{1 \leq i \leq r} \frac{n}{p_i} + \sum_{1 \leq i < j \leq r} \frac{n}{p_i p_j} - \sum_{1 \leq i < j < k \leq r} \frac{n}{p_i p_j p_k} + \dots + (-1)^r \frac{n}{p_1 p_2 \dots p_r} \\ &= n \prod_{i=1}^r \left(1 - \frac{1}{p_i}\right). \end{aligned}$$

We note that Euler's formula is just one of the examples of the use of inclusion-exclusion principle in number theory. The principle can be used for various questions including divisibilities, for example, to estimate the number of primes in arithmetic sequences. Meanwhile, a beautiful generalization of the theory of inclusion-exclusion is known by the name of the theory of the Mobius function.

**Example 2.3.13** Suppose that  $n$  students are planning to participate in Secret Santa Christmas gift exchange. For that each student randomly draws a name from a bag that contains all names of all students participating (and then he or she will secretly prepare a gift for the student whose name he or she found). A draw is considered to be successful, if every student received draws someone else's name (finding your own name would mean giving a gift to yourself which is not desirable). What is the probability that the draw is successful without anybody needing to redraw? How does this probability change as  $n \rightarrow \infty$ ?

## 2.4 Two more methods of proof/disproof

In this section we will provide two more simple, but powerful ideas of combinatorial proof: Pigeonhole Principle and Invariance Principle. These can be used to proof or disprove various combinatorial statements.

### 2.4.1 Pigeonhole Principle

The simplest version of pigeonhole principle states that if  $n$  items are put into  $m$  containers, with  $n > m$ , then at least one container must contain more than one item. For example, given that the population of Suzhou is greater than the maximum number of hairs that can be present on a human's head, then by the pigeonhole principle it follows that there must be at least two people in Suzhou who have the same number of hairs on their heads.

The principle is sometimes also called Dirichlet's box principle and has several generalizations that can be stated in various ways. In this course we will refer to Pigeonhole principle as follows:

**(Generalized) Pigeonhole Principle**

Let  $n$  and  $m$  be positive integers. If  $n$  objects are distributed among  $m$  containers then at least one of the containers contain  $\lceil \frac{n}{m} \rceil$  objects.



Figure 2.2:  $n=10$  pigeons in  $m=9$  holes (source: Wikipedia)

Note that  $\lceil x \rceil$  denotes the ceiling function, which is the smallest integer bigger or equal to  $x$  (for instance  $\lceil 5.1 \rceil = 6$ ).

Note also, that the Pigeonhole Principle follows easily from Generalized Pigeonhole principle, as for any  $n > m$  we have  $\frac{n}{m} > 1$ , hence there exists a container that contains more at least  $\lceil \frac{n}{m} \rceil \geq 2$  objects. We further illustrate the (Generalized) Pigeonhole Principle as follows:

**Example 2.4.1** *It is known that the number of people in Suzhou is more than 7000000, i.e.  $n \geq 7000001$ . It is also known that the number of hairs on each persons head is at most  $m = 500000$ . Hence, there exists a subset of at least*

$$\lceil \frac{n}{m} \rceil \geq \lceil \frac{7000001}{500000} \rceil = \lceil 14.000002 \rceil = 15$$

*people who have the same number of hair on their head.*

Here are couple of easy Pigeonhole Principle exercises to get acquainted with the basic idea.

**Exercise 2.4.2** *Show that among three people, there are two of the same sex.*

**Exercise 2.4.3** *Show that among thirteen people, there are two born in the same month.*

**Exercise 2.4.4** *In a group of 30 people, must 3 be born on the same month?*

**Exercise 2.4.5** *In a group of 30 people, must 4 be born on the same month?*

**Exercise 2.4.6** *How many people should be in the lecture theater to ensure that two people have the same birthday?*

**Exercise 2.4.7** *Prove that if  $qs+1$  pearls are put into  $s$  boxes, then there is a box which contains at least  $q+1$  pearls.*

**Exercise 2.4.8** *Consider the fraction  $5/20483$ . What is the maximal length of the period when written as the decimal periodic expansion?*

**Exercise 2.4.9** Is the number  $0.101001000100001000001\dots$  (where each string of 0's is one longer than the previous one) rational or irrational?

**Exercise 2.4.10** If we have 12 distinct 2-digit numbers, we can always find two of them whose difference is a two digit number of the form  $n = \overline{AA}$  (a two-digit number with both digits the same).

**Exercise 2.4.11** Given a set of 52 integers, show that there exists two whose sum or difference is divisible by 100.

**Exercise 2.4.12** Show that if 101 integers are chosen from 1 to 200 inclusive, there must be two with the property that one is divisible by the other.

We will now proceed to couple of deeper applications revealing the strength and beauty of the principle. Consider the following sequence of length 10:

$$3, 5, 8, 10, 6, 1, 9, 2, 4, 7.$$

Let us look for increasing subsequence of maximal length. It is not hard to see that  $(3, 5, 8, 10)$ ,  $(3, 5, 8, 9)$ ,  $(3, 5, 6, 7)$ ,  $(3, 5, 6, 9)$  and  $(1, 2, 4, 7)$  are all the maximal increasing sequences with length 4. Next, let's look for the decreasing subsequences of maximal length. Here, the best we can do is achieving length 3 with subsequences  $(8, 6, 1)$ ,  $(8, 6, 2)$ ,  $(8, 6, 4)$ ,  $(10, 6, 1)$ ,  $(10, 6, 2)$ ,  $(10, 6, 4)$ ,  $(10, 9, 2)$ ,  $(10, 9, 4)$ ,  $(10, 9, 7)$ . Is it possible to find an arrangement of integers from 1 to 10 that simultaneously avoids both an increasing subsequence of length 4 and a decreasing subsequence of length 4? The following theorem shows that this is not possible.

**Theorem 2.4.13** Let  $p, q$  be integers. Every sequence  $x_1, x_2, \dots, x_{pq+1}$  must contain either increasing subsequence of length  $p + 1$  or decreasing subsequence of length  $q + 1$ .

**Proof.** Suppose, that  $x_1, x_2, \dots, x_{pq+1}$  is a sequence. If this sequence contains an increasing sequence of length  $p + 1$ , then we are done. Hence, we can assume from now onwards that the longest increasing sequence is of size at most  $p$ . For each integer  $i$  with  $1 \leq i \leq pq + 1$ , denote  $a_i$  to be the length of the longest increasing subsequence with the first term  $x_i$ . Clearly,  $1 \leq a_i \leq p$  for all  $i$ . But now, we have  $pq + 1$  terms  $a_1, a_2, \dots, a_{pq+1}$ , each of them taking one of the  $p$  values from the set  $\{1, 2, \dots, p\}$ . Hence, by pigeonhole principle, one of these values must be taken  $q + 1$  times!

Hence we have a subsequence  $i_1 < i_2 < \dots < i_{q+1}$  such that  $a_{i_1} = a_{i_2} = \dots = a_{i_{q+1}}$ . By definition of  $a_i$  it is not hard to see that  $(x_{i_1}, x_{i_2}, \dots, x_{i_{q+1}})$  is a decreasing subsequence of length  $q + 1$  and hence we are done. ■

**Exercise 2.4.14** Suppose we have a sequence containing of 26 numbers. Prove that it contains either increasing subsequence of length 6 or a decreasing sequence of length 6.

The theorem above is an example of applications of the Pigeonhole Principle to a beautiful area of combinatorics called Ramsey Theory. Ramsey Theory, is a branch of mathematics that comes with it's own philosophy saying that a total disorder is impossible, and every disorder must contain some order. Here, we saw our first example of such a statement. In our example, we showed that however disordered sequence we have, if it is long enough, it contains a long ordered subsequence (monotonically increasing, or monotonically decreasing). Ramsey Theory



is all about finding patterns in chaotic mathematical structures. Examples of results include finding arithmetic progressions in various subsets of natural numbers, or certain types of nice subgraphs within big chaotic graphs. The latter, i.e. finding well-structured graphs using Ramsey-theoretical tools is one of the main research interests of your teacher.

We finish this section with one more exercise, which asks to find a simple order in a reasonably disordered system.

**Exercise 2.4.15** *A plane is coloured blue and red in any way. Prove that there exists a rectangle (all angles 90 degree) with all four vertices of the same colour.*

## 2.4.2 Principle of Invariance

Our last but not least strategy of combinatorial problem solving is the search for invariants, and it is called the Principle of Invariance. The word “invariant” means “not changing”. The principle is applicable to algorithms, games, transformations: some tasks that are performed repeatedly. What stays the same? What remains invariant? The principle of invariance is a heuristic principle, which will become clearer when working through examples, but the basic heuristic approach can be stated as follows.

### Principle of Invariance

If there is a repetition, look for what does not change!

The recognition of invariants is an important problem solving skill, and it can lead to proofs or disproofs of many mathematical and algorithmic problems. We start with the couple of problems.

**Example 2.4.16** *Suppose the numbers  $1, 2, \dots, 10$  are written on a white-board. Alice takes any two numbers  $a, b$  written on the board and replaces them by their sum. For example, if Alice initially selects numbers 7, and 8 she replaces them by  $7 + 8 = 15$ , then after the first step the numbers  $1, 2, 3, 3, 4, 5, 6, 9, 10, 15$  are written on the board. Suppose Alice continues the process until only one number is left on the board. What are the possible values of this number?*

**Proof.** In this problem, the invariant is the sum of all the numbers on the board, let's say  $S$ . If Alice erases the numbers  $a$  and  $b$  and writes  $a + b$  instead, then the new sum is  $S - a - b + (a + b) = S$ , thus indeed  $S$  is an invariant. Since the initial sum was  $1 + 2 + \dots + 10 = 55$ , the final number must be equal to 55. ■

Usually, a process can end in more than one possible outcome, in which case the invariants can tell us more about how the possible outcomes look like, or which outcomes are impossible.

**Example 2.4.17** *Suppose the numbers  $1, 2, \dots, 10$  are written on a white-board. Alice takes any two number  $a, b$  written on the board and replaces them by their non-negative difference  $|a - b|$ . For example, if Alice initially selects numbers 5 and 8, she replaces them by  $|8 - 5| = 3$ , then after the first step the numbers  $1, 2, 3, 3, 4, 6, 7, 9, 10$  are written on the board. Suppose Alice continues the process until only one number is left on the board. Prove that this number is odd.*

**Proof.** If Alice chooses the numbers  $a$  and  $b$  with  $a \geq b$ , then the sum  $S$  changes to  $S - a - b + (a - b) = S - 2b$ . Therefore the sum always changes by an even number, which means that the

parity of  $S$  is invariant. Initial parity is odd, hence throughout the process we must have  $S = 1 \pmod{2}$ . Hence the final number must be odd. ■

A very important application of invariants is in proving that certain states can never be reached by the process.

**Example 2.4.18** *A circle is divided into 6 sectors. Then numbers 1, 0, 1, 0, 0, 0 are written in the sectors (let's say clockwise). You may choose any two neighbouring numbers and increase them by 1. Is it possible to equalize all numbers by a sequence of such steps?*

**Proof.** Suppose  $a_1, a_2, \dots, a_6$  are the six numbers written in the sectors. Then  $I = a_1 - a_2 + a_3 - a_4 + a_5 - a_6$  is an invariant. Initially,  $I = 2$ , thus the goal  $I = 0$  cannot be reached. ■

Because the problem solving can only be learned by solving problems, here we provide a number of exercises for you to try.

**Exercise 2.4.19** *What happens in the previous example if a circle is divided into 5 sectors (instead of 6), with numbers 1, 0, 1, 0, 0 written. Does the same conclusion hold?*

**Exercise 2.4.20** *Suppose positive integers  $1, 2, \dots, 4n - 1$  are written on the board. In one move you may replace any two integers by their difference. Prove that an even integer will be left after  $4n - 2$  steps.*

**Exercise 2.4.21** *Each of the numbers  $a_1, a_2, \dots, a_n$  is 1 or  $-1$  and we have*

$$S = a_1 a_2 a_3 a_4 + a_2 a_3 a_4 a_5 + \dots + a_n a_1 a_2 a_3 = 0$$

*Prove that  $n$  is divisible by 4.*

**Exercise 2.4.22** *Given a natural number, the following operations can be performed:*

- 1) add 6;
- 2) divide by 2, if the number is even;
- 3) change the order of the digits of the number (as long as 0 does not appear as the first digit).

*Is it possible, that after a certain number of operations, starting with the number 21, one obtains the number 2023?*

**Exercise 2.4.23** *Around a circle, 5 ones and 4 zeros are arranged in any order. Then, between any two equal digits, you write 0 and between any two different digits 1. Finally, the original digits are wiped out, leaving only the new 9 digits around the circle. If this process is repeated indefinitely, prove that you can never get 9 zeros.*

Here we give two more involved examples of invariance principle.

**Example 2.4.24** *Can  $10 \times 10$  board be covered by  $1 \times 4$  tiles (without overlapping, of course)?*

**Proof.** Let us label the cells of the board from  $(i, j)$  with  $0 \leq i, j \leq 9$  according to their position with respect to  $x - y$  axis of the plane and let us colour the cells

- red if  $i + j = 0 \pmod 4$ ,
- blue if  $i + j = 1 \pmod 4$ ,
- green if  $i + j = 2 \pmod 4$  and
- yellow if  $i + j = 3 \pmod 4$ .

This is the colouring such that every  $1 \times 4$  tile covers one cell of each colour. However, there are  $1 + 5 + 9 + 7 + 3 = 25$ ,  $2 + 6 + 10 + 6 + 2 = 26$ ,  $3 + 7 + 9 + 5 + 1 = 25$  and  $4 + 8 + 8 + 4 = 24$  cells coloured red, blue, green and yellow respectively. In particular, since there are only 24 yellow cells, and each  $1 \times 4$  tile has to contain one yellow cell, we can have at most 24 tiles, hence we cannot cover the whole board. ■

**Example 2.4.25** Consider all lattice squares  $(x, y)$  with  $x, y$  nonnegative integers. Assign to each its lower left corner as a label. We shade squares  $(0, 0), (1, 0), (0, 1), (2, 0), (1, 1), (0, 2)$  and place a chip on each of the six squares. Consider the following operation:

- If  $(x, y)$  is occupied, but  $(x + 1, y)$  and  $(x, y + 1)$  are free, you may remove the chip from  $(x, y)$  and replace it by two chips, one of each of  $(x + 1, y)$  and  $(x, y + 1)$ .

The goal is to remove all the chips from the shaded squares. Is this possible?

**Proof.** No. To each square labelled  $(x, y)$  assign the weight  $2^{-(x+y)}$ . The total weight of all squares on the board, can be calculated to be equal to 4. The seven shaded squares have total weight of  $2\frac{3}{4}$  while the unshaded squares have total weight of  $4 - 2\frac{3}{4} = 1\frac{1}{4}$ . Note that the sum of the weights of the squares on which the chips is placed initially is  $2\frac{3}{4}$  and it remains invariant during the procedure. Since the unshaded squares have total weight less than  $2\frac{3}{4}$  it is impossible to remove all the chips from the shaded squares. ■

**Example 2.4.26** The 15 puzzle (also called Game of Fifteen, Mystic Square, Gem Puzzle, and many others) is a sliding puzzle having fifteen square tiles numbered 1-15 in a frame that is 4 tiles high and 4 tiles wide, leaving one unoccupied position. Tiles in the same row and column of the open position can be moved by sliding them horizontally or vertically, respectively. The goal of the puzzle is to place the tiles in numerical order. The famous challenge, posed by 19th century puzzle author and chess player Sam Lloyd was to find a sequence of moves that could solve the puzzle when the initial position had numbers 15 and 14 reversed. Such sequence turned out not to exist. Can you prove why?



Figure 2.3: LLOYD's 15-14 puzzle

We finish this section by highlighting the importance of the principle of invariance for algorithms and computer science. One such application relies on a slightly more general notion called 'monovariance' which is used to ensure the algorithmic procedure terminates (does not run for ever). It also gives ideas how to solve a question by providing a procedure (algorithm) which starts from arbitrary position and at each step makes a non-trivial step getting closer to the required solution. In all these cases monovariants play an important role providing a bound on the number of steps required to finish the procedure.

**Definition 2.4.27 (Monovariant)** *A monovariant is a quantity that only changes in one direction. When there is a repetition, it either always increases or always decreases.*

**Example 2.4.28** *In the Parliament of Siskinia, each member has at most three enemies. Prove that the house can be separated into two houses, so that each member has at most one enemy in his own house.*

**Proof.** Initially, we split the members into two houses arbitrarily. Let  $S$  be the total sum of all the enemies each member has in his own house. Now suppose  $A$  has at least two enemies in his house. Then  $A$  has at most one enemy in the other house. If  $A$  switches houses, the number  $S$  will decrease. As  $S$  is an integer that cannot be negative,  $S$  can be decreased only finitely many times, and hence the procedure must terminate. ■

The proof above is one of the first examples in this course on algorithmic approach to problem solving. Clearly, having a piece of paper or a computer, one can now follow the procedure and partition the members into two houses where each has at most one enemy. And it is not clear at all if one could have done this question without the algorithmic approach (can you?). This shows the usefulness and potential irreplaceability of algorithmic approach for certain types of problems. We will see more of this when we cover graphs. Below we give two more exercises, one easy, and one harder that shows how invariants/monovariants can be used to prove that the given procedures terminate.

**Exercise 2.4.29 (Termination of the Euclidean Algorithm)**

*Given two positive integers  $a > b$ , one can find their greatest common divisor  $GCD(a, b)$  by using the Euclidean Algorithm as follows.*

- Divide  $a$  by  $b$  to get a quotient  $q_1$  and a remainder  $r_1$ , i.e.

$$a = q_1b + r_1 \text{ with } 0 \leq r_1 < b.$$

- If  $r_1 = 0$ , then  $GCD(a, b) = b$  and stop the procedure. Otherwise, divide  $b$  by  $r_1$  to get another quotient  $q_2$  with remainder  $r_2$ , i.e.

$$b = q_2r_1 + r_2 \text{ with } 0 \leq r_2 < r_1.$$

- If  $r_2 = 0$ , then  $GCD(a, b) = r_1$  and stop the procedure. Otherwise, divide  $r_1$  by  $r_2$  to get a new quotient  $q_3$  and remainder  $r_3$  and continue.

- More formally, for  $n \geq 2$ , if  $r_n = 0$ :

– Then  $GCD(a, b) = r_{n-1}$  and stop the procedure.

- Otherwise, divide  $r_n$  by  $r_{n-1}$  to get a new quotient  $q_{n+1}$  and a remainder  $q_{n+1}$ , i.e.

$$r_n = r_{n-1}q_{n+1} + r_{n+1} \text{ with } 0 \leq r_{n+1} < r_n.$$

Continue with  $n$  replaced by  $n + 1$ .

Prove that this algorithm terminates.

**Exercise 2.4.30** Consider a sequence of  $n$  integers. Given two **consecutive** decreasing terms in the sequence  $b$  and  $c$ , with  $b > c$ , one can replace this pair  $(b, c)$  by either  $(c + 1, b)$  or  $(b - 1, b)$ . [For instance, if we have a sequence  $1, 8, 7, 3, 5, 2$ , then  $(7, 3)$  is a pair of consecutive and decreasing terms of the sequence. Hence we can replace  $(7, 3)$  by either  $(4, 7)$  or  $(6, 7)$ , obtaining either a sequence  $1, 8, 4, 7, 5, 2$  or a sequence  $1, 8, 6, 7, 5, 2$ , respectively. Note that  $(1, 8)$  is consecutive, but not decreasing so we couldn't have performed the operation for this pair.] Prove that no matter in which order the steps are carried out, the procedure must terminate.

**Remark 2.4.31** We remark that apart from ensuring the termination of algorithms, invariants also play an important role in ensuring the correctness of algorithms (that the answers given by the algorithms are indeed correct). These invariants are known as 'loop invariants' (occurring in the computer execution of 'while' or 'for' loops) and a more detailed discussion of them can be found in Susanna Epp's book 'Discrete Mathematics'.

## Chapter 3

# Recursion

One core example of a recursion is a *recurrence relation*, which is an equation according to which the  $n$ 'th term of a sequence of numbers is equal to some combination of the previous terms. A classical example, of a recurrence relation is the Fibonacci sequence, defined by

$$F(n) = F(n-1) + F(n-2)$$

for all  $n > 1$  with the base case  $F(0) = 0$ ,  $F(1) = 1$ .

The recurrence relations, such as Fibonacci numbers, have a number of applications to the fields such as biology and economics, where the key variables (population growth, interest rate, GDP, etc.) depend on and can be modelled in terms of the past values of the variables. The recurrence relations are also of fundamental importance in analysis of algorithms in computer science. If an algorithm is designed so that it will break a problem into smaller subproblems, its running time is described by a recurrence relation. It is also an important part of enumerative combinatorics, where the number of combinatorial objects (for instance the numbers of various partitions of a set) often satisfy certain recurrence relations.

Apart from recurrence relations, a recursion has a broader meaning, when a more general object (not necessarily a number!) is defined by smaller similar objects. For example, a geometrical shape such as Sierpinski triangle pictured above can be defined as a union of three smaller Sierpinski triangles, which in turn is a union of 9 smaller Sierpinski triangles and so on... A much more important example of recursion is that of an algorithmic problem, which can be split into several subproblems, such that knowing the solutions of the subproblems, we could obtain the solution to the problem. The subproblems then can be split into further subsubproblems and so on, until we reach a base case which can be solved trivially. We will explore this broader algorithmic notion of recursion, by covering a couple of recursive algorithms. The broader notion of recursion is also illustrated in the remark below.

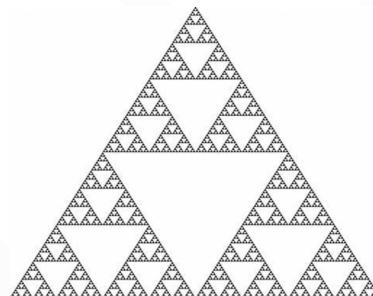


Figure 3.1: Sierpinski triangle

**Remark 3.0.1** *To understand what is a recursive joke, look at the Remark 3.1.7.*

### 3.1 Recurrence relations and generating functions

In this section we will be solving recurrence relations.

### 3.1.1 The auxiliary equation method

In this section we will concentrate on so-called homogeneous second order linear recurrence relations with constant coefficients of the form

$$a_n = Aa_{n-1} + Ba_{n-2},$$

with two given constants  $A$  and  $B$  such that  $B \neq 0$ , and also where  $a_0$  and  $a_1$  are given.

It turns out that geometric series, or indeed a combination of geometric series, are usually the solutions of such recurrences. Indeed, simply trying  $a_n = r^n$  for some  $r \neq 0$  gives us  $r^n = Ar^{n-1} + Br^{n-2}$ , i.e.  $r^2 = Ar + B$ . Thus,  $a_n = r^n$  is a solution of the recurrence relation precisely when  $r$  is the root of the so-called **auxiliary equation**:

$$x^2 = Ax + B.$$

The solution to the auxiliary equation then follows easily.

**Theorem 3.1.1** *Let  $\alpha$  and  $\beta$  be the roots of the auxiliary equation given above. Then*

(i) *if  $\alpha \neq \beta$ , there are constants  $C_1, C_2$  such that for all  $n \geq 0$ , we have*

$$a_n = C_1\alpha^n + C_2\beta^n;$$

(ii) *if  $\alpha = \beta$ , there are constants  $C_1, C_2$  such that for all  $n \geq 0$ , we have*

$$a_n = (C_1 + nC_2)\alpha^n.$$

**Proof.** (i) Since  $\alpha^n$  and  $\beta^n$  are solutions of the recurrence relation, any linear combination  $C_1\alpha^n + C_2\beta^n$  would also satisfy the recurrence (check this!). We only need to choose suitable constants  $C_1$  and  $C_2$  to satisfy initial conditions:

$$\begin{aligned} a_0 &= C_1 + C_2 \\ a_1 &= C_1\alpha + C_2\beta \end{aligned}$$

A unique solution for  $C_1$  and  $C_2$  can always be found, since the matrix  $\begin{pmatrix} 1 & 1 \\ \alpha & \beta \end{pmatrix}$  is invertible.

(ii) In general, if  $\alpha^n$  is a solution of a recurrence relation, then  $n\alpha^n$  is not necessarily a solution. However, when  $\alpha$  is a repeated root, from the auxiliary equation we have that

$$x^2 - Ax - B = (x - \alpha)^2$$

and so  $A = 2\alpha$  and  $B = -\alpha^2$ . Hence, in this case we can verify that  $a'_n = n\alpha^n$  does satisfy the recurrence:

$$Aa'_{n-1} + Ba'_{n-2} = A(n-1)\alpha^{n-1} + B(n-2)\alpha^{n-2} = 2(n-1)\alpha^n - (n-2)\alpha^n = n\alpha^n = a'_n.$$

Hence, once again since  $\alpha^n$  and  $n\alpha^n$  satisfy the recurrence, so does any linear combination  $a_n = (C_1 + nC_2)\alpha^n$ . We just need to choose suitable constants to satisfy initial conditions:

$$\begin{aligned} a_0 &= C_1 \\ a_1 &= (C_1 + C_2)\alpha \end{aligned}$$

which clearly can always be done. This finishes the proof. ■

**Exercise 3.1.2** Recall that the Fibonacci sequence is defined by  $F_0 = 0$ ,  $F_1 = 1$  and

$$F_n = F_{n-1} + F_{n-2}.$$

for  $n \geq 2$ . Show that

$$F_n = \frac{1}{\sqrt{5}} \left( \frac{1 + \sqrt{5}}{2} \right)^n - \frac{1}{\sqrt{5}} \left( \frac{1 - \sqrt{5}}{2} \right)^n.$$

Deduce that  $\frac{F_{n+1}}{F_n} \rightarrow \frac{1+\sqrt{5}}{2}$ , the golden ratio.

**Exercise 3.1.3** Solve the recurrence relation

$$a_n = 4a_{n-1} - 4a_{n-2}$$

for  $n \geq 3$ , with initial conditions  $a_1 = 1$ ,  $a_2 = 3$ .

The auxiliary equation method extends to higher order recurrences in the obvious way.

**Exercise 3.1.4** Solve the third order recurrence relation

$$a_n = 6a_{n-1} - 11a_{n-2} + 6a_{n-3}.$$

for  $n \geq 4$ , with initial conditions  $a_1 = 3$ ,  $a_2 = 6$ ,  $a_3 = 14$ .

Finally, let us generalize a bit further and briefly consider **non-homogeneous** case of linear recurrences, e.g. recurrences of the form

$$a_n = Aa_{n-1} + Ba_{n-2} + t_n$$

where  $t_n$  is some function of  $n$ . The solution of this can be obtained by the following procedure:

- (i) solving the homogeneous case (i.e. the recurrence obtained by replacing  $t_n$  with 0),
- (ii) adding **any particular solution** of the non-homogeneous case.

**Example 3.1.5** Solve the first order non-homogeneous recurrence relation

$$a_n = -a_{n-1} + 3 \cdot 2^{n-1}$$

for  $n \geq 1$ , with initial condition  $a_0 = 3$ .

**Proof.** The homogeneous equation  $a_n = -a_{n-1}$  has auxiliary equation  $x = -1$ , and so the solution to the homogeneous recurrence is  $a_n^H = C(-1)^n$  for any constant  $C$ . Since  $t_n = 3 \cdot 2^{n-1}$ , it is sensible to try a particular solution of the form  $D2^n$  for some constant  $D$ . Solving

$$D2^n = -D2^{n-1} + 3 \cdot 2^{n-2},$$

we obtain  $D = 1$ , and so we found a particular solution  $a_n^P = 2^n$ . Hence, the solution that satisfies the non-homogeneous recurrence relation (but not necessarily initial conditions) is

$$a_n = a_n^H + a_n^P = C(-1)^n + 2^n.$$

The initial condition now fix  $C = 2$ , and so  $a_n = 2(-1)^n + 2^n$ . ■

**Exercise 3.1.6** Solve the second order non-homogeneous recurrence relation

$$a_n = 4a_{n-1} - 3a_{n-2} + 2^n$$

for  $n \geq 3$  with initial conditions  $a_1 = 1$ ,  $a_2 = 11$ .

**Remark 3.1.7** To understand what is a recursive joke, look at the Remark 3.0.1.



### 3.1.2 Generating functions

The generating function of a sequence  $a_0, a_1, a_2, \dots$  is defined to be

$$f(x) = \sum_{i=0}^{\infty} a_i x^i.$$

For example, the generating function of the Fibonacci sequence is

$$x + x^2 + 2x^3 + 3x^4 + 5x^5 + \dots$$

Sometimes, given a recurrence relation, it is possible to find the generating function of the sequence and then find the coefficients  $a_n$  by reading off the coefficient of  $x^n$ . We will illustrate this method by re-doing the last example from the previous section.

**Example 3.1.8** *By finding the generating function, solve*

$$a_n = -a_{n-1} + 3 \cdot 2^{n-1}$$

for  $n \geq 1$ , with initial condition  $a_0 = 3$ .

**Proof.** By definition of generating function we have  $f(x) = \sum_{n=0}^{\infty} a_n x^n$ . Using the  $a_0 = 3$  and the recurrence relation for all entries  $a_n$  with  $n \geq 1$ , we obtain

$$\begin{aligned} f(x) &= 3 + \sum_{n=1}^{\infty} a_n x^n \\ &= 3 + \sum_{n=1}^{\infty} (-a_{n-1} + 3 \cdot 2^{n-1}) x^n \\ &= 3 - x \sum_{n=1}^{\infty} a_{n-1} x^{n-1} + 3x \sum_{n=1}^{\infty} 2^{n-1} x^{n-1} \\ &= 3 - x f(x) + 3x \frac{1}{1-2x} \\ &= \frac{3-3x}{1-2x} - x f(x) \end{aligned}$$

Hence, we have found that our generating function is

$$f(x) = \frac{3-3x}{(1-2x)(1+x)}.$$

Since we obtained a rational function, we would like to use the method of partial fractions to find the constants  $A$  and  $B$  such that

$$f(x) = \frac{A}{1-2x} + \frac{B}{1+x}.$$

It's not hard to see that  $A = 1$  and  $B = 2$  satisfies the relation, and hence

$$\begin{aligned} f(x) &= \frac{1}{1-2x} + \frac{2}{1+x} \\ &= (1 + 2x + 2^2 x^2 + 2^3 x^3 + \dots) + 2(1 - x + x^2 - x^3 + \dots) \end{aligned}$$

It is clear that the coefficient in front of  $x^n$  is  $2^n + 2(-1)^n$  which is the same as we obtained in the previous section using the auxiliary equation method. ■

Let us do another example

**Example 3.1.9** *By finding the generating function, solve*

$$a_n = 4a_{n-1} - 4a_{n-2}$$

for  $n \geq 3$ , with initial conditions  $a_1 = 1, a_2 = 3$ .

**Proof.** Since the sequence starts from the entry  $a_1$ , our generating function will have no constant term and will be defined as  $f(x) = \sum_{n=1}^{\infty} a_n x^n$ . Once, again, using  $a_1 = 1, a_2 = 3$  and the recurrence relation, we can rewrite our equation as:

$$\begin{aligned} f(x) &= x + 3x^2 + \sum_{n=3}^{\infty} a_n x^n \\ &= x + 3x^2 + \sum_{n=3}^{\infty} (4a_{n-1} - 4a_{n-2}) x^n \\ &= x + 3x^2 + 4x \sum_{n=3}^{\infty} a_{n-1} x^{n-1} - 4x^2 \sum_{n=3}^{\infty} a_{n-2} x^{n-2} \\ &= x + 3x^2 + 4x(f(x) - x) - 4x^2 f(x) \\ &= x - x^2 + (4x - 4x^2)f(x) \end{aligned}$$

Hence, once again, we obtain  $f(x)$  is a rational function

$$f(x) = \frac{x - x^2}{(1 - 2x)^2} = -\frac{1}{4} + \frac{1}{4(1 - 2x)^2}$$

Note that

$$\left(\frac{1}{1-x}\right)^2 = 1 + 2x + 3x^2 + 4x^3 + \dots$$

[This can be shown by differentiating termwise  $\frac{1}{1-x} = 1 + x + x^2 + x^3 + \dots$ , or by termwise multiplying out the brackets in  $(\frac{1}{1-x})^2 = (1 + x + x^2 + \dots)(1 + x + x^2 + \dots)$ .]

Hence, it follows that

$$\frac{1}{(1-2x)^2} = 1 + 2(2x) + 3(2x)^2 + 4(2x)^3 + \dots$$

Thus,

$$f(x) = -\frac{1}{4} + \frac{1}{4}(1 + 2 \cdot 2x + 3 \cdot 2^2 x^2 + 4 \cdot 2^3 x^3 + \dots).$$

We can now clearly see that the coefficient in front of  $x_n$  is  $a_n = (n+1)2^{n-2}$ . ■

**Remark 3.1.10** *Once we have found the power series in the argument above, we can show that the series converges uniformly for any  $|x| < \frac{1}{2}$ , and so the manipulations above are justified analytically. But in fact there is a theory of formal power series, according to which it is legitimate to do such manipulations without any regard to questions of convergence. This is important in cases where the series don't converge or it is not possible to find enough information about the series to deduce the convergence. One can formally consider the power series as a sequence  $(a_0, a_1, a_2, \dots)$  and define addition and multiplication of the sequences accordingly. For instance, addition and multiplication can be defined as:*

$$(a_0, a_1, a_2, \dots) + (b_0, b_1, b_2, \dots) = (a_0 + b_0, a_1 + b_1, a_2 + b_2, \dots), \text{ and}$$

$$(a_0, a_1, a_2, \dots) \times (b_0, b_1, b_2, \dots) = (c_0, c_1, c_2, \dots),$$

respectively, where  $c_n = \sum_{i=0}^n a_i b_{n-i}$ . From multiplication rule, it would follow that

$$(1, c, c^2, c^3, \dots) \times (1, -c, 0, 0, \dots) = (1, 0, 0, \dots),$$

which would justify the formula we use for geometric series

$$1 + cx + c^2x^2 + c^3x^3 + \dots = \frac{1}{1 - cx},$$

and show that division operation can also be defined in this formal series setting. With a bit more work, standard tools such as differentiation can also be justified. You can read more about this in Peter Cameron's "Combinatorics: topics, techniques, algorithms" book.

### 3.1.3 Catalan, Bell and Stirling Numbers

In this section we will discuss several of the most important combinatorial numbers, which occur in various counting problems, often involving recursively defined objects.

We start with Catalan numbers named after the Belgian mathematician Eugène Charles Catalan (1814-1894). Sloane and Plouffe in their work on "Encyclopedia of integer sequences", remarks that Catalan numbers, which are 1, 2, 5, 14, 42, ..., is possibly the second most frequently occurring numbers in combinatorics, after binomial coefficients. Meanwhile Stanley's "Enumerative Combinatorics" book lists 66 different combinatorial interpretations of these numbers! While abundant in combinatorics and computer science, it also appears in other fields of study. Peter Cameron in "Combinatorics: topics, techniques and algorithms" remarks that two of his colleagues approached him with queries about Catalan numbers that came up in their research. One studied non-linear dynamics, the other Lie superalgebras.

Here is a typical application:

In how many ways can a sum of  $n$  terms be bracketed so that  
it can be calculated by adding two terms at a time?

For example, if  $n=4$ , there are five possibilities:

$$\begin{aligned}
&(((a + b) + c) + d), \\
&((a + (b + c)) + d), \\
&(a + ((b + c) + d)), \\
&(a + (b + (c + d))), \\
&((a + b) + (c + d)).
\end{aligned}$$

We have enclosed the entire expression in an extra pair of brackets so that each addition has its pair of brackets.

Let  $C_n$  be the number of ways of bracketing the sum of  $n$  terms. To obtain a recurrence relation for  $C_n$ , note that any bracketed expression has a form  $(E_1 + E_2)$ , where  $E_1$  and  $E_2$  are bracketed expressions with  $i$  and  $n - i$  terms, for some  $1 \leq i \leq n - 1$ . There are  $C_i$  choices for bracketing  $E_1$  and  $C_{n-i}$  for bracketing  $E_2$ . Hence, applying product and sum rules, we obtain the recurrence relation for Catalan numbers

$$C_n = \sum_{i=1}^{n-1} C_i C_{n-i},$$

for  $n > 1$  and  $C_1 = 1$ .

Since this recurrence relation is non-linear, we will use generating function to determine the Catalan numbers. Let  $f(x) = \sum_{n=1}^{\infty} C_n x^n$ , and we proceed as before, obtaining:

$$f(x) = x + \sum_{n=2}^{\infty} \sum_{i=1}^{n-1} C_i C_{n-i} x^n = x + f^2(x)$$

Hence,

$$f(x) = \frac{1}{2} \left( 1 \pm (1 - 4x)^{\frac{1}{2}} \right).$$

Since, by definition of  $f$  it follows that  $f(0) = 0$ , we have to choose the minus sign. The other coefficients of the power series can be obtained by differentiation:

$$\begin{aligned}
C_n &= \frac{f^{(n)}(0)}{n!} \\
&= -\frac{1}{2} \cdot \frac{1}{2} \cdot \frac{-1}{2} \cdot \frac{-3}{2} \cdot \dots \cdot \frac{-(2n-3)}{2} (-4)^n / n! \\
&= \frac{1 \cdot 3 \cdot 5 \cdot \dots \cdot (2n-3) \cdot 2^{n-1}}{n!} \\
&= \frac{(2n-2)!}{n!(n-1)!}
\end{aligned}$$

Thus, we deduce the following theorem:

**Theorem 3.1.11 (Catalan numbers)** *For any  $n \in \mathbb{N}$  we have*

$$C_n = \frac{1}{n} \binom{2n-2}{n-1}.$$

$n$	1	2	3	4	5	6	7	8	9	10
$C_n$	1	1	2	5	14	42	132	429	1430	4862

We note that some authors define the Catalan numbers from as  $C_n = \frac{1}{n+1} \binom{2n}{n}$  for  $n \geq 0$ . This gives the same sequence, but the enumeration starts from 0, rather than from 1.

**Exercise 3.1.12** A rooted tree is a graph theoretical tree with a distinguished vertex called the root. The vertices in a rooted tree form a hierarchy, with the root at the highest level, and the level of every other vertex determined by its distance from the root. Some familiar terms are often used to describe relationships between vertices in a rooted tree: If  $v$  and  $w$  are adjacent vertices and  $v$  lies closer to the root than  $w$ , then  $v$  is the parent of  $w$ , and  $w$  is a child of  $v$ . Likewise, one may define siblings, grandparents, cousins, and other family relationships in a rooted tree.

We say that a rooted tree is strictly binary if every parent vertex has exactly two children. How many strictly binary trees are there with  $k$  parent vertices? Do not take symmetry into account: if two trees are mirror images of one another, count both configurations. Figure 3.2 shows that there are five trees with three parent vertices.

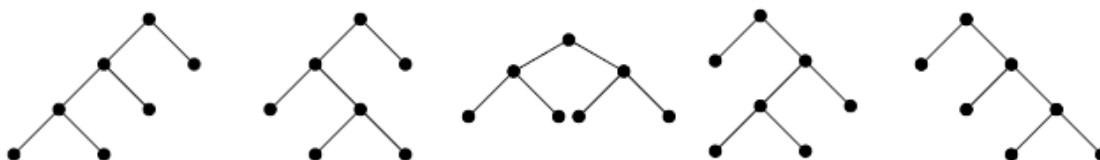


Figure 3.2: Strictly binary trees with three parent vertices

**Exercise 3.1.13** Consider two questions:

- How many solutions are there for

$$a_1 + a_2 + \dots + a_{2n} = 0, \text{ with } a_i \in \{-1, 1\} \text{ such that}$$

$$a_1 + a_2 + \dots + a_j \geq 0 \text{ for all } 1 \leq j \leq 2n.$$

- A mountain ridgeline of size  $2n$  is a connected line consisting of  $n$  ascending steps  $/$  and  $n$  descending  $\backslash$ , which starts at the horizon and can never dip below the horizon (see the diagram below). How many mountain ridgelines of size  $2n$  are there?



A. Can you spot a relationship between the two questions?

B. Define a mountain of size  $2m$  to be a mountain ridgeline of size  $2m$ , which touches the horizon exactly twice: at the start and at the end, and it is strictly above the horizon for all the other points of the mountain ridgeline. For example, the diagram above consists of 5 mountains, with sizes 10, 2, 6, 2, 2, respectively, which together add up to mountain ridgeline of size 22. Denote the  $R_m$  to be the number of mountain ridgelines of size  $2m$  and let  $M_m$  be the number of mountains of size  $2m$ . Find the relationship between the numebers  $R_m$  and  $M_m$ .

C. Use the previous part, to solve the two questions.

We end the section with couple other frequently occuring combinatorial numbers.

### Definition 3.1.14 (Stirling and Bell numbers)

- Stirling cycle number  $s(n, k) = \begin{bmatrix} n \\ k \end{bmatrix}$ , also called Stirling number of the first kind, is the number of permutations of an  $n$ -element set with exactly  $k$  cycles.
- Stirling partition number  $S(n, k) = \begin{Bmatrix} n \\ k \end{Bmatrix}$ , also called Stirling number of the second kind, is the number of partitions of  $n$  objects into exactly  $k$  classes.
- Bell number  $B_n$  is the number of partitions of an  $n$ -element set.

**Example 3.1.15** How many ways are there to split 5 students into two groups, three groups, any number of groups?

**Proof.** The short answer is  $\begin{Bmatrix} 5 \\ 2 \end{Bmatrix}$ ,  $\begin{Bmatrix} 5 \\ 3 \end{Bmatrix}$  and  $B_5$ . Let us try to evaluate these numbers. For splitting fiive students into two groups, fix one student, say student A, and consider how many ways are there to add other students to a group with student A. It follows that:

$$\begin{Bmatrix} 5 \\ 2 \end{Bmatrix} = \binom{4}{0} + \binom{4}{1} + \binom{4}{2} + \binom{4}{3} = 15$$

For making three groups, we see that there are two possible sizes of the groups:  $1 + 1 + 3$  or  $1 + 2 + 2$ . Hence the number can be obtained using the multinomial coefficient:

$$\begin{Bmatrix} 5 \\ 3 \end{Bmatrix} = \frac{1}{2} \cdot \binom{5}{1, 1, 3} + \frac{1}{2} \cdot \binom{5}{1, 2, 2} = 10 + 15 = 25$$

By similar arguments, one can find that  $\begin{Bmatrix} 5 \\ 1 \end{Bmatrix} = 1$ ,  $\begin{Bmatrix} 5 \\ 4 \end{Bmatrix} = 10$ ,  $\begin{Bmatrix} 5 \\ 5 \end{Bmatrix} = 1$ , hence

$$B_5 = \sum_{k=1}^5 \begin{Bmatrix} 5 \\ k \end{Bmatrix} = 1 + 15 + 25 + 10 + 1 = 52.$$

■

While one may obtain a recurrence relation and a generating function for Bell numbers, no closed formula for Bell number is known. We list the first ten values of the Bell numbers:

$n$	1	2	3	4	5	6	7	8	9	10
$B_n$	1	2	5	15	52	203	877	4140	21147	115975

## 3.2 Recursive algorithms and complexity

“Languages come and go, but algorithms stand the test of time.”  
(Donald Knuth)

The concept of algorithms existed more than a thousand years ago, and the word itself is derived from the name of the 9th-century Persian mathematician Muhammad ibn Musa al-Khwarizmi. This word describes a sequence of elementary steps, which transforms the given input data into output results. The algorithm is correct, if for all the initial input data, it finishes the work and produces correct output results.

What is considered to be a good algorithm? The quality of the algorithm is usually described by its computational complexity - the amount of resources required to run it. The amount of resources required to run the algorithm generally varies with the input, the complexity is typically expressed as a function  $n \rightarrow f(n)$ , where  $n$  is the size of input, and  $f(n)$  is the worst-case complexity - the maximum of the amount of resources that are needed over all inputs of size  $n$ . Time complexity is generally expressed as the number of required elementary operations on an input of size  $n$ , where elementary operation are assumed to take a constant amount of time on a given computer and change only by a constant factor when run on a different computer. Space complexity is generally expressed as the amount of memory required by an algorithm on an input of size  $n$ .



Figure 3.3: Mohammad ibn Musa al-Khwarizmi

To classify the algorithms according to their run time or space complexity the computer scientists use big- $O$  notation. The letter  $O$  is used because the growth rate of the function is also referred as the order of the function. A description of a function in terms of big  $O$  notation usually only provides an upper bound on the growth rate of the function, and there are other symbols like  $o, \Omega, \omega$  and  $\Theta$  to describe other kinds of asymptotic growth rates (strict upper bound, lower bound, strict lower bound, exact bound, respectively).

We note that big- $O$  and little- $o$  notations are also used in other branches of mathematics, for example Analysis/Calculus when estimating the size of the remainder term in Taylor series expansion of a function. The only difference is that in Analysis/Calculus we more often take the limit as  $x \rightarrow c$  for some finite value  $c$  (i. e. we are interested in local behavior around a some point), while in computer science, one usually considers the limit as  $x \rightarrow \infty$ . The formal definition for big- $O$  notation is as follows.

**Definition 3.2.1 (Big- $O$ )** *Let  $f$  and  $g$  be two functions, defined on some unbounded set of positive real numbers, and such that  $g(x) > 0$  for all large enough values of  $x$ . One writes*

$$f(x) = O(g(x)) \text{ as } x \rightarrow \infty$$

*if there exists  $M > 0$  and a real number  $x_0$ , such that*

$$|f(x)| \leq Mg(x) \text{ for all } x \geq x_0.$$

**Example 3.2.2** *Consider the function  $f(n) = n^4 - 2n^3 + 5$ , then  $f(n) = O(n^4)$ .*

**Proof.** Observe that

$$|n^4 - 2n^3 + 5| \leq n^4 + |2n^3| + 5 \leq n^4 + 2n^4 + 5n^4 \leq 8n^4.$$

Hence by definition the result follows. ■

**Exercise 3.2.3** Let  $f = O(n^2)$  and  $g = O(n^3)$ . Find the big-O estimate of the functions  $f + g$ ,  $f \times g$  and  $f^2 + 5g + 7$ .

Let us also formally define the lower bound 'Big-Omega' and the precise bound 'Big-Theta'.

**Definition 3.2.4 (Big-Ω, Big-Θ)** Let  $f$  and  $g$  be two functions, defined on some unbounded set of positive real numbers, and such that  $g(x) > 0$  for all large enough values of  $x$ . One writes:

$$f(x) = \Omega(g(x)) \text{ as } x \rightarrow \infty$$

if there exists  $M > 0$  and a real number  $x_0$ , such that

$$f(x) \geq Mg(x) \text{ for all } x \geq x_0.$$

If  $f(x) = O(g(x))$  and  $f(x) = \Omega(g(x))$  then one writes  $f(x) = \Theta(g(x))$ , and may refer to this by saying that  $f$  is of order  $g$ .

**Exercise 3.2.5** Show that  $f(n) = n^4 - 2n^3 + 5 = \Theta(n^4)$ .

Consider the following problem.

Given a sequence  $(x_1, x_2, x_3, \dots, x_n)$ , find the largest strictly increasing subsequence.

Suppose, for example, we have a sequence,

$$(9, 5, 6, 2, 8, 4, 1, 9, 7, 5, 3, 6).$$

In this case the longest strictly increasing subsequence has length 4, and this can be achieved by subsequences  $(5, 6, 8, 9)$  and  $(2, 4, 5, 6)$ .

Algorithmically, we can approach this problem by considering all possible subsets of  $(x_1, x_2, \dots, x_n)$  and finding the one that gives us the largest strictly increasing sequence. However, this naive approach would be way too difficult for computer to handle even for a moderately long input sequences. Indeed, given a sequence of length  $n$ , we know that this sequence has  $2^n$  subsets. For  $n = 20$  the computer would need to go through  $2^{20} \approx 10^6$  subsets, which the computer would be able to handle, possibly in around a millisecond or so (depending on the speed of the computer). Yet, adding any other point in the sequence, i.e. going from 20-entry sequence to 21-entry sequence, would double the number of subsets to be worked through and as a result, no matter what computer is being used, the time would double at each addition of an extra entry. So a computer which solves the problem for  $n = 20$  in one millisecond, will spend 2 milliseconds for  $n = 21$ , 4 milliseconds for  $n = 22$ , etc., around a second for  $n = 30$ , and around  $4 \times 10^{13}$  years for  $n = 100$ . This number is so large, that it is likely to never be handled even with the most powerful computers on earth. Even if a million computers, or a computer that is a million times more powerful than a standard PC was employed, it would still take them more than a million years to finish the task for  $n = 100$ . For  $n = 200$  it would take a trillion times longer...



What we have seen above is so-called exponential complexity explosion, where the algorithm provided has time complexity  $\Theta(n2^n)$ . To justify complexity, note that we would need to check  $2^n$  subsets, and for each subset there would be at most  $n$  elementary operations performed checking whether the numbers in the subset form an increasing subsequence. Hence by the Product Rule at most  $n2^n$  elementary operations performed. This shows the upper bound  $O(n2^n)$ . On the other hand, at least half of the subsets have size not smaller than  $\frac{n}{2}$ , thus at least  $\frac{n}{2} \cdot \frac{2^n}{2} = \frac{1}{4}n2^n$  operations are performed, proving the lower bound  $\Omega(n2^n)$ . Hence the complexity is  $\Theta(n2^n)$  as claimed. Can we do better? Yes, actually we can find a polynomial time algorithm that solves this problem much faster, using recursion in a clever way.

The trick is to use inductive/recursive reasoning. Let's look at the last element  $a_n$ . The right question to ask is the following: what is length of the longest strictly increasing sequence that ends in  $a_n$ ? Such sequence is obtained by appending  $a_n$  to the longest strictly increasing sequence that ends with  $a_k$  for some  $a_k < a_n$ . Let us denote

$L(k)$  = 'The length of the longest strictly increasing sequence that ends in  $a_k$ '.

With this notation, we have:

$$L(n) = 1 + \max_{\substack{1 \leq k \leq n-1, \\ a_k < a_n}} L(k).$$

To reconstruct a sequence, let us always record the 'previous term' that follows before  $a_n$  in the longest strictly increasing sequence ending in  $a_n$ :

$$p(n) = r \text{ such that } 'a_r < a_n' \text{ and } 'L(r) = \max_{\substack{1 \leq k \leq n-1, \\ a_k < a_n}} L(k)'.$$

We illustrate this algorithm in the table below for our sequence (9, 5, 6, 2, 8, 4, 1, 9, 7, 5, 3, 6).

$k$	1	2	3	4	5	6	7	8	9	10	11	12
$a_k$	9	5	6	2	8	4	1	9	7	5	3	6
$L(k)$	1	1	2	1	3	2	1	4	3	3	2	4
$p(k)$	-	-	2	-	3	4	-	5	6	6	4	10

Note that we filled the third and fourth row of this table inductively left to right, starting with  $L(1) = 1$  and  $p(1) = -$  as the sequence is of length one - no previous member. Then filling in  $L(2) = 1$  since  $5 < 9$ , so the longest sequence ending in  $a_2 = 5$  is of length one again. Then, finally  $L(3) = 2$  since  $5 < 6$  and  $L(2) = 1$ , so  $L(3) = 1 + L(2) = 2$ , and we record that we achieve the longest sequence ending in  $a_3$  through the second entry:  $p(3) = 2$ . Having filled in the table, the third row suggests that there are two sequences of length 4, one ending in the 12th entry and the other in the 8th entry. We can recover the sequences, by looking at the values  $p(k)$ . For example, the 4 in the 12th row tell us to go to 10th row, which in turn, asks us to go to 6th row, and 6th row points us to the 4th, giving us an increasing sequence  $(a_4, a_6, a_{10}, a_{12}) = (2, 4, 5, 6)$ .

What is the complexity of the algorithm? Well, to calculate the value  $L(k), p(k)$  we have to find the maximum of the values  $L(1), L(2), \dots, L(k-1)$  and store the index  $r$  corresponding to the maximum (making sure that  $a_r < a_k$ ). This takes  $k-1$  elementary steps. Hence, in total, we will take  $1 + 2 + \dots + n-1 = \frac{n(n-1)}{2} = O(n^2)$  elementary steps. To recover and output the longest sequence, we might need to go through the table once more, so further  $O(n)$  steps,

which gives the total time complexity  $O(n^2)$ . Note that there is also a space complexity of  $O(n)$  as we use  $3n$  cells of computer memory in total throughout our calculation (to store values of  $a_1, \dots, a_n, L(1), \dots, L(n), p(1), \dots, p(n)$ ). In fact, it is not hard to see that the lower complexity bounds coincide with the upper and that we do indeed have time and space complexities of  $\Theta(n^2)$  and  $\Theta(n)$ , respectively.

Note that with this new updated algorithm of complexity  $\Theta(n^2)$ , it would take less than a millisecond to find the longest subsequence in the sequence of length  $n = 100$  and  $n = 1000$  - something that would have taken centuries with our previous algorithm. In fact, it would take around 17 minutes to handle a sequence of length  $n = 1000000$  on average computer which seems reasonable given the size of the problem. Overall, the polynomial time is regarded as a good time complexity for practical applications.

We end up this section with the following exercise:

**Exercise 3.2.6** *Given a sequence of  $(x_1, x_2, \dots, x_n)$  let us say that  $x_k$  is the peak, if*

- *either  $x_k \geq x_{k-1}$  and  $x_k \geq x_{k+1}$  for  $k \in \{2, 3, \dots, n-1\}$ ,*
- *or  $k = 1$  and  $x_1 \geq x_2$ ,*
- *or  $k = n$  and  $x_n \geq x_{n-1}$ .*

*Intuitively,  $x_k$  is a peak if it is larger or equal to any of its neighbouring (one or two) entries.*

- (a) Show that any sequence has a peak.*
- (b) Suggest a linear time  $\Theta(n)$  algorithm for finding a peak.*
- (c) Is there an algorithm faster than linear time to find a peak?*

**Remark 3.2.7** *A common algorithm design tactic is to divide a problem into sub-problems of the same type as the original, solve those sub-problems and combine the results. This is often referred to as the divide-and-conquer method (suggested approach for part (c) of the exercise above). When combined with a lookup table (like the one we built when looking for longest subsequence) that stores the results of previously solved subproblems (to avoid solving them repeatedly and incurring extra computation time), it can be referred to as dynamical programming.*

## Chapter 4

# Basic concepts in graph theory

“The study of Euler’s works will remain the best school for different fields of mathematics, and nothing else can replace it”  
(Carl Friedrich Gauss)

The city of Königsberg (“King’s Mountain”) grew up around the fortress built in 1255 in Prussia, which was then a part of Germany. The city was laid out across a fork in the river Pregel, with seven bridges connecting the different parts of the city. People who lived in the city often wondered idly over coffee whether it would be possible to make a journey through the city, crossing all of the bridges, but without crossing any of the bridges twice. People struggled to find a solution to this problem, and it took a Swiss mathematician Leonhard Euler, who lived in nearby St Petersburg - to find the answer. In 1736 he wrote an article about it. His work on the “Königsberg Bridge problem” is considered by many to be the beginning of the field of graph theory.

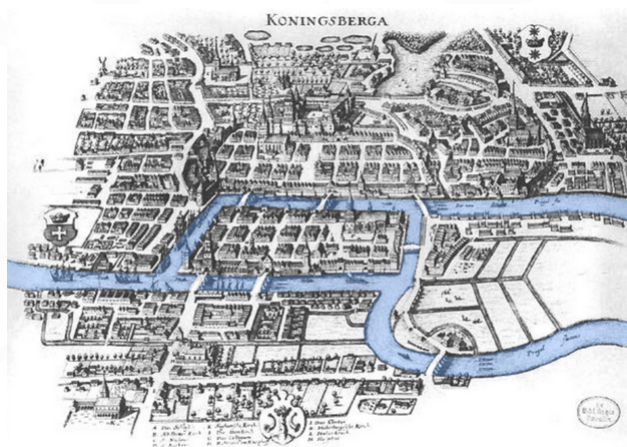


Figure 4.1: Königsberg

**Exercise 4.0.1** *Is it possible to build the eight bridge, so that one can complete a journey, walking through each of the eight bridges exactly once?*

### 4.1 Graphs and their relatives

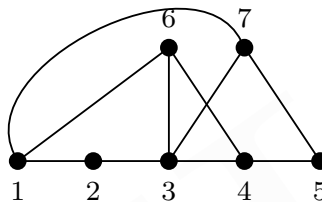
To solve Königsberg bridge problem, Euler represented each of the four lands (south and north banks of the river, as well as two islands of the river) by a point in the plane and connected the four points with seven lines to represent the bridges connecting the corresponding lands. He then studied the properties of the diagram obtained, which we nowadays refer to as a representation of a multigraph (‘multi’ comes from the fact some pairs of points are connected with multiple lines). Let us define these new objects formally.

We begin with the formal definition of a graph.

**Definition 4.1.1** A **graph**, also known as a **simple graph**, is a pair  $G = (V, E)$ , where

- $V$  is a finite set whose elements are called **vertices** (singular: *vertex*), and
- $E$  is a finite set of unordered pairs of vertices, whose elements are called **edges**.

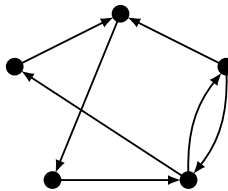
**Example 4.1.2** The sets  $V = \{1, 2, 3, 4, 5, 6, 7\}$  and  $E = \{\{1, 2\}, \{2, 3\}, \{3, 4\}, \{4, 5\}, \{6, 1\}, \{6, 3\}, \{6, 4\}, \{7, 1\}, \{7, 3\}, \{7, 5\}\}$  define a graph with 7 vertices and 10 edges. A natural visual representation of any graph can be obtained by representing each vertex by a point, and each edge by a line connecting two points. Thus our graph can be represented as follows:



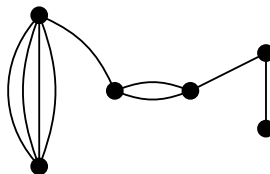
In our course, we will refer to such a diagram, consisting of points and lines connecting the pairs of points, simply as 'a graph', rather than a 'visual representation of a graph'.

**Remark 4.1.3** By altering our definition in various ways, we can obtain similar structures. For instance,

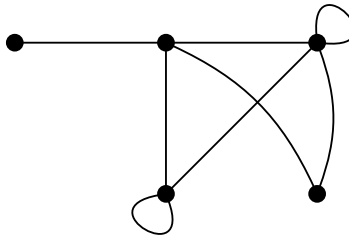
- by replacing the set  $E$  with a set of ordered pairs of vertices, we obtain a **directed graph** or a **digraph**, also known as **oriented graph**. Each edge of a digraph has a specific orientation.



- by allowing  $E$  to contain both directed and undirected edges, we obtain a **mixed graph**.
- by allowing  $E$  to contain repeated elements in the set of edges, technically replacing  $E$  with a multiset, we obtain a **multigraph**.



- by allowing edges to connect a vertex to itself, i.e. allowing "loops", we obtain a **pseudo-graph**.



- by allowing edges to be arbitrary subsets of vertices (rather than just pairs) gives us a **hypergraph**.

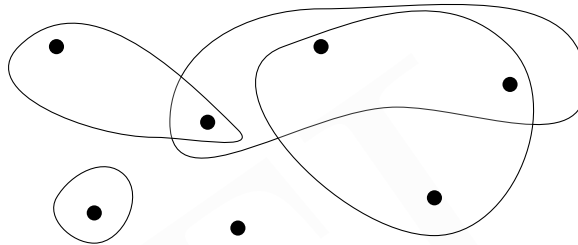


Figure 4.2: Hypergraph with 7 vertices and 4 edges

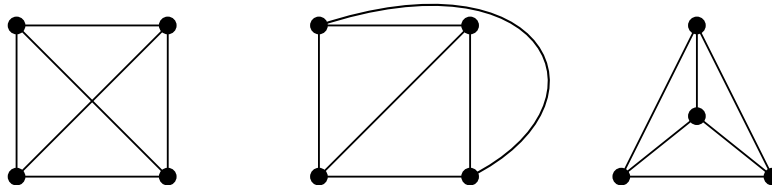
- by allowing  $V$  and  $E$  to be infinite sets, we obtain an **infinite graph**.

**Remark 4.1.4** Throughout the course, as stated in Definition 4.1.1 by a 'graph' we will mean a 'simple graph' (finite, undirected, without loops and multiple edges), unless stated otherwise.

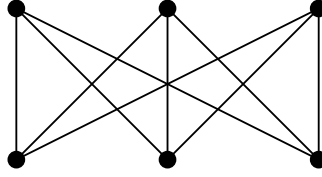
**Exercise 4.1.5** Draw the multigraph for the Königsberg bridge problem, with the 4 vertices corresponding to land regions and seven edges corresponding to the bridges connecting the respective land regions. Explain why the route visiting each bridge exactly once is impossible.

**Exercise 4.1.6** Eight people are seated around a circular table. Each person shakes hands with everyone at the table except the person sitting directly across the table. Draw a graph that models this situation. How many handshakes were there?

**Exercise 4.1.7** Explain whether the following graphs below should be considered as different or as the same.



**Exercise 4.1.8** In the previous exercise, the second and the third drawings are obtained from the first by representing edges differently and changing the positions of vertices in the plane. Note that in both case we were able to avoid crossings of the edges. Can you redraw the following graph below on the plane in such a way that no two edges cross?



[One can interpret this problem as finding a way to connect three houses to three utilities: gas, water and electricity, so that the lines (which can turn any number of times) never cross.]

## 4.2 Adjacency, neighbourhood, vertex degree, isomorphism, complement

The vertex set of a graph  $G$  is denoted by  $V(G)$  and the edge set is denoted by  $E(G)$ . We may refer to these sets simply by  $V$  and  $E$  if the graph under consideration is clear from the context. The **order** of the graph  $G$  is the cardinality of its vertex set, i.e. the number  $n = |V(G)|$ , and the **size** of  $G$  is the cardinality of its edge set, i.e. the number  $m = |E(G)|$ . In fact, letters  $n$  and  $m$  are commonly used to denote the order and the size of the graphs in the literature, and I would strongly encourage you to use these letters in your proofs as well.

For notational convenience, instead of representing an edge as  $\{u, v\}$ , we denote this simply as  $uv$ . Given two vertices  $u$  and  $v$ , if  $uv \in E$ , we say one of the following:

- $u$  is **adjacent** to  $v$ ,
- $u$  is **connected** to  $v$ ,
- $u$  is a **neighbour** of  $v$ .

Similarly, if  $uv \notin E$ , then  $u$  and  $v$  are said to be **nonadjacent** (**not connected**, **non-neighbours**). If  $uv \in E$ ,  $u$  and  $v$  are called the **end vertices** of the edge  $uv$ . Furthermore, if an edge  $e$  has a vertex  $v$  as an end vertex, we say that

- $v$  is **incident** to  $e$ .

The **neighbourhood** of a vertex  $v \in V(G)$ , denoted by  $N(v)$  is the set of vertices adjacent to  $v$ , i.e.

$$N(v) = \{u \in V(G) \mid uv \in E(G)\}.$$

The **degree** of  $v \in V(G)$ , denoted  $\deg(v)$  is the number of edges incident to  $v$ . Alternatively,  $\deg(v) = |N(v)|$ . If  $\deg(v) = 0$ , then the vertex  $v$  is called **isolated**. If  $\deg(v) = 1$ , then the vertex and the only edge incident to  $v$  are called **pendant**. The **maximum vertex degree** and **the minimum vertex degree** are denoted by  $\Delta(G)$  and  $\delta(G)$ , respectively. If all the degrees of the vertices of a graph  $G$  are equal to  $k$ , i.e. if  $\delta(G) = \Delta(G) = k$ , the graph  $G$  is said to be **k-regular** or **regular** (when  $k$  is not specified).

**Remark 4.2.1** The order, size, maximum and minimum vertex degrees are the first examples of **graph parameters** we encounter. Formally, a graph parameter is a function that associates a number with each graph, such as its order, size, etc. Knowing the values of certain parameters is of crucial importance for various applications, especially when the graphs are large.

We are now ready for the first result in graph theory:

**Lemma 4.2.2 (Handshake Lemma)** For any graph  $G$  we have

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

**Proof.** When summing the degrees of the vertices of  $G$ , each edge is counted twice, once for each of its two incident vertices. Hence the result holds. ■

**Exercise 4.2.3** There are 9 people in the conference room. Each person claims that he/she has shaken hands with exactly 3 of the others. Prove that at least one of them is lying.

**Exercise 4.2.4** What happens in the previous example if there are 8 people. Can each of them shake their hands with exactly three others?

**Exercise 4.2.5** Give an example of the following or explain why no such example exists:

- (a) a graph of order 7 whose vertices have degrees 1,1,1,2,2,3,3;
- (b) a graph of order 7 whose vertices have degrees 1,2,2,2,3,3,7;
- (c) a graph of order 4 whose vertices have degrees 1,3,3,3.

**Definition 4.2.6 (Isomorphism)**

- Two graphs  $G_1 = (V_1, E_1)$  and  $G_2 = (V_2, E_2)$  are **isomorphic** if there is a bijection  $f : V_1 \rightarrow V_2$  which preserves the adjacency, i.e.  $uv \in E_1$  if and only if  $f(u)f(v) \in E_2$ .

**Exercise 4.2.7** Find all pairwise non-isomorphic graphs with 2,3,4,5 vertices.

**Exercise 4.2.8** Find all pairwise non-isomorphic graphs of order 6 whose vertices have degrees 2,2,3,3,4,4.

**Definition 4.2.9 (Complement, self-complementary graphs)**

- The **complement** of a graph  $G = (V, E)$  is a graph with vertex set  $V$  and the edge set  $E'$  such that  $e \in E'$  if and only if  $e \notin E$ . The complement of a graph  $G$  is denoted  $\overline{G}$  and sometimes is called  $\text{co-}G$ .
- A graph  $G$  is **self-complementary** if  $G$  is isomorphic to its complement.

**Exercise 4.2.10** Find self-complementary graphs with 4,5,6 vertices.

**Exercise 4.2.11** Is it possible for a self-complementary graph with 100 vertices to have exactly one vertex of degree 50?

### 4.3 Paths, cycles, trees and Kruskal's algorithm

#### Definition 4.3.1 (Walk, path, cycle)

- A **walk** in a graph is a sequence of (not necessarily distinct) vertices  $v_1, v_2, \dots, v_k$  such that  $v_i v_{i+1} \in E$  for each  $i = 1, 2, \dots, k-1$ . Such a walk is sometimes called  $v_1 - v_k$  **walk**, and  $v_1$  and  $v_k$  are the **end vertices** of the walk.
- A **path** is a walk with all the vertices distinct.
- A **cycle** is a walk  $v_1, v_2, \dots, v_{k+1}$  with  $k \geq 3$  such that  $v_1, v_2, \dots, v_k$  are distinct and  $v_1 = v_{k+1}$ .
- The **length** of a walk (resp. path, cycle) is the number of edges, counting repetitions.

**Lemma 4.3.2** In a graph  $G$  with vertices  $u$  and  $v$ , every  $u - v$  walk, contains a  $u - v$  path.

**Proof.** We will prove by induction of the length of the walk. For the base case, it is clear that if the walk has length 0 or 1 (no edges, or just one edge), then the walk must be a path. Hence the base case holds. So consider a walk

$$u = v_0, v_1, \dots, v_k = v$$

of length  $k \geq 2$ , and assume the result is true for all walks of length less than  $k$ . If the vertices of the walk are distinct, then we are done, as the walk itself is the desired path. Otherwise, let  $i < j$  be any two integers such that  $v_i = v_j$ . But then

$$u = v_0, v_1, \dots, v_i, v_{j+1}, \dots, v_k = v$$

is a walk with length strictly less than  $k$ . By induction, this walk contains a  $u - v$  path and hence we are done. ■

**Exercise 4.3.3** Let  $G$  be a graph and  $x, y, z \in V(G)$ . Prove the following statements:

- (a) If there exists a  $x - y$  walk and a  $y - z$  walk, then there exists a  $x - z$  walk;
- (b) If there exists a  $x - y$  path and a  $y - z$  path, then there exists a  $x - z$  path.

#### Definition 4.3.4 (Connected graph)

- A graph is **connected** if every pair of distinct vertices is joined by a path. If a graph is not connected, we say it is **disconnected**.

#### Exercise 4.3.5

- (a) Is it true that the complement of a connected graph is necessarily disconnected?
- (b) Prove that the complement of a disconnected graph is necessarily connected.
- (c) Prove that if a graph is connected if and only if for every partition of its vertex set into two non-empty sets  $A$  and  $B$  there is an edge  $ab \in E(G)$  such that  $a \in A$  and  $b \in B$ .

The graph that is disconnected, informally speaking, consist of "several connected pieces". We will call these pieces "connected components", which we now define formally as follows:



**Definition 4.3.6 (Subgraph, connected components)**

- We say that a graph  $G_1 = (V_1, E_1)$  is a **subgraph** of a graph  $G_2 = (V_2, E_2)$  if  $V_1 \subseteq V_2$  and  $E_1 \subseteq E_2$ , i.e.  $G_1$  can be obtained from  $G_2$  by deleting some vertices and some edges.
- Each maximal connected subgraph of a graph is called a **connected component**.

**Definition 4.3.7 (Tree, forest, spanning tree)**

- A **tree** is a connected graph with no cycles.
- A **forest** is a graph with no cycles.
- A **spanning tree** of  $G$  is a subgraph of  $G$  which forms a tree on  $V(G)$  vertices.

By definition above it follows that every connected component of the forest is a tree. One can also see that a connected forest is a tree. Also note that a graph  $G$  may have no spanning trees, one spanning tree or many different spanning trees.

**Exercise 4.3.8** Find all pairwise non-isomorphic forests of order 6 which have

- exactly one connected component;
- exactly two connected components;
- at least three connected components.

**Exercise 4.3.9** Let  $F$  be a forest, consider the three numbers  $n = V(F)$ -number of vertices in the forest,  $m = E(F)$  - number of edges in the forest,  $c = c(F)$ - number of connected components in the forest. Looking at your results from the previous exercise, can you suggest a relationship between these three numbers?

**Theorem 4.3.10** The  $T$  be a tree. Then

$$|E(T)| = |V(T)| - 1$$

**Proof.** We will prove the statement by induction on  $n = |V(T)|$ . If  $n = 1$ , the tree has one vertex and no edges, so the required equation holds. Let  $n > 1$  and suppose, by induction, that the equation holds for all trees of size  $n - 1$ . Let  $T$  be a tree on  $V(T) = n$  vertices.

We will first show that  $T$  must contain a vertex of degree 1. For this purpose, consider a path  $v_1, v_2, \dots, v_k$  of maximal length. As  $n > 1$  and the graph is connected, the graph contains an edge, so  $k \geq 2$ . We claim, that  $v_k$  is a vertex of degree 1. Indeed,  $v_k$  is connected to  $v_{k-1}$ , but to none of  $v_1, v_2, \dots, v_{k-2}$  as otherwise we have a cycle. Similarly,  $v_k$  is not connected to any vertex from  $V(G) \setminus \{v_1, v_2, \dots, v_k\}$  since that would contradict the maximality of the path. Hence  $\deg(v_k) = 1$ .

Let us remove the vertex  $v = v_k$  of degree 1, together with the pendant edge  $v$  is incident to, from the graph  $G$ . In this way we obtain a graph  $G'$  with  $n - 1$  vertices. Note that by removing a vertex we did not create a cycle so  $G'$  is acyclic, and also  $G'$  is still connected (check this!). Hence,  $G'$  is a tree on  $n - 1$  vertices, and by induction it has  $n - 2$  edges. But then, since  $G$  has one more edge than  $G'$  (the pendant edge incident to vertex  $v$ ) we conclude that  $G$  has  $n - 1$  edges in total, and so we are done. ■

**Exercise 4.3.11** Using the theorem above, provide a rigorous proof of the relationship you conjectured in Exercise 4.3.9.

We will now illustrate one of many real-life applications of trees. Suppose we want to connect certain cities with electricity cables. Not all pairs of cities can be connected by cables, and for those who can be connected, there is an associated non-negative cost of installing the cables. Our task is connect all the cities by cables forming a single connected component, and minimizing the cost. Clearly any network containing a cycle is not cost-optimal, as one can remove an edge from the cycle reducing the cost, but keeping the same connectivity. Hence we are looking for a minimum cost spanning tree. We formalize this discussion as follows.

**Definition 4.3.12 (Cost/weight function, tree cost, minimum cost spanning tree)**

- Given a graph  $G$  a **cost function** (or a **weight function**) on  $E(G)$  is a function  $w : E(G) \rightarrow \mathbb{R}_{\geq 0}$ , where  $\mathbb{R}_{\geq 0}$  denotes the non-negative real numbers.
- Given a graph  $G$  and a cost function  $w$  on  $E(G)$ , we define a **tree cost** for any tree  $T$  which is a subgraph of  $G$  to be

$$w(T) = \sum_{e \in E(T)} w(e).$$

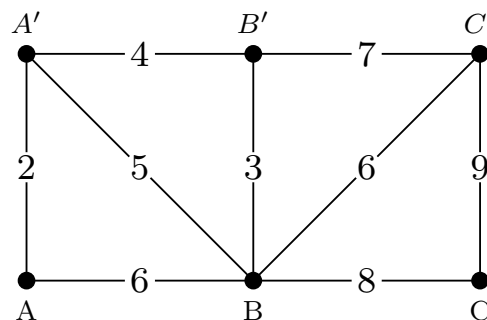
- We say that a spanning tree  $T$  of  $G$  is a **minimum cost spanning tree** if  $w(T) \leq w(T')$  for any other spanning tree of  $G$ .

The greedy algorithm, also known as Kruskal's algorithm, is one of the best known efficient algorithms (although not the only one) for finding the minimum cost spanning tree.

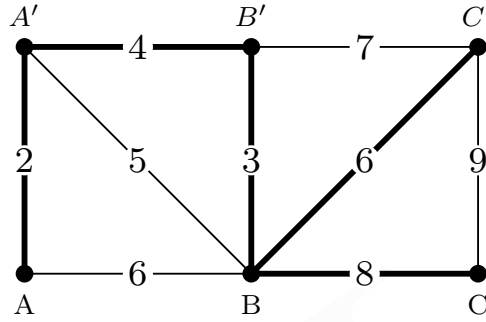
**Algorithm 4.3.13 (Kruskal's algorithm)**

- Choose an edge of smallest weight.
- At each stage, choose from the edges not yet chosen the edge of smallest weight whose inclusion will not create a cycle.
- Continue until a spanning tree is obtained.

**Example 4.3.14** Consider the following graph.



The algorithm, first suggests to choose the edge  $AA'$ , since it has the smallest weight 2. Then proceed choosing  $BB'$  and  $A'B'$  of weights 3 and 4. Then, we consider the edge  $A'B$  since it has weight 5, but we can't include it as it would create a cycle with the edges already chosen. For similar reason, we can't pick  $AB$  of weight 6. So the next edge we choose is  $BC'$  of weight 6. Since once again, we can't pick the edge  $B'C'$  of weight 7 as it would create a cycle, we pick our last edge to be  $BC$  of size 8, which completes the spanning tree. Our spanning tree consists of 5 edges  $AA'$ ,  $BB'$ ,  $A'B'$ ,  $BC'$ ,  $BC$  and has the total weight of  $2 + 3 + 4 + 6 + 8 = 23$ .



Note that it is not clear whether the tree obtained by the algorithm is of the minimal weight among all the possible trees. Maybe picking the edges in some other way one can achieve a better solution? Surprisingly, it turns out that this greedy approach does indeed always output the desired minimum weight spanning tree as justified by the theorem below.

**Theorem 4.3.15 (Correctness of Kruskal's algorithm)** *The output tree  $T$  obtained by Kruskal's algorithm is a minimum cost spanning tree.*

**Proof.** Suppose not, and consider, for contradiction a tree  $T^*$  with cost smaller than  $T$ . Let

$$E(T) = \{e_1, e_2, \dots, e_m\}$$

be the edges of  $T$  listed in the increasing weight order  $w(e_1) \leq w(e_2) \leq \dots \leq w(e_m)$ , i.e. in the order they were discovered by Kruskal's algorithm. As  $T^* \neq T$  (otherwise they would have the same cost), some of the edges of the two trees must differ, and let  $k$  be the smallest positive integer such that  $e_k \notin E(T^*)$ , i.e.  $e := e_k$  is the edge of the smallest weight in  $T$  that does not belong to the tree  $T^*$ . Let  $T^+$  be the graph obtained by adding the edge  $e$  to the graph  $T^*$ . Then  $T^+$  has a unique cycle  $C$ , consisting of the edge  $e$  and the edges of the unique path joining the endpoints of  $e$  in the tree  $T^*$ . Since  $e \in E(T)$ , and  $T$  is acyclic,  $C$  must not be fully contained in  $T$  and hence there must be an edge  $e^* \notin E(T)$ . Also note that since  $e \in E(T)$ , we have that  $e^*$  must be different from  $e$ .

Hence we have two edges  $e$ , belonging to  $T$  but not  $T^*$  and  $e^*$  belonging to  $T^*$  but not to  $T$ . Here comes one of the crucial observations. Note that

$$\{e_1, e_2, \dots, e_{k-1}, e\} \subseteq E(T)$$

and

$$\{e_1, e_2, \dots, e_{k-1}, e^*\} \subseteq E(T^*).$$

Hence, since  $T$  and  $T^*$  are trees, neither  $e$  nor  $e^*$  forms a cycle with the edges from  $\{e_1, e_2, \dots, e_{k-1}\}$ . Hence, since at the  $k$ 'th step of Kruskal's algorithm the edge  $e$  was chosen, we conclude that

$$w(e) \leq w(e^*).$$

Finally, since  $C$  is the unique cycle of  $T^+$ , removing the edge  $e^*$  gives us a tree, let's call it  $T^{**}$ . Note that

$$w(T^{**}) = w(T^*) + w(e) - w(e^*) \leq w(T^*) < w(T).$$

Note that  $T^{**}$  is a tree which still has the cost smaller than  $T$ , but it has one more edge in common with  $T$  than  $T^*$  had. Repeating this process we eventually change  $T^{**}$  into  $T$  and would obtain  $w(T) \leq w(T^*) < w(T)$ , which is a contradiction.

■

We will finish the section with discussing the algorithmic complexity of Kruskal's algorithm.

**Remark 4.3.16 (Running time of Kruskal's algorithm)**

*Here we will provide a simple implementation of Kruskal's algorithm, from the first principles, with running time  $O(E^2)$ . We first start with noting that to run the algorithm, we would need to sort the edges in the increasing order of weight. The simplest way to do this is the following so-called exchange-sort algorithm, which compares every pair of edges, and swaps if they are in decreasing order:*

```

for  $i = 1$  to  $|E| - 1$ 
    for  $j = i + 1$  to  $|E|$ 
        if  $w(e_i) > w(e_j)$ :
             $x \leftarrow e_i$ 
             $e_i \leftarrow e_j$ 
             $e_j \leftarrow x$ 

```

*We note that most programming languages would allow  $\text{swap}(e_i, e_j)$  operation directly (even sorting commands are all pre-built), but we wrote the explicit process in order to estimate the number of elementary operations computer needs to execute the sorting. Here we ask the computer to store the  $i$ 'th edge temporarily in the cell of computer memory assigned to variable  $x$  and execute the swap in three elementary steps described by the pseudocode above. Since comparing two numbers (weights of the edges), and the three elementary steps needed to swap two edges take constant time, and we have  $\binom{E}{2}$  iterations of this procedure, we conclude that the complexity of this sorting algorithm is  $O(E^2)$ .*

*Next, following Kruskal's algorithm, we would need to add edges one-by-one ensuring no cycle is created. For the purpose of checking whether a cycle arises, a good idea is to consider connected components created at each step of the procedure. We illustrate this with the example from before. At the beginning, we have no edges, so we have 6 connected components, corresponding to the vertices  $\{A\}, \{A'\}, \{B\}, \{B'\}, \{C\}, \{C'\}$ . Then we pick the edge  $AA'$ , after which the list of connected components are  $\{A, A'\}, \{B\}, \{B'\}, \{C\}, \{C'\}$ . Adding the edge  $BB'$  gives  $\{A, A'\}, \{B, B'\}, \{C\}, \{C'\}$  and the edge  $A'B'$  leads us to  $\{A, A', B, B'\}, \{C\}, \{C'\}$ . Now, the next edge according to the minimum weight is  $A'B$  which has both endpoints in the same connected component, hence creates the cycle! Same with edge  $AB$ ! Then we add  $BC'$  as it gives*

us  $\{A, A', B, B', C'\}, \{C\}$ , and skip the edge  $B'C'$  as it is now having both endpoints in one of the previously created component (hence a cycle) and we finish the procedure by adding the edge  $BC$ .

Let's consider a simple implementation of the idea above. We will denote the vertices simply by  $\{1, 2, \dots, |V|\}$ , edges  $e_i = (x_i, y_i)$  for  $i = 1, 2, \dots, |E|$  and let  $c(v)$  denote the component of vertex  $v$ . Initially, let each vertex have it's own component, say  $c(v) = v$ , and if an edge joins two different components, we make the component number the same for all vertices in the two components. Our output edges of the minimal spanning tree will be stored in the set  $T$ .

```

 $T \leftarrow \emptyset$                                 { We start with an empty set of edges.}

for  $v = 1$  to  $|V|$ 

     $c(v) \leftarrow v$                             { At the start each vertex is in its own component.}

for  $i = 1$  to  $|E|$ 

    if  $c(x_i) \neq c(y_i)$ :                        { If the endpoints of  $e_i$  belong to different components,}
         $T \leftarrow T \cup \{e_i\}$                 { we choose this edge for our minimal spanning tree and}
        for  $v = 1$  to  $|V|$ 
            if  $c(v) = c(y_i)$ :
                 $c(v) \leftarrow c(x_i)$  { we relabel vertices of component  $c(y_i)$  to component  $c(x_i)$ .}

```

Looking at the code, it's clear that the complexity of this part is  $O(|V||E|)$  and since normally, the graph is connected (otherwise we can just output that there is no spanning tree), we know that  $|E| \geq |V| - 1$  and so the overall complexity is  $O(|E|^2) + O(|E||V|) = O(E^2)$ , as required.

**Remark 4.3.17** The state of art complexity for the Kruskal's algorithm is  $O(|E|\log|E|)$ , but the detailed proof of this fact is outside the scope of this course. To get the best known complexity, we would need to modify our simple sorting algorithm to one of the divide and conquer recursive sorting algorithms, such as Merge-sort. We would also need to improve the second part, by using sets rather than arrays to keep track of the connected components. Nevertheless, we would like to note that a polynomial (quadratic) algorithm we provided is already considered very efficient in graph theory. In fact, for many of the most important graph theory problems the best known precise algorithms run in exponential time, which is a big challenge in computer science.

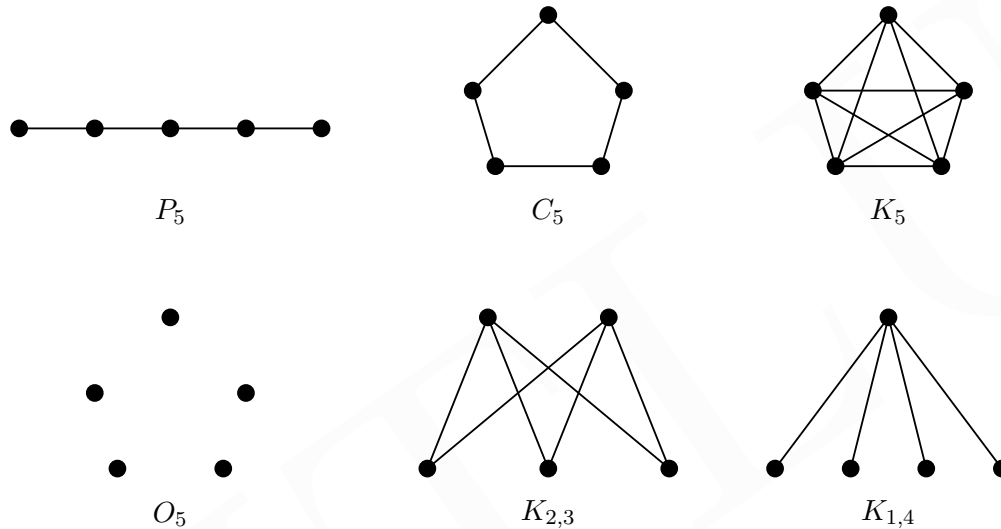
## 4.4 Special graphs and graph operations

Below we list the notation for some very important special graphs:

- $P_n$  is a **chordless path** with  $n$  vertices, i.e.  $V(P_n) = \{v_1, v_2, \dots, v_n\}$  and  $E(P_n) = \{v_1v_2, v_2v_3, \dots, v_{n-1}v_n\}$ .
- $C_n$  is a **chordless cycle** with  $n$  vertices, i.e.  $V(C_n) = \{v_1, v_2, \dots, v_n\}$  and  $E(C_n) = \{v_1v_2, v_2v_3, \dots, v_{n-1}v_n, v_nv_1\}$ .
- $K_n$  is the **complete graph** with  $n$  vertices, i.e. the graph with  $n$  vertices every two of which are adjacent.

- $O_n$  is the **empty (edgeless)** graph with  $n$  vertices, i.e. the graph with  $n$  vertices no two of which are adjacent.
- $K_{n,m}$  is a **complete bipartite graph** which consists of two disjoint empty parts of sizes  $n$  and  $m$  and all possible edges between the parts, i.e. the graph with  $n + m$  vertices  $V(G) = \{a_1, a_2, \dots, a_n\} \cup \{b_1, b_2, \dots, b_m\}$  and  $E(G) = \{a_i b_j : 1 \leq i \leq n, 1 \leq j \leq m\}$ .
- $K_{1,m}$  is called a **star**.

We provide examples of these graphs:



**Exercise 4.4.1** Which of the graphs  $P_n, C_n, K_n, O_n, K_{n,m}, K_{1,n}$  for  $n, m \geq 1$ , are self-complementary?

We have seen a definition of a subgraph in the previous section. The subgraph of a graph is a graph obtained by arbitrary deletion of vertices and edges. A more restricted, but very important notion below, defines a special type of subgraph obtained by vertex deletion only (the edges incident to the deleted vertices, of course, have to be removed, but no further edges are allowed to be deleted):

**Definition 4.4.2 (Induced subgraph)**

- A graph  $H$  is an **induced subgraph** of a graph  $G$  if  $V(H) \subseteq V(G)$  and for any pair of distinct vertices  $v, w \in V(H)$ , we have  $vw \in E(H)$  if and only if  $vw \in E(G)$ .

Note that every induced subgraph of a graph  $G$  is also a subgraph of the graph  $G$ , but the converse is not true. For instance,  $P_5$  is a subgraph of  $C_5$ , but not an induced subgraph of  $C_5$ . Another example can be obtained by observing that every graph of order up to 5 is a subgraph of  $K_5$ , however, only  $K_1, K_2, K_3, K_4$  and  $K_5$  are induced subgraphs of  $K_5$ .

Some convenient notation for (induced) subgraphs are as follows.

**Definition 4.4.3 (Removal of vertex/vertex set/edge)** Let  $G$  be a graph,  $v \in V(G)$ ,  $W \subseteq V(G)$ ,  $e \in E(G)$ . Then,

- $G - v$  denotes the induced subgraph of  $G$  with vertex set  $V(G) \setminus \{v\}$ .

- $G - W$  denotes the induced subgraph of  $G$  with vertex set  $V(G) \setminus W$ .
- $G - e$  denotes the graph with vertex set  $V(G)$  and edge set  $E(G) - e$ .

Also, it might be useful, to have some notation to denote the disjoint unions of graphs:

**Definition 4.4.4 (Graph operations)** Let  $G, H$  be two graphs. Then,

- $G + H$  denotes the disjoint union of graphs  $G$  and  $H$ , i.e. the graph with vertex set  $V(G) \cup V(H)$  and edge set  $E(G) \cup E(H)$ .
- $nG$  denotes the disjoint union of  $n$  copies of graph  $G$ , i.e. the graph  $G + G + \dots + G$  (with  $n$  additions).

**Example 4.4.5** The induced subgraphs of  $P_5$  can now be listed as:  $P_5, P_4, P_3 + K_1, 2K_2, P_3, K_1 + K_2, O_3, O_2, K_2, K_1$ . Note that a single vertex can be described in couple of ways  $K_1 = P_1 = O_1$ , but normally, in the literature,  $K_1$  is used. Writing  $P_3 + v$  or  $P_3 + \{v\}$  instead of  $P_3 + K_1$  is not a correct of the notation, since neither  $v$  nor  $\{v\}$  denote graphs (they denote a vertex and a set containing a vertex, respectively).

**Exercise 4.4.6** List all the induced subgraphs of  $K_{2,3}$  and of  $2P_3$ .

Recall that the complement of  $G$  is denoted by  $\overline{G}$ , which is another graph operation we can use when describing various graphs. For instance,  $K_{n,m}$  could have been defined simply by saying that it is the graph  $\overline{K_n + K_m}$ .

**Exercise 4.4.7** Using the notation of the special graphs, together with operations of disjoint unions and complementations, list all pairwise non-isomorphic the graphs of order 4.

**Remark 4.4.8** It is interesting to note that some of the graphs, have special names, according to the shape they resemble. For instance,  $K_{1,3}$  is often referred to as a 'claw', while the complement of  $P_5$  is referred to as a 'house' (can you see why?). You can possibly guess the definitions of the graphs referred to as a 'chair', a 'diamond' or a 'domino', but it might be harder to guess the definitions of a 'caterpillar', a 'spider' or a 'rising sun'.

## 4.5 Cliques, independent sets and Ramsey theorem

In Section 4.3 we have shown an algorithm how to find a spanning tree in any given graph. In this section we will consider another fundamental question in graph theory of finding large complete subgraphs (known as cliques) or large induced empty subgraphs (known as independent sets).

**Definition 4.5.1 (Clique, independent set)**

- In a graph, a set of pairwise adjacent vertices is called a **clique**. The size of the maximum clique in  $G$  is called the **clique number** of  $G$  and it is denoted by  $\omega(G)$ .
- A set of pairwise non-adjacent vertices is called an **independent set**. The size of maximal independent set is called the **independence number** of  $G$  and is denoted by  $\alpha(G)$ .

**Exercise 4.5.2** Find the clique number and independence number for each of the following graphs:  $K_n, K_{n,n}, P_n, C_n, P_n + C_n$ .

**Theorem 4.5.3** *Among any 6 people, there exists either 3 mutual friends, or 3 mutual strangers.*

**Proof.** Pick any 6 people. Consider a complete graph on 6 vertices, in which any vertex represents a person. Colour the edge between two vertices red, if the corresponding people are friends, and blue otherwise (we will always assume that friendship is a mutual relationship).

Take a vertex, call it  $A$ , and note that the vertex  $A$  is incident to five edges, some of them are red, others are blue. By Pigeonhole principle, three of these five edges must have the same colour. Without loss of generality (WLOG) we can assume that we have three blue edges incident to  $A$  which we can denote by  $AB$ ,  $AC$ ,  $AD$ . Now consider the triangle  $BCD$ . If any of its edges is blue, then together with vertex  $A$ , we have a blue triangle, and so we are done - we found three mutual strangers. On the other hand, if none of its edges are blue, then  $BCD$  is a red triangle and we are again done - we found three mutual friends. Hence, in any case we can find either three mutual friends or three mutual strangers, which completes the proof. ■

Note that the theorem is not true if 6 is replaced by 5. Indeed, given a  $K_5$  we can choose a  $C_5$  subgraph and color its edges red, and the remaining five edges of  $K_5$  (which also form a  $C_5$ ) blue. Then we obtain a coloring of  $K_5$  with two colors, but no red or blue triangle.

This motivates us to define the following:

**Definition 4.5.4 (Ramsey Numbers)** *Let  $n, m \in \mathbb{N}$ . Define*

- $R(n, m)$  *to be the smallest integer  $N$  such that for any red-blue colouring of edges of  $K_N$ , there is either a red  $K_n$  subgraph or a blue  $K_m$  subgraph.*

We have already shown that  $R(3, 3) = 6$ . Also, it is not hard to see that  $R(2, n) = R(n, 2) = n$  (check this!) for any  $n \geq 2$ . The next sensible question to wonder about the value of  $R(4, 4)$ . Does this number even exist? Maybe for any value  $N$  there is a 'nasty' colouring of  $K_N$  with red and blue colours, so that any 4-vertex subgraph of  $K_N$  has both blue and red edges, and no monochromatic (same colour)  $K_4$  can be found? And if  $R(4, 4)$  exists, what about  $R(5, 5)$ ,  $R(10, 10)$ , etc.?

The positive answer to the existence question has been obtained by a British mathematician, philosopher and economist Frank Ramsey in the following theorem, which was a starting point of an important branch in mathematics called "Ramsey Theory". The branch comes with its own philosophy, saying that complete disorder is impossible, and every disorder, if it is large enough contains some big orderly pattern, in this case, a big monochromatic clique.

**Theorem 4.5.5 (Ramsey theorem)** *For any  $n, m \in \mathbb{N}$  we have*

$$R(n, m) \leq 2^{n+m}.$$

**Proof.** We will prove the theorem by induction on  $n + m$ . For  $n = 1$  or  $m = 1$ , the inequality holds since  $R(1, m) = R(n, 1) = 1$  for all  $n, m \in \mathbb{N}$ . Suppose now  $n, m \geq 2$  and the inequality holds for all pairs of integers  $n', m' \in \mathbb{N}$  with  $n' + m' < n + m$ . Consider a complete graph on  $2^{n+m}$  vertices whose edges are coloured either blue or red. Pick an arbitrary vertex  $v$ . Then, since  $v$  is incident to  $2^{n+m} - 1$  edges, each either red or blue, by Pigeonhole Principle  $v$  is incident to either  $2^{n+m-1}$  red edges or to  $2^{n+m-1}$  blue edges.

First consider the case when the neighbourhood of  $v$  induced by the red edges is of size at least  $2^{n+m-1}$ . Then, since  $R(n-1, m) \leq 2^{n+m-1}$ , by induction hypothesis (since  $n-1 + m < n + m$ ) the red neighbourhood of  $v$  contains either a red  $K_{n-1}$  subgraph or a blue  $K_m$  subgraph. If



blue  $K_m$  appears, then we are done, and if red  $K_{n-1}$  appears, then together with the vertex  $v$  we have red  $K_n$  and so we are done as well.

Similarly, if  $v$  has  $2^{n+m-1}$  blue edges, then the blue neighbourhood is of size at least  $R(n, m-1)$  and by induction we can find either red  $K_n$  or a blue  $K_{m-1}$  inside. Once again, if red  $K_n$  is found we are done, and in case of blue  $K_{m-1}$ , we can adjoin vertex  $v$ , to obtain a blue  $K_m$ . Hence we proved that in any red-blue coloured complete graph on  $2^{n+m}$  vertices either a red  $K_n$  or  $K_m$  appears. Therefore, by induction we are done. ■

The lower bound for Ramsey numbers is given by a prominent mathematician Paul Erdős.

**Theorem 4.5.6 (Lower bound for Ramsey numbers)** *For any  $n > 4$ , we have*

$$R(n, n) \geq (\sqrt{2})^n$$

**Proof.** We need to show that there exists a red-blue edge-coloured complete graph on  $(\sqrt{2})^n$  vertices with no monochromatic (same colour)  $K_n$ .

Consider a random edge-colouring of  $K_N$  where each edge is coloured blue with probability  $\frac{1}{2}$  and red with probability  $\frac{1}{2}$ . The probability that a randomly chosen  $K_n$  is monochromatic is  $p = 2(\frac{1}{2})^{n(n-1)/2}$ . Since there are  $\binom{N}{n}$  ways to choose an  $n$ -element subset, we have that the expectation of the number of monochromatic cliques  $X$  (the number of monochromatic  $K_n$ 's) is

$$EX = \binom{N}{n} \cdot 2\left(\frac{1}{2}\right)^{n(n-1)/2}$$

For  $N = \lfloor 2^{n/2} \rfloor$  we have

$$EX \leq \frac{N^n}{2^{n-1}} \times 2 \times \left(\frac{1}{2}\right)^{n(n-1)/2} = 2^{n^2/2 - (n-1) + 1 - n(n-1)/2} = 2^{2-n/2} < 1,$$

for  $n > 4$  Hence, since the expected number of monochromatic cliques is less than 1, there must exist at least one graph on  $\lfloor 2^{n/2} \rfloor$  vertices with no monochromatic  $K_n$ , which is what we needed to prove. ■

**Remark 4.5.7** *We have shown that  $\sqrt{2}^n \leq R(n, n) \leq 4^n$ , and in fact improving any of the numbers  $\sqrt{2}$  and 4 is a hard unsolved open question. If solved, a person would likely be awarded some mathematical prize, possibly a Fields medal.*

**Remark 4.5.8** *Finding precise Ramsey numbers is also a very challenging question, and apart from the trivial values  $R(1, n) = R(n, 1) = 1$  and  $R(2, n) = R(n, 2) = n$ , only a few more precise values are known (see the table below). Note that even  $R(5, 5)$  unknown and  $R(6, 6)$ , as believed by Erdős, would never be found even with the most powerful computers (can you tell why?).*

$r \backslash s$	1	2	3	4	5	6	7	8	9	10
1	1	1	1	1	1	1	1	1	1	1
2		2	3	4	5	6	7	8	9	10
3			6	9	14	18	23	28	36	40–42
4				18	25 <sup>[9]</sup>	36–41	49–61	59 <sup>[13]</sup> –84	73–115	92–149
5					43–48	58–87	80–143	101–216	133–316	149 <sup>[13]</sup> –442
6						102–165	115 <sup>[13]</sup> –298	134 <sup>[13]</sup> –495	183–780	204–1171

It is not hard to see that that looking for monochromatic clique within a 2-coloured complete graph is an equivalent question to finding a complete or an empty induced set within an arbitrary graph. The question is very important theoretically and practically (think about finding websites in the webgraph related to the same topic, or identifying a group of friends in a social network). The next remark reviews the computational complexity of this question.

**Remark 4.5.9 (Complexity of the maximum independent set problem)**

*While an independent set looks a very simple structure, finding a maximum independent set is a hard problem, for which no polynomial time algorithm is known. Note that for a graph with  $n$  vertices has  $2^n$  vertex subsets and a naive brute force algorithm would have to check all of them. We can do slightly better than this with a recursive algorithm as follows.*

*Recall that  $\alpha(G)$  denotes the maximum number of vertices in an independent set of the graph  $G$ . Given a graph  $G$ , we pick a vertex  $v$  and consider any independent set  $S$ . If  $S$  contains  $v$ , then the set  $S$  consist of  $v$  and an independent set of  $G - (N(v) \cup \{v\})$ . Otherwise, if  $S$  does not contain  $v$ ,  $S$  is an independent set of  $G - v$ . This gives us the following recursion:*

$$\alpha(G) = \max(1 + \alpha(G - (N(v) \cup \{v\})), \alpha(G - v))$$

*Let's denote  $f(n)$  the maximum number of steps needed to find the largest independent set of a graph on  $n$  vertices, with maximum taken over all  $n$ -vertex graphs. We note that making new graphs  $G - v$  and  $G - v - N(v)$  from the graph  $G$  (which one can consider as dealing with matrices of up to  $n^2$  entries) may require up to  $cn^2$  elementary operations. After this we apply recursion which finds the largest independent set of  $G - v$  in at most  $f(n - 1)$  elementary steps, and the largest independent set of  $G - v - N(v)$  in at most  $f(n - 2)$  steps (provided  $N(v) \neq \emptyset$  in which case we already found the largest independent set in previous step). Hence,*

$$f(n) \leq cn^2 + f(n - 1) + f(n - 2).$$

*It's a (non-homogeneous variant of) Fibonacci sequence, and hence we know that  $f(n) = O(1.619^n)$  where the precise value of the exponential growth is  $\frac{1+\sqrt{5}}{2} = 1.61803\dots$*

*In fact, we could go a bit further. What if we modified an algorithm to always choose a vertex of degree at least 2. It's not hard to see that for graphs, for which we cannot choose such vertex, i.e. whose every degree is at most 1, independent set could be found in linear time. Hence, our algorithm will have worst case complexity if the vertex has degree at equal to 2 which gives the new equation for the modified algorithm to be*

$$f(n) \leq cn^2 + f(n - 1) + f(n - 3).$$

*The complexity of the new algorithm is now improved to  $f(n) = O(1.466^n)$  where the precise growth of  $c = 1.46557\dots$  is the positive root of  $c^3 = c^2 + 1$ .*

*We could push a little bit further, insisting that a vertex we pick has a degree 3, giving us complexity of  $O(1.39^n)$ , however, perhaps surprisingly, this is the point where we have to stop and cannot do any better. In fact, the reason why we could have assumed that we have a vertex of degree 2, was because degree-at-most-1 graphs are so simple, they can be done in linear time. Similarly, we could have assumed that we have a degree 3 graph, because the degree-at-most-2 graphs are again simple and we could provide a linear time algorithm to find an independent set in them (check this!). However, degree-at-most-3 graphs are no longer simple, and no-one knows an efficient algorithm to find a maximum independent set in them. In fact, it is possible to show (but outside the scope of this course) that they are as complex as the general graphs, by*

*showing that if one could find an efficient algorithm for degree-at-most-3 graphs, then one would find an efficient algorithm for all graphs. As none of such efficient algorithms have been found yet, our approach cannot be improved any further, and we must stop here.*

In this section we've seen discussions/proofs of three results that have been very influential to the development of mathematics and computer science. Firstly, the upper bound for Ramsey numbers is in fact the first result of this type, that grew out later into a branch called "Ramsey theory" which concerns with finding large ordered systems within large chaos. The ordered patterns within chaos do not have to be cliques or independent sets, but, for instance, can be "Star forests and related graphs" as a recent result published in Journal of Discrete Mathematics by your teacher reveals. Secondly, the lower bound result by Erdős gave birth to so-called "Probabilistic method" and subsequently many crucial applications to "Randomised algorithms". Finally, the third result of finding maximum independent set is an example of a famous  $P \neq NP$  problem, which is one of the most famous problem in computer science. The importance of it lies in the fact, that if one managed to find independent set in polynomial time, they would be able to solve a big list of other problems in polynomial time, which would have enormous impact on computer science and its applications.



Figure 4.3: Frank Ramsey (left picture), Paul Erdős with young Terence Tao (right picture)

**Remark 4.5.10** *Paul Erdős (1913-1996) was one of the most productive mathematicians and producers of mathematical conjectures in the 20th century. He has published around 1500 mathematical papers during his lifetime, a figure that remains unsurpassed. Most of his work centered around discrete mathematics, cracking many of the previously unsolved problems in the field, but he also contributed to number theory, analysis, approximation theory, probability theory. Erdős spent most of his life travelling between scientific conferences, universities and homes of the colleagues all over the world, with most of his belongings fitting in the suitcase. He would typically show up at a colleague's doorstep and announce "my brains are open", staying long enough to collaborate on a few papers before moving on a few days later. In many cases, he would ask the current collaborator about whom should he visit next. Because of the high output and large number of collaborators, mathematicians defined a notion called **Erdős number** to be a collaboration distance from Erdős. Erdős alone was assigned the Erdős number 0, while his immediate collaborators could claim Erdős number 1, their collaborators have Erdős number 2, and so on. For instance, your teacher has Erdős number 3.*

## Chapter 5

# Graph colouring

In 1852, while trying trying to color the map of counties of England, Francis Guthrie, then a student at UCL, noticed that four different colours were required so that no two regions sharing a common border were the same colour. Curiously, Francis conjectured that 4 colours would be sufficient to colour any map. Francis forwarded this question to his mathematics lecturer Prof. Augustus De Morgan, who shared this question among his colleagues and a wider society. This became known as the Four Color Problem. In 1879 Alfred Kempe has proposed a proof for this fact, and for 11 years the mathematical community believed the question was settled, until in 1890 Percy Heawood found a mistake in Kempe's proof. Attempts were made to adapt Kempe's proof, or find a new proof, or find a counterexample, but none of them were successful. The question remained one of the most famous unsolved problems in graph theory and topology for more than half a century, until it was eventually proven in 1976 by Kenneth Appel and Wolfgang Haken using a lengthy computer-aided proof, which was also one of the first known computer-aided proofs in mathematics.



Figure 5.1: Counties of the UK

### 5.1 Planar graphs

A planar map, such as the map of counties of the UK, can be easily transformed into the graph as follows. Each region (county) is represented by a vertex, and we join the two vertices by an edge, if the two regions share a border. It is not hard to see that the graphs obtained in such a way are the graphs that can be drawn in the plane without edges crossing.

**Definition 5.1.1 (Planarity)** *A graph is **G planar** if  $G$  can be drawn on the plane without edges crossing.*

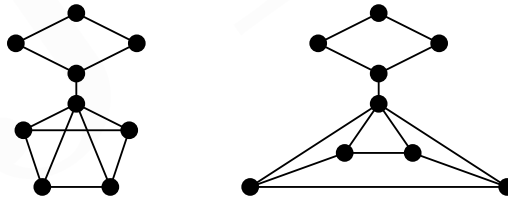
For example,  $K_4$  is planar as it can be drawn in the plane without edges crossing, with some examples of such drawings already seen in Exercise 4.1.7.

**Exercise 5.1.2** Show that the graphs  $K_{2,3}$  and  $K_5 - e$  are planar. Is  $K_{3,3}$  or  $K_5$  planar?

A **planar representation**, or a **planar embedding**/an **embedding** of a graph  $G$  is a drawing of the graph in the plane without edges crossing. Given a planar representation of a graph  $G$ , a **face** (also called **region**) is a maximal section of the plane in which two points can be joined by a curve that does not intersect any part of  $G$ . When we trace around the boundary of a face in  $G$ , we encounter a sequence of vertices and edges, finally returning to our final position. Let  $v_1, e_1, v_2, e_2, \dots, v_d, e_d, v_1$  be the sequence obtained by tracing around a face, then  $d$  is the **degree** of the face. Some edges might be encountered twice because both sides of them are on the same face. A tree is an extreme example of this: each edge is encountered twice. Finally, we will call a region **unbounded** if it is an unbounded set with respect to Euclidean metric (i.e. the Euclidean distances between pairs of points belonging to the region are unbounded).

**Remark 5.1.3** A planar representation traditionally have the vertices represented by points in the Euclidean plane and its edges are represented by curves between these points in the Euclidean plane, and different curves only meet in common endpoints. To avoid unnecessary topological complications, in graph theory it is often assumed that the curves representing edges are **piecewise linear** - consisting of finitely many straight line segments. In fact, it can be shown that any drawing can be straightened out in this way, so the two notions represent the same thing. Thus, in this course, if needed, you can always assume that a planar graph has an embedding whose curves representing edges are piecewise linear.

**Example 5.1.4** Below we see a planar graph (on the left) and its planar embedding (on the right) with 6 faces, whose degrees are 3, 3, 3, 4, 4, 9.



**Exercise 5.1.5** Consider the embedding of the planar graph in the example above.

- Note that the unbounded face has degree 9. Can you draw another planar embedding of the same graph for which the unbounded face has degree less than 9?
- Note that the sum of the degrees of faces is  $3 + 3 + 3 + 4 + 4 + 9$ , which is even. Do we always obtain an even sum for any embedding of any planar graph?

**Theorem 5.1.6 (Euler's formula)** Let  $G$  be a connected planar graph, and let  $V, E$  and  $F$  be the number of vertices, edges and faces (two-dimensional regions) in the planar embedding of  $G$ . Then

$$V - E + F = 2.$$

**Proof.** We will prove the Euler's formula by induction on the number of faces. Since  $G$  is connected,  $G$  has one (unbounded) face if and only if  $G$  is a tree. Then we have  $F = 1$ ,  $V = n$  -the number of vertices,  $E = n - 1$ , and so

$$V - E + F = n - (n - 1) + 1 = 2$$

, and so the base case holds.

Suppose now  $G$  is a connected graph with  $F > 1$  faces, and the assumption holds for all connected graphs with less than  $F$  faces. Since  $F > 1$ ,  $G$  must contain a cycle  $C$ . Let  $e$  be any edge of  $C$ . Then it is clear that  $G - e$  is a planar connected graph (why?), with  $F' = F - 1$  faces,  $E' = E - 1$  edges and  $V' = V$  vertices. Since  $F' < F$ , by induction we have

$$V' - E' + F' = 2.$$

But this gives us  $V - (E - 1) - (F - 1) = 2$ , and so  $V - E + F = 2$ , which finishes the proof. ■

**Theorem 5.1.7**  $K_5$  and  $K_{3,3}$  are not planar.

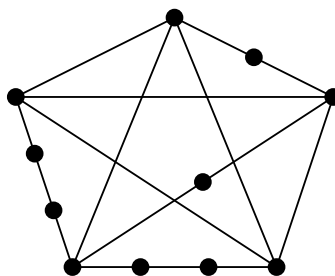
**Proof.** Suppose, for contradiction,  $K_5$  is planar, and consider a planar embedding of  $K_5$ . We know that  $V = 5$ ,  $E = \binom{5}{2} = 10$ . Let  $f_k$  be the number of faces of degree  $k$  in the planar embedding of  $K_5$ . Note that each face has degree at least 3, and each edge is either a boundary separating two faces, contributing degree 1 to each of them, or has the same face on both sides in which case it contributes degree 2 to that one face only. Hence counting the total sum of degrees of all faces we obtain

$$3F \leq \sum_{k \geq 3} k f_k = 2E.$$

Since  $E = 10$ , it follows that we have  $F \leq \frac{20}{3}$ , and so  $F \leq 6$ . But then  $V - E + F \leq 5 - 10 + 6 = 1$  which violates Euler's formula. Hence  $K_5$  is not planar. A similar argument can be used to show that  $K_{3,3}$  is not planar, which is left for the reader as an exercise (prove this!). ■

**Definition 5.1.8 (Subdivision)** A graph  $H$  is said to be a **subdivision** of a graph  $G$  if  $H$  can be obtained from  $G$  by successively deleting an edge  $uv \in E(G)$ , and adding a new vertex  $w^*$  of degree 2, joined to vertices  $u$  and  $v$ .

The following figure represents a subdivision of  $K_5$ .



Since we know that  $K_5$  and  $K_{3,3}$  are not planar, it is not hard to see that any subdivision of these graphs are not planar. Hence, if the graph contains as a subgraph a subdivision of  $K_5$  or  $K_{3,3}$  as a subgraph, the graph is not planar. Surprisingly, as proven by a Polish mathematician Kazimierz Kuratowski, the converse is true and these are the only obstructions for planarity. We provide the statement of the celebrated result below, however, the proof of it, though beautiful and insightful, is slightly too long to fit within the scope of this course.

**Theorem 5.1.9 (Kuratowski's theorem)** *A graph  $G$  is planar if and only if  $G$  does not contain a subdivision of  $K_5$  or  $K_{3,3}$ , as a subgraph.*

## 5.2 Chromatic number and the 4-colour theorem

**Definition 5.2.1 (Colouring, Chromatic number)**

- A **colouring** (or a **vertex colouring** or a **proper vertex colouring**) of a graph  $G$  is a function  $c$  which assigns a color for each vertex of  $G$  such that

$$c(v) \neq c(w) \text{ whenever } vw \in E(G).$$

- Given a graph  $G$ , the **chromatic number**  $\chi(G)$  of  $G$  is the least number of colors needed to colour the graph.

**Exercise 5.2.2** Find  $\chi(K_n)$ ,  $\chi(C_n)$ ,  $\chi(P_n)$ ,  $\chi(K_n + K_m)$ ,  $\chi(\overline{C_n})$ .

**Exercise 5.2.3** A graph  $G$  is said to be **bipartite** if  $\chi(G) \leq 2$ . Prove that a graph  $G$  is bipartite if and only if  $G$  does not contain a cycle of odd length.

Recall that  $\omega(G)$  and  $\alpha(G)$  denote the number of vertices in the largest clique and independent set of  $G$ , respectively. Together with the maximal degree  $\Delta(G)$  we can derive some upper/lower bounds for  $\chi(G)$ .

**Exercise 5.2.4** Prove that  $\omega(G) \leq \chi(G) \leq \Delta(G) + 1$ .

**Exercise 5.2.5** Prove that  $\frac{|V(G)|}{\alpha(G)} \leq \chi(G)$ .

However, in general the chromatic number can be arbitrarily far away from these bounds.

**Exercise 5.2.6** Find a graph  $G$  with

- |  |   |
|--|---|
| (a) $\Delta(G) = 100$ and $\chi(G) = 2$ ;              | (c) $\omega(G) = 2$ and $\chi(G) = 3$ ; |
| (b) $\frac{V(G)}{\alpha(G)} = 2$ and $\chi(G) = 100$ ; | (d) $\omega(G) = 2$ and $\chi(G) = 4$ . |

While for arbitrary graphs the chromatic number can be hard to determine, for some special graphs, such as planar graphs, some progress can be made. We will devote the rest of the section discussing ideas surrounding the celebrated 4-colour theorem. We start with a weaker result, known as 6-colour theorem, i.e. we will show that 6 colours are enough to colour a planar graph.

**Theorem 5.2.7 (6-colour theorem)** *For any planar graph  $G$ ,  $\chi(G) \leq 6$ .*

**Proof.** We will show by induction on the number  $n$  of vertices of  $G$ . If  $n \leq 6$ , then  $G$  can be 6-coloured. Let  $n > 6$  vertices and that any planar graph with less than  $n$  vertices can be 6-coloured. Let  $G$  be any planar graph on  $n$  vertices,  $m$  edges, and whose planar embedding has  $f$  faces.

We can assume that  $G$  is connected and triangulated (each face has degree 3): if not we can add edges until we obtain a connected and triangulated planar graph. As each face has degree 3, and each edge is a border of two faces, we have

$$3F = 2E.$$

Plugging  $F = \frac{2}{3}E$  into Euler's formula we obtain  $V - E + F = V - \frac{1}{3}E = 2$ . Hence,  $\frac{1}{3}E = V - 2$ , and so  $E = 3V - 6$ . By the handshake lemma,  $\sum_{v \in V(G)} d(v) = 2E = 6V - 12$ , and hence the **average degree** of vertices of  $G$  is

$$\frac{\sum_{v \in V(G)} d(v)}{|V(G)|} < 6.$$

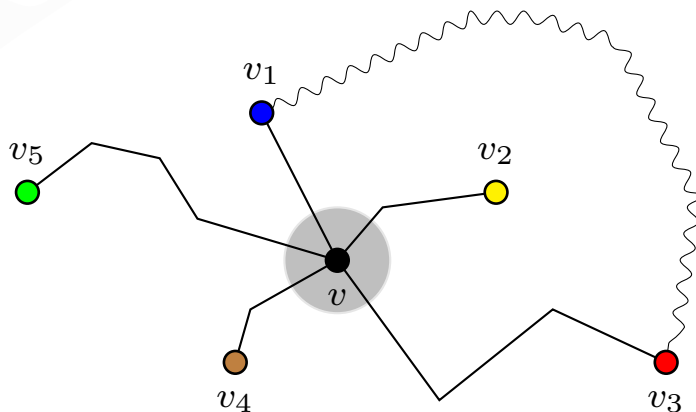
This means that  $G$  has a vertex  $v$  whose degree is at most 5. By induction  $G - v$  can be 6-coloured. Since  $v$  has at most 5 neighbours, one of the 6 colours will not appear among the neighbours of  $v$  in  $G$ . Hence the 6-colouring of  $G - v$  can be extended to a 6-colouring of  $G$  by colouring  $v$  with a colour unused in the neighbourhood of  $v$ . ■

This can be extended to 5-colour theorem, using an argument, known as Kempe's chain.

**Theorem 5.2.8 (5-colour theorem)** *For any planar graph  $G$ ,  $\chi(G) \leq 5$ .*

**Proof.** We proceed by induction on the number of vertices as before. Clearly, any graph on at most 5 vertices can be 5-coloured. Suppose now  $G$  is a planar graph on more than 5 vertices,  $v \in V(G)$  be a vertex of degree at most 5 (which exists by the same argument as before) and suppose, by induction, that  $G - v$  has a 5-colouring  $c$ . Note that if  $v$  has degree at most 4, then as before, we can choose a colour not used in the neighbourhood of  $v$  and use this colour for  $v$  obtaining the colouring of  $G$ . Also note that if  $v$  has a degree 5, but the neighbours of  $v$  are coloured with at most 4 colours, i.e. some colour is repeated among the neighbours of  $v$ , then there is a free colour we can use for colouring  $v$  and we are done as well.

So, the only remaining case is when  $v$  has 5 neighbours, let's say  $N(v) = \{v_1, v_2, v_3, v_4, v_5\}$ , and they are all coloured with 5 distinct colours. Without loss of generality we can assume that  $v_1, v_2, \dots, v_5$  appear in this cyclic order (formally, one can determine the cyclic order of the edges by drawing some small disc around vertex  $v$ , which is small enough that all polygonal edges intersect the disc in straight lines, then the cyclic order of the segments in the intersection with the disc determines the cyclic order of the vertices). Also, without loss of generality, we denote the colours of  $v_1, v_2, \dots, v_5$  by  $c_1$ =blue,  $c_2$ =yellow,  $c_3$ =red,  $c_4$ =brown and  $c_5$ =green, respectively (see picture below).





Note that if  $v_1$  has no red neighbour, then we can re-colour  $v_1$  by red colour, and now, since the neighbourhood of  $v$  has repeated colour red,  $v$  can be coloured blue, and so we are done. Consider another case, when  $v_1$  is adjacent to some red vertices, say  $w_1, w_2, \dots, w_k$  (none of which is  $v_3$ ), but none of  $w_1, w_2, \dots, w_k$  are not adjacent to any further blue vertices (except for  $v_1$ ). Then we can again re-colour  $v_1$  red, re-colour  $w_1, w_2, \dots, w_k$  blue, and we again saved the blue colour for  $v$ , so we are done. In general, consider the set of all vertices  $S$  reachable from vertex  $v_1$  by an alternating red-blue path. If  $S$  does not contain  $v_3$ , then we can re-colour all red vertices in  $S$  by blue colour, all the blue vertices in  $S$  by red colour, and hence, since  $v_1$  and  $v_3$  are both red, we can use blue colour for  $v$ , and so we are done.

However, our argument does not work if  $S$  contains vertex  $v_3$ , i.e. if there is a red-blue path between  $v_1$  and  $v_3$ . Indeed, switching red and blue colours in  $S$  in this case would make  $v_1$  red, and  $v_3$  blue and so the neighbourhood of  $v$  would still have all five colours present, so no free colour available for  $v$ . However, in this case, we can consider the set  $T$  of all vertices reachable from  $v_2$  by a yellow-green path. Since the blue-red path joins  $v_1$  and  $v_3$ , note that there is no yellow-green path joining  $v_2$  to  $v_5$  (otherwise, some edges of yellow-green path, would need to cross some edges of red-blue path, contradicting planarity of the graph). Hence we can switch yellow and green colours in  $T$ , obtaining a colouring with  $v_2$  and  $v_5$  both green, and so we can use colour yellow for  $v$ . This finishes the proof. ■

We note that the red-blue or yellow-green paths used in the previous argument are called Kempe chains. Alfred Kempe, in his original proof, used the idea of switching colours in the chains to prove the 4-colour theorem. However, the argument contained a mistake, which couldn't be corrected. The correct argument was supplied almost a century later, 1976 by Kenneth Appel and Wolfgang Haken. The proof is outside the scope of this course and we only state the result below.

**Theorem 5.2.9 (4-colour theorem)** *For any planar graph  $G$ ,  $\chi(G) \leq 4$ .*

### 5.3 SAT and NP-completeness of Colouring

The question of finding a colouring of graph with minimal number of colours has a number of practical applications. One of the main application is for various scheduling problems, such as designing a university timetable. Here, the vertices could be the classes offered by university, and there is an edge between two vertices if the classes cannot be scheduled at the same time for some reason. Some obvious reasons could be that the two classes have to be taught by the same teacher (hence the teacher cannot be in two places at once), or some student has to attend both classes (hence a clash for a student). By the definition of vertex colouring, the colour classes are clash-free, and so each color class could be scheduled into one separate time-slot. Finding the smallest number of colors is key to ensure that the classes can be delivered during the smallest number of time-slots, and do not extend to, for instance, late evenings or weekends.

One can then ask about the time complexity of the colouring problem. Intuitively, it looks harder than the problem of finding the maximal independent set, that we discussed in the previous chapter. In this section we will try to make the notion of hardness of the problem more rigorous and formal.

To address the problem of computational complexity of algorithmic problems more formally, the theoretical computer scientists consider the notion of a **Turing machine**. The Turing machine is a theoretical model of a simple computer that reads and writes symbols one at a time on an endless tape by strictly following certain rules and used in thought experiments to

examine the abilities and limits of computers. The notion of the Turing machine was introduced by a british mathematician and computer scientists Alan Turing (1912-1954), who is widely considered to be the father of theoretical computer science and artificial intelligence. Apart from his great contribution to science, Alan Turing played a very important role in the 2nd world war, by breaking Enigma codes, which meant deciphering the encrypted German messages and hence providing a crucial intelligence that helped British and their allies to win several decisive battles which led to the eventual defeat of Germany.

We will say that the problem is in the complexity class **NP** (non-deterministic polynomial), if it can be solved by a non-deterministic Turing machine in polynomial time. The algorithm consist of two phases, the first consist of a guess about the solution, which is generated non-deterministically, while the second phase consist of deterministic algorithm that verifies if the guess is a solution to the problem. For instance, the problem of determining whether the graph can be  $k$ -coloured is in  $NP$ , since given a guess of a  $k$ -colouring of the graph, one can check in polynomial time whether or not any two adjacent vertices do indeed have different colours. Also the problem of determining whether the graph contains an independent set of size  $k$  is also in  $NP$  as given any set of vertices, one can check in polynomial time whether or not any two vertices in the set are non-adjacent.

We will say that a problem is in the complexity class **P** (polynomial), if it can be solved in polynomial time by a deterministic Turing machine. In other words, the problem is in  $P$ , if there is a deterministic (without any guessing) algorithm which runs in polynomial time and solves the problem. Examples such as list-sorting algorithms, or Kruskal's algorithm are in  $P$ , as we provided polynomial time algorithms to solve them. Clearly, any problem in  $P$  is also in  $NP$ , which we can denote as  $P \subseteq NP$ . The oposite direction is much more interesting, can any problem in  $NP$ , be solved deterministically in polynomial time? This is known as **P=NP** problem and it is the most famous and important open problem in computer science.

**NP-complete** problems are the hardest problems in  $NP$ , in the sense that if  $Q'$  is any problem in  $NP$  and  $Q$  is an NP-complete problem, then every instance of  $Q'$  is polynomially reducible to an instance of  $Q$ . The surprising thing is that there is an NP-complete problem at all, since it is not clear that a single problem can hold a key for polynomial solvability for all the NP-complete problems. But indeed, there is such a problem, known as Satisfiability problem (SAT), as was the first NP-complete problem discovered by Stephen Cook in 1971. Once the first one was discovered, hundreds of problems have been shown to be NP-complete, thus forming a class of NP-complete problems, which include many important computational questions about graphs, games, algebraic structures, formal logic and more... The problems that are at least as hard as NP-complete problems are known as **NP-hard**, i.e.  $Q''$  is NP-hard, if there is an NP-complete problem  $Q$  such that every instance of  $Q$  can be reduced to an instance of  $Q''$  in polynomial time.

**Exercise 5.3.1** Draw a Venn diagram to illustrate the relationships between  $P$ ,  $NP$ ,  $NP$ -hard, and  $NP$ -complete in each of the following two cases:

- (a) If  $P \neq NP$ ;
- (b) If  $P = NP$ .

**Definition 5.3.2 (Satisfiability problem (SAT))** Let  $x_1, x_2, \dots, x_n$  be a list of (Boolean) variables.

- A **literal** is either a variable  $x_i$ , or its negation, denoted by  $\overline{x_i}$ .

- A **clause** is either a single literal or a disjunction (logical operation OR) of several literals, e.g.  $x_1 \vee \overline{x_2} \vee x_3$ .
- A formula is in **conjunctive normal form (CNF)** if it is a conjunction (logical operation AND) of clauses, e.g.  $(x_1 \vee \overline{x_2} \vee x_3) \wedge (\overline{x_1} \vee \overline{x_2}) \wedge (x_2 \vee x_3)$ .
- We say that a CNF formula is **satisfiable**, if there is an assignment of TRUE/FALSE values to variables  $x_1, x_2, \dots, x_n$ , such that the overall computation of the formula gives a TRUE value. For instance, assigning values  $x_1 = \text{TRUE}$ ,  $x_2 = \text{FALSE}$ ,  $x_3 = \text{TRUE}$  satisfies the formula provided above.
- The **satisfiability problem (SAT)** asks whether a given CNF formula is satisfiable.

**Exercise 5.3.3** Which of the following CNF formulas are satisfiable?

- (a)  $x_1 \wedge (\overline{x_1} \vee x_2) \wedge \overline{x_2}$ ;
- (b)  $(x_1 \vee \overline{x_2}) \wedge (x_2 \vee \overline{x_3}) \wedge (x_3 \vee \overline{x_1}) \wedge (\overline{x_1} \vee \overline{x_2} \wedge \overline{x_3})$ .
- (c)  $(x_1 \vee \overline{x_2}) \wedge (x_2 \vee \overline{x_3}) \wedge (x_3 \vee \overline{x_1}) \wedge (x_1 \vee x_2 \vee x_3) \wedge (\overline{x_1} \vee \overline{x_2} \wedge \overline{x_3})$ .
- (d)  $(x_1 \vee x_2) \wedge (x_1 \wedge \overline{x_3}) \wedge (\overline{x_1} \vee \overline{x_2}) \wedge (\overline{x_1} \vee x_2 \vee x_3) \wedge (x_2 \wedge \overline{x_3})$ ;

**Exercise 5.3.4** Show that SAT belongs to the complexity class NP.

As we mentioned before, SAT was the first problem proved to be NP-complete by Stephen Cook in 1971. We state the celebrated theorem below.

**Theorem 5.3.5** SAT is NP-complete.

**Proof.** The proof is outside the scope of the course. For those of you interested, a proof can be found in Herbert S. Wilf's book 'Algorithms and complexity'. ■

Building on SAT, showing the NP-completeness is now much easier, as it is now enough to simply compare a given problem in NP with SAT: if the problem is as hard as SAT, it is NP-complete. We will illustrate this with an example of a problem called **3-satisfiability**, or **3SAT**. The 3SAT is the restriction of SAT, where we ask whether a given CNF formula satisfiable only for formulas whose clauses have at most 3 literals each. Interestingly, this problem is as hard as SAT itself as we will show now.

**Theorem 5.3.6** 3SAT is NP-complete.

**Proof.** We will show how an instance of SAT can be reduced in polynomial time to an instance of 3SAT. Let  $f = C_1 \wedge C_2 \wedge \dots \wedge C_m$  be an instance of SAT, with  $C_i$ 's representing clauses of CNF formula  $f$ . If all clauses have 3 literals, then we are done. So assume that for some  $i$  we have a clause  $C_i = (x_1 \vee x_2 \vee \dots \vee x_k)$  with at least 4 literals, i.e.  $k \geq 4$ . We now introduce a new literal  $z$  and replace  $C_i$  by a disjunction of two clauses

$$C_i^* = (x_1 \vee x_2 \vee \dots \vee x_{k-2} \vee z) \wedge (x_{k-1} \vee x_k \vee \overline{z}).$$

It is not hard to see that  $f$  is satisfiable if and only if  $f^* = C_1 \wedge \dots \wedge C_{i-1} \wedge C_i^* \wedge C_{i+1} \wedge \dots \wedge C_m$  is satisfiable. Indeed, if  $f$  is satisfiable, then there is an assignment of true/false values for which

all clauses, including  $C_i$  is true. In this assignment one of the literals  $x_1, x_2, \dots, x_k$  must be true, and depending whether the true literal is among  $\{x_1, \dots, x_{k-2}\}$  or among  $\{x_{k-1}, x_k\}$  the value of  $z$  can be set true or false to make  $C_i^*$  true, hence extending the satisfying assignment of  $f$  to the one for  $f^*$ , and showing that  $f^*$  is satisfiable. On the other hand, if  $f^*$  is satisfiable, then there is an assignment for which all clauses of  $f^*$  are true, and we can easily see that in this assignment to make  $C_i^*$  true, one of  $x_1, x_2, \dots, x_k$  must be true, hence giving an assignment for which  $f$  is true. So  $f$  is satisfiable if and only if  $f^*$  is satisfiable.

Note that a clause  $C_i$  of length  $k$  was replaced by two clauses of length 3 and length  $k - 1$ . Clearly, if the original formula consisted of a total of  $n$  literals, repeating this operation less than  $n$  times we will eventually obtain a formula whose all clauses are of size at most 3. Hence, in polynomial time, we reduced an instance of SAT into 3SAT so that the original instance is satisfiable if and only if the 3SAT instance is satisfiable. This shows that 3SAT is NP-complete. ■

Note that the above reduction shows that if we had an efficient (polynomial time) algorithm to solve 3SAT, then we would also have an efficient (polynomial time) algorithm to solve SAT. So in this sense, 3SAT is as hard as SAT. Either by recalling that SAT is NP-complete, or by just observing that SAT is a generalization of 3SAT, we also get that SAT is as hard as 3SAT. Hence, from computational perspective, 3SAT is of equivalent hardness with SAT, formally both being NP-complete problems. In contrast, 2SAT, where each clause is only allowed to have at most 2, is known to be solvable in polynomial (even linear) time. Now we will show that graph colouring is also NP-complete.

**Graph colouring problem** given a graph  $G$  and an integer  $k$  asks whether or not graph  $G$  can be coloured with  $k$  colours. Similar to SAT or 3SAT, this problem asks only for yes-or-no answer, and such problems are formally called **decision problems**. Most problems in graph theory, can be made into decision problems such as asking: Can this graph be 5-coloured? Is there a set of 32 independent vertices? Is there a path of length  $\leq 10$  miles?

In comparison to decision problems, the questions can also be phrased as **optimization problems**: What is the smallest number of colours with which  $G$  can be coloured? What is the size of the largest independent set of vertices in  $G$ ? What is the length of the shortest path between two cities?

Clearly, any fast algorithm for optimization problem, can be used for decision problem. Conversely, if we find a fast algorithm for decision problem, with a bit of work, we can convert it to a fast algorithm for optimization problem. For instance, suppose we have a fast algorithm for graph colouring, and what we want is a solution to the optimization problem of finding the chromatic number. Suppose we are given a graph  $G$  on 100 vertices. We may ask: can  $G$  be 50-coloured? If so, then the chromatic number lies between 1 and 50 and we may proceed by asking if  $G$  can be 25-coloured. If not, we know the chromatic number lies between 26 and 50. Continuing this way, after  $O(\log(n))$  steps we have found a chromatic number for a graph on  $n$  vertices. The extra multiplicative factor of  $\log(n)$ , will not alter the polynomial versus non-polynomial running time distinction. Hence, a fast way for decision problem results in fast way for optimization problem.

When proving NP-completeness, we usually restrict to decision problems, because their hardness are often equivalent to the hardness of corresponding optimization problems, but the decision problems are easier to handle in proofs. Below we will prove that the decision problem of graph colouring is NP-complete, from which it would then immediately follow that the optimization problem of finding chromatic number is also NP-complete.

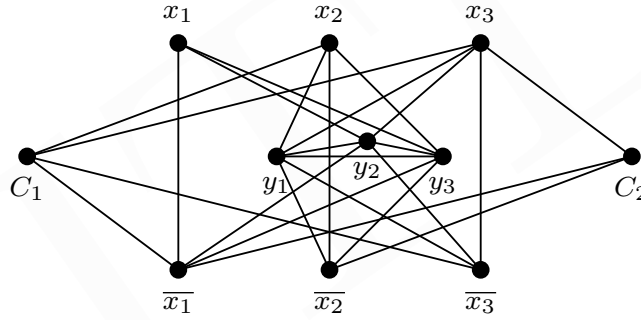
**Theorem 5.3.7** *Graph colouring is NP-complete.*

**Proof.** Given an instance of 3SAT with CNF formula involving  $n$  variables and  $k$  clauses (with at most three literals each), we will construct in polynomial time a graph  $G$  such that  $G$  is  $n + 1$ -colourable if and only if the given CNF formula is satisfiable. We will assume that  $n \geq 4$ , as other cases can be solved trivially.

Let  $G$  have  $3n + k$  vertices  $\{x_1, x_2, \dots, x_n\}$ ,  $\{\bar{x}_1, \bar{x}_2, \dots, \bar{x}_n\}$ ,  $\{y_1, y_2, \dots, y_n\}$  and  $\{C_1, C_2, \dots, C_k\}$ . The edges are as follows:

- $x_i$  is adjacent to  $\bar{x}_i$  for all  $i$ ;
- $y_i$  is adjacent to  $y_j$  for all  $i \neq j$ ;
- $y_i$  is adjacent to  $x_j$  and  $\bar{x}_j$  for all  $i \neq j$ ;
- $C_i$  is adjacent to  $x_i$  if the clause  $C_i$  does not contain literal  $x_i$ ;
- $C_i$  is adjacent to  $\bar{x}_i$  if the clause  $C_i$  does not contain literal  $\bar{x}_i$ ;

To illustrate our construction, below is a graph on 11 vertices corresponding to the CNF formula  $f = (x_1 \vee \bar{x}_2) \wedge (x_1 \vee x_2 \vee \bar{x}_3)$ :



We claim that the graph we just defined is  $n + 1$ -colourable if and only if the CNF formula  $f = C_1 \wedge C_2 \wedge \dots \wedge C_k$  is satisfiable. In other words, we are trying to show that we can use a graph colouring algorithm to find a satisfying assignment for an instance of the 3SAT problem!

Note first that to colour the graph  $G$  we need at least  $n$  colours, as vertices  $\{y_1, y_2, \dots, y_n\}$  form a clique. Without loss of generality we can assume that  $y_i$  is coloured with colour  $i$ . Secondly, that  $x_i$  and  $\bar{x}_i$  are both joined to all vertices  $y_j$  with  $j \neq i$ . It follows that in any colouring  $x_i$  and  $\bar{x}_i$  cannot use any of the colours  $\{1, 2, \dots, i - 1, i + 1, \dots, n\}$ . Hence, in any  $n + 1$ -colouring,  $x_i$  and  $\bar{x}_i$  can only use the two colours:  $i$  and  $n + 1$ , and since they are adjacent, exactly one of them must be coloured with colour  $i$ , and the other must have colour  $n + 1$ .

As we assume that  $n \geq 4$  and each clause  $C_k$  is non-adjacent to at most 3 literals from  $\{x_1, x_2, \dots, x_n, \bar{x}_1, \bar{x}_2, \dots, \bar{x}_n\}$ , we must have that  $C_k$  is adjacent to the pair  $x_p, \bar{x}_p$ , for some  $p$ , and hence it must be adjacent to colour  $n + 1$  (in any  $n + 1$ -colouring of  $G$ , if such exists). Thus, only colours  $\{1, 2, \dots, n\}$  are possible for  $C_k$ . Note further, that it is possible to choose a colour for  $C_k$ , if and only if,  $C_k$  is not adjacent to all the possible colours  $\{1, 2, \dots, n\}$ , which happens if and only if one of the vertices that  $C_k$  is not joined to (corresponding to a literal in  $C_k$ ) is coloured with a colour from  $\{1, 2, \dots, n\}$ . In other words, it is possible to assign a colour for  $C_k$  if and only if not all literals belonging to  $C_k$  are coloured with the colour  $n + 1$ .

Given an  $n + 1$ -colouring of  $G$ , we assign the value TRUE for a literal  $x_i$  or  $\bar{x}_i$  if the corresponding vertex is coloured with one of the colours  $\{1, 2, \dots, n\}$ , and we assign the value FALSE if the the vertex corresponding to the literal is coloured with the colour  $n + 1$ . From the discussion above it is clear that any valid  $n + 1$ -colouring will correspond to an assignment satisfying the formula (as the set of vertices corresponding to literals of any one given clause will not all be coloured with colour  $n + 1$ ). On the other hand, it is not hard to see that a satisfying assignment easily leads to an  $n + 1$ -colouring of  $G$ . Hence, we conclude that the graph is  $n + 1$ -colourable if and only if the corresponding CNF formula is satisfiable.

Noting that given a CNF formula the graph can be constructed in polynomial time, finishes the proof of NP-completeness of graph colouring. ■

By means of ingenious transformations, like the one we have just seen in the proof above, hundreds of problems of theoretical and practical importance have been proven to be NP-complete. We list a couple of them here:

- **Independent set** problem given a graph  $G$  and an integer  $k$  asks whether a graph  $G$  has an independent set of size  $k$ .
- **Vertex cover** problem given a graph  $G$  and an integer  $k$  asks whether there exists a set of  $k$  vertices which are incident to all of the edges of  $G$ .
- **Hamilton path** problem given a graph  $G$  asks whether there exists a path which visits every vertex of  $G$  exactly once.
- **Subgraph isomorphism** problem given two graphs  $G$  and  $H$  asks whether  $G$  contains a subgraph that is isomorphic to  $H$ .
- **Partition** problem given a finite multiset of positive integers whose sum is  $S$  asks whether there exists a subset whose sum is  $S/2$ .
- **Travelling Salesman** problem given a list of cities and the distances between each pair of cities, find the shortest possible route that visits each city exactly once and returns to the original city.
- **Knapsack** problem given a set of items, each with a weight and a value, asks to determine the number of each item to include in a collection so that a total weight is less than or equal to a given limit, and the total value is as large as possible.

**Remark 5.3.8 (Fastest known exact algorithms for graph colouring)** *We note that a naive brute-force search for  $k$ -colouring of a graph  $G$  on  $n$  vertices, trying all possible colourings and checking every pair of vertices whether the colouring is valid has running time  $O(n^2 k^n)$ . Using dynamic programming, one can do much better and decide the  $k$ -colorability in time  $O(2.4423^n)$ . The best known state-of-art exact algorithm with running time  $O(2^n n)$  has been published in 2009 by A. Björklund, T. Husfeldt and M. Koivisto in the article “Set partitioning via inclusion-exclusion”, with the algorithm relying on an advanced use of the method of inclusion-exclusion.*

## 5.4 Interval graphs and chromatic polynomial

While the problem of graph colouring is NP-complete in general, it can become polynomial time solvable when restricted to certain types of graphs. Here we will provide one such class useful for an application to taxi scheduling problem.

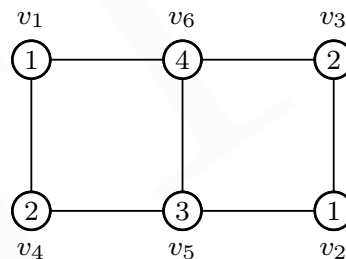
**Definition 5.4.1** Let  $\mathcal{I} = \{I_1, I_2, \dots, I_n\}$  be a set of intervals of a real line defined by  $I_i = \{x \in \mathbb{R} : a_i \leq x < b_i\}$  where  $a_i$  and  $b_i$  define the start and the end values of the interval  $I_i$ . The **interval graph** of  $\mathcal{I}$  is a graph  $G$  with  $V(G) = \{v_1, v_2, \dots, v_n\}$  and  $v_i v_j \in E(G)$  if and only if  $I_i \cap I_j \neq \emptyset$ . In other words, the vertices of the graph represent the intervals, and the two vertices are adjacent if and only if the corresponding intervals overlap.

**Exercise 5.4.2** Consider  $n$  bookings for taxi made, each booking a taxi for a specific time-slot represented by an interval. Explain why the smallest number of taxi cars needed to fulfil all the orders is precisely the chromatic number of the interval graph.

**Definition 5.4.3** Given a graph  $G$  and an ordering of vertices  $v_1, v_2, \dots, v_n$  a **greedy algorithm** for colouring proceeds as follows:

- Colour  $v_1$  with colour 1, i.e. let  $c(v_1) = 1$ .
- Having coloured  $v_1, v_2, \dots, v_i$ , colour  $v_{i+1}$  with the smallest available colour not used among the neighbours of  $v_{i+1}$  which have already been coloured.

**Example 5.4.4** Let's apply the greedy colouring to the following graph with vertex ordering  $v_1, v_2, \dots, v_6$  as in the picture below. The greedy algorithm first assigns colour 1 for vertices  $v_1$  and  $v_2$ . Since  $v_3$  and  $v_4$  are adjacent to colour 1, the next smallest available colour for these vertices are is colour 2. Now,  $v_5$  is adjacent to colours 1 and 2, so must be colour 3. Finally, since  $v_6$  neighbours already have colours 1, 2, 3,  $v_6$  is assigned colour 4.



We note that the number of colours obtained by greedy algorithm is not necessarily the smallest number of colours, but it depends on the order in which the vertices have been coloured.

**Exercise 5.4.5** Suggest a different ordering of the vertices of the same 6-vertex graph in the previous example so that the greedy algorithm produces a colouring with

- 2 colours;
- 3 colours.

**Exercise 5.4.6** Let  $n \in \mathbb{N}$ . Find an example of a graph  $G$  and a suitable vertex ordering of  $G$  such that  $\chi(G) = 2$  but the greedy algorithm uses  $n$  colours for colouring  $G$ .

Surprisingly, the greedy algorithm with a suitable ordering gives optimal answer for interval graphs, hence solving taxi scheduling problem efficiently in polynomial time.

**Theorem 5.4.7** Let  $G$  be an interval graph, whose vertices  $v_1, v_2, \dots, v_n$  such that the left-endpoints of the corresponding intervals are in increasing order, i.e.  $a_1 \leq a_2 \leq \dots \leq a_n$  with  $a_i$ 's as in the Definition 5.4.1. Then the Greedy algorithm with this ordering provides an optimal colouring of the graph  $G$  with  $\chi(G)$  colours. Moreover,  $\chi(G) = \omega(G)$ .

**Proof.** The greedy algorithm uses colour 1 for  $v_1$ , and for  $i > 1$  it colours  $v_i$  with the smallest available colour not used among the neighbours of  $v_i$  that have been coloured (i.e. neighbours of  $v_i$  among the vertices  $v_1, v_2, \dots, v_{i-1}$ ). Note that if  $v_j$  is a neighbour of  $v_i$  with  $j < i$ , then the interval of  $v_j$  must intersect the interval of  $v_i$ , and since  $a_j \leq a_i$ , the interval of  $v_j$  must contain the point  $a_i$ . Since this is true for every coloured neighbour of  $v_i$ , we note that the neighbours of  $v_i$  all contain the point  $a_i$  and hence they form a clique. Since  $v_i$  forms a clique with its coloured neighbours, the number of coloured neighbours must be at most  $\omega(G) - 1$ , and hence the smallest available colour for  $v_i$  is at most  $\omega(G)$ . Since this is true for every vertex  $v_i$ , we obtained a colouring of  $G$  with at most  $\omega(G)$  colours. Since every colouring of  $G$  has to use at least  $\omega(G)$  colours, we conclude that the colouring is optimal with  $\chi(G) = \omega(G)$ . ■

**Exercise 5.4.8** Show that for any graph  $G$ , there always exists an ordering of vertices of  $G$  such that the greedy algorithm produces an optimal solution. Use this to suggest a new algorithm for graph colouring and discuss its complexity.

Given a graph  $G$ , one may ask, how many ways are there to colour  $G$  with  $x$  colours. We will denote this number by  $P_G(x)$  and explore its properties in the exercises below.

**Exercise 5.4.9** Show that

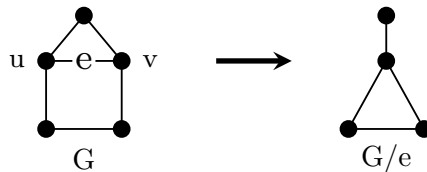
$$(a) P_{K_n}(x) = x(x-1) \dots (x-n+1);$$

$$(b) P_{O_n}(x) = x^n;$$

$$(c) P_{P_n}(x) = x(x-1)^{n-1};$$

$$(d) P_{K_{1,n}}(x) = x(x-1)^n.$$

**Definition 5.4.10** Given a graph  $G$  and  $e = uv \in E(G)$ , a **contraction** of  $e$ , written as  $G/e$ , is a graph obtained from  $G$  by replacing the vertices  $u$  and  $v$  with one new vertex adjacent to all neighbours of  $u$  and  $v$ .



**Exercise 5.4.11 (Deletion-contraction relation)** Let  $G$  be a graph and  $e \in E(G)$ . Prove that

$$P_G = P_{G-e} - P_{G/e}.$$

**Exercise 5.4.12** Prove that for any graph  $G$ ,  $P_G(x)$  is a polynomial in  $x$ .



Hence  $P_G(x)$  is called the **chromatic polynomial** of  $G$ .

**Exercise 5.4.13** Find the chromatic polynomial of the cycle  $C_n$ .

**Exercise 5.4.14** Let  $G$  be a graph on  $n$  vertices and  $m$  edges. Show that the leading terms of  $P_G(x)$  are  $t^n - mt^{n-1} + \dots$ .

**Exercise 5.4.15** Find all the graphs  $G$  whose polynomial is  $P_G(x) = x(x-1)^{n-1}$ .

**Exercise 5.4.16** Suppose  $G$  and  $H$  are two graphs with disjoint set of vertices, i.e.  $V(G) \cap V(H) = \emptyset$ . Show that:

(a)  $P_{G+H} = P_G \times P_H$

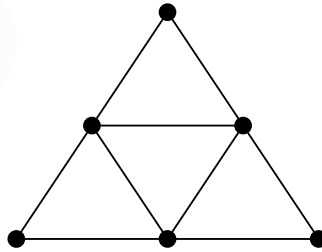
(b) Let  $v \in V(G)$  and  $w \in V(H)$  be two vertices, and let  $K$  be a graph obtained by gluing vertex  $v$  of  $G$  to vertex  $w$  of  $H$ . I.e. let  $K$  be a graph obtained from  $G + H$  by adding an edge  $vw$  and then contracting this edge. Show that

$$P_K = \frac{P_G \times P_H}{x}.$$

(c) Let  $S \subseteq V(G)$ ,  $S' \subseteq V(H)$  be two subsets inducing cliques of sizes  $s = |S| = |S'|$ . Show that if  $K$  is a graph obtained by gluing  $G$  and  $H$  along the cliques  $S$  and  $S'$  (for instance, a "house graph" pictured in Definition 5.4.10 can be viewed as a graph obtained by gluing two simpler graphs  $C_3$  and  $C_4$  along a size-2 clique), then

$$P_K = \frac{P_G \times P_H}{x(x-1) \times \dots \times (x-s+1)}.$$

**Exercise 5.4.17** Find the chromatic polynomial of the following graph:



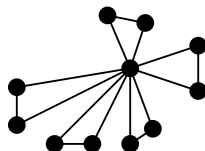
## Chapter 6

# Introduction to spectral methods

Paul Erdős believed in and liked to talk about The Book, in which God maintains the perfect proofs for mathematical theorems. There were a number of proofs that Erdős discovered himself that contained brilliant ideas, clever insights and wonderful observations. We will now present one such wonderful proof, that Erdős believed belongs to The Book. In this proof, graph theory meets linear algebra and even number theory. The joint insights provide an elegant resolution to simple looking but, in fact, very hard problem.

**Theorem 6.0.1** *Suppose in a group of people any two people have precisely one common friend. Then there is always a person (the “politician”) who is everybody’s friend.*

**Proof.** Before we proceed to the proof, we remark that a graph where any two people have precisely one common friend exists. Such a graph is a windmill graph:



Also, it is not hard to see that whenever there is a single person who knows everyone, a windmill graph is the only possibility. Indeed, since any person who is not the politician must have exactly one friend with the politician, the non-politicians must pair-up and hence form the wind-mill graph. Thus, proving that there always must be a politician, would immediately imply that the windmill graphs are the only graphs for which any two vertices have exactly one common neighbour.

We will prove the statement by contradiction. Suppose that  $G$  is a counterexample graph, satisfying the precisely-one common neighbour condition, but which has no vertex adjacent to

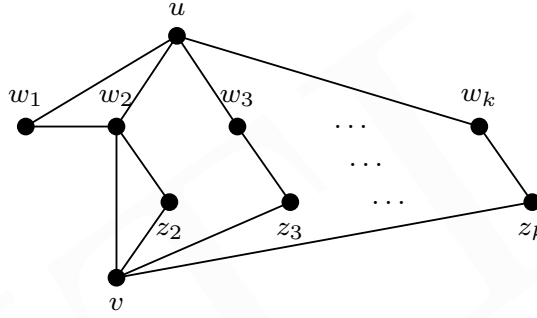


Figure 6.1: Friendship problem from “Proofs from The Book” by M. Aigner and G. Ziegler

all other vertices. To derive a contradiction we will proceed in three steps. The first part is graph theory, the second is linear algebra, the third is number theory.

- (1) We claim that  $G$  is a regular graph, i.e.  $d(u) = d(v)$  for all vertices  $u, v \in V(G)$ .

We first prove that any two non-adjacent vertices  $u, v \in V(G)$  have equal degree. Suppose  $d(u) = k$  and  $w_1, w_2, \dots, w_k$  are the neighbours of  $u$ . First of all,  $v$  must be adjacent to exactly one neighbour of  $u$ , say  $w_2$ . Secondly,  $w_2$  has to have exactly one common neighbour with  $u$ , say  $w_1$ . Finally,  $v$  and  $w_1$  already has a common neighbour  $w_2$ , but  $v$  must also have a common neighbour with each  $w_i$  for  $i = 2, 3, \dots, k$ . We remind that no vertex  $z \neq u$  can be adjacent to two neighbours of  $u$ , otherwise  $z$  would share two common neighbours with  $u$ . From this it follows that the common neighbours of  $v$  and  $w_i$  for  $i = 2, 3, \dots, k$ , which we can denote  $z_2, z_3, \dots, z_k$  must all be pairwise distinct vertices not belonging to the neighbourhood of  $u$ . Since  $v$  is adjacent to  $w_2, z_2, z_3, \dots, z_k$  it follows that  $d(v) \geq d(u)$ . By symmetry, we also deduce  $d(u) \geq d(v)$  and hence  $d(u) = d(v)$ .



To complete the proof of (1), note that any vertex different from  $w_2$  is non-adjacent to either  $u$  or  $v$ , hence has degree  $k$ . Since  $w_2$  also has a non-neighbour, it has degree  $k$  as well, completing the proof that  $G$  is  $k$ -regular.

Note that each neighbour of  $u$  is adjacent to precisely one other neighbour of  $u$ , and has exactly  $k - 2$  neighbours outside  $\{u\} \cup N(u)$ . As any vertex outside  $\{u\} \cup N(u)$  is adjacent to exactly one vertex of  $N(u)$ , we obtain that the number of vertices in  $G$  is

$$n = 1 + k + k(k - 2) = k^2 - k + 1.$$

When  $k = 1$  or  $k = 2$ , we obtain  $n = 1$  or  $n = 3$ , hence  $G = K_1$  or  $G = K_3$ , respectively, both of which are trivial windmill graphs. Hence, we will assume that  $k > 2$ .

- (2) The next step is a beautiful application of some standard results from linear algebra. We first build an  $n \times n$  matrix  $A$  of the graph  $G$  where  $a_{i,j} = 1$  if the  $i$ 'th vertex of  $G$  is adjacent to  $j$ 'th vertex of  $G$  (in some fixed ordering of vertices of  $G$ ), and we place  $a_{i,j} = 0$  otherwise. In particular, the diagonal entries  $a_{i,i} = 0$  for all  $i = 1, 2, \dots, n$ .

Can we find eigenvalues of  $A$ ? Well, we can see that  $k$  is an eigenvalue, with eigenvector  $(1, 1, \dots, 1)$ , because the graph is  $k$ -regular, hence there are exactly  $k$  entries '1' in each column. To find other eigenvalues, consider the matrix  $A^2$ . Note that, since every vertex has degree  $k$ , every diagonal entry of  $A^2$  is  $k$ , and since any two vertices have exactly one common neighbour, every non-diagonal entry is 1, hence:

$$A^2 = \begin{pmatrix} k & 1 & \cdots & 1 \\ 1 & k & \cdots & 1 \\ \vdots & \vdots & \ddots & \vdots \\ 1 & 1 & \cdots & k \end{pmatrix} = (k-1)I + J,$$

where  $I$  is the identity matrix and  $J$  is the matrix of all 1's. It's clear that  $(1, 1, \dots, 1)$  is an eigenvector of  $A^2$  with eigenvalue  $k + n - 1 = k^2$ . Also, for any vector  $v$  orthogonal to  $(1, 1, \dots, 1)$ , we have  $A^2v = (k-1)Iv + Jv = (k-1)v$ , and hence any such vector is an eigenvector of  $A^2$  with eigenvalue  $k-1$ . It follows that  $A^2$  has eigenvalue  $k^2$  (of multiplicity 1), and  $k-1$  (of multiplicity  $n-1$ ).

Since  $A$  is symmetric,  $A$  is diagonalizable, and so its eigenvalues are  $k$  (of multiplicity 1) and  $\pm\sqrt{k-1}$ . Suppose  $r$  of the eigenvalues are equal to  $\sqrt{k-1}$  and  $s$  of them are equal to  $-\sqrt{k-1}$ . Since the sum of the eigenvalues of  $A$  is the trace of  $A$  (which is 0) we obtain:

$$k + r\sqrt{k-1} + s(-\sqrt{k-1}) = 0.$$

- (3) We will now use number theory to show that there are no integers  $r, k, s \in \mathbb{N}$  with  $k > 2$ , satisfying the above equation. As  $r \neq s$ , we can rewrite the equation as

$$\sqrt{k-1} = \frac{k}{s-r}.$$

Now, if the square root  $\sqrt{m}$  of a positive integer  $m$  is a rational number, it must be an integer (prove this!). Hence, writing  $\sqrt{k-1} = h$  for some integer  $h$  we obtain:

$$h(s-r) = h^2 + 1$$

Now, since both  $h^2$  and  $h^2 + 1$  is divisible by  $h$ , we obtain that 1 is divisible by  $h$  and so  $h = 1$ . Thus  $k = 2$ , which we already excluded. So we arrived at contradiction, and hence the proof is complete.

■

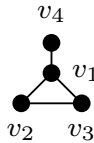
As shown in the brilliant proof by Erdős, graphs can be naturally represented by matrices and the machinery learned in linear algebra can then be used. We will provide couple of definitions to talk about this more formally, and will finish the course with one more example of use of matrices from the movie "Good Will Hunting".

## 6.1 Formal definitions and terminology for matrices of graphs

Let  $G$  be a graph with vertex set  $V(G) = \{v_1, v_2, \dots, v_n\}$ .

- The **adjacency matrix** of  $G$  is the  $n \times n$  matrix  $A$  with

$$A_{ij} = \begin{cases} 1 & \text{if } v_i v_j \in E(G) \\ 0 & \text{otherwise} \end{cases}$$

For example, for the graph  we have  $A = \begin{pmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix}$

Note that  $(A^2)_{ij} = \sum_{k=1}^n A_{ik}A_{kj}$ . Thus it is not hard to see that

$$(A^2)_{ij} = \text{number of walks of length 2 from } v_i \text{ to } v_j.$$

**Exercise 6.1.1** Show that for any integer  $k$ , we have

$$(A^k)_{ij} = \text{number of walks of length } k \text{ from } v_i \text{ to } v_j.$$

**Exercise 6.1.2** Show that

- (a) The trace  $\text{tr}(A^2)$  is equal to twice the number of edges in  $G$ .
- (b) The trace  $\text{tr}(A^3)$  is equal to six times the number of triangles in  $G$ .

Note that  $A$  is always a real symmetric matrix, with 0's in the main diagonal, i.e.  $A_{ii} = 0$  for all  $i$ . Since  $A$  is real symmetric,  $A$  is diagonalizable: it has basis of eigenvectors  $e_1, e_2, \dots, e_n$  with eigenvalues  $\lambda_1 \geq \lambda_2 \geq \dots \geq \lambda_n$ . We also note that  $e_1, e_2, \dots, e_n$  can be taken to be orthonormal basis.

- The multiset of eigenvalues (given with their multiplicity)  $\{\lambda_1, \lambda_2, \dots, \lambda_n\}$  is called the **spectrum of  $G$** .
- The largest modulus of the eigenvalues of  $G$ , i.e.  $\max\{|\lambda_1|, |\lambda_2|, \dots, |\lambda_n|\}$  is called the **spectral radius of  $G$**  (or spectral radius of  $A$ , where  $A$  is an adjacency matrix of  $G$ ).

There are various relationships between spectrum and the properties of graphs explored by the branch of mathematics called spectral graph theory. We provide one such example here and couple more can be found in the last exercise sheet.

- The **distance** between two vertices  $x$  and  $y$ , denoted by  $d(x, y)$  is the length of the shortest path between them.
- The **diameter** of a connected graph  $G$ , denoted  $\text{diam}(G)$  is

$$\max_{x, y \in V(G)} d(x, y).$$

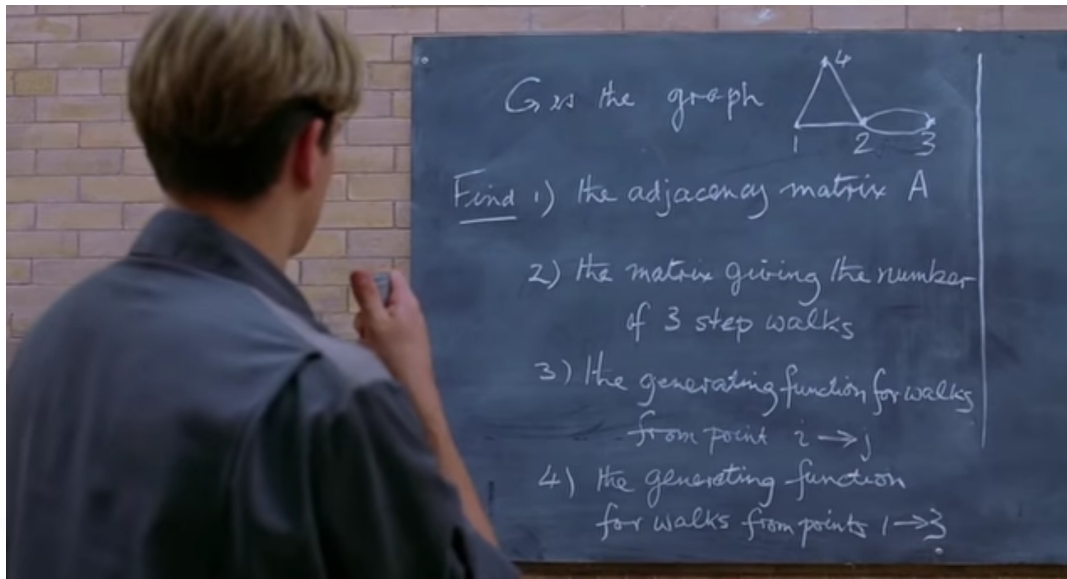
For example, a clique  $K_n$  is the only graph on  $n$  vertices whose diameter is 1. The cycles  $C_4$ ,  $C_5$  and non-trivial windmill graphs discussed previously have diameter 2, while  $P_4$  have diameter 3. We present an interesting relationship between the diameter of  $G$  and the number of distinct eigenvalues in the spectrum of  $G$ .

**Theorem 6.1.3** Let  $G$  be a connected graph with diameter  $d$ . Then  $G$  has at least  $d+1$  distinct eigenvalues.

**Proof.** Let  $A$  be an adjacency matrix of  $G$  and let  $x, y \in V(G)$  be some vertices of  $G$  with  $d(x, y) = d$ . Suppose  $\theta_1, \theta_2, \dots, \theta_t$  be all distinct eigenvalues of  $A$ . Then note that  $(A - \theta_1 I)(A - \theta_2 I) \dots (A - \theta_t I) = 0$ . (To see this, recall that  $A$  is diagonalizable and so any vector  $v \in \mathbb{R}^n$  can be expressed as a sum of eigenvectors.) Hence,  $A^t$  can be expressed as a linear combination of  $I, A, A^2, \dots, A^{t-1}$ . If  $t \leq d$ , multiplying by  $A^{d-t}$  we obtain that  $A^d$  can be expressed as a linear combination of  $I, A, A^2, \dots, A^{d-1}$ . However,  $(A^d)_{xy} \neq 0$  while  $(A^k)_{xy} = 0$  for all  $k = 0, 1, \dots, d-1$ , which is a contradiction. Hence,  $t > d$  which finishes the proof. ■

## 6.2 Good Will Hunting problems

We finish the course by working through the problems from "Good Will Hunting". Give a go yourself, before reading the notes below!



- 1) Note that  $G$  is a multigraph. For multigraphs we can generalize our notion of adjacency matrix by recording the number of edges between the two vertices. Thus the adjacency matrix is:

$$A = \begin{pmatrix} 0 & 1 & 0 & 1 \\ 1 & 0 & 2 & 1 \\ 0 & 2 & 0 & 0 \\ 1 & 1 & 0 & 0 \end{pmatrix}$$

- 2) As we've seen in exercises before, the number of 3-step walks can be calculated by finding the cube of our matrix

$$A^3 = \begin{pmatrix} 2 & 7 & 2 & 3 \\ 7 & 2 & 12 & 7 \\ 2 & 12 & 0 & 2 \\ 3 & 7 & 2 & 2 \end{pmatrix}$$

- (3) We know that the number of walks from  $i$  to  $j$  is of length  $n$  is  $(A^n)_{ij}$ . Hence the generating function is

$$g_{ij}(x) = \sum_{n=0}^{\infty} (A^n)_{ij} x^n = \left( \sum_{n=0}^{\infty} A^n x^n \right)_{ij}.$$

The last equality can be justified formally by establishing the natural isomorphism between the ring of power series over matrices and the ring of matrices of power series.

Note that we have

$$I + Ax + A^2x^2 + A^3x^3 + \dots = (I - Ax)^{-1}.$$

This can be justified by treating these as 'formal series' and using the arguments mentioned in generating series section. Alternatively, this can also be justified because the series of matrices converges when  $x$  belongs to some small neighbourhood of 0. What does it mean to say that a sequence of matrices converge, you may ask? The  $n \times n$  matrices can be viewed as vectors with  $n^2$  entries, and a sequence of matrices define  $n^2$  sequences of real numbers. We say that the sequence of matrices converges, if and only if it converges pointwise at each of its  $n^2$  components, i.e. each of the  $n^2$  sequences of real numbers converges.

Let  $\mu$  be the spectral radius of the graph of  $A$ . Then, it's not hard to see that for any  $x \in \mathbb{R}$  with  $|x| < \frac{1}{\mu}$ ,  $A^n x^n \rightarrow 0$  as  $n \rightarrow \infty$ . Indeed, for any eigenvector  $v$ , we have  $A^n x^n v = \lambda^n x^n v \rightarrow 0$  as  $n \rightarrow \infty$ . Hence, as any vector  $w$  can be written as a linear sum of eigenvectors, we obtain that  $A^n x^n w \rightarrow 0$  as  $n \rightarrow \infty$  for any vector  $w$ , and so  $A^n x^n \rightarrow 0$  as  $n \rightarrow \infty$ . Moreover, it's not hard to see that  $I - Ax$  is invertible when  $|x| < \frac{1}{\mu}$ , and that  $(I - Ax)(I + Ax + \dots + A^{n-1}x^{n-1}) = I - A^n x^n$ . Hence, when  $|x| < \frac{1}{\mu}$  we have

$$I + Ax + A^2x^2 + \dots = \lim_{n \rightarrow \infty} (I + Ax + \dots + A^{n-1}x^{n-1}) = \lim_{n \rightarrow \infty} (I - A^n x^n)(I - Ax)^{-1} = (I - Ax)^{-1}.$$

Collecting all the entries together and using Cramer's rule for calculating entries of an inverse matrix we obtain the answer, where  $I^{ij}$  and  $A^{ij}$  denotes the matrices obtained from  $I$  and  $A$  by omitting the  $i$ 'th row and  $j$ 'th column:

$$g_{ij}(x) = \sum_{n=0}^{\infty} (A^n)_{ij} x^n = \left( \sum_{n=0}^{\infty} A^n x^n \right)_{ij} = ((I - Ax)^{-1})_{ij} = (-1)^{i+j} \frac{\det(I^{ij} - A^{ij}x)}{\det(I - Ax)}$$

- (4) Having obtained a general formula for walks between any two points, it is not hard to substitute  $i = 1$  and  $j = 3$  for walks from 1 to 3:

$$g_{13}(x) = (-1)^{1+3} \frac{\det(I^{13} - A^{13}x)}{\det(I - Ax)} = \frac{\det \begin{pmatrix} -x & 1 & -x \\ 0 & -2x & 0 \\ -x & -x & 1 \end{pmatrix}}{\det \begin{pmatrix} 1 & -x & 0 & -x \\ -x & 1 & -2x & -x \\ 0 & -2x & 1 & 0 \\ -x & -x & 0 & 1 \end{pmatrix}}$$

Calculating the determinants we obtain

$$g_{13}(x) = \frac{2x^3 + 2x^2}{4x^4 - 2x^3 - 7x^2 + 1} = \frac{(x+1)x^2}{(x+1)(4x^3 - 6x^2 - x + 1)} = \frac{2x^2}{4x^3 - 6x^2 - x + 1}$$

Now, to obtain the coefficients of the power series, one should repeatedly differentiate  $g(x) := \frac{2x^2}{4x^3 - 6x^2 - x + 1}$  and evaluate at 0, which looks hard. To aid the differentiation, we can use a trick by setting  $f(x) = 2x^2$  and  $h(x) = 4x^3 - 6x^2 - x + 1$ . Then we have  $g(x)h(x) = f(x)$ . Note that we know the values

$$f(0) = 0, f'(0) = 0, f''(0) = 4, f'''(0) = 0, f^{(4)}(0) = 0, f^{(5)}(0) = 0, f^{(6)}(0) = 0$$

and

$$h(0) = 1, h'(0) = -1, h''(0) = -12, h'''(0) = 24, h^{(4)}(0) = 0, h^{(5)}(0) = 0, h^{(6)}(0) = 0.$$

Hence we can solve for  $g$  as follows:

$$g(0)h(0) = f(0) \rightarrow g(0) = 0$$

$$g'(0)h(0) + h'(0)g(0) = f'(0) \rightarrow g'(0) = 0$$

$$g''(0)h(0) + 2g'(0)h'(0) + h''(0)g(0) = f''(0) \rightarrow g''(0) = 4$$

continuing this method we obtain

$$g'''(0) = 12, g^{(4)}(0) = 336, g^{(5)}(0) = 2160, g^{(6)}(0) = 67680.$$

Dividing by the appropriate factorials we obtain

$$g_{13}(x) = \frac{2x^2}{4x^3 - 6x^2 - x + 1} = 2x^2 + 2x^3 + 14x^4 + 18x^5 + 94x^6 + \dots$$

which is the answer that Will wrote on the board.

THE END!