Ve203 Discrete Mathematics

Lecturer: Zach McKenzie

University of Michigan - Shanghai Jiaotong University
Joint Institute

Fall Term 2017

Course Information

- **Lecturer:** Zach McKenzie
- **Room:** 409
- **Email:** `zachiri.mckenzie@sjtu.edu.cn`
- **Office Hours:** Mondays 10am – 12noon (in 409)
- **Teaching Assistants:** See CANVAS for their contact information
- **Recitation classes:** Teaching Assistants will lead a weekly recitation class that begins in the second week.

Use of Wikipedia and Other Sources; Honor Code Policy

When faced with a particularly difficult problem, you may want to refer to other textbooks or online sources such as Wikipedia. Here are a few guidelines:

- Outside sources may treat a similar sounding subject matter at a much more advanced or a much simpler level than this course. This means that explanations you find are much more complicated or far too simple to help you. For example, when looking up the "induction axiom" you may find many high-school level explanations that are not sufficient for our problems; on the other hand, wikipedia contains a lot of information relating to formal logic that is far beyond what we are discussing here.
- If you do use any outside sources to help you solve a homework problem, you are not allowed to just copy the solution; this is considered a violation of the Honor Code.

Use of Wikipedia and Other Sources; Honor Code Policy

- The correct way of using outside sources is to understand the contents of your source and then to write in your own words and without referring back to the source the solution of the problem. Your solution should differ in style significantly from the published solution. If you are not sure whether you are incorporating too much material from your source in your solutions, then you must cite the source that you used.
- You may cooperate with other students in finding solutions to assignments, but you must write your own answers. Do not simply copy answers from other students. It is acceptable to discuss the problems orally, but you may not look at each others' written notes. Do not show your written solutions to any other student. This would be considered a violation of the Honor Code.

Course Grade

The course contains four grade components:

- Three examinations,
- Course work.

The course grade will be calculated from these components using the following weighting:

- First midterm exam: 25% (based on 25 marks in the exam)
- Second midterm exam: 25% (based on 25 marks in the exam)
- Final exam: 25% (based on 25 marks in the exam)
- Course work: 25% (details to follow)

LaTeX Policy and Textbook

As engineers, you are strongly encouraged to familiarize yourselves with a mathematical typesetting program called LaTeX. This is open-source software, and there are various implementations available. I suggest that you use Baidu or Google to find a suitable implementation for your computer and OS.

While the use of LaTeX is **optional**, there will be a **10% bonus to the awarded marks** for those assignments handed in as typed LaTeX manuscripts.

The main textbook for this course is

- Rosen, K. H., *Discrete Mathematics and its Applications*, 6^{th} Ed., McGraw-Hill International Edition 2007.

Logic

Mathematical Logic, developed in the mid-1800s, gives mathematicians the tools to mathematicaly analyse the structure of language and arguments, and even treat notions such as “arguments”, “truth”, and “proof”, that seemingly belong to the realm of philosophy, as mathematical notions. The development of mathematical logic helped mathematicians identify the foundamental assumptions that underpin mathematics and what constitutes a rigorous “proof” of a mathematical statement. Today, mathematical logic has found wide ranging applications:

- It underpins much of the work that is done in computional liguistics (with applications like speach recognition software)
- It provides the basis for formal verification tools that are essential for checking the correctness of computer software and hardware.
- It has allowed us to treat notions such as “algorithm” and “proof” as mathematical objects giving us insight into the limitations of computers and mathematics.

The Natural Numbers

Despite the wide ranging applications of logic, this course will focus on using mathematical logic to analyse mathematical statements and arguments. For this reason, most of our examples will be based on numbers. For now, we assume that the set of natural numbers

$$\mathbb{N} := \{0, 1, 2, 3, \ldots\}$$

has been constructed. In particular, we assume that we know what a *set* is! If n is a natural number, we write $n \in \mathbb{N}$. (We will later discuss naive set theory and give a formal construction of the natural numbers.)
We also assume that on $\mathbb{N}$ we have defined the operations of addition $+ \colon \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ and multiplication $\cdot \colon \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ and that their various properties (commutativity, associativity, distributivity) hold.

The Natural Numbers

Definition

Let $m, n \in \mathbb{N}$ be natural numbers.

(i) *We say that n is greater than or equal to m, writing $n \geq m$, if there exists some $k \in \mathbb{N}$ such that $n = m + k$. If we can choose $k \neq 0$, we say n is greater than m and write $n > m$.*

(ii) *We say that m divides n, writing $m \mid n$, if there exists some $k \in \mathbb{N}$ such that $n = m \cdot k$.*

(iii) *If $2 \mid n$, we say that n is even.*

(iv) *If there exists some $k \in \mathbb{N}$ such that $n = 2k + 1$, we say that n is odd.*

(v) *Suppose that $n > 1$. If there does not exist any $k \in \mathbb{N}$ with $1 < k < n$ such that $k \mid n$, we say that n is prime.*

It can be proven that every number is either even or odd and not both. We also assume this for the purposes of our examples.

Propositional Logic

Propositional logic represents a first-attempt at analysing the structure of statements and arguments.

Definition

*A **proposition** is a declarative sentence. I.e. a statement that is either true (T) or false (F), but not both.*

Example

- *"$2 + 2 = 4$" is a true proposition*
- *"27 is a prime number" is a false proposition*
- *"$x > 5^3$" is NOT a proposition (the variable x is not specified)*
- *"$7 + 6$" is NOT a proposition*

Just as letters are used to represent variables in mathematical formulae, we will also use letters ($A, B, C, p, q, \ldots$) in propositional logic to denote **propositional variables**.

Propositional Logic

Compound expressions or compound propositions are are built up from propositions using **connectives**. There are five conectives availble to us in propositional logic: one unary connective (the connective prefixes a single proposition) $\neg$ (not), and four binary connective $\vee$ (disjunction), $\wedge$ (conjunction), $\Rightarrow$ (implication) and $\iff$ (biconditional). Propositional logic is **truth functional**, which means that the truth value (T or F) assigned to a compound expression is completely determined by the truth values of the propositions that appear in the compund expression.

Definition

Let A be a (compound) proposition. The proposition $\neg A$ (pronounced "It is not the case that A") is the statement that is true if A is false, and false if A is true.

Example

If A is the proposition A: $2 > 3$, then the negation of A is $\neg A$: $2 \not> 3$.

Negation and Conjunction

The behaviour of the connective $\neg$ is convieniently represented using a **truth table**:

A	$\neg A$
T	F
F	T

We can now use truth tables to define the behaviour of the remaining connectives.

Definition

Let A and B be two (compound) propositions. Then we define the **conjunction** *of A and B, written $A \wedge B$, by the following truth table:*

A	B	$A \wedge B$
T	T	T
T	F	F
F	T	F
F	F	F

Disjunction

Definition

Let A and B be two propositions. Then we define the **disjunction** *of A and B, written $A \vee B$, by the following truth table:*

A	B	$A \vee B$
T	T	T
T	F	T
F	T	T
F	F	F

The disjunction $A \vee B$ is spoken "A or B". It is true only if either A or B is true, false otherwise.

Example

- *Let A: $2 > 0$ and B: $1 + 1 = 1$. Then $A \wedge B$ is false and $A \vee B$ is true.*
- *Let A be a proposition. Then the compound expression "$A \vee (\neg A)$" is always true, and "$A \wedge (\neg A)$" is always false.*

Proofs using Truth Tables

How do we prove that "$A \vee (\neg A)$" is an always true expression? We are claiming that $A \vee (\neg A)$ will be a true expression, regardless of whether the proposition A is true or not. To prove this, we go through all possibilities using a truth table:

A	$\neg A$	$A \vee (\neg A)$
T	F	T
F	T	T

Since the column for $A \vee (\neg A)$ only lists T for "true", we see that $A \vee (\neg A)$ is always true. A compound expression that is always true is called a **tautology**.

Correspondingly, the truth table for $A \wedge (\neg A)$ is

A	$\neg A$	$A \wedge (\neg A)$
T	F	F
F	T	F

so $A \wedge (\neg A)$ is always false. A compound expression that is always false is called a **contradiction**.

Implication

Definition

Let A and B be two propositions. Then we define the **implication** *of B and A, written $A \Rightarrow B$, by the following truth table:*

A	B	$A \Rightarrow B$
T	T	T
T	F	F
F	T	T
F	F	T

In the expression $A \Rightarrow B$, A is called the **antecedent** *and B is called the* **consequence**.

We read "$A \Rightarrow B$" as "A implies B", "if A, then B" or "A only if B". (The last formulation refers to the fact that A can not be true unless B is true.) Note that an implication $A \Rightarrow B$ is false (F) only when the antecedent (A) is true (T) and the consequence (B) is false (F).

Implication

Example

Consider the following infinite list of implications:

$$A_n\colon n \text{ is prime} \Rightarrow n \text{ is odd}, \qquad n \in \mathbb{N}, \tag{1}$$

- A_2 *is the expression* $(2\ is\ prime) \Rightarrow (2\ is\ odd)$. *Since "2 is prime" is true and "2 is odd" is false,* A_2 *is false*
- A_3 *is the expression* $(3\ is\ prime) \Rightarrow (3\ is\ odd)$. *Since "3 is prime" is true and "3 is odd" is true,* A_3 *is true*
- A_4 *is the expression* $(4\ is\ prime) \Rightarrow (4\ is\ odd)$. *Since "4 is prime" is false and "4 is odd" is false,* A_4 *is true*
- A_9 *is the expression* $(9\ is\ prime) \Rightarrow (9\ is\ odd)$. *Since "9 is prime" is false and "9 is odd" is true,* A_9 *is true*

Biconditional

Definition

Let A and B be two propositions. Then we define the **biconditional** *of A and B, written $A \Leftrightarrow B$, by the following truth table:*

A	B	$A \Leftrightarrow B$
T	T	T
T	F	F
F	T	F
F	F	T

We read "$A \Leftrightarrow B$" as "A is equivalent to B" or "A if and only if B". Some textbooks abbreviate "if and only if" by "iff". If A and B are both true or both false, then they are equivalent. Otherwise, they are not equivalent. In propositional logic, "equivalence" is the closest thing to the "equality" of arithmetic.

Equivalence

On the one hand, logical equivalence is strange; two propositions A and B do not need to have anything to do with each other to be equivalent. For example, the statements "$2 > 0$" and "$100 = 99 + 1$" are both true, so they are equivalent.

On the other hand, we use equivalence to manipulate compound propositions.

Definition

Two compound propositions A and B are called **logically equivalent** *if $A \Leftrightarrow B$ is a tautology. We then write $A \equiv B$.*

Example

The two de Morgan rules are the tautologies

$$\neg(A \vee B) \Leftrightarrow (\neg A) \wedge (\neg B), \qquad \neg(A \wedge B) \Leftrightarrow (\neg A) \vee (\neg B).$$

In other words, they state that $\neg(A \vee B)$ is logically equivalent to $(\neg A) \wedge (\neg B)$ and $\neg(A \wedge B)$ is logically equivalent to $(\neg A) \vee (\neg B)$.

Contraposition

An important tautology is the equivalence of the proposition $A \Rightarrow B$ with its **contrapositive**: $\neg B \Rightarrow \neg A$.

$$(A \Rightarrow B) \Leftrightarrow (\neg B \Rightarrow \neg A).$$

For example, for any natural number n, the statement "$n > 0 \Rightarrow n^3 > 0$" is equivalent to "$n^3 \not> 0 \Rightarrow n \not> 0$" This principle is used in proofs by contradiction.
We prove the contrapositive using a truth table:

A	B	$\neg A$	$\neg B$	$\neg B \Rightarrow \neg A$	$A \Rightarrow B$	$(A \Rightarrow B) \Leftrightarrow (\neg B \Rightarrow \neg A)$
T	T	F	F	T	T	T
T	F	F	T	F	F	T
F	T	T	F	T	T	T
F	F	T	T	T	T	T

Some Logical Equivalences

The following logical equivalences can be established using truth tables or by using previously proven equivalences. Here T is the compound statement that is always true, $T : A \vee (\neg A)$ and F is the compound statement that is always false, $F : A \wedge (\neg A)$

Equivalence	Name
$A \wedge T \equiv A$	Identity for $\wedge$
$A \vee F \equiv A$	Identity for $\vee$
$A \wedge F \equiv F$	Dominator for $\wedge$
$A \vee T \equiv T$	Dominator for $\vee$
$A \wedge A \equiv A$	Idempotency of $\wedge$
$A \vee A \equiv A$	Idempotency of $\vee$
$\neg(\neg A) \equiv A$	Double negation

Some Logical Equivalences

Equivalence	Name
$A \wedge B \equiv B \wedge A$	Commutativity of $\wedge$
$A \vee B \equiv B \vee A$	Commutativity of $\vee$
$(A \wedge B) \wedge C \equiv A \wedge (B \wedge C)$	Associativity of $\wedge$
$(A \vee B) \vee C \equiv A \vee (B \vee C)$	Associativity of $\vee$
$A \vee (B \wedge C) \equiv (A \vee B) \wedge (A \vee C)$	Distributivity
$A \wedge (B \vee C) \equiv (A \wedge B) \vee (A \wedge C)$	Distributivity
$A \vee (A \wedge B) \equiv A$	Absorption
$A \wedge (A \vee B) \equiv A$	Absorption

These laws include all that are necessary for a *boolean algebra generated by* $\wedge$ *and* $\vee$ (identity element, commutativity, associativity, distributivity). Hence the name *boolean logic* for this calculus of logical statements.

Some Logical Equivalences

We omitted de Morgan's laws from the previous table.We now list some equivalences involving the implication.

Equivalence
$A \Rightarrow B \equiv \neg A \vee B \equiv \neg B \Rightarrow \neg A$
$(A \Rightarrow B) \wedge (A \Rightarrow C) \equiv A \Rightarrow (B \wedge C)$ $(A \Rightarrow B) \vee (A \Rightarrow C) \equiv A \Rightarrow (B \vee C)$
$(A \Rightarrow C) \wedge (B \Rightarrow C) \equiv (A \vee B) \Rightarrow C$ $(A \Rightarrow C) \vee (B \Rightarrow C) \equiv (A \wedge B) \Rightarrow C$
$(A \Leftrightarrow B) \equiv ((\neg A) \Leftrightarrow (\neg B))$ $(A \Leftrightarrow B) \equiv (A \Rightarrow B) \wedge (B \Rightarrow A)$ $(A \Leftrightarrow B) \equiv (A \wedge B) \vee ((\neg A) \wedge (\neg B))$ $\neg(A \Leftrightarrow B) \equiv A \Leftrightarrow (\neg B)$

Manipulating propositional expressions

One of the first things one learns in mathematics is to manipulate mathematical expressions using equations. For example, using the equation $(x+1)^2 = x^2 + 2x + 1$, one can seee that

$$x^2 + 3x + 1 = (x+1)^2 + x$$

Propositional expressions can be manipulated in the same fashion using known logical equivalences. This provides a way of proving new logical equivalences.

Example

$\neg(A \Rightarrow B)$ *is equivalent to* $A \wedge \neg B$ *because*

$$\begin{aligned} \neg(A \Rightarrow B) &\equiv \neg(\neg A \vee B) \\ &\equiv \neg\neg A \wedge \neg B \\ &\equiv A \wedge \neg B \end{aligned}$$

Manipulating propositional expressions

Example

The expression $(A \wedge B) \Rightarrow (A \vee B)$ *is a tautology:*

$$\begin{aligned}(A \wedge B) \Rightarrow (A \vee B) &\equiv \neg(A \wedge B) \vee (A \vee B) \\ &\equiv (\neg A \vee \neg B) \vee (A \vee B) \\ &\equiv (A \vee \neg A) \vee (B \vee \neg B) \\ &\equiv T \vee T \\ &\equiv T\end{aligned}$$

Note that the order of operation for the propositional connectives is: bracketed expressions, then $\neg$, then $\wedge$ and $\vee$, then $\Rightarrow$ and $\iff$. Brackets should always be used to distinguish between $\wedge$ and $\vee$, and $\Rightarrow$ and $\iff$.

Arguments in propositional logic

Propositional logic gives us a tool that we can use to analyse the structure of arguments. Consider:

$$\text{If } \sqrt{2} > \frac{3}{2}, \text{ then } (\sqrt{2})^2 > \left(\frac{3}{2}\right)^2. \text{ We know that } \sqrt{2} > \frac{3}{2}.$$

$$\text{Therefore } (\sqrt{2})^2 > \left(\frac{3}{2}\right)^2.$$

This is an example of an argument. We have asserted some facts, call premises, that we know to be true, and using these premises we determined that a new fact, called the conclusion, is true. It is enlightening to examine what this looks like in propositional logic. Let A be the proposition "$\sqrt{2} > \frac{3}{2}$" and let B be the proposition "$(\sqrt{2})^2 > \left(\frac{3}{2}\right)^2$".Then, the two premises of the argument are $A \Rightarrow B$ and A, and the conclusion is B.

$$\begin{array}{ll} & A \Rightarrow B \\ & A \\ \hline \therefore & B \end{array}$$

Arguments

Definition

An **argument** *is a finite sequence of propositions. All propositions except for the final statement are called* **premises** *while the final statement is called the* **conclusion**. *We say that an argument is* **valid** *if the truth of all premises implies the truth of the conclusion.*

From the definition of an argument it is clear that an argument consisting of a sequence of premises $P_1, \ldots, P_n$ and a conclusion C is valid of and only if

$$(P_1 \wedge P_2 \wedge \cdots \wedge P_n) \Rightarrow C \tag{2}$$

is a tautology, i.e., a true statement for any values of the premises an the conclusion.We can write this argument as:

$$\begin{array}{l} P_1 \\ P_2 \\ \vdots \\ P_n \\ \hline \therefore \quad C \end{array}$$

Arguments

The symbol $\therefore$ is pronounced "therefore". You may only use this symbol when constructing a logical argument in the notation above. Do not use it as a general-purpose abbreviation of "therefore".

On an earlier slide we saw the argument:

$$\begin{array}{ll} & A \Rightarrow B \\ & A \\ \hline \therefore & B \end{array}$$

To check that this argument is valid we need to verify that $((A \Rightarrow B) \wedge A) \Rightarrow B$ is a tautology.

A	B	$A \Rightarrow B$	$(A \Rightarrow B) \wedge A$	$((A \Rightarrow B) \wedge A) \Rightarrow B$
T	T	T	T	T
T	F	F	F	T
F	T	T	F	T
F	F	T	F	T

Hypothetical Syllogisms

Certain basic valid arguments in mathematics are given latin names and called **rules of inference**. A **syllogism** is an argument that has exactly two premises. We first give three **hypothetical syllogisms**, i.e., syllogisms involving the implication "$\Rightarrow$".

Rule of Inference	Name
$\begin{array}{l} A \Rightarrow B \\ A \\ \hline \therefore \quad B \end{array}$	Modus (Ponendo) Ponens *Mode that affirms (by affirming)*
$\begin{array}{l} A \Rightarrow B \\ \neg B \\ \hline \therefore \quad \neg A \end{array}$	Modus (Tollendo) Tollens *Mode that denies (by denying)*
$\begin{array}{l} A \Rightarrow B \\ B \Rightarrow C \\ \hline \therefore \quad A \Rightarrow C \end{array}$	Transitive Hypothetical Syllogism

Hypothetical Syllogisms

Example

(i) *Modus ponendo ponens:*

If 3 is both prime and greater than 2, then 3 is odd
3 is both prime and greater than 2

$\therefore$ *3 is odd.*

(ii) *Modus tollendo tollens:*

If 4 is both prime and greater than 2, then 4 is odd
4 is not odd

$\therefore$ *4 is not both prime and greater than 2.*

(iii) *Transitive hypothetical syllogism:*

If 5 is greater than 4, then 5 is greater than 3
If 5 is greater than 3, then 5 is greater than 2

$\therefore$ *If 5 is greater than 4, then 5 is greater than 2.*

Disjunctive and Conjunctive Syllogisms

There are two important syllogisms involving the disjunction "$\vee$" and the conjunction "$\wedge$":

Rule of Inference	Name
$\begin{array}{l} A \vee B \\ \neg A \\ \hline \therefore \quad B \end{array}$	Modus Tollendo Ponens *Mode that affirms by denying*
$\begin{array}{l} \neg(A \wedge B) \\ A \\ \hline \therefore \quad \neg B \end{array}$	Modus Ponendo Tollens *Mode that denies by affirming*
$\begin{array}{l} A \vee B \\ \neg A \vee C \\ \hline \therefore \quad B \vee C \end{array}$	Resolution

Disjunctive and Conjunctive Syllogisms

Example

(i) *Modus tollendo ponens:*

$$\begin{array}{ll} & \textit{4 is odd or even} \\ & \textit{4 is not odd} \\ \hline \therefore & \textit{4 is even.} \end{array}$$

(ii) *Modus ponendo tollens:*

$$\begin{array}{ll} & \textit{4 is not both even and odd} \\ & \textit{4 is even} \\ \hline \therefore & \textit{4 is not odd.} \end{array}$$

(iii) *Resolution:*

$$\begin{array}{ll} & \textit{4 is even or 4 is greater than 2} \\ & \textit{4 is odd or 4 is prime} \\ \hline \therefore & \textit{4 is greater than 2 or 4 is prime.} \end{array}$$

Some Simple Arguments

Finally, we give some seemingly obvious, but nevertheless useful, arguments:

Rule of Inference	Name
A B $\therefore\ A \wedge B$	Conjunction
$A \wedge B$ $\therefore\ A$	Simplification
A $\therefore\ A \vee B$	Addition

Examples for these are left to the reader!

Validity and Soundness

The previous rules of inference are all *valid arguments*. In the examples we gave, the arguments always led to a correct conclusion. This was, however, only because all the premises were true statements. It is possible for a valid argument to lead to a wrong conclusion if one or more of its premises are false.

If, in addition to being valid, an argument has only true premises, we say that the argument is **sound**. In that case, its conclusion is true.

Example

The following argument is valid (it is based on the rule of resolution), but not sound:

$$\begin{array}{ll} & \textit{4 is even or 4 is prime} \\ & \textit{4 is odd or 4 is prime} \\ \hline \therefore & \textit{4 is prime.} \end{array}$$

(The second premise is false, so the conclusion doesn't have to be true.)

Non Sequitur

The term **non sequitur** (latin for "it does not follow") is often used to describe logical fallacies, i.e., inferences that invalid because they are not based on tautologies.Some common fallacies are listed below:

Rule of Inference	Name
$\begin{array}{l} B \\ A \Rightarrow B \\ \hline \therefore \quad A \end{array}$	Affirming the Consequent
$\begin{array}{l} \neg A \\ A \Rightarrow B \\ \hline \therefore \quad \neg B \end{array}$	Denying the Antecedent
$\begin{array}{l} A \vee B \\ A \\ \hline \therefore \quad \neg B \end{array}$	Affirming a Disjunct

Non Sequitur

Example

(i) *Affirming the consequent:*

If 9 is prime, then it is odd
9 is odd

$\therefore$ *9 is prime.*

(ii) *Denying the antecedent*

If 9 is prime, then it is odd
9 is not prime

$\therefore$ *9 is not odd.*

(iii) *Affirming a disjunct:*

2 is even or 2 is prime
2 is even

$\therefore$ *2 is not prime.*

Predicate Logic

Consider the following argument:

every prime number that is greater than 2 is odd
7 is a prime number that is greater than 2

$\therefore$ 7 is odd.

This looks like a valid argument. However, if we try to analyse this argument using propositional logic, then the best we can do is assign the propositional variables A, B and C to the statements "every prime number that is greater than 2 is odd", "7 is a prime number that is greater than 2" and "7 is odd" respectively. But then the argument looks like:

$$\begin{array}{l} A \\ B \\ \hline \therefore \quad C \end{array}$$

which is not a valid argument!

Propositional logic is not rich enough to formalise statements in the form "for every ..." and "there exists ...". To formalise these statements we need quantifiers and predicates.

Predicates

Definition

A **predicate** *is a declaritive sentence involving variables. I.e. a statement invloving variables such that when the variables a substituted with appropriate individuals we obtain a proposition. The* **arity** *of a predicate is the number of distinct variables appearing in the predicate. We call a predicate of arity* 1 *a unary predicate, a predicate of arity* 2 *a binary predicate, . . .*

Example

- *"*$x > 5$*" is a unary predicate*
- *"*$x^2 + y$*" is NOT a predicate*
- *"*x *is the least prime greater than* y*" is a binary predicate*

Predicate logic

Predicate logic consists of **basic predicate variables** $A, B, P, Q, \ldots$, **variables** $x, y, z, \ldots$, and **constants** $a, b, c, \ldots$. If A is, for example, a binary predicate, then we write $A(x, y)$ to indicate that x and y are the variables appearing in A. The variables $x, y, z, \ldots$ take values from a universe called the **domain of discourse**. The constants $a, b, c, \ldots$ are specific individuals from the domian of discourse. Replacing all the variables by constants in a predicate or compound expression yields a declaritive statement (a proposition). Compound expressions in predicate logic are built-up by:

- Forming expressions using the connectives of propositional logic: $\neg$, $\wedge$, $\vee$, $\Rightarrow$ and $\Longleftrightarrow$
- Replacing variables by constant symbols or other variables
- **Bounding** variables in a compound expression using the **logical quantifier** $\forall$, that reads "for all ...", or the **logical quantifier** $\exists$, that reads "there exists ..."

Predicate logic

Example

Let the domain of discourse be $\mathbb{N}$. *Let* $P(x)$ *be the predicate "*x *is prime" and let* $Q(x, y)$ *be the predicate "*$x < y$*".*

1. $P(y) \wedge Q(x, y)$ *is the expression "*y *is prime and* $x < y$*"*
2. $P(y) \wedge Q(100, y)$ *is the expression "*y *is prime and* $100 < y$*"*
3. $\exists y(P(y) \wedge Q(100, y))$ *is the expression "there exists* y*, such that* y *is prime and* $100 < y$*"*
4. $\forall x \exists y(P(y) \wedge Q(x, y))$ *is the expression "for all* x*, there exists* y*, such that* y *is prime and* $x < y$*"*

The expressions (1) and (2) involve variables that are not yet specified. Such expressions involving unspecified variables are called **compound predicates** or **formulae**. In expressions (3) and (4) all of the variables have been replaced by constants or bound by quantifiers. These expressions are propositions and are called **sentences**.

Predicate logic

In order to assess the truth or falsity of a sentence ϕ in predicate logic, we need to specify the domain of discourse M (a collection of objects or set) in which all of the basic predicates have an interpretation. On the previous slide, the domain of discourse was $\mathbb{N}$ and the interpreation of the two predicates $P(x)$ and $Q(x, y)$ were given at the beginnning of the course.

Definition

Let M be a domain and let $A(x)$ be a predicate or formula. We define the quantifier $\forall$ by

$$\forall x A(x) \text{ is true in } M \iff A(x) \text{ is true for all } x \in M$$

We define the quantifier $\exists$ by

$$\exists x A(x) \text{ is true in } M \iff A(x) \text{ is true for at least one } x \in M$$

We often abbreviate "$\forall x A(x)$ is true in M" and "$\exists x A(x)$ is true in M" by $\underset{x \in M}{\forall} A(x)$ and $\underset{x \in M}{\exists} A(x)$. This notation also allows us to further restrict the domain of the quantifiers.

Predicate logic

We may also write $(\forall x \in M)A(x)$ or $(\exists x \in M)A(x)$ to specify or further restrict the domain of a quantifier.

Example

Let the domain be the real numbers ($\mathbb{R}$). Then

- $\forall x(x > 0 \Rightarrow x^3 > 0)$ *is a true statement;*
- $\forall x(x > 0 \Leftrightarrow x^2 > 0)$ *is a false statement;*
- $\exists x(x > 0 \Leftrightarrow x^2 > 0)$ *is a true statement.*

Sometimes mathematicians put a quantifier at the end of a statement form; this is known as a **hanging quantifier**. Such a hanging quantifier will be interpreted as being located just before the expression:

$$\exists y(y + x^2 > 0) \qquad\qquad \forall x$$

is equivalent to $\exists y \forall x(y + x^2 > 0)$.

Contraposition and Negation of Quantifiers

We do not actually need the quantifier $\exists$ since

$$\begin{aligned} \underset{x\in M}{\exists} A(x) \Leftrightarrow \quad & A(x) \text{ is true for at least one } x \in M \\ \Leftrightarrow \quad & A(x) \text{ is not false for all } x \in M \\ \Leftrightarrow \quad & \neg \underset{x\in M}{\forall} \neg A(x) \end{aligned} \tag{3}$$

The equivalence (**??**) is called **contraposition of quantifiers**. It implies that the negation of $(\exists x \in M)A(x)$ is equivalent to $(\forall x \in M)\neg A(x)$. For example,

$$\neg\Big((\exists x \in \mathbb{R})x^2 < 0\Big) \qquad \Leftrightarrow \qquad (\forall x \in \mathbb{R})(x^2 \not< 0).$$

Conversely,

$$\neg(\forall x \in M)A(x) \qquad \Leftrightarrow \qquad (\exists x \in M)\neg A(x).$$

Vacuous Truth

If the domain of the universal quantifier $\forall$ is the empty set $M = \emptyset$, then the statement $(\forall x \in M)A(x)$ is defined to be true regardless of the predicate $A(x)$. It is then said that $A(x)$ is **vacuously true**.

Example

Let M be the set of real numbers x such that $x = x + 1$. Then the statement

$$\forall_{x \in M} \; x > x$$

is true.

This convention reflects the philosophy that a universal statement is true unless there is a counterexample to prove it false. While this may seem a strange point of view, it proves useful in practice.
This is similar to saying that "All pink elephants can fly" is a true statement, because it is impossible to find a pink elephant that can't fly. It is also consistent with behaviour of compound expressions involving $\Rightarrow$, which are true if the antecedent is false, regarless of the consequence.

Quantifier order

Example

Let the domain be the real numbers ($\mathbb{R}$).

- $\forall x \forall y(x^2 + y^2 - 2xy \geq 0)$ *is equivalent to* $\forall y \forall x(x^2 + y^2 - 2xy \geq 0)$. *Therefore, one often writes* $\forall x, y(x^2 + y^2 - 2xy \geq 0)$.
- $\exists x \exists y(x + y > 0)$ *is equivalent to* $\exists y \exists x(x + y > 0)$. *One often writes* $\exists x, y(x + y > 0)$.
- $\forall x \exists y(x + y > 0)$ *is a true statement.*
- $\exists x \forall y(x + y > 0)$ *is a false statement.*

The order of the quantifiers is important if they are different.

Examples from Calculus

Let the domain of discourse be $\mathbb{R}$ and let I be an interval in $\mathbb{R}$. Then a function $f : I \longrightarrow \mathbb{R}$ is said to be **continuous** on I if and only if

$$\underset{\varepsilon>0}{\forall}\ \underset{x\in I}{\forall}\ \underset{\delta>0}{\exists}\ \underset{y\in I}{\forall} \quad |x-y|<\delta \Rightarrow |f(x)-f(y)|<\varepsilon.$$

The function f is **uniformly continuous** on I if and only if

$$\underset{\varepsilon>0}{\forall}\ \underset{\delta>0}{\exists}\ \underset{x\in I}{\forall}\ \underset{y\in I}{\forall} \quad |x-y|<\delta \Rightarrow |f(x)-f(y)|<\varepsilon.$$

It is easy to see that a function that is uniformly continuous on I must also be continuous on I.
If I is a closed interval, $I = [a, b]$, it can also be shown that a continuous function is also uniformly continuous. However, that requires techniques from calculus and is not obvious just by looking at the logical structure of the definitions.

Examples from Calculus

Negating complicated expressions can be done step-by-step. For example, the statement that f is not continuous on I is equivalent to

$$\neg\Big(\underset{\varepsilon>0}{\forall}\ \underset{x\in I}{\forall}\ \underset{\delta>0}{\exists}\ \underset{y\in I}{\forall}\quad |x-y|<\delta \Rightarrow |f(x)-f(y)|<\varepsilon\Big)$$

$$\Leftrightarrow \Big(\underset{\varepsilon>0}{\exists}\ \neg\ \underset{x\in I}{\forall}\ \underset{\delta>0}{\exists}\ \underset{y\in I}{\forall}\quad |x-y|<\delta \Rightarrow |f(x)-f(y)|<\varepsilon\Big)$$

$$\Leftrightarrow \Big(\underset{\varepsilon>0}{\exists}\ \underset{x\in I}{\exists}\ \neg\ \underset{\delta>0}{\exists}\ \underset{y\in I}{\forall}\quad |x-y|<\delta \Rightarrow |f(x)-f(y)|<\varepsilon\Big)$$

$$\Leftrightarrow \Big(\underset{\varepsilon>0}{\exists}\ \underset{x\in I}{\exists}\ \underset{\delta>0}{\forall}\ \neg\ \underset{y\in I}{\forall}\quad |x-y|<\delta \Rightarrow |f(x)-f(y)|<\varepsilon\Big)$$

$$\Leftrightarrow \Big(\underset{\varepsilon>0}{\exists}\ \underset{x\in I}{\exists}\ \underset{\delta>0}{\forall}\ \underset{y\in I}{\exists}\quad (|x-y|<\delta)\wedge\neg(|f(x)-f(y)|<\varepsilon)\Big)$$

$$\Leftrightarrow \Big(\underset{\varepsilon>0}{\exists}\ \underset{x\in I}{\exists}\ \underset{\delta>0}{\forall}\ \underset{y\in I}{\exists}\quad (|x-y|<\delta)\wedge(|f(x)-f(y)|\geq\varepsilon)\Big)$$

Examples from Calculus

Example

The Heaviside function $H : \mathbb{R} \longrightarrow \mathbb{R}$,

$$H(x) := \begin{cases} 1 & if\ x \geq 0, \\ 0 & if\ x < 0 \end{cases}$$

is not continuous on $I = \mathbb{R}$. *To see this, we need to show that there exists an* $\varepsilon > 0$ *(take* $\varepsilon = 1/2$*) and an* $x \in \mathbb{R}$ *(take* $x = 0$*) such that for any* $\delta > 0$ *there exists* $y \in \mathbb{R}$ *such that*

$$|x - y| = |y| < \delta \qquad and \qquad |H(x) - H(y)| = |1 - H(y)| \geq \varepsilon = \frac{1}{2}.$$

Given any $\delta > 0$ *we can choose* $y = -\delta/2$. *Then* $|y| = \delta/2 < \delta$ *and* $|1 - H(y)| = 1 > 1/2$. *This proves that* H *is not continuous on* $\mathbb{R}$.

Tautologies in predicate logic

Now that we are equipped with a calculus for compund expressions involving predicates and quantifiers, determining the truth or falsity of sentences becomes much more complicated. For example, the truth of a unary predicate bound by a quantifier depends on the truth of that predicate applied to individuals in the domain discourse. The following generalises the notion of tautology to sentences in predicate logic:

Definition

Let A be a predicate logic sentence. A is a **tautology** *if for every non-empty domain of of discourse M that is equipped with interpretations of the predicate symbols in A, A is true in M.*

Example

The sentence $A : \exists x \forall y Q(y, x)$ is not a tautology because A is not true in $\mathbb{N}$ with $Q(x, y)$ interpreted as "$x \leq y$".

Rules of inference for quantified expressions

Showing that sentences are tautologies of predicate logic is difficult, so instead we introduce rules of inference that allow us to deal with expressions involving quantifiers. These are taken as primitive (they are assumed):

Rule of Inference	Name
$\dfrac{\underset{x\in M}{\forall} P(x)}{\therefore \quad P(x_0) \text{ for any } x_0 \in M}$	Universal Instantiation
$\dfrac{P(x) \text{ for any arbitrarily chosen } x \in M}{\therefore \quad \underset{x\in M}{\forall} P(x)}$	Universal Generalization
$\dfrac{\underset{x\in M}{\exists} P(x)}{\therefore \quad P(x_0) \text{ for a certain (unknown) } x_0 \in M}$	Existential Instantiation
$\dfrac{P(x_0) \text{ for some (known) } x_0 \in M}{\therefore \quad \underset{x\in M}{\exists} P(x)}$	Existential Generalization

Rules of inference for quantified expressions

In addition to these rules of inference, every valid argument in propositional logic remains valid in predicate logic.

The fact these rules of inference yield all the tautologies of predicate logic, and conversely that all of these rules of inference are valid, are deep results of mathematical logic known as the Completeness and Soundness Theorems.

Constructing Arguments

Often, complex arguments can be broken down into syllogisms. As an example, we give a logical proof of the following theorem:

Theorem

Let $n \in \mathbb{N}$ be a natural number and suppose that n^2 is even. Then n is even.

Proof.

We use the following premises:

$$P_1: \underset{n\in\mathbb{N}}{\forall} \neg(n \text{ even} \wedge n \text{ odd}),$$

$$P_2: n \text{ odd} \Rightarrow n^2 \text{ odd},$$

$$P_3: n^2 \text{ even} \wedge (n \text{ even} \vee n \text{ odd})$$

and we wish to arrive at the conclusion

$$C: n \text{ even}.$$

Constructing Arguments

Proof (continued).

Premise P_2 can be easily checked: if n is odd, there exists some k such that $n = 2k + 1$, so

$$n^2 = (2k + 1)^2 = 2(2k^2 + 2k) + 1 = 2k' + 1$$

where $k' = 2k^2 + 2k$. Hence n^2 is also odd. We have

$$\frac{P_3 \colon n^2 \text{ even} \wedge (n \text{ even} \vee n \text{ odd})}{\therefore \quad P_4 \colon n^2 \text{ even}.}$$

by the Rule of Simplification. By Universal Instantiation, we obtain

$$\frac{P_1 \colon \underset{n \in \mathbb{N}}{\forall} \neg(n \text{ even} \wedge n \text{ odd})}{\therefore \quad P_5 \colon \neg(n^2 \text{ even} \wedge n^2 \text{ odd}).}$$

Constructing Arguments

Proof (continued).

Furthermore, by Modus Ponendo Tollens,

$$\begin{array}{ll} & P_4 : n^2 \text{ even} \\ & P_5 : \neg(n^2 \text{ even} \wedge n^2 \text{ odd}) \\ \hline \therefore \quad & P_6 : \neg(n^2 \text{ odd}). \end{array}$$

Using Modus Tollendo Tollens,

$$\begin{array}{ll} & P_6 : \neg(n^2 \text{ odd}) \\ & P_2 : n \text{ odd} \Rightarrow n^2 \text{ odd} \\ \hline \therefore \quad & P_7 : \neg(n \text{ odd}). \end{array}$$

Simplification yields

$$\begin{array}{ll} & P_3 : n^2 \text{ even} \wedge (n \text{ even} \vee n \text{ odd}) \\ \hline \therefore \quad & P_8 : n \text{ even} \vee n \text{ odd}. \end{array}$$

Constructing Arguments

Proof (continued).

Finally, Modus Tollendo Ponens gives

$$\begin{array}{ll} & P_7 : \neg(n \text{ odd}) \\ & P_8 : n \text{ even} \vee n \text{ odd} \\ \hline \therefore \quad & C : n \text{ even}. \end{array}$$

This completes the proof. □

Of course, this proof could have been shortened and simplified if we had replaced "odd" with "not even" throughout, and we might have formulated premise P_3 slightly differently (as two separate premises) to avoid using the rule of simplification. However, our goal was to illustrate the usage of a wide variety of rules of inference and that writing down a logically valid proof is in most cases extremely tedious; in most mathematics, many of the mentioned rules of inference are used implicitly without being stated.

Proofs by contradiction

One indirect proof method that often arises in mathematics is **proof by contradiction** (this proof method sometimes goes by the latin name *reductio ad absurdum*). If we want to prove a sentence A is true using a proof by contradiction, then we start by assuming that A is false (i.e. $\neg A$ is true). If, using this assumption, we are able to prove a contradiction (a sentence $B \wedge \neg B$) using a valid argument, then, since we know that $B \wedge \neg B$ is always FALSE, we can conclude that our argument must not be sound, and so $\neg A$ must be false. Therefore we can conclude that A is true.

Theorem

There are infinitely many primes.

Proof by contradiction

Proof.

Suppose that there are NOT infintely many primes. Therefore there must be finitely many primes:

$$p_1 < \cdots < p_n$$

and hence a largest prime p_n. Let

$$m = \prod_{1 \leq i \leq n} p_i + 1.$$

By our assumption, m is not prime, so there exists p_j such that $p_j \mid m$. I.e. $m = p_j k$ for some integer k. But now,

$$1 = p_j \left(k - \prod_{1 \leq i \neq j \leq n} p_i \right),$$

which shows that $p_j \mid 1$. This is a contradiction. Therefore, we can conclude that there are infinitely many primes. $\square$

Naive Set Theory: Sets via Predicates

Set Theory is the study of collections of mathematical objects. A **set** is a collection of objects that contains no information about order, and in which repetitions of objects are ignored. In pure set theory every object is itself a set, but it is often useful, and formally harmless, to also think of other mathemetical entities such as numbers and shapes, or even physical entities, as objects. We will see later that mathematical entities such as the natural numbers can be represented as sets, and mathematicians have provided evidence that every mathematical entity can be represented using sets.

We will see later that allowing any collection of objects (or sets) to be a set leads to an inconsistent theory. This means that if we were going to be rigorous we would need to set out rules that tell us which collections are sets. However, as collections that lead to inconsistency are not generally encountered in applications of set theory, in this course we will assume that every collection that can be desribed is set - this is called **Naive Set Theory**. In contrast, laying out definite rules for which collections are sets is referred to as **Axiomatic Set Theory**.

Curly brackets ($\{\ldots\}$) are used to indicate that some collection of objects are collected into a set.

Definition

Let X be a set and let x be an object. We write $x \in X$ to indicate that x is a member of X.

If $P(x)$ is predicate then the set of all objects x that satisfy $P(x)$ is written:

$$X = \{x \mid P(x)\}$$

I.e. $x \in X$ if and only if $P(x)$

Definition

Two sets X and Y are equal ($X = Y$) if for all x, $x \in X$ if and only if $x \in Y$.

Notation for Sets

We define the empty set $\emptyset := \{x \mid x \neq x\}$. The empty set has no elements, because the predicate $x \neq x$ is never true.

We may also use the notation $X = \{x_1, x_2, \ldots, x_n\}$ to denote a set. In this case, X is understood to be the set

$$X = \{x \mid (x = x_1) \vee (x = x_2) \vee \cdots \vee (x = x_n)\}.$$

We will frequently use the convention

$$\{x \in A \mid P(x)\} = \{x \mid x \in A \wedge P(x)\}$$

Example

The set of even nonnegative integers is

$$\{n \in \mathbb{N} \mid \exists_{k \in \mathbb{N}} n = 2k\}$$

Subsets

If every object $x \in X$ is also an element of a set Y, we say that X is a **subset** of Y, writing $X \subseteq Y$; in other words,

$$X \subseteq Y \Leftrightarrow \forall x(x \in X \Rightarrow x \in Y).$$

Note that $X = Y$ if and only if $X \subseteq Y$ and $Y \subseteq X$.We say that X is a

proper subset of Y if $X \subseteq Y$ but $X \neq Y$. In that case we write $X \subset Y$.

Some authors write $\subset$ for $\subseteq$ and $\subsetneq$ for $\subset$. Pay attention to the convention used when referring to literature.

Examples of Sets and Subsets

Example

1. *For any set X, $\emptyset \subseteq X$. Since $\emptyset$ does not contain any elements, for every x, the antecedant of the implication $x \in \emptyset \Rightarrow x \in X$ is false. So, the implication is true, which shows that $\emptyset \subseteq X$.*
2. *Consider the set $A = \{a, b, c\}$ where a, b, c are arbitrary objects, for example, numbers. The set*

$$B = \{a, b, a, b, c, c\}$$

is equal to A,

$$x \in A \quad \Leftrightarrow \quad (x = a) \vee (x = b) \vee (x = c) \quad \Leftrightarrow \quad x \in B.$$

If $C = \{a, b\}$, then $C \subseteq A$ and in fact $C \subset A$. Setting $D = \{b, c\}$ we have $D \subset A$ but $C \not\subseteq D$ and $D \not\subseteq C$.

3. *Let $A = \{\emptyset, \{\{\emptyset\}\}\}$, $B = \{\emptyset\}$ and $C = \{\{\emptyset\}\}$. $B \subseteq A$, but $B \notin A$, and $C \in A$, but $C \not\subseteq A$.*

Powerset and Cardinality

If a set X has a finite number of elements, we define the **cardinality** of X to be this number, denoted by $\#X$, $|X|$ or $\mathrm{card}(X)$.

Definition

If X is a set, then the **powerset** *of X, denoted $\mathcal{P}(X)$, is the set of all subsets of X. I.e.*

$$\mathcal{P}(X) := \{A \mid A \subseteq X\}$$

This means the expressions "$A \in \mathcal{P}(X)$" and "$A \subseteq X$" are equivalent.

Example

The power set of $\{a, b, c\}$ is

$$\mathcal{P}(\{a, b, c\}) = \{\emptyset, \{a\}, \{b\}, \{c\}, \{a, b\}, \{b, c\}, \{a, c\}, \{a, b, c\}\}.$$

$|\{a, b, c\}| = 3$ *and* $|\mathcal{P}(\{a, b, c\})| = 8$.

Operations on Sets

Let A and B be sets.

Definition

- *The* **union** *of A and B is the set:* $A \cup B = \{x \mid x \in A \vee x \in B\}$
- *The* **intersection** *of A and B is the set:* $A \cap B = \{x \mid x \in A \wedge x \in B\}$
- *The* **difference** *is:* $A \backslash B = \{x \in A \mid x \notin B\}$ *some authors write this* $A - B$

If $A \subseteq M$, then $M \backslash A$ is called the complement of A. Some authors write this A^c when M is clear from the context.
If $A \cap B = \emptyset$, then we say that A and B are disjoint.

Example

Let $A = \{a, b, c\}$ *and* $B = \{c, d\}$. *Then*

$$A \cup B = \{a, b, c, d\}, \qquad A \cap B = \{c\}, \qquad A \setminus B = \{a, b\}.$$

Operations on Sets

Logical equivalences immediately lead to several rules for set operations. For example, the distributive laws for $\wedge$ and $\vee$ imply

- $A \cap (B \cup C) = (A \cap B) \cup (A \cap C)$
- $A \cup (B \cap C) = (A \cup B) \cap (A \cup C)$

Other such rules are, for example,

- $(A \cup B) \setminus C = (A \setminus C) \cup (B \setminus C)$
- $(A \cap B) \setminus C = (A \setminus C) \cap (B \setminus C)$
- $A \setminus (B \cup C) = (A \setminus B) \cap (A \setminus C)$
- $A \setminus (B \cap C) = (A \setminus B) \cup (A \setminus C)$
- $A \setminus B = B^c \cap A$
- $(A \setminus B)^c = A^c \cup B$

Some of these will be proved in the recitation class and the exercises.

Operations on Sets

Occasionally we will need the following notation for the union and intersection of a finite number $n \in \mathbb{N}$ of sets:

$$\bigcup_{k=0}^{n} A_k := A_0 \cup A_1 \cup A_2 \cup \cdots \cup A_n,$$

$$\bigcap_{k=0}^{n} A_k := A_0 \cap A_1 \cap A_2 \cap \cdots \cap A_n.$$

This notation even extends to $n = \infty$, but needs to be properly defined:

$$x \in \bigcup_{k=0}^{\infty} A_k \quad :\Leftrightarrow \quad \exists_{k \in \mathbb{N}} \; x \in A_k,$$

$$x \in \bigcap_{k=0}^{\infty} A_k \quad :\Leftrightarrow \quad \forall_{k \in \mathbb{N}} \; x \in A_k.$$

Operations on Sets

In particular,

$$\bigcap_{k=0}^{\infty} A_k \subseteq \bigcup_{k=0}^{\infty} A_k.$$

Example

Let $A_k = \{0, 1, 2, \ldots, k\}$ *for* $k \in \mathbb{N}$. *Then*

$$\bigcup_{k=0}^{\infty} A_k = \mathbb{N}, \qquad \bigcap_{k=0}^{\infty} A_k = \{0\}.$$

To see the first statement, note that $\mathbb{N} \subseteq \bigcup_{k=0}^{\infty} A_k$ *since* $x \in \mathbb{N}$ *implies* $x \in A_x$ *implies* $x \in \bigcup_{k=0}^{\infty} A_k$. *Furthermore,* $\bigcup_{k=0}^{\infty} A_k \subseteq \mathbb{N}$ *since* $x \in \bigcup_{k=0}^{\infty} A_k$ *implies* $x \in A_k$ *for some* $k \in \mathbb{N}$ *implies* $x \in \mathbb{N}$.

For the second statement, note that $\bigcap_{k=0}^{\infty} A_k \subseteq \mathbb{N}$. *Now* $0 \in A_k$ *for all* $k \in \mathbb{N}$. *Thus* $\{0\} \subseteq \bigcap_{k=0}^{\infty} A_k$. *On the other hand, for any* $x \in \mathbb{N} \setminus \{0\}$ *we have* $x \notin A_{x-1}$ *whence* $x \notin \bigcap_{k=0}^{\infty} A_k$.

Operations on Sets

More generally, if A is a set and $X \subseteq \mathcal{P}(A)$, then

$$\bigcup X = \{x \in A \mid (\exists y \in X)(x \in y)\} \text{ and } \bigcap X = \{x \in A \mid (\forall y \in X)(x \in y)\}$$

Example

Let

$$X = \{A \in \mathcal{P}(\mathbb{N}) \mid (\exists k \in \mathbb{N})(\forall n \in \mathbb{N})(n \in A \vee n = k)\}$$

Then

$$\bigcup X = \mathbb{N} \; and \; \bigcap X = \emptyset$$

Note here that $X \subseteq \mathcal{P}(A)$, but both $\bigcup X$ and $\bigcap X$ are subsets of A (these operations strip off $\{\cdots\}$).

Ordered Pairs

A set does not contain any information about the order of its elements, e.g.,

$$\{a, b\} = \{b, a\}.$$

Thus, there is no such a thing as the "first element of a set". However, sometimes it is convenient or necessary to have such an ordering. This is achieved by defining an **ordered pair**, denoted by

$$(a, b)$$

and having the property that

$$(a, b) = (c, d) \qquad \Leftrightarrow \qquad (a = c) \wedge (b = d). \tag{4}$$

We define

$$(a, b) := \{\{a\}, \{a, b\}\}.$$

It is not difficult to see that this definition guarantees that (**??**) holds.

Cartesian Product of Sets

If A, B are sets and $a \in A$, $b \in B$, then we denote the set of all ordered pairs by

$$A \times B := \{(a, b) \mid a \in A \wedge b \in B\}.$$

$A \times B$ is called the **cartesian product** of A and B.In this manner we can

define an **ordered triple** (a, b, c) or, more generally, an ordered n-**tuple** $(a_1, \ldots, a_n)$ and the n-fold cartesian product $A_1 \times \cdots \times A_n$ of sets A_k, $k = 1, \ldots, n$.If we take the cartesian product of a set with itself, we may

abbreviate it using exponents, e.g.,

$$\mathbb{N}^2 := \mathbb{N} \times \mathbb{N}.$$

Problems in Naive Set Theory

The fact that Naive Set Theory allows the unrestricted formation of sets, means that a set A can be defined with the property that $A \in A$. For example: the set of all sets, the set of all infinite sets, ... Unrestricted self-reference has long been known to be problematic. For example,

- Epimenides Paradox (6^{th}-century BC)
- Barber Paradox: A male barber in a hamlet shaves all those, and only those, who do not shave themselves. Who shaves the barber?

In 1902 Bertrand Russell formalised these paradoxes in Naive Set Theory to show that Naive Set Theory is inconsistent.

Russell's Paradox

Theorem

The set of all sets that are not members of themselves is not a set. I.e.

$$R := \{x \mid x \notin x\} \; is \; not \; a \; set.$$

Proof.

The proof is by contradiction. Suppose that R is a set. If $R \in R$, then $R \notin R$ by the definition of R, which is a contradiction. If $R \notin R$, then $R \in R$ by the definition of R, which is also a contradiction. $\square$

Russell's Paradox

We will simply ignore the existence of such contradictions and build on Naive Set Theory. There are further paradoxes in naive set theory, such as **Cantor's Paradox** and the **Burali-Forti Paradox**. All of these are resolved if naive set theory is replaced by a modern axiomatic set theory such as **Zermelo-Fraenkel Set Theory**.

Further Information:

- *Set Theory*, Stanford Encyclopedia of Philosophy, `http://plato.stanford.edu/entries/set-theory/`
- P.R. Halmos, **Naive Set Theory**, Available here: `http://link.springer.com/book/10.1007/978-1-4757-1645-0`
- T. Jech, **Set Theory: The Third Millennium Edition, Revised and Expanded**, Available here: `http://link.springer.com/book/10.1007/3-540-44761-X`

Relations

Definition

A set R is called a **relation** *if R only contains ordered pairs.*

This means that a reation describes a relationship between members of sets. Let R be a relation. We define the **domain** of R to be the set

$$\text{dom } R = \{x \mid \exists y((x, y) \in R)\}$$

And we define the **range** of R to be the set

$$\text{ran } R = \{y \mid \exists x((x, y) \in R)\}$$

If $(a, b) \in R$ then we say that a and b are related (by R). Sometimes we write aRb instead of $(a, b) \in R$.
Sometimes we want to refer to the objects that R is relating. If M is a set and $R \subseteq M \times M$, then we say that R is a relation on M.

Relations

Example

- $R = \emptyset$ *is a relation (it vacuously contains only ordered pairs) with* $\operatorname{dom} R = \operatorname{ran} R = \emptyset$
- *If* A *and* B *are sets, then* $R = A \times B$ *is a relation with* $\operatorname{dom} R = A$ *and* $\operatorname{ran} R = B$
- $$R = \{(a, b) \in \mathbb{N} \times \mathbb{N} \mid \exists_{k \in \mathbb{N}} (a = b + k)\}$$

  is a relation with $\operatorname{dom} R = \operatorname{ran} R = \mathbb{N}$. *This is the relation* $\geq$ *on* $\mathbb{N}$.
- $$R = \{(a, b) \in \mathbb{N} \times \mathbb{N} \mid a, b \textit{ are both even or both odd}\}$$

  is a relation with $\operatorname{dom} R = \operatorname{ran} R = \mathbb{N}$. *This relation is equivalence* $(\operatorname{mod} 2)$.

Attributes of Relations

- $$R = \{(1,3),(1,2),(2,3),(4,3),(4,2)\}$$

  is a relation with $\mathrm{dom}\ R = \{1,2,4\}$ and $\mathrm{ran}\ R = \{2,3\}$

Definition

We say that a relation R on a set M is

(i) **reflexive** *if for all $a \in M$, $(a,a) \in R$*

(ii) **symmetric** *if for all $a, b \in M$, if $(a,b) \in R$, then $(b,a) \in R$*

(iii) **antisymmetric** *if for all $a, b \in M$, if $(a,b) \in R$ and $(b,a) \in R$, then $a = b$*

(iv) **asymmetric** *if for all $a, b \in M$, if $(a,b) \in R$, then $(b,a) \notin R$*

(v) **transitive** *if for all $a, b, c \in M$, if $(a,b) \in R$ and $(b,c) \in R$, then $(a,c) \in R$*

Attributes of Relations

Example

- *The relation* $R = \emptyset$ *on* $\emptyset$ *is reflexive, symmetric, antisymmetric, asymmetric and transitive. If* $M \neq \emptyset$ *then* R *on* M *is symmetric, antisymmetric, asymmetric and transitive.*
- $$R = \{(a, b) \in \mathbb{N} \times \mathbb{N} \mid \underset{k \in \mathbb{N}}{\exists} (a = b + k)\}$$

  is reflexive, antisymmetric and transitive.
- $$R = \{(a, b) \in \mathbb{N} \times \mathbb{N} \mid a, b \ \textit{are both even or both odd}\}$$

  is reflexive, symmetric and transitive.
- $R = \{(1, 2), (3, 4)\}$ *is antisymmetric, asymmetric and transitive.*

Equivalence Relations

Definition

Let R be a relation on a set M. If R is reflexive, symmetric and transitive, then we say that R is an **equivalence relation** *on M. If R is an equivalence relation on M and $a \in M$, the define the* **equivalence class of** *a to be*

$$[a]_R = \{b \in M \mid (a, b) \in R$$

We sometimes just write $[a]$ when R is clear from the context.

If R is an equivalence relation on a set M, then for all $a, b \in M$,

$$\text{either } [a]_R \cap [b]_R = \emptyset \text{ or } [a]_R = [b]_R$$

In other words, the equivalence classes partition the set M.

Equivalence Relations

Example

$$R = \{(a, b) \in \mathbb{N} \times \mathbb{N} \mid a, b \ are\ both\ even\ or\ both\ odd\}$$

is an equivalence relation on $\mathbb{N}$ *and the equivalence classes partition* $\mathbb{N}$ *into two sets:*

$$[0]_R = \{n \in \mathbb{N} \mid n \ is\ even\} \ and\ [1]_R = \{n \in \mathbb{N} \mid n \ is\ odd\}$$

The set $[0]_R$ *is often written as* $2\mathbb{N}$ *and the set* $[1]_R$ *is often written as* $2\mathbb{N} + 1$.

Equivalence Relations

Example

For $n \in \mathbb{N}$ we define the integer sum $I(n)$ as the sum of all integers that compose the number, e.g. $I(125) = 1 + 2 + 5 = 8$, $I(78) = 7 + 8 = 15$.

$$R = \{(a, b) \in \mathbb{N} \times \mathbb{N} \mid I(a) = I(b)$$

is an equivalence relation on $\mathbb{N}$ and the equivalence classes partition $\mathbb{N}$ into infinitely many sets:

$$\begin{aligned} [1]_R &= \{n \in \mathbb{N} \mid I(n) = 1\} \\ [11]_R &= \{n \in \mathbb{N} \mid I(n) = 2\} \\ [111]_R &= \{n \in \mathbb{N} \mid I(n) = 3\} \\ &\vdots \end{aligned}$$

Orders

Definition

Let R be a relation on a set M. If R is reflexive, antisymmetric and transitive, then R is called a **partial order**. *We often write a partial order together with its domain, (M, R), and say that (M, R) is a* **partially ordered set** *or* **poset**.

Note that the requirement that the axiom of reflexivity in the specification of partial order is sometimes replaced by irreflexivity (for all $a \in M$, $(a, a) \notin R$). For this reason, we also give the following definition:

Definition

Let R be a relation on a set M. If R is asymmetric and transitive, then R is called a **strict partial order**. *We say that (M, R) is a* **strict partially ordered set** *or* **strict poset**.

Note that if R is asymmetric, then for all $a \in M$, $(a, a) \notin R$.

Orders

Definition

Let (M, R) be a partially ordered set. If for all $x, y \in M$, $(x, y) \in R$ or $(y, x) \in R$, then R is called a **linear order** *or* **total order**, *and we say that (M, R) is a* **linearly ordered set** *or* **totally ordered set**.

Example

- *Let $A = \{1, 2, 3, 4\}$ and $R = \{(1, 1), (2, 2), (3, 3), (4, 4), (2, 1), (1, 3), (2, 3)\}$. Then R is a partial order but not a linear order*
- *If $S = \{(1, 1), (2, 2), (3, 3), (4, 4), (2, 1), (1, 3), (3, 2)\}$, then (A, S) is not a partially ordered set*
- *If $R' = \{(1, 1), (2, 2), (3, 3), (4, 4), (2, 1), (1, 3), (2, 3), (3, 4), (2, 4), (1, 4)\}$, then (A, R') is a linearly ordered set*
- *$(\mathbb{N}, \leq)$ is a linearly ordered set*
- *$R = \{(n, m) \in (\mathbb{N} \backslash \{0\})^2 \mid n \mid m\}$ is a partial order*

Orders

Definition

Let R be a linear order on a set M. We say that R is a **well-order** *if for all $A \subseteq M$, if $A \neq \emptyset$, then there exists $x \in A$, such that for all $y \in A$, if $(y, x) \in R$ then $y = x$. We also say that (M, R) is a* **well-ordered set**.

This says that every nonempty $A \subseteq M$ has a least element according to R.

Example

- *If R is a linear order on M and M is finite then R is a well-order*
- *The linear order $\leq$ on $\mathbb{N}$ is a well-order*
- *The linear order $\leq$ on $\mathbb{R}$ is not a well-order*
- *The linear order $\geq$ on $\mathbb{N}$ is not a well-order*

Note that there are corresponding definitions of **strict linear order** and **strict well-order** that impose additional conditions on a strict partial order in the same way that the definitions of linear order and well order impose additional conditions on a partial order.

Lattices

Definition

Let $(L, \preceq)$ be a poset and let $S \subseteq L$. We say that $x \in L$ is an **upper bound** *on S if for all $y \in S$, $y \preceq x$. We say that $x \in L$ is a* **lower bound** *on S if for all $y \in S$, $x \preceq y$.*

Definition

Let $(L, \preceq)$ be a poset and let $S \subseteq L$. We say that $x \in L$ is a **least upper bound (l.u.b.)** *on S if x is an upper bound on S and for all y, if y is an upper bound on S, then $x \preceq y$. We say that $x \in L$ is a* **greatest lower bound (g.l.b.)** *on S if x is a lower bound on S and for all y, if y is a lower bound on S then $y \preceq x$.*

Example

- *Consider $A = 2\mathbb{N} + 1$ as a subset of $(\mathbb{N}, \leq)$. A has no upper bounds. Both 0 and 1 are lower bounds on A and 1 is the greatest lower bound.*
- *Considering $\{2, 3\}$ as a subset of $(\mathbb{N}\backslash\{0\}, |)$, 1 is the greatest lower bound and 6 is the least upper bound.*

Lattices

Definition

Let $(L, \preceq)$ be a poset. We say that $(L, \preceq)$ is a **lattice** *if for all $x, y \in L$, the set $\{x, y\}$ has both a l.u.b. and a g.l.b. If $(L, \preceq)$ is a lattice and $x, y \in L$, then we write $x \vee y$ for the l.u.b. of $\{x, y\}$ and $x \wedge y$ for the g.l.b. of $\{x, y\}$.*

Example

- *The posets $(\mathbb{N}, \leq)$, $(\mathbb{Q}, \leq)$ and $(\mathbb{R}, \leq)$ are all lattices. In fact, if $\preceq$ is a linear order on M, then $(M, \preceq)$ is a lattice. In all these cases, if $x, y \in M$, then*

$$x \vee y = \max(x, y) \ and \ x \wedge y = \min(x, y)$$

- *The poset $(\mathbb{N}\backslash\{0\}, |)$ is a lattice. If $x, y \in \mathbb{N}\backslash\{0\}$ then $x \vee y$ is the least common multiple of x and y (denoted $\operatorname{lcm}(x, y)$) and $x \wedge y$ is the greatest common divisor of x and y (denoted $\gcd(x, y)$).*

Lattices

Example

- *If* $A = \{1, 2, 3, 4\}$ *and* $R = \{(1, 1), (2, 2), (3, 3), (4, 4), (1, 3), (1, 2), (2, 4), (3, 4), (1, 4)\}$, *then* (A, R) *is a lattice.*
- *If* $A = \{1, 2, 3, 4\}$ *and* $R = \{(1, 1), (2, 2), (3, 3), (4, 4), (1, 2), (2, 4), (3, 4), (1, 4)\}$, *then* (A, R) *is not a lattice because* $\{2, 3\}$ *has no lower bound.*
- *If* A *is a set, then* $(\mathcal{P}(A), \subseteq)$ *is a poset. Moreover,* $(\mathcal{P}(A), \subseteq)$ *is a lattice. If* $x, y \in \mathcal{P}(A)$, *then*

$$x \vee y = x \cup y \ \ and \ x \wedge y = x \cap y$$

Complete Lattices

Definition

Let $(L, \preceq)$ be a lattice. We say that $(L, \preceq)$ is **complete** *if for every $X \subseteq L$, X has both a least upper bound and a greatest lower bound. If $(L, \preceq)$ is a complete lattice and $X \subseteq L$, then we use $\bigvee X$ to denote the least upper bound of X and $\bigwedge X$ to denote the greatest lower bound of X.*

- If $(L, \preceq)$ is a nonempty finite lattice then $(L, \preceq)$ is complete
- If A is any set, then $(\mathcal{P}(A), \subseteq)$ is a complete lattice. If $X \subseteq \mathcal{P}(A)$, then

$$\bigvee X = \bigcup X \text{ and } \bigwedge X = \bigcap X$$

- The lattice $(\mathbb{Q}, \leq)$ is not complete. One reason is that $\{x \in \mathbb{Q} \mid x^2 \leq 2\}$ has no l.u.b.
- Is the lattice $(\mathbb{R}, \leq)$ complete?

Complete Lattices

- If $(L, \preceq)$ is a complete lattice, then $(L, \preceq)$ has a maximal element given by $\bigvee L$. This maximal element is sometimes denoted $\mathbb{1}$.
- If $(L, \preceq)$ is a complete lattice, then $(L, \preceq)$ has a minimal element given by $\bigwedge L$. This minimal element is sometimes denoted $\mathbb{0}$.

Example

- *The linear order* $(\mathbb{Z}, \leq)$ *can be turned into a complete lattice* $(\mathbb{Z}^*, \leq^*)$ *where*

$$\mathbb{Z}^* = \mathbb{Z} \cup \{-\infty, \infty\}$$
$$\leq^* = \{(-\infty, x), (x, \infty), (\infty, \infty), (-\infty, -\infty), (-\infty, \infty) \mid x \in \mathbb{Z}\} \cup \leq$$

  In this complete lattice $\mathbb{1} = \infty$ *and* $\mathbb{0} = -\infty$.
- *The lattice* $(\mathbb{N}\backslash\{0\}, |)$ *is not complete. However, if we define that* $n \mid 0$ *for all* $n \in \mathbb{N}$*, then* $(\mathbb{N}, |)$ *is a complete lattice. In this complete lattice* $\mathbb{1} = 0$ *and* $\mathbb{0} = 1$.

Chain Complete Posets

Definition

Let $(P, \preceq)$ be a partial order. We say that $X \subseteq P$ is a **chain** *if $(X, \preceq)$ is a linear order.*

- If $(P, \preceq)$ is a linear order, then every $X \subseteq P$ is a chain
- The set $X = \{3^n \mid n \in \mathbb{N}\}$ is a chain in the lattice $(\mathbb{N}, |)$

Definition

Let $(P, \preceq)$ be a partial order. We say that $(P, \preceq)$ is **chain complete** *if for all $X \subseteq P$, if X is a chain then X has a least upper bound.*

Note that again this definition ensures the existence of a unique least element. Why?

- What about greatest elements?

Chain Complete Posets

Example

- *Every complete lattice is a chain complete poset*
- *Let* $P = \mathbb{N} \cup \{\mathbf{a}, \mathbf{b}, \mathbf{c}\}$ *and define*

$$\begin{aligned} \preceq &= \{(n, n), (n, \mathbf{a}), (\mathbf{b}, n) \mid n \in 2\mathbb{N}\} \cup \\ &= \{(n, n), (n, \mathbf{a}), (\mathbf{c}, n) \mid n \in 2\mathbb{N} + 1\} \cup \\ &= \{(\mathbf{b}, \mathbf{a}), (\mathbf{c}, \mathbf{a})\} \end{aligned}$$

Then $(P, \preceq)$ *is NOT a chain complete poset, because the empty chain has no least upper bound.*

Functions

Definition

Let $f \subseteq A \times B$ (so f is a relation). We say that f is a function, and write $f : A \longrightarrow B$, if $\mathrm{dom}\ f = A$ and for all $x \in A$ and for all $y, z \in B$, if $(x, y) \in f$ and $(x, z) \in f$, then $y = z$.

- If f is a function, then f is a relation and the sets $\mathrm{dom}\ f$ and $\mathrm{ran}\ f$ are the same $\mathrm{dom}\ f$ and $\mathrm{ran}\ f$ that you saw in Calculus
- If $f : A \longrightarrow B$ is a function, then we often write $f(a) = b$ instead of $(a, b) \in f$ and use $f(a)$ to denote b– this makes sense because for all $a \in A$, if the exists $b \in B$ with $(a, b) \in f$, then this b is unique

Example

- *Let $A = \{1, 2, 3\}$. The relation $f = \{(1, 2), (2, 2), (3, 1)\}$ is a function, $f : A \longrightarrow A$, with $\mathrm{dom}\ f = A$ and $\mathrm{ran}\ f = \{1, 2\}$.*
- *The relation $f = \{(1, 3), (2, 3), (1, 2), (3, 1)\}$ is not a function*

Functions

Example

- *The relation* $f = \{(x, y) \in \mathbb{N}^2 \mid y = x \cdot x\}$ *is a function:* $f : \mathbb{N} \longrightarrow \mathbb{N}$ *When the domain and range of a function* f *are collections of numbers, then one sometimes expresses the relationship described by a function using formula (in the traditional mathematical sense) and variables. So* f *can be expressed as* $f(x) = x^2$.
- *The relation* $f = \{(x, y) \in \mathbb{R}^2 \mid x = y^4\}$ *is not a function because* $(1, -1) \in f$ *and* $(1, 1) \in f$

If $f : A \longrightarrow B$ is a function and $C \subseteq A$ then we use $f``C$ as shorthand for the set $\{y \mid \exists x(x \in C \wedge (x, y) \in f)\}$

Definition

Let $f : A \longrightarrow B$ *be a function. We say that* f *is* **injective** *or* **one-to-one** *if for all* $x, y \in A$ *and for all* $z \in B$, *if* $(x, z) \in f$ *and* $(y, z) \in f$, *then* $x = y$.

Injective Functions

Example

- *The function* $f = \{(1,3),(2,1),(3,2)\}$ *is injective*
- *The function* $f = \{(1,2),(2,1),(3,2)\}$ *is NOT injective*
- *The function* $f : \mathbb{R} \longrightarrow \mathbb{R}$ *defined by* $f(x) = x^3$ *is injective*
- *The function* $f : \mathbb{R} \longrightarrow \mathbb{R}$ *defined by* $f(x) = x^2$ *is NOT injective*
- *In fact, any (strictly) increasing function* $f : \mathbb{R} \longrightarrow \mathbb{R}$ *is injective*

Definition

Let f *and* g *be functions with* $\operatorname{ran} f \subseteq \operatorname{dom} g$*. Then define* $g \circ f$ *to be the relation*

$$g \circ f = \{(x,y) \mid \exists z((x,z) \in f \wedge (z,y) \in g)\}$$

Lemma

If f *and* g *are functions with* $\operatorname{ran} f \subseteq \operatorname{dom} g$ *then* $g \circ f$ *is a function*

Composing Functions

Using our notation for functions, the result of applying $g \circ f$ to a $((g \circ f)(a))$ is $g(f(a))$.

Example

- *If* $f = \{(1,2),(2,1),(3,2)\}$ *and* $g = \{(1,3),(2,1),(3,2)\}$, *then*

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

- *If* $f : \mathbb{R} \longrightarrow \mathbb{R}$ *is defined by* $f(x) = x + 6$ *and* $g : \mathbb{R} \longrightarrow \mathbb{R}$ *is defined by* $g(x) = x^2$, *then*

$$(g \circ f)(x) = (x+6)^2$$
$$(f \circ g)(x) = x^2 + 6$$

Inverses

Definition

Let $f \subseteq A \times B$ be a function. The **inverse** *of f, written f^{-1} is the relation*

$$f^{-1} = \{(x, y) \in B \times A \mid (y, x) \in f\}$$

Example

- *If* $f = \{(1, 2), (2, 1), (3, 2)\}$ *then* $f^{-1} = \{(2, 1), (1, 2), (2, 3)\}$
- *If* $f = \{(1, 3), (2, 1), (3, 2)\}$ *then* $f^{-1} = \{(3, 1), (1, 2), (2, 3)\}$
- *If* $f : \mathbb{R} \longrightarrow \mathbb{R}$ *is defined by* $f(x) = x^2$, *then*

$$f^{-1} = \{(x, y) \in \mathbb{R}^2 \mid y^2 = x\}$$

- *If* $f : \mathbb{R} \longrightarrow \mathbb{R}$ *is defined by* $f(x) = e^x$, *then*

$$f^{-1} = \{(x, y) \in [0, \infty) \times \mathbb{R} \mid y = \ln(x)\}$$

Inverse Functions

Definition

Let A be a set. The **identity function on** *A is the function $\mathrm{id}_A : A \longrightarrow A$ defined by*

$$\mathrm{id}_A = \{(x, y) \in A \times A \mid x = y\}$$

Lemma

Let $f : A \longrightarrow B$ be a function. The relation f^{-1} is a function with $\mathrm{dom}\ f^{-1} = \mathrm{ran}\ f$ and $\mathrm{ran}\ f^{-1} = A$ if and only if f is injective. Moreover, f^{-1} is injective and $f \circ f^{-1} = f^{-1} \circ f = \mathrm{id}_A$.

Example

If $f : [0, \infty) \longrightarrow \mathbb{R}$ is defined by $f(x) = x^2$, then $f^{-1} = \{(x, y) \in \mathbb{R}^2 \mid y^2 = x\}$, which is a function that we often write as $f^{-1}(x) = \sqrt{x}$, and $f(f^{-1}(x)) = f^{-1}(f(x)) = x$- i.e. $f \circ f^{-1} = f^{-1} \circ f = \mathrm{id}_{[0,\infty)}$.

Surjective Functions

Definition

Let $f : A \longrightarrow B$ be a function. We say that f is **surjective** *or* **onto** *if for all $x \in B$, there exists $y \in A$, such that $(y, x) \in f$.*

Definition

If a function $f : A \longrightarrow B$ is both injective and surjective, then we say that f is a bijection.

Note that the definition of a surjection makes reference to B where $f : A \longrightarrow B$. This is a bit strange, and it means that any function f is a function $f : \mathrm{dom}\ f \longrightarrow \mathrm{ran}\ f$ that is surjective.

- The function $f : \mathbb{R} \longrightarrow \mathbb{R}$ defined by $f(x) = e^x$ is not surjective.
- Let $A = \{1, 2, 3\}$. The function $f : A \longrightarrow A$ where $f = \{(1, 3), (2, 1), (3, 2)\}$ is surjective. Moreover, f is a bijection.

Cardinality Revisited

Lemma

If $f : A \longrightarrow B$ and $g : B \longrightarrow C$ are bijections, then $g \circ f$ is a bijection.

In an earlier slide I defined the cardinality or size of a set for finite sets. Is there a more general notion of size of a set?

Definition

Let A and B be sets. We say that A and B have the same **cardinality**, *and write $|A| = |B|$, if there exists a function $f : A \longrightarrow B$ that is a bijection.*

Definition

Let A and B be sets. We write $|A| \leq |B|$ if there exists a function $f : A \longrightarrow B$ that is an injection.

Note that if $|A| \leq |B|$, then $|A| = |C|$, for some $C \subseteq B$ (namely $C = \mathrm{ran}\ f$ where $f : A \longrightarrow B$ is an injection).

We are writing $\leq$ for the "less than or equals to" relation on cardinalities of sets, but be careful! Is this relationship a linear order? Is it even a partial order? Is it antisymmetric?

Cardinality

Example

(i) *The function* $f : \mathbb{N} \longrightarrow \mathbb{N}$ *defined by* $f(n) = 2n$ *shows that* $|\mathbb{N}| = |2\mathbb{N}|$

(ii) *The function* $f : \mathbb{N} \longrightarrow \mathbb{N}$ *defined by* $f(n) = n + 1$ *shows that* $|\mathbb{N}| = |\mathbb{N}\backslash\{0\}|$. *If the domain of* f *is restricted to the set* $2\mathbb{N}$ *(this is written* $f \restriction 2\mathbb{N}$*) then this function shows that* $|2\mathbb{N}| = |2\mathbb{N} + 1|$.

(iii) *Let* $A = \{1, 2, 4, 5\}$ *and* $B = \{a, b, c, d\}$. *The function* $f = \{(1, b), (2, a), (4, d), (5, c)\}$ *shows that* $|A| = |B|$.

(iv) *The function*

$$f(n) = \begin{cases} 0 & if\ n = 0 \\ n + 1 & if\ n > 0 \end{cases}$$

shows that $|\mathbb{N}| = |\mathbb{N}\backslash\{1\}|$

Cardinality

Example

(v) *Every nonzero integer (element of* $\mathbb{Z}\backslash\{0\}$*) has a unique representation in the form* $(-1)^k \cdot n$ *where* $k \in \{0,1\}$ *and* $n \in \mathbb{N}\backslash\{0\}$*. Moreover, any two distinct pairs* (k_1, n_1) *and* (k_2, n_2) *where* $k_1, k_2 \in \{0,1\}$ *and* $n_1, n_2 \in \mathbb{N}\backslash\{0\}$ *yield distinct nonzero integers* $(-1)^{k_1} \cdot n_1$ *and* $(-1)^{k_2} \cdot n_2$*. This means that the function*

$$f((-1)^k n) = \begin{cases} 0 & if\ n = 0 \\ 2n + k & if\ n \neq 0 \end{cases}$$

shows that $|\mathbb{Z}| = |\mathbb{N}\backslash\{1\}|$.

The examples (iv) and (v) above show the following:

Theorem

$|\mathbb{Z}| = |\mathbb{N}|$

Countable Sets

There are many senses in which the natural numbers ($\mathbb{N}$) are the smallest infinite set. This leads to the following definition:

Definition

Let A be a set. We say that A is **countable** *if $|A| \leq |\mathbb{N}|$. We say that A is* **countably infinite** *if A is countable and A is infinite.*

Note that there is a small gap here: I have not given you a formal definition of what it means to be infinite! I hope, however, that for the purposes of this course you will be able to recognise an infinite set when you see one.

Example

(i) *Any finite set is countable, but NOT countably infinite.*

(ii) *On the previous slide we showed that $\mathbb{Z}$ is countable, and $\mathbb{Z}$ is countably infinite*

(iii) *If $A \subseteq \mathbb{N}$, then A is countable. If A is also infinite, then A is countably infinite*

Countable Sets

Lemma

If $f : A \longrightarrow B$ and $g : B \longrightarrow C$ are injective functions, then $g \circ f$ is an injective function.

Lemma

If B is a countable set and $A \subseteq B$, then A is countable.

Example

If $A \subseteq \mathbb{Z}$, then A is countable.

Cantor's Pairing Function

Theorem

$|\mathbb{N} \times \mathbb{N}| = |\mathbb{N}|$

This is an important result and there are many proofs. Most of the proofs that I can think of, off the top of my head (as I am writing this), make use of the fact that the relation $\leq$ on cardinals is antisymmetric. However, there is also a function, known as Cantor's Pairing Function, that leads to a direct proof.

Definition

Cantor's Pairing Function *is the function* $\pi : \mathbb{N} \times \mathbb{N} \longrightarrow \mathbb{N}$ *defined by*

$$\pi(x, y) = \frac{1}{2}(x + y)(x + y + 1) + y$$

$$\pi(0,0) = 0 \qquad \pi(2,0) = 3$$
$$\pi(0,1) = 2 \qquad \pi(0,2) = 5$$
$$\pi(1,0) = 1 \qquad \pi(1,1) = 4$$

Cantor's Pairing Function

Theorem

Cantor's Pairing Function $\pi : \mathbb{N} \times \mathbb{N} \longrightarrow \mathbb{N}$ *is a bijection.*

We will not prove this Theorem here, and, as I mentioned, there are other proofs that $|\mathbb{N} \times \mathbb{N}| = |\mathbb{N}|$. However, the simplicity and elegance of Cantor's Pairing Function means that it has found applications in computer science and the study of the foundations of mathematics. Note that Cantor's pairing function is a degree 2 polynomial. In fact, it is known that it is the only degree 2 polynomial pairing function!

We have seen that $\mathbb{Z}$ and $\mathbb{N} \times \mathbb{N}$ are countable. In the assignments, hopefully, you will see that even $\mathbb{Q}$ is countable. Are there any sets that are not countable?

Definition

Let A and B be sets. We write $|A| < |B|$ if there exists a function $f : A \longrightarrow B$ that is an injection, and there does not exist a bijection $g : A \longrightarrow B$.

This does not say that there exists an injection that is not a bijection!

Cantor's Theorem

Theorem

If A is a set, then there is no injection $f : \mathcal{P}(A) \longrightarrow A$.

Proof.

This is a proof by contradiction. Let A be a set. Suppose that $f : \mathcal{P}(A) \longrightarrow A$ is an injection. Since f is an injection, $f^{-1} : \operatorname{ran} f \longrightarrow \mathcal{P}(A)$ is a bijection. Let $Z = \{x \in \operatorname{ran} f \mid x \notin f^{-1}(x)\}$. Note that $Z \subseteq A$, and let $z = f(Z)$.
Now, if $z \in f^{-1}(z) = Z$, then $z \notin f^{-1}(z)$, which is a contradiction. And, if $z \notin f^{-1}(z)$, then $z \in Z = f^{-1}(z)$, which is a contradiction. □

Corollary

If A is a set, then $|A| < |\mathcal{P}(A)|$.

Proof.

The function $f = \{(x, \{x\}) \in A \times \mathcal{P}(A) \mid x \in A\}$ is an injection. □

Uncountable Sets

In particular:

$$|\mathbb{N}| < |\mathcal{P}(\mathbb{N})| < |\mathcal{P}(\mathcal{P}(\mathbb{N}))| < \cdots$$

Definition

We say that a set is **uncountable** *if it is not countable.*

So, Cantor's Theorem gives us many examples of uncountable sets...

Morphisms and Isomorphisms

In this slide, rather than give you concrete definitions, I am going to introduce two concepts that occur repeated in different contexts in mathematics.

We have seen that two sets A and B are the same size (have the same cardinality) if there is a bijection $f : A \longrightarrow B$. Another way to think about this is that the abstract entities A and B are essentially the same — they are both sets, so they contain no information about repetition or order, and since there is a bijection between them, we can obtain one from the other by simply relabeling the elements. We have seen that by using relations we can represent more complicated mathematical structures— e.g. a poset is a set together with a relation that is reflexive, antisymmetric and transitive; (A, R). Now, suppose that we have two abstract structures (A, R) and (B, S) and a bijection $f : A \longrightarrow B$ such that for all $x, y \in A$,

$$(x, y) \in R \text{ if and only if } (f(x), f(y)) \in S$$

Then (A, R) and (B, S) are essentially the same structure – (B, S) can be obtained from (A, R) by replacing each $x \in A$ with $f(x)$ and replacing R by S.

Morphisms and Isomorphisms

If $\mathcal{R}$ and $\mathcal{S}$ are structures, and f is a bijection between the domains of $\mathcal{R}$ and $\mathcal{S}$ that preserves all of the structure associated with these domains, then f is called an **isomorphism** and the structures $\mathcal{R}$ and $\mathcal{S}$ are said to be **isomorphic**.

Mathematicians are often also interested in functions that preserve structure between one entity $\mathcal{R}$ and another entity $\mathcal{S}$, but are not necessarily bijections. These maps are called **morphisms** and in specific contexts **homomorphisms**. E.g. if (A, R) and (B, S) are structures where A and B are sets (the domains), and S and R are relations, then a homomorphism from (A, R) to (B, S) would be a function $f : A \longrightarrow B$ such that for all $x, y \in A$

$$\text{if } (x, y) \in R, \text{ then } (f(x), f(y)) \in S$$

As I said, here I am just introducing concepts rather than giving concrete definitions!

Order-preserving Functions

Definition

Let $(P_1, \preceq_1)$ and $(P_2, \preceq_2)$ be partial orders. We say that a function $f : P_1 \longrightarrow P_2$ is **order-preserving** *if for all $x, y \in P_1$,*

$$if\ x \preceq_1 y,\ then\ f(x) \preceq_2 f(y)$$

We might write $f : (P_1, \preceq_1) \longrightarrow (P_2, \preceq_2)$ when we need to specify which orders we are talking about.

Example

(i) *Let $a \in \mathbb{N}$ with $a \neq 0$. The function $f : \mathbb{N} \longrightarrow \mathbb{N}$ defined by $f(x) = ax$ is order-preserving from $(\mathbb{N}, |)$ to $(\mathbb{N}, |)$*

(ii) *If $f : \mathbb{R} \longrightarrow \mathbb{R}$ is an increasing function, then f is order-preserving from $(\mathbb{R}, \leq)$ to $(\mathbb{R}, \leq)$*

(iii) *The function $f : \mathbb{Z} \longrightarrow \mathbb{Z}$ given by $f(n) = n - 1$ is order-preserving from $(\mathbb{Z}, \leq)$ to $(\mathbb{Z}, \leq)$, but $g : \mathbb{Z} \longrightarrow \mathbb{Z}$ defined by $g(n) = -n$ is NOT*

Fixed points

Definition

Let A be a set and let $f : A \longrightarrow A$ be a function. We say that $x \in A$ is a **fixed point** *of f if $f(x) = x$.*

Example

(i) *The function $f : \mathbb{R} \longrightarrow \mathbb{R}$ defined by $f(x) = e^{-x}$ has fixed point somewhere in the interval $(0, 1)$ (this is an easy applictaion of the Intermediate Value Theorem)*

(ii) *Let $a \in \mathbb{N}$ with $a \neq 0$. The function $f : \mathbb{N} \longrightarrow \mathbb{N}$ defined by $f(n) = an$ has 1 fixed point 0 if $a \neq 1$, and infinitely many fixed points given by n for all $n \in \mathbb{N}$ if $a = 1$.*

(iii) *The function $f : \mathbb{Z} \longrightarrow \mathbb{Z}$ defined by $f(x) = x + 1$ has no fixed points.*

(iv) *The function $f : \mathcal{P}(\mathbb{N}) \longrightarrow \mathcal{P}(\mathbb{N})$ defined by $f(X) = X \backslash \{0\}$ has the property that if $A \subseteq \mathbb{N}$ is such that $0 \notin A$, then A is a fixed point of f.*

Tarski-Knaster Theorem

Theorem

(Tarski-Knaster) Let $(L, \preceq)$ *be a complete lattice. If*

$$f : (L, \preceq) \longrightarrow (L, \preceq)$$

is an order-preserving function, then f *has a fixed point.*

Corollary

Let $(L, \preceq)$ *be a complete lattice. If*

$$f : (L, \preceq) \longrightarrow (L, \preceq)$$

is an order-preserving function, then f *has a least fixed point.*

Schröder-Bernstein Theorem

Theorem

(Schröder-Bernstein) Let A and B be sets. If there exists $f : A \longrightarrow B$ that is injective and $g : B \longrightarrow A$ that is injective, then there exists a bijection $h : A \longrightarrow B$.

Corollary

The $\leq$ relation on cardinalities is antisymmetric. I.e. if $|A| \leq |B|$ and $|B| \leq |A|$, then $|A| = |B|$.

A flawed definition of $\mathbb{N}$

I have mentioned that set theory is an example of a foundational theory. There are plausible claims that it is capable of encoding or interpreting all of the mathematics that we use in our understanding of the world. In particular, we can define the natural number ($\mathbb{N}$), the rational numbers ($\mathbb{Q}$) and real number ($\mathbb{R}$) in set theory, and represent all of the operations and relations that we use in conjunction with these structures (such as $+$, $\cdot$ and $\leq$) using sets of ordered pairs. Any honest attempt to investigate the claim like "set theory is capable of interpreting all of the mathematics that we use in our understanding of the world" would need to work in an Axiomatic Set Theory, since Naive set theory is inconsistent. Despite the fact that we are working in an inconsistent theory, I will attempt to present a construction here that gives you an idea of how the natural numbers are defined in set theory.

A flawed definition of $\mathbb{N}$

- Imagine that we are working in pure set theory (the only objects are sets)
- Let V be the set of all sets and let L be the set of all sets that have $\emptyset$ as a member. I.e.

$$L = \{x \in V \mid \emptyset \in x\}$$

- $(L, \subseteq)$ is a complete lattice.
- Define the **successor operation** $S : V \longrightarrow V$ by

$$S(x) = x \cup \{x\} \text{ for all } x \in V$$

- Define $F : L \longrightarrow L$ such that for all $A \in L$,

$$F(A) = A \cup S\text{“}A$$

A flawed definition of $\mathbb{N}$

- Now, for all $A, B \in L$, if $A \subseteq B$, then $S\text{“}A \subseteq S\text{“}B$. So, F is an order-preserving function on the complete lattice $(L, \subseteq)$.
- Therefore, by the Tarski-Knaster Theorem, F has a least fixed point.
- Let $\mathbb{N}_{\text{def}}$ be the the least fixed point of F.
- So, what is the set $\mathbb{N}_{\text{def}}$? It is the $\subseteq$-least set X such that

$$\emptyset \in X, S(\emptyset) \in X, S(S(\emptyset)) \in X, \ldots$$

- Now, by defining:

$$\begin{aligned} 0 &:= \emptyset \\ 1 &:= S(\emptyset) = \{\emptyset\} \\ 2 &:= S(S(\emptyset)) = \{\emptyset, \{\emptyset\}\} \\ &\ldots \end{aligned}$$

the set $\mathbb{N}_{\text{def}}$ interprets the natural numbers.

A flawed definition of $\mathbb{N}$

Now, we have the set:

$$\mathbb{N}_{\text{def}} = \{\emptyset, \{\emptyset\}, \{\emptyset, \{\emptyset\}\}, \ldots\}$$

This defintion of the natural numbers has the nice property that for all $n \in \mathbb{N}_{\text{def}}$, the set n has n elements. This allows us to define $+ : \mathbb{N}_{\text{def}} \times \mathbb{N}_{\text{def}} \longrightarrow \mathbb{N}_{\text{def}}$ and $\cdot : \mathbb{N}_{\text{def}} \times \mathbb{N}_{\text{def}} \longrightarrow \mathbb{N}_{\text{def}}$ by: for all $n, m, k \in \mathbb{N}_{\text{def}}$

$$\begin{array}{lll} n = m \cdot k & \text{if and only if} & |n| = |k \times m| \\ n = m + k & \text{if and only if} & \text{there exists sets } A, B \\ & & \text{such that } A \cap B = \emptyset, \\ & & |A| = m \text{ and } |B| = k \\ & & \text{and } |A \cup B| = |n| \end{array}$$

It can be seen that these definitions mean that for all $n \in \mathbb{N}_{\text{def}}$, $S(n) = n + 1$.

A flawed definition of $\mathbb{N}$

What are the flaws?

- We defined $\mathbb{N}_{\text{def}}$ by appealing to the lattice of all sets x such that $\emptyset \in x$. In light of Russell's Paradox, we should be wary about appealling the "large" sets like this.
- Is the $\mathbb{N}_{\text{def}}$ that we obtain from this construction even infinite? I mean, are there even any infinite sets. In the context of Naive set theory, this problems seems almost impossible to even comprehend...

If we move to the axiomatic set theory known as ZFC, which is the the most popular axiomatisation of set theory, then this definition remains the same except for the following details:

- ZFC includes an axiom to ensure the existence of an infinite set
- the construction proceeds using a restricted, but sufficiently large set lattice $(L, \subseteq)$

A flawed definition of $\mathbb{N}$

All of the properties that we would expect of $\mathbb{N}$ are now provable properties of $\mathbb{N}_{\text{def}}$:

- The operations $+$ and $\cdot$ on $\mathbb{N}_{\text{def}}$ are commutative and associative, $\cdot$ distributes over $+$, and 0 and 1 are identities for $+$ and $\cdot$ respectively— all of these properties follow from facts about cardinalities of sets
- The order $\leq$ is a well ordering of $\mathbb{N}_{\text{def}}$— this follows from the fact that $\mathbb{N}_{\text{def}}$ is the least fixed point in the right lattice
- Every $n \in \mathbb{N}_{\text{def}}$ except 0 is the successor of some $k \in \mathbb{N}_{\text{def}}$, i.e. $n = k + 1$
- $\mathbb{N}_{\text{def}}$ satisfies the **principle of induction**: If a property $P(x)$ is such that $P(0)$ holds, and for all $n \in \mathbb{N}_{\text{def}}$, if $P(n)$ holds, then $P(n+1)$ holds, then for all $n \in \mathbb{N}_{\text{def}}$, $P(n)$ holds

Principle of Induction in $\mathbb{N}_{\text{def}}$

Why does the principle of induction hold for $\mathbb{N}_{\text{def}}$?

- Suppose the principle of induction does not hold.
- Let $P(x)$ be a property such that $P(0)$ holds, and for all $n \in \mathbb{N}_{\text{def}}$, if $P(n)$ hold, then $P(n+1)$ holds, but it is not the case that for all $n \in \mathbb{N}_{\text{def}}$, $P(n)$ holds
- Let $A = \{n \in \mathbb{N}_{\text{def}} \mid P(n)\}$, so

$$A \subset \mathbb{N}_{\text{def}} \text{ and } \emptyset \in A \text{ and}$$

$$\text{for all } n, \text{if } n \in A, \text{ then } S(n) \in A$$

- But this means that $S\text{“}A \subseteq A$ and so $A \cup S\text{“}A = A$, which contradicts the fact that $\mathbb{N}_{\text{def}}$ is a least fixed point for this operation

Induction Arguments

I have shown you how to define the natural numbers in set theory. One can also define other familiar mathematical structures in set theory such as the integers ($\mathbb{Z}$), the rationals ($\mathbb{Q}$) and the real numbers ($\mathbb{R}$). You can find the details in most introductory books on axiomatic set theory. You may, if you wish, now forget the definition of $\mathbb{N}_{\text{def}}$ and return to thinking of the natural numbers as the Platonic entities with all of the properties that you have become so familiar with over your years of studying and doing mathematics. For my part, I will return to simply referring to $\mathbb{N}$.

The principle of on induction gives us an important tool for proving new results. Let be $P(n)$ be a property. We can show that $P(n)$ holds for all $n \in \mathbb{N}$ using the following argument structure:

1. Show that $P(0)$ holds
2. Show that for arbitrary $n \in \mathbb{N}$, $P(n) \Rightarrow P(n+1)$, i.e. if $P(n)$ holds, then so does $P(n+1)$

It then follows, by the Principle of Induction, that for all $n \in \mathbb{N}$, $P(n)$ holds

Induction Arguments

This argument structure also work for showing that a property $P(n)$ holds for every $n \in \mathbb{N}$ with $n \geq n_0$. In this case:

1. Show that $P(n_0)$ holds
2. Show that for arbitrary $n \in \mathbb{N}$ with $n \geq n_0$, $P(n) \Rightarrow P(n+1)$, i.e. if $n \geq n_0$ and $P(n)$ holds, then so does $P(n+1)$

If we can do this, then it follows from the Principle of Induction that:

$$\{n \in \mathbb{N} \mid n < n_0\} \cup \{n \in \mathbb{N} \mid (n \geq n_0) \wedge P(n)\} = \mathbb{N}$$

Therefore, for all $n \in \mathbb{N}$ with $n \geq n_0$, $P(n)$ holds.

Examples of Induction Arguments

Theorem

For all $n \in \mathbb{N}$ with $n \geq 1$,

$$\sum_{k=1}^{n}(2k-1) = n^2$$

Proof.

We will prove this result by induction on the property $P(n)$ that says "$\sum_{k=1}^{n}(2k-1) = n^2$". To see that $P(1)$ holds, observe that

$$\sum_{k=1}^{1}(2k-1) = 2 \cdot 1 - 1 = 1 = 1^2$$

Now, let $n \geq 1$ and assume that $P(n)$ holds (we are proving the implication $P(n) \Rightarrow P(n+1)$). We need to prove that $P(n+1)$ holds.

Proof.

(Continued) We have

$$\sum_{k=1}^{n+1}(2k-1) = \sum_{k=1}^{n}(2k-1) + 2(n+1) - 1$$

And, since $P(n)$ holds,

$$\sum_{k=1}^{n+1}(2k-1) = n^2 + 2(n+1) - 1 = n^2 + 2n + 1 = (n+1)^2$$

And the result now follows by induction. □

Link between induction and the well-orderedness of $(\mathbb{N}, \leq)$

There is an intimate link between the Principle of Induction and the fact $(\mathbb{N}, \leq)$ is a well-order. Sometimes it is convientient to phase induction arguments in terms of this well order. The following structure outlines a proof by a contradiction that a property $P(n)$ holds for all $n \in \mathbb{N}$ with $n \geq n_0$:

1. Show that $P(n_0)$ holds
2. Suppose that $\{n \in \mathbb{N} \mid n \geq n_0 \wedge \neg P(n)\}$ is nonempty and let n' be the least element of this set
3. Let $m \geq n_0$ be such that $n' = m + 1$
4. Show that the fact that $P(m)$ holds implies that $P(n')$ holds, thus obtaining a contradiction

Example of Induction

Theorem

Let $(L, \preceq)$ be a lattice. If $X \subseteq L$ is finite with $|X| \geq 2$, then X has a least upper bound.

Proof.

We will prove this result by induction on the property $P(n)$ that says "every $X \subseteq L$ with $|X| = n$ has a least upper bound". It is clear that $P(2)$ holds- this is the definition of a lattice! Suppose that $m > 2$ is least such that there exists

$$X = \{x_1, \ldots, x_m\} \subseteq L$$

and X does not have a least upper bound. Since m is least, $X' = \{x_1, \ldots x_{m-1}\}$ has a least upper bound y. And, since $(L, \preceq)$ is a lattice, $y \vee x_m$ exists. Moreover, $y \vee x_m$ is an upper bound for X. Now, if u is an upper bound for X, then u is an upper bound for X' and so $y \preceq u$. And $x_m \preceq u$, so u is an upper bound for $\{y, x_m\}$. But this means that $y \vee x_m \preceq u$, and we have shown that $y \vee x_m$ is a least upper bound for X, which is a contradiction. $\square$

Strong Induction

Sometimes induction arguments take a form that is referred to as **strong induction**. An argument by strong induction that shows that a property $A(n)$ holds for all $n \in \mathbb{N}$ with $n \geq n_0$ proceeds as follows:

(i) Show that $A(n_0)$ holds

(ii) Show that for all $n \geq n_0$, if for all $n_0 \leq k \leq n$, $A(k)$ holds, then $A(n+1)$ holds

(iii) Conclude that for all $n \in \mathbb{N}$ with $n \geq n_0$, $A(n)$ holds

Step (ii) appears to be easier that the corresponding step in a usual induction argument because we are allowed to assume that $A(k)$ holds for all $n_0 \leq k \leq A(n)$. However, the validity of strong induction follows from the principle of induction. In fact, the above argument is just an argument by induction on the property $P(n)$ that says "for all $n_0 \leq k \leq n$, $A(k)$ holds". Note that $P(n_0)$ just says that $A(n_0)$ holds. And, if $P(n)$ holds, then $P(n+1)$ follows from $A(n+1)$.

Example of Strong Induction

Theorem

For all $n \in \mathbb{N}$ with $n \geq 2$, n is either prime or the product of primes.

Proof.

We proceed by strong induction on property $A(n)$ that say "n is prime or the product of primes". $A(2)$ holds, because 2 is prime. Let $n \in \mathbb{N}$ with $n \geq 2$ and assume that for all $2 \leq k \leq n$, k is prime or the product of primes. Now, consider $n+1$. If $n+1$ is prime, then we are done. So, suppose $n+1$ is not prime and therefore $n+1 = a \cdot b$ where $2 \leq a, b \leq n$. But now, by the strong induction hypothesis, both a and b are either prime or the product of primes. So, $n+1$ is the product of primes. The theorem now follows by (strong) induction. $\square$

Recursive Definitions

The "inductive" property of the natural numbers also allows us to define functions on $\mathbb{N}$. We can define a function $f : \mathbb{N} \longrightarrow \mathbb{N}$ by specifying:

(i) The value of $f(0)$ and maybe some other initial values of f such as $f(1)$

(ii) A rule that allows us to obtain the value of $f(n+1)$ from the values of $f(n)$, $f(n-1)$, ...

A definition in this form is called a **recursive definition**.

Example

- *The* **factorial function** $(\cdot)! : \mathbb{N} \longrightarrow \mathbb{N}$ *is defined by* $0! = 1$ *and for all* $n \in \mathbb{N}$ *with* $n > 0$, $n! = n(n-1)!$
- *The* **Fibonacci sequence** $f : \mathbb{N} \longrightarrow \mathbb{N}$ *is defined by* $f(0) = 0$ *and* $f(1) = 1$, *and for all* $n \in \mathbb{N}$ *with* $n \geq 2$, $f(n) = f(n-1) + f(n-2)$

Note that if we are given a recursive definition of a function $f : \mathbb{N} \longrightarrow \mathbb{N}$, then we can prove by induction on the property $P(n)$ saying "f is uniquely defined on every input $k \leq n$" that f is defined uniquely.

Recursively Defined Functions

But... Previously I told you that functions are sets, and sets are collections of objects satisfying a property. So, what is the property that specifies the ordered pairs in function defined by a recursive definition? Such a property can be found, but one can also "get hold" of the set corresponding to a recursively defined function using the Tarski-Knaster Theorem.

Suppose that a function $f : \mathbb{N} \longrightarrow \mathbb{N}$ is defined by: $f(0) = n_0$ and $(n+1, f(n+1)) = G(n, f(n))$. I.e. $G(\cdot, \cdot)$ is a rule that takes inputs n and $f(n)$ and outputs $(n+1, f(n+1))$, and we can think of G as a function $G : \mathbb{N} \times \mathbb{N} \longrightarrow \mathbb{N} \times \mathbb{N}$.

- Let $X = \{R \in \mathcal{P}(\mathbb{N} \times \mathbb{N}) \mid (0, n_0) \in R\}$
- It should be clear that $(X, \subseteq)$ is a complete lattice
- Define $F : X \longrightarrow X$ by $F(R) = R \cup G\text{“}R$
- Now, $F : (X, \subseteq) \longrightarrow (X, \subseteq)$ is order preserving (check), and so, by the Tarski-Knaster Theorem, there exists a $\subseteq$-least f in X such that $F(f) = f$
- I will leave it to you to convince yourself that this f is the function given by the recursive definition

More General Recursive Definitions

This realisation of a recursively defined function as the least fixed point of an order preserving function on a complete lattice should make you realise that that you have seen something that looks like a recursive definition before. Recall that, in set theory, the natural numbers were defined to be the $\subseteq$-least set $\mathbb{N}_{\text{def}}$ such that $0 \in \mathbb{N}_{\text{def}}$ and for all n, if $n \in \mathbb{N}_{\text{def}}$, then $n + 1 \in \mathbb{N}_{\text{def}}$ (this is what it meant to be a fixed point of the successor operation). Moreover, it was this recursive definition that gave us the Principle of Induction.

We are now in a position to generalise the notion of a recursive definition and in doing so obatining a more general Principle of Induction. Suppose that $C_1, \ldots, C_n$ are a finite list of rules that allow us to build new objects from existing objects (C. is for constructor). And suppose that B is some collection of basic objects. We can define a set A to be the $\subseteq$-least set such that $B \subseteq A$ and A is closed under applications of the rules $C_1, \ldots, C_n$. In other words, if $a_1, \ldots, a_m \in A$ and b can be obtained by applying the rules $C_1, \ldots, C_n$ to $a_1, \ldots, a_m$, then $b \in A$. We call such a definition a **recursive definition** of the set A.

Recursively Defined Sets and Structural Induction

- It should be reasonably clear that if A is recursively defined to be the $\subseteq$-least set such that $B \subseteq A$ and A is closed under applications of the rules $C_1, \ldots, C_n$, then A can be realised, using the Tarski-Knaster Theorem, as the least fixed point of an order preserving function on a complete lattice
- Moreover, the fact that A is the $\subseteq$-least set that conatins B and is closed under the rules $C_1, \ldots, C_n$ means that every element of A can be obatined by repeated applications of the rules $C_1, \ldots, C_n$ to elements of B
- This is analogous to natural number being the $\subseteq$-least set that is closed under the "successor" rule

Principle of Structural Induction

Principle of Structural Induction: Let B be a set and let $C_1, \ldots, C_n$ be construction rules. Let A be recursively defined to be the $\subseteq$-least set such that $B \subseteq A$ and A is closed under the rules $C_1, \ldots, C_n$. Let $P(x)$ be a property. If

(i) for all $b \in B$, $P(b)$ holds

(ii) for all $a_1, \ldots a_m$ and c and $1 \leq i \leq n$, if $P(a_1), \ldots, P(a_m)$ all hold and c is obtained from $a_1, \ldots, a_m$ by a single application of the rule C_i, then $P(c)$ holds

Then $P(x)$ holds for every element of A.

$\subseteq$-least Property of Recursively Defined Sets: Let B be a set and let $C_1, \ldots, C_n$ be construction rules. Let A be recursively defined to be the $\subseteq$-least set such that $B \subseteq A$ and A is closed under the rules $C_1, \ldots, C_n$. If X is such that $B \subseteq X$ and X is closed under the rules $C_1, \ldots, C_n$, then, since A is $\subseteq$-least, $A \subseteq X$.

Recursively Defined Sets

Example

- *The collection of well-formed compound propositional expressions formed from a set of atomic propositions* $B = \{A_1, A_2, \ldots\}$ *can be defined to be the* $\subseteq$*-least collection of compound propositions that contains* B *and such that if* P *and* Q *are well-formed compound propositional expressions, then so are* $\neg P$, $P \vee Q$, $P \wedge Q$, $P \Rightarrow Q$ *and* $P \iff Q$.
- *Let* $S \subseteq \mathbb{N}$ *be the* $\subseteq$*-least set such that* $3 \in S$ *and if* $x, y \in S$, *then* $x + y \in S$. *Then* $S = \{n \in \mathbb{N} \mid 3 \mid n\}$.

Proof.

Let $X = \{n \in \mathbb{N} \mid 3 \mid n\}$. *The fact that* $S \subseteq X$ *follows from the fact that* S *is* $\subseteq$*-least:* $3 \in X$ *and for all* $x, y \in X$, $x + y \in X$. *The fact that* $X \subseteq S$ *follows by induction:* $X = \{n \in \mathbb{N} \mid (\exists k \in \mathbb{N} \backslash \{0\})(n = 3k)\}$. *Now,* $3 \in S$ *and if* $3k \in S$, *then* $3(k + 1) = 3k + 3 \in S$. □

A question from Assignment 1

Theorem

Let $B = \{A_1, A_2, \ldots\}$ be a set of atomic propositions. Every well-formed compound propositional expression formed from atomic propositions in B is logically equivalent to a compound expression that only involves atomic propositions from B and the connectives $\vee$ and $\neg$.

Proof.

We prove this result by structural induction. The result clearly holds for every atomic proposition in B. Suppose that the result holds for the compound expressions P and Q. Let P' and Q' be compound expressions involving only atomic propositions in B and the connectives $\vee$ and $\neg$ such that $P \equiv P'$ and $Q \equiv Q'$. Now,

$$\neg P \equiv \neg P'$$

$$P \vee Q \equiv P' \vee Q'$$

A question from Assignment 1

Proof.

$$P \wedge Q \equiv \neg(\neg P' \vee \neg Q')$$

$$P \Rightarrow Q \equiv \neg P' \vee Q'$$

$$P \iff Q \equiv (P \Rightarrow Q) \wedge (Q \Rightarrow P) \equiv \neg(\neg(\neg P' \vee Q') \vee \neg(\neg Q' \vee P'))$$

We have shown that if the Theorem holds for the compound expressions P and Q, then the theorem also holds for $\neg P$, $P \vee Q$, $P \wedge Q$, $P \Rightarrow Q$ and $P \iff Q$. Therefore the result follows by structural induction. $\square$

Binomial Theorem

Definition

Let A be a finite set and let $0 \leq k \leq |A|$. Define

$$\mathcal{P}_k(A) = \{x \in \mathcal{P}(A) \mid |x| = k\}$$

For all $n \in \mathbb{N}\backslash\{0\}$, we will use $[n]$ to denote the set

$$\{0, \ldots, n-1\}$$

and $[0] = \emptyset$. $[n]$ will be the canonical set of size n.

Definition

Let $n \in \mathbb{N}$ and $0 \leq k \leq n$. Define $\binom{n}{k}$ to be the cardinality of the set $\mathcal{P}_k([n])$.

- For all $n \in \mathbb{N}$, $\binom{n}{0} = \binom{n}{n} = 1$

Binomial Theorem

- For all $n \in \mathbb{N}$, $\binom{n}{1} = n$
- What about $\binom{n}{2}$?

Lemma

For all $n \in \mathbb{N}$ and for all $0 \leq k \leq n$, $\binom{n}{k} = \binom{n}{n-k}$

Proof.

The function $F : \mathcal{P}_k([n]) \longrightarrow \mathcal{P}_{n-k}([n])$ defined by: for all $x \in \mathcal{P}_k([n])$, $F(x) = [n] \backslash x$ is a bijection. $\square$

Binomial Theorem

Theorem

For all $n \in \mathbb{N}$ with $n \geq 1$ and for all $0 < k \leq n$,

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

Proof.

Let

$$A = \{x \cup \{n\} \mid x \in \mathcal{P}_{k-1}([n])\} \text{ and } B = \mathcal{P}_k([n])$$

Note that $A \cap B = \emptyset$, because n is not a member of any element of B. Moreover, $|A| = \binom{n}{k-1}$ and $|B| = \binom{n}{k}$. Now, since $A \cup B = \mathcal{P}_k([n+1])$, we have

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

□

Pascal's Triangle

This gives us a recursive definition of $\binom{n}{k}$.

$n = 0$					1					
$n = 1$				1		1				
$n = 2$			1		2		1			
$n = 3$		1		3		3		1		
$n = 4$	1		4		6		4		1	

This table is called **Pascal's Triangle**

Binomial Theorem

Theorem

(Binomial Theorem) For all $n \in \mathbb{N}$ with $n \geq 1$ and for all numbers x and y,

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

Proof.

We will prove this result by induction on the property $P(n)$ that says

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

Note that

$$(x+y)^1 = x + y = \binom{1}{0} x + \binom{1}{1} y$$

and so $P(1)$ holds.

Binomial Theorem

Proof.

(Continued) Assume that $P(n)$ holds. We need to prove that $P(n+1)$ holds.

$$(x+y)^{n+1} = (x+y)(x+y)^n = (x+y)\left(\sum_{k=0}^{n}\binom{n}{k}x^{n-k}y^k\right)$$

since $P(n)$ holds. Now, we expand the right hand side of this equation and analse the coefficients of each of the $x^p y^q$ terms where $0 \leq p, q \leq n+1$. The coefficients of the terms x^{n+1} and y^{n+1} are both 1, which is equal to $\binom{n+1}{0}$ and $\binom{n+1}{n+1}$. For all $0 < k \leq n$, the coefficient of the term $x^{n+1-k}y^k$ is

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

Binomial Theorem

Proof.

(Continued.) Putting this together, we get:

$$(x+y)^{n+1} = \sum_{k=0}^{n+1} \binom{n+1}{k} x^{n+1-k} y^k$$

which shows that $P(n+1)$ holds. This proves the induction step, and the result now follows by induction. $\square$

Groups

Definition

A **group** *is a pair* $(G, \cdot)$ *where* G *is a set and* $\cdot : G \times G \longrightarrow G$*, called the* **group operation** *and pronounced as "the product", that satisfies:*

(i) *for all* $x, y, z \in G$*,*

$$x \cdot (y \cdot z) = (x \cdot y) \cdot z \ \textit{(Associativity)}$$

(ii) *there exists* $e \in G$*, called an* **identity***, such that for all* $x \in G$*,*

$$x \cdot e = e \cdot x = x$$

and for all $x \in G$*, there exists* $y \in G$ *such that*

$$x \cdot y = y \cdot x = e$$

where the element y *is called an* **inverse** *of* x

Note that we are writing $x \cdot y$ *insetead of* $\cdot(x, y)$

Groups

Lemma

Let $(G, \cdot)$ be a group. The element $e \in G$ such that for all $x \in G$, $x \cdot e = e \cdot x = x$ is unique. Moreover, for all $x \in G$, there exists a unique $y \in G$ such that $x \cdot y = y \cdot x = e$.

In light of this result, we often use e_G or just e to denote the unique identity element in a group $(G, \cdot)$. If $(G, \cdot)$ is a group and $x \in G$, then we will write x^{-1} for the unique inverse of x in $(G, \cdot)$.

Definition

Let $(G, \cdot)$ be a group. We say that $(G, \cdot)$ is **abelian**, *if for all $x, y \in G$, $x \cdot y = y \cdot x$.*

Example

- $(\mathbb{Z}, +)$ - *the integers with the operation addition- is an abelian group*
- $(\mathbb{Z}, \cdot)$ - *the integers with the operation multiplication- is not a group, because there is no identity*

Examples of Groups

Example

- $(\mathbb{Q}\backslash\{0\}, \cdot)$- *the rationals (without zero) with multiplication- is an abelian group*
- *Let* $X = \{f : \mathbb{N} \longrightarrow \mathbb{N} \mid f \ is\ a\ bijection\}$. *Then* $(X, \circ)$ *is a group. Consider*

$$f(n) = \begin{cases} 0 & n = 1 \\ 1 & n = 0 \\ n & n \neq 0, 1 \end{cases} \quad and\ g(n) = \begin{cases} 1 & n = 2 \\ 2 & n = 1 \\ n & n \neq 1, 2 \end{cases}$$

  $g \circ f(1) = 0$ *and* $f \circ g(1) = 2$, *so* $f \circ g \neq g \circ f$ *and* $(X, \circ)$ *is not abelian*
- *Let* $X = \{f : \mathbb{R} \longrightarrow \mathbb{R} \mid f \ is\ linear\}$. *Then* $(X, \circ)$ *is a group. To see that* $(X, \circ)$ *is not abelian consider* $f(x) = x + 1$ *and* $g(x) = 2x$. *However, if* $X' = \{f \in X \mid f(0) = 0\}$, *then* $(X', \circ)$ *is an abelian group.*

Algebra in Groups

We are all familiar with performing algebraic manipulations on numbers. For example, if $a, b \in \mathbb{Z}$ and $4 + a = 4 + b$, then we know that $a = b$. Or, if $a, b \in \mathbb{Q}\backslash\{0\}$ and $\frac{1}{3}a = \frac{1}{3}b$, then $a = b$. By studying groups we are able to study a broad range of mathematical entities that are endowed with the properties that allow us to perform such algebraic manipulations. We are able to identify the properties that give rise certain behaviour that we are familiar with when we are dealing with numbers, and identify this behaviour in other mathematical structures.

Lemma

Let $(G, \cdot)$ be a group. If $a, b, c \in G$ and $a \cdot b = a \cdot c$, then $b = c$

Proof.

Let $a, b, c \in G$ and suppose that $a \cdot b = a \cdot c$. Now,

$$b = e \cdot b = (a^{-1} \cdot a) \cdot b = a^{-1} \cdot (a \cdot b) = a^{-1} \cdot (a \cdot c) = (a^{-1} \cdot a) \cdot c = e \cdot c = c$$

□

Algebra in Groups

Corollary

Let $(G, \cdot)$ be a group and $a \in G$. If $a \cdot a = a$, then $a = e$.

Example

Let $(G, \cdot)$ be a group and let $x \in G$. We know that there exists $y \in G$ such that $x \cdot y = y \cdot x = e$. However, this clause is partially redundant. If $y \in G$ is such that $y \cdot x = e$, then $x \cdot y = e$.

Proof.

Let $y \in G$ be such that $y \cdot x = e$. Now,

$$\begin{aligned} x \cdot y &= (x \cdot y) \cdot e = x \cdot (y \cdot e) \\ &= x \cdot (e \cdot y) = x \cdot ((y \cdot x) \cdot y) \\ &= x \cdot (y \cdot (x \cdot y)) = (x \cdot y) \cdot (x \cdot y) \end{aligned}$$

and so it follows that $x \cdot y = e$ □

The Symmetric Group

Definition

Let $X = \{f : [n] \longrightarrow [n] \mid f \ is\ a\ bijection\}$. *The group* $(X, \circ)$ *is call the* **symmetric group on** n **elements** *and is written* S_n.

The symmetric groups are extremely important in the study of finite groups and groups in general. In fact, the symmetric groups are, in some sense, universal: Every finite group "appears" as part of an S_n for some n.

The definition of S_n that I have given you is very cumbersome and it is necessary to introduce some special notation that allows us to better visualise and deal with the elements of S_n in the context of studying groups.

Definition

(Cycle Notation) Let $n \in \mathbb{N}$. *Let* $m \leq n$ *and let* $k_1, \ldots, k_m < n$ *all distinct. We use* $(k_1 k_2 \cdots k_m)$ *to denote the bijection* $f : [n] \longrightarrow [n]$ *defined by*

$$f(x) = \begin{cases} f(k_i) = k_{i+1} & if\ i < m \\ f(k_m) = k_1 & \\ f(x) = x & if\ x\ is\ not\ any\ of\ the\ k_i s \end{cases}$$

Cycles

We will use juxtaposition instead of the symbol $\circ$ to represent composition. I.e. we write $(k_1 \cdots k_m)(p_1 \cdots p_q)$ instead of $(k_1 \cdots k_m) \circ (p_1 \cdots p_q)$.

Example

- *In* S_4 (102) *is the bijection*

$$0 \mapsto 2$$
$$1 \mapsto 0$$
$$2 \mapsto 1$$
$$3 \mapsto 3$$

and this bijection can also be written as (021), (210) *and* $(3)(021)$ *...*

Cycles

Example

- *In* S_4 $(132)(21)$ *is the bijection*

$$0 \mapsto 0$$
$$1 \mapsto 1$$
$$2 \mapsto 3$$
$$3 \mapsto 2$$

  and this bijection can also be written as (23), (32) *and* $(23)(01)(10)\ldots$ *(Remember that composition acts on the right first, so cycles need to be read from right to left)*
- *In* S_n *the identity bijection* $\mathrm{id}_{[n]} : [n] \longrightarrow [n]$ *can be written as* (k) *for any* $k \in [n]$*, or, if you like, as the empty cycle* $()$*.* $\mathrm{id}_{[n]}$ *is the identity element of* S_n*, so this bijection can also be written as* e_{S_n} *or just* e *when the context is clear.*

Cycles

Example

- *In* S_4 $(12)(01)(12)$ *is the bijection*

$$0 \mapsto 2$$
$$1 \mapsto 1$$
$$2 \mapsto 0$$
$$3 \mapsto 3$$

and this bijection can also be written as (02) *and* $(03)(32)(03)\ldots$

Cycles

Cycle notation given us an easy way of writing down and computing the products of elements in the groups S_n.

Example

- *The product of* $(023)(13)$ *and* $(12)(13)$ *in* S_4 *is the bijection* $(023)(13)(12)(13)$. *Reading from right to left, this is the bijection that sends:* $0 \mapsto 2$, $1 \mapsto 1$, $2 \mapsto 0$ *and* $3 \mapsto 3$. *So, we can see that* $(023)(13)(12)(13) = (02)$
- *The product of* $(12)(13)$ *and* $(023)(13)$ *in* S_4 *is the bijection* $(12)(13)(023)(13)$. *Reading from right to left, this is the bijection that sends:* $0 \mapsto 1$, $1 \mapsto 0$, $2 \mapsto 2$ *and* $3 \mapsto 3$. *So, we can see that* $(12)(13)(023)(13) = (01)$. *These two examples show that* S_4 *is NOT an abelian group.*
- *The product of* (124) *and* (142) *in* S_5 *is* $(124)(142)$. *Reading from right to left, this is the bijection that sends:* $0 \mapsto 0$, $1 \mapsto 1$, $2 \mapsto 2$, $3 \mapsto 3$ *and* $4 \mapsto 4$. *So* $(124)(142) = e$ *in* S_5, *i.e.* (142) *is the inverse of* (124) *in* S_5

The Symmetric Group

Theorem

Let $n \in \mathbb{N}\backslash\{0\}$. The group S_n is not abelian if and only if $n \geq 3$.

Proof.

$\Leftarrow$: Suppose $n \geq 3$. In S_n the product of (01) and (012) is $(01)(012) = (12)$, and the product of (012) and (01) is $(012)(01) = (02)$. Therefore $(01)(012) \neq (012)(01)$, and so S_n is not abelian.

$\Rightarrow$: We prove the contrapositive, i.e. "if $0 < n < 3$, then S_n is abelian". This is just an argument by "brute force". S_1 is the group that only contains one element– the identity – and so S_1 is an abelian group. S_2 contains exactly two elements: the identity and the bijection (01). It is easy to check that this is an abelian group. □

Note that if $(G, \cdot)$ is group, then $G \neq \emptyset$, because G must have an identity element $e \in G$. In the above proof, we see that S_1 is a group whose only element is the identity element, i.e. $S_1 = \{e\}$. We call S_1, and any other group that consists only of an identity element, the **trivial group**.

Cycles

We have seen that the representation of bijections using cycles is not unique. It is often useful to choose cycles with certain properties to represent the bijections that we are considering.

Definition

Let $k_1, \ldots, k_m$ all be distinct natural numbers. We say that $(k_1 k_2 \cdots k_m)$ is a cycle of **length** *m, or that $(k_1 k_2 \cdots k_m)$ is an m-***cycle***.*

For example: (023) is a 3-cycle and (0342) is a 4-cycle

Definition

We say that cycles $(k_1 \cdots k_m)$ and $(p_1 \cdots p_q)$ are **disjoint** *if $\{k_1, \ldots, k_m\} \cap \{p_1, \ldots, p_q\} = \emptyset$*

For example: (01) and (243) are disjoint, and (213) and (1567) are not disjoint

Lemma

If α and β are disjoint cycles then $\alpha\beta = \beta\alpha$ in S_n

Cycles

Theorem

Every element of S_n can be written as a product of disjoint cycles.

Proof.

This proof gives a procedure or "algorithm" for writing an element of S_n as a product of disjoint cycles an argues that this algorithm works. Let $f : [n] \longrightarrow [n]$ be a bijection. If f is the identity, then $f = (0)$, and we are done. So assume that f is not the identity and let $x_0 \in [n]$ be such that $f(x_0) \neq x_0$. We will write $f^q(x)$ for the result of applying f q times to x. Let $k_0 = x_0$, and let $k_1 = f(x_0)$. Continue applying f, obtaining successive $k_m = f^m(x_0)$, until $f^q(x_0) = x_0$. This must always eventually happen, otherwise $f^q(x_0) = f^p(x_0)$ for some least $0 < p < q$. And this would mean that $f(f^{q-1}(x_0)) = f(f^{p-1}(x_0))$ with $f^{q-1}(x_0) \neq f^{p-1}(x_0)$, which would contradict the fact that f is a bijection. This process yields a cycle $(k_0 \cdots k_{q-1})$ that sends the elements $k_0, \ldots, k_{q-1}$ to the same things that f does.

Cycles

Proof.

(Continued.) Now, if there exists $x_1 \in [n]$ such that for all $0 \leq i \leq q-1$, $x_1 \neq k_i$, and $f(x_1) \neq x_1$, then repeat this process. The result of applying f to x_1 with never yield one of the ks, because f is a bijection. So, this process will eventually yield a second cycle that is disjoint from the first. Continue this process until all the elements of $[n]$ have been dealt with. □

This proof gives us a procedure that allows us to transform any bijection into a product of disjoint cycles.

Example

- $(124)(352) = (12354)$
- $(05)(132)(21)(143)(560) = (1423)(56) = (56)(1423)$
- $(45)(12)(31)(54)(02)(32)(45) = (013)(45) = (45)(013)$

Cycles

Theorem

Let $n \geq 2$. Every element of S_n can be written as the product of 2-cycles.

Proof.

It suffices to show that every cycle $(k_1 \cdots k_m)$ with $m \geq 2$ can be written as the product of 2-cycles. We do this by induction on m. The result clearly holds for $m = 2$. Suppose that the cycle $(k_1 \cdots k_m)$ can be written as the product of 2-cycles. The fact that $(k_1 \cdots k_{m+1})$ can be written as the product of 2-cycles now follows from the observation that

$$(k_1 \cdots k_{m+1}) = (k_1 k_{m+1})(k_1 \cdots k_m)$$

□

We can "unravel" this induction argument to obtain a procedure for writing bijections as the product of 2-cycles.

Example

$(2143) = (23)(214) = (23)(24)(21)$

Cycles

Example

$(0213)(123) = (03)(01)(02)(13)(12) = (02)$

Definition

Let $\sigma \in S_n$. If σ can be written as a product of an odd number of 2-cycles, then we say that σ is **odd**. *If σ can be written as a product of an even number of 2-cycles, then we say that σ is* **even**.

Theorem

Every element of S_n is either even or odd, but not both.

Example

- $(1032) = (12)(13)(10)$, *so* (1032) *is odd*
- *The identity is even*

Order

Definition

Let $(G, \cdot)$ be a group and let $x \in G$. For $n \in \mathbb{N}$, recursively define x^n by: $x^0 = e$ and $x^{n+1} = x \cdot x^n$.

Be careful, this means that in the group $(\mathbb{Z}, +)$ we are writing 1^n for $\underbrace{1 + \cdots + 1}_{n \text{ times}}$

Example

- *In the group $(\mathbb{Z}, +)$, $2^4 = 2 + 2 + 2 + 2 = 8$*
- *In S_4, $(012)^3 = (012)(012)(012) = e$*
- *In the group $(\mathbb{Q}\backslash\{0\}, \cdot)$, $(\frac{1}{2})^3 = \frac{1}{2} \cdot \frac{1}{2} \cdot \frac{1}{2} = \frac{1}{8}$*

Definition

Let $(G, \cdot)$ be a group and let $x \in G$. If there exists an $n \geq 1$ such that $x^n = e$, then we say that x has **finite order** *and we define the* **order** *of x to be the least $n \geq 1$ such that $x^n = e$. If x does not have finite order, then we say that x has* **infinite order**.

Order

Example

- *We have seen that* $(012)^3 = e$ *in* S_4*. Checking that* $(012)(012) = (021) \neq e$ *confirms that the order of* (012) *in* S_4 *is* 3
- *In the group* $(\mathbb{Z}, +)$*, the element* 6 *has infinite order because for all* $n \in \mathbb{N}\backslash\{0\}$, $6^n = \underbrace{6 + \cdots + 6}_{n \text{ times}} \neq 0$
- *In the group* $(\mathbb{C}\backslash\{0\}, \cdot)$ *- the complex numbers with multiplication- the element* 3 *has infinite order and the element* i *has order* 4*:* $i^2 = -1 \neq 1$, $i^3 = -i \neq 1$ *and* $i^4 = 1$*.*

Be careful: If $(G, \cdot)$ is a group and $x, y \in G$, then

$$(x \cdot y)^n = \underbrace{x \cdot y \cdots x \cdot y}_{n \text{ times}} \text{ and } x^n \cdot y^n = \underbrace{x \cdots x}_{n \text{ times}} \cdot \underbrace{y \cdots y}_{n \text{ times}}$$

In general, we can only conclude that these two expressions are equal if $(G, \cdot)$ is abelian.

Order

Theorem

If $(G, \cdot)$ is a finite group, then every element of G has finite order.

Proof.

Assume, for a contradiction, that $(G, \cdot)$ is a finite group and $x \in G$ has infinite order. Therefore none of the elements $x, x^2, x^3, \ldots$ are equal to e. Not all of these element can be distinct, otherwise G would be infinite, so there must exist $n > k \geq 1$ such that $x^n = x^k$. Multiplying the left hand side of this equation by x^{-1} k times we get $x^{n-k} = e$, which is a contradiction. □

Recall the exclusive OR operator in propositional logic:

A	B	$A \oplus B$
T	T	F
T	F	T
F	T	T
F	F	F

Order

Example

Let $A = \{T, F\}$ *and let* $X = \{f \mid f : \mathbb{N} \longrightarrow A\}$. *Define* $\cdot : X \times X \longrightarrow X$ *by: for all* $f, g, h \in X$,

$$f \cdot g = h \text{ if and only if for all } n \in \mathbb{N}, f(n) \oplus g(n) = h(n)$$

- $(X, \cdot)$ *is an abelian group*
- *The identity of* $(X, \cdot)$ *is the function* $f : \mathbb{N} \longrightarrow A$ *defined by: for all* $n \in \mathbb{N}$, $f(n) = F$
- $(X, \cdot)$ *is infinite. In fact,* X *is uncountable.*
- *For* $g \in X$, $g \cdot g = e$. *So, every element of* $(X, \cdot)$ *that is not the identity has order* 2

Subgroups

Definition

Let $(G, \cdot)$ be a group. We say that $H \subseteq G$ is a subgroup of $(G, \cdot)$, and write $H \leq G$ or $(H, \cdot) \leq (G, \cdot)$, if $e \in H$ and for all $x, y \in H$, $x \cdot y^{-1} \in H$.

Lemma

Let $(G, \cdot)$ be a group and let $H \subseteq G$. Then $H \leq G$ if and only if $(H, \cdot)$ is a group.

Proof.

$\Leftarrow$ is clear.

$\Rightarrow$: Suppose $H \leq G$. H is closed under the operation $\cdot$. The identity, e, of G is such that $e \in H$ and this will also be the identity of H. If $x \in H$, then $e \cdot x^{-1} = x^{-1} \in H$, and so H has inverses. $\square$

Example

- *If $(G, \cdot)$ is a group, then both G and the trivial group $\{e\}$ are subgroups of $(G, \cdot)$*

Subgroups

Example

- *The set of even integers* $2\mathbb{Z} = \{n \in \mathbb{Z} \mid (\exists k \in \mathbb{Z})(n = 2k)\}$ *is a subgroup of* $(\mathbb{Z}, +)$
- $H = \{e, (012), (021)\}$ *is a subgroup of* S_3*, but* $H' = \{e, (01), (012)\}$ *is not a subgroup of* S_3
- *Let* $X = \{f \mid f : \mathbb{R} \longrightarrow \mathbb{R}\}$*. Then* $(X, +)$*– the set* X *with the operation "addition of functions"– is a group. And* $X' = \{f : \mathbb{R} \longrightarrow \mathbb{R} \mid f(0) = 0\}$ *is subgroup of* $(X, +)$*. But* $X'' = \{f : \mathbb{R} \longrightarrow \mathbb{R} \mid f(0) = 1\}$ *is not a subgroup of* $(X, +)$*.*

Definition

Let $(G, \cdot)$ *be a group. If* G *is finite, then we call the cardinality of* G *the* **order** *of* G*.*

The Dihedral Groups

Let $n \geq 3$. Consider a regular n-gon with vertices $v_0, \ldots, v_{n-1}$ at positions $p_0, \ldots, p_{n-1}$ on a plane. The edges of the regular n-gon connect vertices v_k to v_{k+1} for $0 \leq k < n-1$ and v_{n-1} to v_0.

Any $\sigma \in S_n$ can be seen as acting on our regular n-gon by: for all $k \in [n]$, moving the vertex v_k to position $\sigma(k)$ and not changing which vertices have lines between them. Some of the bijections in S_n will mangle our regular n-gon by stretching or crossing the edges. However, some $\sigma \in S_n$ will act on our regular n-gon to give an identical regular n-gon with the vertices possibly in a different place. The bijections $\sigma \in S_n$ that yield a regular n-gon when they act on the regular n-gon described above are called **symmetries** of the regular n-gon.

For example, the bijection $\sigma \in S_n$ that rotates the vertices clockwise is a symmetry of the regular n-gon. And the bijection $\sigma \in S_n$ that reflects the vertices through an axis of symmetry is a symmetry of the regular n-gon.

The Dihedral Groups

Definition

Let $n \geq 3$. The **Dihedral Group** *D_n is the subgroup of S_n of all symmetries of a regular n-gon.*

Example

- *D_3 is the subgroup of S_3 of symmetries of an equilateral triangle. One should convince oneself that every bijection $\sigma \in S_3$ is a symmetry of an equilateral trinagle, and so $D_3 = S_3$*
- *D_4 is the subgroup of S_4 of symmetries of a square. $D_4 \neq S_4$, e.g. the bijection (01) crosses two parallel edges. One can check that:*

$$D_4 = \left\{ \begin{array}{c} e, (01)(23), (0123), (02)(13), (0321), \\ (01)(23)(0123), (01)(23)(02)(13), (01)(23)(0321) \end{array} \right\}$$

Some people write D_{2n} instead of D_n (so D_3 is D_6, and D_4 is D_8). Here is why...

The Dihedral Groups

Theorem

Let $n \geq 3$. The group D_n has order $2n$.

Proof.

Consider a regular n-gon with vertices $v_0, \ldots, v_{n-1}$ at positions $p_0, \ldots, p_{n-1}$ on a plane. The edges of the regular n-gon connect vertices v_k to v_{k+1} for $0 \leq k < n-1$ and v_{n-1} to v_0. We will count the number of symmetries of this n-gon. Consider the vertex v_0. This vertex can be sent to any of the positions $p_0, \ldots, p_{n-1}$, so there are n choices for where to send v_0. Once we have chosen where to send v_0, we need to choose where to send v_1. But there are only two choices of where to send v_1, because v_1 must remain adjacent to v_0 so we can only send it to one of the two positions that are adjacent to the position where we sent v_0. Therefore, in total, there are $2n$ choices for where to send the vertices v_0 and v_1. Once we have chosen where to send v_0 and v_1, the positions of all of the other vertices are determined by which vertices they are adjacent to.

The Dihedral Groups

Proof.

(Continued.) All the possible choices of where to send v_0 and v_1 yield distinct symmetries of the regular n-gon and all symmetries are determined by the positions of v_0 and v_1, so we have counted all of the possible symmetries of a regular n-gon. □

We have just shown that S_n has a subgroup of order $2n$. Is there any restrictions on the possible orders of a subgroup of a finite group?

Lagrange's Theorem

Definition

Let $(G, \cdot)$ be a group. Let $H \leq G$ and let $a \in G$. We define the a **left coset** *of H, denoted aH, to be*

$$aH = \{a \cdot x \mid x \in H\}$$

We define the a **right coset** *of H, denoted Ha, to be*

$$Ha = \{x \cdot a \mid x \in H\}$$

Theorem

(Lagrange's Theorem) Let $(G, \cdot)$ be a finite group. If $H \leq G$, then the order of H divides the order of G.

The Division Algorithm

For this section, we will extend the definition of divisibility ($|$) to $\mathbb{Z}$.

Definition

Let $x, y \in \mathbb{Z}$. We say that x divides y, and write $x \mid y$, if there exists a $k \in \mathbb{Z}$ such that $k \cdot x = y$.

Note that $(\mathbb{Z}, |)$ is no longer a partial order: $-1 \mid 1$ and $1 \mid -1$, but $1 \neq -1$.

Theorem

(Division Algorithm) Let $a \in \mathbb{Z}$ and let $b \in \mathbb{N}$ with $b \neq 0$. There exists a unique $q, r \in \mathbb{Z}$ such that

$$a = q \cdot b + r \ \ and \ 0 \leq r < b$$

The number q is called the **quotient** *and r is called the* **remainder** *in the division of a by b.*

For example, if $a = 25$ and $b = 4$, then we can write $25 = 6 \cdot 4 + 1$ and so the quotient is 6 and the remainder is 1

Proof of The Division Algorithm

Proof.

(Uniqueness) Suppose that there exists $r, \tilde{r}, q, \tilde{q} \in \mathbb{Z}$ with $0 \leq r, \tilde{r}, < b$ and

$$a = q \cdot b + r = \tilde{q} \cdot b + \tilde{r}$$

Without loss of generality, we can assume that $q \leq \tilde{q}$. Therefore

$$r - \tilde{r} = b \cdot (\tilde{q} - q) \geq 0$$

and so $\tilde{r} \leq r$ and $0 \leq r - \tilde{r} < b$. But this means that $b \cdot (q - \tilde{q}) < b$, which implies that $q - \tilde{q} < 1$. Therefore, since $q, \tilde{q} \in \mathbb{Z}$, $q = \tilde{q}$. And the fact that $q = \tilde{q}$ implies that $r = \tilde{r}$.

Proof of The Division Algorithm

Proof.

(Existence) Let

$$S = \{n \in \mathbb{N} \mid (\exists x \in \mathbb{Z})(n = a - x \cdot b)\}$$

We can see that $S \neq \emptyset$, because when $x = -|a|$, $a + |a| \cdot b \geq a + |a| \geq 0$. Therefore, let $r \in S$ be the $\leq$-least element of S. There exists $q \in \mathbb{Z}$ such that $r = a - q \cdot b$. If $r \geq b$, then
$a - (q + 1) \cdot b = a - q \cdot b - b = r - b \geq 0$, which contradicts the fact that r is least. Therefore $0 \leq r < b$ and $q \in \mathbb{Z}$ with $a = q \cdot b + r$. $\square$

Application of the Division Algorithm

Theorem

If $n \in \mathbb{Z}$ is odd, then there exists $k \in \mathbb{N}$ such that $n^2 = 8k + 1$.

Proof.

Assume that n is odd. By the Division Algorithm, there exists $q \in \mathbb{Z}$ such that n is equal to $4q$, $4q + 1$, $4q + 2$ or $4q + 3$. Since n is odd, n is equal to either $4q + 1$ or $4q + 3$. If $n = 4q + 1$, then

$$n^2 = (4q + 1)^2 = 16q^2 + 8q + 1 = 8(2q^2 + q) + 1$$

And, if $q < 0$ then $2q + 1 < 0$ and $2q^2 + q > 0$. If $q \geq 0$, then $2q^2 + q \geq 0$. If $n = 4q + 3$, then

$$n^2 = (4q + 3)^2 = 16q^2 + 24q + 9 = 8(2q^2 + 3q + 1) + 1$$

And, if $q < -1$, then $2q + 3 < 0$ and $2q^2 + 3q > 0$. If $q = -1$, then $2q^2 + 3q + 1 = 0$. And, if $q \geq 0$, then $2q^2 + 3q + 1 \geq 0$. So, in all cases we have written n^2 in the form $4k + 1$ where $k \in \mathbb{N}$. □

Generated Subgroups

Definition

Let $(G, \cdot)$ be a group and let $A \subseteq G$. We define the subgroup generated by A, denoted $\langle A \rangle_G$, to be the $\subseteq$-least $H \subseteq G$ such that $A \cup \{e\} \subseteq H$ and for all $x, y \in H$, $x \cdot y^{-1} \in H$.

- $\langle A \rangle_G$ is a recursively defined set!
- The closure conditions (constructors) ensure that $\langle A \rangle_G \leq G$
- Moreover, if $H \leq G$ with $A \subseteq H$, then $\langle A \rangle_G \subseteq H$ and so $\langle A \rangle_G \leq H$
- If $A \subseteq G$ is finite with $A = \{a_1, \ldots, a_n\}$, then we will often write $\langle a_1, \ldots, a_n \rangle_G$ instead of $\langle A \rangle_G$
- We will often write $\langle A \rangle$ or $\langle a_1, \ldots, a_n \rangle$ instead of $\langle A \rangle_G$ and $\langle a_1, \ldots, a_n \rangle_G$

Example

- $\langle (01)(23), (0123) \rangle_{S_4} = D_4 \leq S_4$

Generated Subgroups

Example

- *Consider* $(\mathbb{Z}, +)$.

$$\langle 2 \rangle = 2\mathbb{Z} \leq \mathbb{Z}$$

- *Consider* $(\mathbb{R}\backslash\{0\}, \cdot)$.

$$\langle \mathbb{Z}\backslash\{0\} \rangle = \mathbb{Q}\backslash\{0\} \leq \mathbb{R}$$

- *Consider* S_n. *If* $A = \{\sigma \in S_n \mid \sigma \ is \ a \ 2\text{-}cycle\}$, *then* $\langle A \rangle = S_n$

The Cyclic Groups

Definition

Let $(G, \cdot)$ be a group and let $n \in \mathbb{N}\backslash\{0\}$. Let $a \in G$ have order n. We call the the group $\langle a \rangle \leq G$ the **Cyclic Group of order** *n and denote this group C_n. Let $b \in G$ have infinite order. We call the the group $\langle b \rangle \leq G$ the* **Cyclic Group of infinite order** *and denote this group C_∞.*

This looks a bit strange as it appears that we are giving the same name to lots of different groups. However, it should be clear, or become clear soon, that all of these 'different' groups really have the same structure— We will make this precise soon!

The Cyclic Groups

Lemma

Let $(G, \cdot)$ be a group. If $a \in G$, then

$$\langle a \rangle = \{a^m \mid m \in \mathbb{Z}\}$$

(Where, for all $k \in \mathbb{N}$, $a^{-k} = (a^{-1})^k$)

Proof.

Let $a \in G$ and let $H = \{a^m \mid m \in \mathbb{Z}\}$. Now, $e, a \in H$ and for all $x, y \in H$, $x \cdot y^{-1} \in H$ so $H \leq G$, and by the $\subseteq$-least property of $\langle a \rangle$, $\langle a \rangle \leq H$. Conversely, a straightforward induction argument shows that for all $k \in \mathbb{N}$, $a^k, a^{-k} \in \langle a \rangle$. Therefore $H \leq \langle a \rangle$. □

Lemma

Let $n \in \mathbb{N}\backslash\{0\}$ or $n = \infty$. The group C_n is abelian.

Proof.

Let $(G, \cdot)$ be a group and let $a \in G$. Let $x, y \in \langle a \rangle$. Therefore, there exists $m, k \in \mathbb{Z}$, such that $x = a^m$ and $y = a^k$. Now, since $a \cdot a = a \cdot a$ and $a \cdot a^{-1} = a^{-1} \cdot a$, $x \cdot y = a^m \cdot a^k = a^{m+k} = a^k \cdot a^m = y \cdot x$. □

The Cyclic Groups

Lemma

Let $(G, \cdot)$ be a group and let $n \in \mathbb{N}\backslash\{0\}$. If $a \in G$ has order n, then $|\langle a \rangle| = n$.

Proof.

Let $a \in G$ have order n. We will show that the elements of $\langle a \rangle$ are exactly $e, a, a^2, \ldots, a^{n-1}$. It is clear that $e, a, a^2, \ldots, a^{n-1} \in G$. We will first show that each of these elements is distinct. Suppose, for a contradiction, that there exists $0 \leq p < q < n$ such that $a^p = a^q$. But this implies that $a^{q-p} = e$ with $0 < q - p < n$, which contradicts the fact that the order of a is n.

We now turn to showing that every element of $\langle a \rangle$ is equal to one of $e, a, a^2, \ldots, a^{n-1}$. Let $g \in \langle a \rangle$. Therefore there exists $m \in \mathbb{Z}$ such that $g = a^m$. Now, using the Division Algorithm, we can write $m = qn + r$ where $0 \leq r < n$. Therefore

$$g = a^m = a^{qn+r} = (a^n)^q \cdot a^r = e \cdot a^r = a^r$$

Cyclic Groups in the Symmetric Group

Lemma

Let $n \in \mathbb{N}\backslash\{0\}$ and let $m \leq n$. Let $k_1, \ldots, k_m \in [n]$ be distinct. The m-cycle $(k_1 \cdots k_m)$ has order m in S_n.

Proof.

Observe that, in S_n, $(k_1 \cdots k_m)^m = e$. And, if $0 < k < m$, then $(k_1 \cdots k_m)^k$ sends k_1 to k_{1+k}. Therefore $(k_1 \cdots k_m)^k \neq e$. □

Theorem

Let $n \in \mathbb{N}\backslash\{0\}$. For all $0 < k \leq n$, $C_k \leq S_n$

We also get a refinement of Lagrange's Theorem:

Theorem

If $(G, \cdot)$ is a finite group and $x \in G$, then the order of x divides the order of G

Proof.

Let $(G, \cdot)$ be a finite group and let $x \in G$. The order of x is equal to the order of $\langle x \rangle$, which divides the order of G. □

Group of order p

Theorem

Let p be prime. Let $(G, \cdot)$ be a finite group of order p. Then $(G, \cdot)$ is the the group C_p.

Proof.

Let $g \in G$ with $g \neq e$. We know that the order of g must be finite and greater than 1, and the order of g must divide $|G| = p$. Therefore the order of g must be exactly p, and $\langle g \rangle$ must be all of G. $\square$

Corollary

If $(G, \cdot)$ is a finite group with order p, then the only subgroups of G are the trivial group and G.

An important consequence of Lagrange's Theorem

Theorem

Let $(G, \cdot)$ be a group and let $g \in G$ have order n. If there exists $m, k \in \mathbb{N}\backslash\{0\}$ with $n = mk$, then the order of g^m is k.

Proof.

Let $m, k \in \mathbb{N}\backslash\{0\}$ with $n = mk$. Now, $(g^m)^k = g^{mk} = g^n = e$. If $0 < q < k$ is such that $(g^m)^q = e$, then $g^{mq} = e$. But $mq < mk = n$, which is a contradiction. □

Theorem

If $(G, \cdot)$ is a finite group with order n, then for all $g \in G$, $g^n = e$.

Proof.

Let $(G, \cdot)$ be a finite group with order n. Let $g \in G$. We know that the order of g must be finite, so let k be the order of g. Now, k must divide n, so the exists $m \in \mathbb{N}$ such that $n = mk$. So $g^n = g^{mk} = (g^k)^m = e^m = e$. □

Example

For all $\sigma \in D_5$, $\sigma^{10} = e$.

There is no converse to Lagrange's Theorem

Let $(G, \cdot)$ be a finite group. We have seen that if $H \leq G$, then the order of H must divide the order of G. And, if $g \in G$, then the order of g must divide the order of G. Is it the case that for all $k \mid |G|$, there exists $g \in G$ with order k?

Example

Let A_4 be the group of all even bijections in S_4. There is no $\sigma \in A_4$ with order 6.

Proof.

By viewing A_4 as a subgroup of S_4, we can see that every element of A_4 can be written as a product of disjoint cycles. The only products of disjoint cycles in S_4 that are even are 3-cycles and products of two disjoint 2-cycles. A 3-cycle has order exactly 3 and the product of any two disjoint 2-cycles has order 2. □

Does A_4 have a subgroup of order 6?

Theorem

If $(G, \cdot)$ is a group of order 6, then there exists $g \in G$ with order 2.

Proof.

Let $(G, \cdot)$ be a group of order 6. Suppose, for a contradiction, that for all $g \in G$, the order of g is not 2. If $g \in G$ with $g \neq e$, then the order of g must be 3 or 6. If $g \in G$ is such that the order of g is 6, then the order of g^3 must be 2. So, for all $g \in G$ with $g \neq e$, the order of g must be 3. Let $p \in G$ with $p \neq e$. Therefore $|\langle p \rangle| = 3$ and $p^4 = p$, so $\langle p^2 \rangle = \langle p \rangle$. Let $q \in G \backslash \langle p \rangle$. Therefore $|\langle q \rangle| = 3$ and $\langle q^2 \rangle = \langle q \rangle$. It follows that $\langle q \rangle \cap \langle p \rangle = \{e\}$. This means that there are exactly 5 elements in $\langle p \rangle \cup \langle q \rangle$. Let $g \in G$ be such that $g \notin \langle p \rangle \cup \langle q \rangle$. Since g has order 3, it follows that $g^2 \neq e$. But this means that g^2 is equal to one of the elements p, p^2, q, q^2. And this means that $\langle g \rangle$ contains at least 4 elements, which is a contradiction. $\square$

Isomorphisms and Homomorphisms

The notions of isomorphism and homomorphism are extremely important in the study of groups.

Definition

Let $(G, \cdot)$ and $(K, \star)$ be groups. We say that $f : G \longrightarrow K$ is a **group homomorphism** *or just a* **homomorphism** *if for all $a, b \in G$,*

$$f(a \cdot b) = f(a) \star f(b)$$

Definition

Let $(G, \cdot)$ and $(K, \star)$ be groups. We say that $f : G \longrightarrow K$ is a **group isomorphism** *or just an* **isomorphism** *if f is a group homomorphism that is a bijection. If there exists an isomorphism between the groups $(G, \cdot)$ and $(K, \star)$, then we say that $(G, \cdot)$ and $(K, \star)$ are isomorphic and write $G \cong K$ or $(G, \cdot) \cong (K, \star)$.*

If $(G, \cdot)$ and $(K, \star)$ are isomorphic, then they are essentially the same group. An isomorphism $f : G \longrightarrow K$ is a relabelling of the elements of G that yields the group $(K, \star)$.

Examples

Theorem

Let $(G, \cdot)$ be a group. Let $g, h \in G$ both have order n. Then $\langle g \rangle \cong \langle h \rangle$.

Proof.

Define $f : \langle g \rangle \longrightarrow \langle h \rangle$ by: $f(g) = h$ and for all $0 \leq k < n$, $f(g^k) = f(g)^k$. So, f is a well-defined function, and, by definition, f preserves the group product. It is clear that the function f sends $e \mapsto e$, $g \mapsto h$, $\ldots$, $g^{n-1} \mapsto h^{n-1}$, and so f is a bijection. □

Example

Let $(G, \cdot)$ be any group and let $H = \{e\}$, i.e. H is the trivial subgroup of $(G, \cdot)$. The function $f : G \longrightarrow H$ defined by: for all $x \in G$, $f(x) = e$, is a homomorphism. The function $g : H \longrightarrow G$ defined by: $g(e) = e$, is also a homomorphism. The homomorphism f is surjective but not injective, and the homomorphism g is injective, but not surjective.

Examples

Theorem

Consider the group $(\mathbb{Z}, +)$. *If* $n \in \mathbb{N}\backslash\{0\}$, *define*

$$n\mathbb{Z} = \{m \in \mathbb{Z} \mid (\exists k \in \mathbb{Z})(m = nk)\}$$

Then $n\mathbb{Z} \leq \mathbb{Z}$ *and* $n\mathbb{Z} \cong \mathbb{Z}$

Proof.

Define $f : \mathbb{Z} \longrightarrow n\mathbb{Z}$ by: for all $x \in \mathbb{Z}$, $f(x) = nx$. Now, f is a bijection and for all $x, y \in \mathbb{Z}$,

$$f(x + y) = n(x + y) = nx + ny = f(x) + f(y)$$

□

Examples

Example

Let $n \in \mathbb{N}$ with $n \geq 2$. Let $(G, \cdot)$ be a group and let $a \in G$ have order n. Let $H = \langle a \rangle$, i.e. H is (isomorphic to) C_n. Consider the group $(\mathbb{Z}, +)$. Define $f : \mathbb{Z} \longrightarrow H$ by: for all $x \in \mathbb{Z}$, $f(x) = a^x$. Then f is a homomorphism because for all $x, y \in \mathbb{Z}$,

$$f(x + y) = a^{x+y} = a^x \cdot a^y$$

Congruency

Definition

Let $n \in \mathbb{N}\backslash\{0\}$. For all $a, b \in \mathbb{Z}$, define

$$a \equiv b \pmod{n} \text{ if and only if } n \mid a - b$$

The relation $\ldots \equiv \ldots \pmod{n}$ is an equivalence relation on $\mathbb{Z}$, and so, for all $a \in \mathbb{Z}$, we use $[a]_n$ to denote the equivalence class of a. Define

$$\mathbb{Z}/n\mathbb{Z} = \{[a]_n \mid a \in \mathbb{Z}\}$$

and define $\oplus_n : \mathbb{Z}/n\mathbb{Z} \times \mathbb{Z}/n\mathbb{Z} \longrightarrow \mathbb{Z}/n\mathbb{Z}$ by: for all $a, b \in \mathbb{Z}$,

$$[a]_n \oplus_n [b]_n = [a + b]_n$$

Define $\otimes_n : \mathbb{Z}/n\mathbb{Z} \times \mathbb{Z}/n\mathbb{Z} \longrightarrow \mathbb{Z}/n\mathbb{Z}$ by: for all $a, b \in \mathbb{Z}$,

$$[a]_n \otimes_n [b]_n = [ab]_n$$

Congruency

Note that we will sometimes omit the subscript n from $\oplus_n$ and $\otimes_n$ when the context is clear.

Lemma

If $n \in \mathbb{N}\backslash\{0\}$, *then* $(\mathbb{Z}/n\mathbb{Z}, \oplus_n)$ *is group.*

Let $(G, \cdot)$ be a finite group. Since $\cdot : G \times G \longrightarrow G$, the operation $\cdot$ can be represented using a table in a similar fashion to the way logical operations were represented using truth tables. For example, the group $(\mathbb{Z}/4\mathbb{Z}, \oplus_4)$ can be represented using the following table:

$\oplus_4$	$[0]_4$	$[1]_4$	$[2]_4$	$[3]_4$
$[0]_4$	$[0]_4$	$[1]_4$	$[2]_4$	$[3]_4$
$[1]_4$	$[1]_4$	$[2]_4$	$[3]_4$	$[0]_4$
$[2]_4$	$[2]_4$	$[3]_4$	$[0]_4$	$[1]_4$
$[3]_4$	$[3]_4$	$[0]_4$	$[1]_4$	$[2]_4$

This tabular representation of a finite group $(G, \cdot)$ is called a **Cayley Table**. Note that the conditions required of a group (the group axioms) impose certain conditions on the Cayley Table of a group $(G, \cdot)$. For example, the identity element must appear in every row and column of the table.

Congruency

Lemma

If $n \in \mathbb{N}\backslash\{0\}$, then $(\mathbb{Z}/n\mathbb{Z}, \oplus_n)$ is abelian with order n. Moreover, $(\mathbb{Z}/n\mathbb{Z}, \oplus_n) = C_n$.

Proof.

First note that the elements $[0]_n, \ldots, [n-1]_n$ are all distinct, because if $0 \leq k < l < n$, then $0 \leq l - k < n$ and so n does not divide $l - k$. Now, let $k \in \mathbb{Z}$. By the Division Algorithm, there exists $q \in \mathbb{Z}$ and $r \in [n]$ such that $k = qn + r$. Therefore $n \mid (k - r)$, and $[k]_n = [r]_n$.
To see that $(\mathbb{Z}/n\mathbb{Z}, \oplus_n) = C_n$, observe that $(\mathbb{Z}/n\mathbb{Z}, \oplus) = \langle [1]_n \rangle$. □

Is $(\mathbb{Z}/n\mathbb{Z}, \otimes_n)$ a group?
No! $(\mathbb{Z}/n\mathbb{Z}, \otimes_n)$ does not have inverses: for all $k \in [n]$ with, $[k]_n \otimes [1]_n = [k]_n$, but there does not exist $k \in [n]$ such that $[k]_n \otimes [0]_n = [1]_n$.

Congruency

- So, the element $[0]_n$ appears to prevent $(\mathbb{Z}/n\mathbb{Z}, \otimes_n)$ from being a group. Let $G_n = \mathbb{Z}/n\mathbb{Z}\backslash\{[0]_n\}$. Is $(G_n, \otimes_n)$ a group?
- The Cayley Table of $(G_3, \otimes_3)$ is

$\otimes_3$	$[1]_3$	$[2]_3$
$[1]_3$	$[1]_3$	$[2]_3$
$[2]_3$	$[2]_3$	$[1]_3$

  and one can verify that this is a group.
- However, $\otimes_6$ is not even a function from $G_6 \times G_6$ to G_6, because

$$[2]_6 \otimes_6 [3]_6 = [2 \cdot 3]_6 = [6]_6 = [0]_6 \notin G_6$$

- It is clear that if $(G_n, \otimes_n)$ is a group then $[1]_n$ must be the identity of this group and for all $[k]_n \in G_n$, there must exist $[m]_n \in G_n$ such that

$$[k]_n \otimes_n [m]_n = [km]_n = [1]_n$$

Congruency

- I.e. for all $[k]_n \in G_n$, there must exists $x \in \mathbb{Z}$ such that

$$kx \equiv 1 \pmod{n}$$

Definition

For all $n \in \mathbb{N}$ with $n \geq 2$, define

$$(\mathbb{Z}/n\mathbb{Z})^* = \{[k]_n \in \mathbb{Z}/n\mathbb{Z} \mid (\exists x \in \mathbb{Z})(kx \equiv 1 \pmod{n})\}$$

Theorem

Let $n \in \mathbb{N}$ with $n \geq 2$. Then $((\mathbb{Z}/n\mathbb{Z})^*, \otimes_n)$ is a group.

Proof.

We first need to show that $\otimes_n$ is a function from $(\mathbb{Z}/n\mathbb{Z})^* \times (\mathbb{Z}/n\mathbb{Z})^*$ to $(\mathbb{Z}/n\mathbb{Z})^*$. Let $[k]_n, [m]_n \in (\mathbb{Z}/n\mathbb{Z})^*$. Therefore there exists $x, y \in \mathbb{Z}$ such that

$$kx \equiv 1 \pmod{n} \text{ and } my \equiv 1 \pmod{n}$$

Therefore $n \mid (kx - 1)$ and $n \mid (my - 1)$. Let $p, q \in \mathbb{Z}$ be such that $kx - 1 = np$ and $my - 1 = nq$. So $kxmy = n^2pq + np + nq + 1$, and $kmxy - 1 = n(npq + p + q)$. Therefore $n \mid (kmxy - 1)$, and so $kmxy \equiv 1 \pmod{n}$. Since $xy \in \mathbb{Z}$, this is shows that

$$[km]_n = [k]_n \otimes_n [m]_n \in (\mathbb{Z}/n\mathbb{Z})^*$$

Now, $[1]_n \in (\mathbb{Z}/n\mathbb{Z})^*$ and for all $[k]_n \in (\mathbb{Z}/n\mathbb{Z})^*$, $[k]_n \otimes_n [1]_n = [k]_n$. Finally, we need to show that inverses exist. Let $[k]_n \in (\mathbb{Z}/n\mathbb{Z})^*$. Therefore there exists $x \in \mathbb{Z}$ such that $kx \equiv 1 \pmod{n}$. But then it must also be the case that $[x]_n \in (\mathbb{Z}/n\mathbb{Z})^*$ and

$$[k]_n \otimes_n [x]_n = [kx]_n = [1]_n$$

Congruency

Example

Note that

$$6 \mid 24 = (5 \cdot 5 - 1), \ so\ [5]_6 \in (\mathbb{Z}/6\mathbb{Z})^*$$
$$6 \mid 0 = (1 - 1), \ so\ [1]_6 \in (\mathbb{Z}/6\mathbb{Z})^*$$

However, if $6 \mid y$*, then* y *must be even. And, since for all* $x \in \mathbb{Z}$*,* $2x - 1$ *and* $4x - 1$ *are odd, this shows that* $[2]_6, [4]_6 \notin (\mathbb{Z}/6\mathbb{Z})^*$*. By the same reasoning, if* $6 \mid y$*, then* y *must be a multiple of* 3*. And, for all* $x \in \mathbb{Z}$*,* $3x - 1$ *can not be a multiple of* 3*, so* $[3]_6 \notin (\mathbb{Z}/6\mathbb{Z})^*$*. Therefore* $(\mathbb{Z}/6\mathbb{Z})^* = \{[1]_6, [5]_6\}$ *and the Cayley Table of* $((\mathbb{Z}/6\mathbb{Z})^*, \otimes_6)$ *is*

$\otimes_6$	$[1]_6$	$[5]_6$
$[1]_6$	$[1]_6$	$[5]_6$
$[5]_6$	$[5]_6$	$[1]_6$

It should be reasonably obvious that $((\mathbb{Z}/6\mathbb{Z})^*, \otimes_6) \cong ((\mathbb{Z}/3\mathbb{Z})^*, \otimes_3) \cong C_2$

Congruency

- How can we find the elements of $(\mathbb{Z}/n\mathbb{Z})^*$?
- What is the order of the group $((\mathbb{Z}/n\mathbb{Z})^*, \otimes_n)$?

Lemma

Let $n \in \mathbb{N}$ with $n \geq 2$. If $1 < m \leq n$ is such that there exists $1 < d \leq m$ with $d \mid m$ and $d \mid n$, then $[m]_n \notin (\mathbb{Z}/n\mathbb{Z})^*$

Proof.

Let $1 < m \leq n$ be such that there exists $1 < d \leq m$ with $d \mid m$ and $d \mid n$. Therefore there exists $k, l \in \mathbb{Z}$ such that $n = dk$ and $m = dl$. Suppose, for a contradiction, that $[m]_n \in (\mathbb{Z}/n\mathbb{Z})^*$. So, there exists $x \in \mathbb{Z}$, with $mx \equiv 1 \pmod{n}$. Therefore $n \mid (mx - 1)$, and so $dk \mid (dlx - 1)$. But this means that there exists $y \in \mathbb{Z}$ such that $dky = dlx - 1$. And so $1 = d(lx - ky)$, which is a contradiction. □

Are these the only elements of $\mathbb{Z}/n\mathbb{Z}$ that are not in $(\mathbb{Z}/n\mathbb{Z})^*$? In order to show that $[m]_n \in (\mathbb{Z}/n\mathbb{Z})^*$ we need to find $x \in \mathbb{Z}$ such that $mx \equiv 1 \pmod{n}$. I.e. we need to find $x, y \in \mathbb{Z}$ such that $mx + ny = 1$.

GCD Revisited

Recall that for $a, b \in \mathbb{N}$, $\gcd(a, b)$ was meet of a and b ($a \wedge b$) in the lattice $(\mathbb{N}, |)$. We now wish to extend this definition to $\mathbb{Z}$, but since $(\mathbb{Z}, |)$ is not a lattice we need to specify that for all $a, b \in \mathbb{Z}$, $\gcd(a, b) \in \mathbb{N}$ in order to ensure that $\gcd(a, b)$ is well-defined.

Definition

Let $a, b \in \mathbb{Z}$ with $|a| + |b| \neq 0$. We say that $d \in \mathbb{N}$ is the **greatest common divisor** *of a and b, and write this element $\gcd(a, b)$, if*

(i) *$d \mid a$ and $d \mid b$,*

(ii) *and for all $c \in \mathbb{Z}$, if $c \mid a$ and $c \mid b$, then $c \mid d$*

Example

- $\gcd(13, -23) = 1$
- $\gcd(-26, -14) = 2$

We could have defined $\gcd(a, b)$ to be $|a| \wedge |b|$ in the lattice $(\mathbb{N}, |)$

Linear Diophantine Equations

Named after the Ancient Greek mathematician Diophantus (circ. 300AD), a **Diophantine Equation** is a polynomial equation in two or more variables for which an integer solution is sought.

Definition

A **linear Diophantine equation in two variables** *is an equation in the form*

$$ax + by = c \ \textit{where}\ a, b, c \in \mathbb{Z}\ \textit{are constants with}\ |a| + |b| \neq 0$$

A **solution** *is a pair* $(x_0, y_0) \in \mathbb{Z} \times \mathbb{Z}$ *with* $ax_0 + by_0 = c$.

This means that in order to show that $[m]_n \in (\mathbb{Z}/n\mathbb{Z})^*$, we show that the linear Diophantine equation $mx + ny = 1$ has a solution.

The result we proved on the previous slide shows that if $\gcd(m, n) > 1$, then $[m]_n \notin (\mathbb{Z}/n\mathbb{Z})^*$.

Bézout's Lemma

Theorem

Let $a, b \in \mathbb{Z}$ with $|a| + |b| \neq 0$. Then there exists $x, y \in \mathbb{Z}$ such that $\gcd(a, b) = ax + by$.

Proof.

Consider

$$S = \{n \in \mathbb{N}\backslash\{0\} \mid (\exists x, y \in \mathbb{Z})(n = ax + by)\}$$

If $y = b$ and $x = a$, then $ax + by = a^2 + b^2 > 0$, which shows that $S \neq \emptyset$. Therefore, let $d \in S$ be the $\leq$-least element of S. We will show that $\gcd(a, b) = d$. Let $x_0, y_0 \in \mathbb{Z}$ be such that $ax_0 + by_0 = d$. Using the Division Algorithm we can find $q \in \mathbb{Z}$ and $0 \leq r < b$ such that $a = qd + r$. But this means that

$$r = a - qd = a - q(ax_0 + by_0) = a(1 - qx_0) + b(-qy_0)$$

which means that $r = 0$, otherwise we would have contradicted the leastness of d. Therefore $d \mid a$. The identical argument replacing a with b shows that $d \mid b$.

Bézout's Lemma

Proof.

(Continued.) Let $c \in \mathbb{Z}$ be such that $c \mid a$ and $c \mid b$. We need to show that $c \mid d$. Let $k, l \in \mathbb{Z}$ such that $a = ck$ and $b = cl$. Therefore

$$d = ckx_0 + cly_0 = c(kx_0 + ly_0)$$

which shows that $c \mid d$. □

Corollary

Let $n \in \mathbb{N}$ with $n \geq 2$. For all $m \in \mathbb{Z}$,

$$[m]_n \in (\mathbb{Z}/n\mathbb{Z})^* \textit{ if and only if } \gcd(m, n) = 1$$

This gives a a characterisation of the elements of $(\mathbb{Z}/n\mathbb{Z})^*$. Do we need to check every $m \in \mathbb{Z}$ to see if $[m]_n \in (\mathbb{Z}/n\mathbb{Z})^*$?

The existence of multiplicative inverses

Lemma

Let $a \in \mathbb{Z}$ and $b \in \mathbb{N}\backslash\{0\}$. If $q, r \in \mathbb{Z}$ with $a = qb + r$, then $\gcd(a, b) = \gcd(b, r)$

Proof.

Let $q, r \in \mathbb{Z}$ with $a = qb + r$. Let $d = \gcd(a, b)$. Now, $d \mid a$ and $d \mid b$, so $d \mid r = a - qb$. Suppose that $c \in \mathbb{Z}$ is such that $c \mid r$ and $c \mid b$. Therefore $c \mid a = qb + r$, and so $c \mid d = \gcd(a, b)$. So $\gcd(b, r) = d$. □

Corollary

Let $n \in \mathbb{N}$ with $n \geq 2$.

$$(\mathbb{Z}/n\mathbb{Z})^* = \{[m]_n \mid (m < n) \wedge (\gcd(m, n) = 1)\}$$

Proof.

Let $m \in \mathbb{Z}$. Using the division algorithm we can obtain $q \in \mathbb{Z}$ and $0 \leq r < n$ such that $m = qn + r$. Therefore $[r]_n = [m]_n$ and $\gcd(m, n) = \gcd(r, n)$. □

The existence of multiplicative inverses

Example

$12 = 3 \cdot 2^2$, *so* $0 \leq a < 12$ *and* $\gcd(a, 12) = 1$ *if and only if* $a = 1, 5, 7, 11$. *Therefore* $(\mathbb{Z}/12\mathbb{Z})^* = \{[1]_{12}, [5]_{12}, [7]_{12}, [11]_{12}\}$. *The Cayley Table for* $((\mathbb{Z}/12\mathbb{Z})^*, \otimes_{12})$ *is*

$\otimes_{12}$	$[1]_{12}$	$[5]_{12}$	$[7]_{12}$	$[11]_{12}$
$[1]_{12}$	$[1]_{12}$	$[5]_{12}$	$[7]_{12}$	$[11]_{12}$
$[5]_{12}$	$[5]_{12}$	$[1]_{12}$	$[11]_{12}$	$[7]_{12}$
$[7]_4$	$[7]_{12}$	$[11]_{12}$	$[1]_{12}$	$[5]_4$
$[11]_4$	$[11]_{12}$	$[7]_{12}$	$[5]_{12}$	$[1]_4$

Note that $((\mathbb{Z}/12\mathbb{Z})^*, \otimes_{12}) \neq C_4$, *because every element* $((\mathbb{Z}/12\mathbb{Z})^*, \otimes_{12})$ *has order* 2.

Definition

Let $a, b \in \mathbb{Z}$ *with* $|a| + |b| \neq 0$. *We say that* a *and* b *are* **relatively prime** *if* $\gcd(a, b) = 1$

The existence of multiplicative inverses

Example

For all $0 < m < 5$, $\gcd(m, 5) = 1$. *Therefore*

$$(\mathbb{Z}/5\mathbb{Z})^* = \{[1]_5, [2]_5, [3]_5, [4]_5\}$$

The Cayley Table for $((\mathbb{Z}/5\mathbb{Z})^*, \otimes_5)$ *is*

$\otimes_5$	$[1]_5$	$[2]_5$	$[3]_5$	$[4]_5$
$[1]_5$	$[1]_5$	$[2]_5$	$[3]_5$	$[4]_5$
$[2]_5$	$[2]_5$	$[4]_5$	$[1]_5$	$[3]_5$
$[3]_5$	$[3]_5$	$[1]_5$	$[4]_5$	$[2]_5$
$[4]_5$	$[4]_5$	$[3]_5$	$[2]_5$	$[1]_5$

Note that $([2]_5)^2 = [4]_5$ *and* $([2]_5)^3 = [3]_5$ *(of course* $([2]_5)^4 = [1]_5$*). This means that* $\langle [2]_5 \rangle = ((\mathbb{Z}/5\mathbb{Z})^*, \otimes_5)$*, and so* $((\mathbb{Z}/5\mathbb{Z})^*, \otimes_5)$ *is (isomorphic to* C_4*).*

Euler's Totient Function

Definition

Euler's Totient Function, *denoted* φ, *is the function defined on all* $n \in \mathbb{N}$ *with* $n \geq 2$ *by*

$$\varphi(n) = |(\mathbb{Z}/n\mathbb{Z})^*|$$

In other words, $\varphi(n)$ is the number of $0 < m < n$ such that m and n are relatively prime.

Lemma

If $p \in \mathbb{N}$ *is prime, then* $\varphi(p) = p - 1$.

Proof.

Let $p \in \mathbb{N}$ be prime. For all $0 < m < p$, $\gcd(m, p) = 1$. □

Example

On the previous slide we showed that $\varphi(12) = 4$.

Euler's Totient Function

The following results are important for understanding the relation

$$x \equiv y \pmod{n}$$

Theorem

(Euler's Theorem) Let $a, n \in \mathbb{N}$ *with* $n \geq 2$ *and* $\gcd(a, n) = 1$*. Then* $a^{\varphi(n)} \equiv 1 \pmod{n}$

Proof.

This follows immediately form the fact that the order of $[a]_n$ in the group $((\mathbb{Z}/n\mathbb{Z})^*, \otimes_n)$ must divide $\varphi(n)$ (which is the order of $((\mathbb{Z}/n\mathbb{Z})^*, \otimes_n)$). □

Theorem

(Fermat's Little Theorem) If $a, p \in \mathbb{N}$*,* p *is prime and* $\gcd(a, p) = 1$*, then* $a^{p-1} \equiv 1 \pmod{p}$

Proof.

$\varphi(p) = p - 1$ and $\gcd(a, p) = 1$. □

Euler's Totient Function

The following results allow us to compute Euler's Totient Function:

Theorem

(Euler's Product Formula)

$$\varphi(n) = n \cdot \prod_{p \in A} \left(1 - \frac{1}{p}\right)$$

Where A is the set of distinct primes that divide n

Unfortunately, we will probably not be able to prove that this formula works in this course.

Example

$21 = 7 \cdot 3$, *so*

$$\varphi(21) = 21 \cdot \left(1 - \frac{1}{3}\right)\left(1 - \frac{1}{7}\right) = 21 \cdot \frac{2}{3} \cdot \frac{6}{7} = 12$$

Euler's Totient Function

Example

$25 = 5^2$, *so*

$$\varphi(25) = 25 \cdot \left(1 - \frac{1}{5}\right) = 4 \cdot 5 = 20$$

Consider 13. *Since* $5 \nmid 13$, $\gcd(25, 13) = 1$. *Therefore* $13^{20} \equiv 1 \pmod{25}$

Example

$26 = 2 \cdot 13$, *so*

$$\varphi(26) = 26 \cdot \left(1 - \frac{1}{2}\right)\left(1 - \frac{1}{13}\right) = 12$$

Therefore the order $((\mathbb{Z}/26\mathbb{Z})^*, \otimes_{26})$ *is* 12. *Suppose that we want to find the order of* $[3]_{26}$ *in* $((\mathbb{Z}/26\mathbb{Z})^*, \otimes_{26})$. *By Lagrange's Theorem, the only possible orders are* $1, 2, 3, 4$ *and* 6. *Now,* $3^2 = 9$ *and* $3^3 = 27$, *so* $3^3 \equiv 1 \pmod{26}$. *Therefore the order of* $[3]_{26}$ *is* 3.

Bézout's Lemma

Corollary

Let $a, b \in \mathbb{Z}$ with $|a| + |b| \neq 0$. Then $\gcd(a, b) = 1$ if and only if there exists a solution to the Diophantine equation $ax + by = 1$.

Proof.

Note that $\gcd(a, b) = \gcd(a, |b|)$. If there exists a solution to the Diophantine equation $ax + by = 1$, then $[a]_{|b|} \in (\mathbb{Z}/|b|\mathbb{Z})^*$. □

Corollary

Let $a, b \in \mathbb{Z}$ with $|a| + |b| \neq 0$. If $\gcd(a, b) = d$, then

$$\gcd\left(\frac{a}{d}, \frac{b}{d}\right) = 1$$

Proof.

Note that $\frac{a}{d}, \frac{b}{d} \in \mathbb{Z}$ and there exists a solution (x_0, y_0) to the Diophantine equation $ax + by = d$. Therefore (x_0, y_0) is also a solution to the Diophantine equation $\frac{a}{d}x + \frac{b}{d}y = 1$ □

Bézout's Lemma

This last result tells us that we can always simplify fraction. I.e. any rational number can be written as $\frac{a}{b}$ where $a, b \in \mathbb{Z}$ and $\gcd(a, b) = 1$.

Theorem

$\sqrt{2}$ is irrational.

Proof.

Suppose, for a contradiction, that $\sqrt{2} = \frac{a}{b}$ with $a, b \in \mathbb{Z}$ and $\gcd(a, b) = 1$. Therefore $\frac{a^2}{b^2} = 2$, and so $a^2 = 2b^2$. So a^2 is even, and so is a. Let $k \in \mathbb{Z}$ be such that $a = 2k$. Now, $a^2 = 4k^2 = 2b^2$, so $b^2 = 2k^2$. But this implies that b is even, which contradicts the fact that $\gcd(a, b) = 1$. $\square$

Consequences of Bézout's Lemma

Theorem

Let $a, b, c \in \mathbb{Z}$ with $\gcd(a, b) = 1$. If $a \mid c$ and $b \mid c$, then $ab \mid c$.

Proof.

Assume that $a \mid c$ and $b \mid c$. Therefore there exists $r, s \in \mathbb{Z}$ such that $c = ar$ and $c = bs$. Since $\gcd(a, b) = 1$, there exists $x, y \in \mathbb{Z}$ such that $ax + by = 1$. Therefore

$$c = c(ax + by) = cax + cby = bsax + arby = ab(sx + ry)$$

so $ab \mid c$. □

Consequences of Bézout's Lemma

Theorem

(Euclid's Lemma)Let $a, b, c \in \mathbb{Z}$ with $\gcd(a, b) = 1$. If $a \mid bc$, then $a \mid c$.

Proof.

Assume that $r \in \mathbb{Z}$ is such that $bc = ar$. Since $\gcd(a, b) = 1$, there exists $x, y \in \mathbb{Z}$ such that $ax + by = 1$. Therefore

$$c = c(ax + by) = acx + bcy = acx + ary = a(cx + ry)$$

so $a \mid c$. □

Theorem

Let $p \in \mathbb{N}$ and and let $a, b \in \mathbb{Z}$. If p is prime and $p \mid ab$, then $p \mid a$ or $p \mid b$.

Proof.

Let p be prime and $p \mid ab$. If $p \nmid a$, then $\gcd(p, a) = 1$ and we can apply Euclid's Lemma. □

Fundamental Theorem of Arithmetic

Theorem

Let $p \in \mathbb{N}$ be prime. If $a_1, \ldots, a_n \in \mathbb{Z}$ and $p \mid a_1 \cdots a_n$, then there exists $1 \leq k \leq n$ such that $p \mid a_k$.

Proof.

We prove this result by induction on n. The statement is true, by the previous Theorem, when $n = 2$. Suppose that the statement is true when $n = m \geq 2$. Let $a_1, \ldots, a_m, a_{m+1} \in \mathbb{Z}$ and suppose that $p \mid a_1 \cdots a_m \cdot a_{m+1}$. So, by the previous Theorem again, $p \mid a_1 \cdots a_m$ or $p \mid a_{m+1}$. If $p \mid a_1 \cdots a_m$, then, by the induction hypothesis, we can find $1 \leq k \leq m$ such that $p \mid a_k$. This completes the proof. □

Theorem

Let $p, q_1, \ldots, q_n \in \mathbb{N}$ be primes. If $p \mid q_1 \cdots q_n$, then there exists $1 \leq k \leq n$ such that $p = q_k$.

Proof.

By the previous result, if $p \mid q_1 \cdots q_n$, then there exists $1 \leq k \leq n$ such that $p \mid q_k$, and so $p = q_k$. □

Fundamental Theorem of Arithmetic

Theorem

(Fundamental Theorem of Arithmetic) If $n \in \mathbb{N}$ with $n \geq 2$, then n can be uniquely factored into a product of primes.

Proof.

We showed previously that for every $n \in \mathbb{N}$ with $n \geq 2$, n is either prime, or the product of primes. Therefore we need to show the uniqueness of this factorisation. We prove uniqueness by induction. Suppose that $n = 2$ does not have a unique factorisation. It is impossible that $2 = p$ with $p \neq 2$, so $2 = p_1 \cdots p_m$ with $p_1, \ldots p_m$ prime and $m \geq 2$. Since, $p_1 \mid 2$, it follows, by the previous result that $p_1 = 2$. But now $1 = p_2 \cdots p_m$, which is a contradiction.

Let $n \in \mathbb{N}$ with $n \geq 2$ be least such that n does not have a unique factorisation into primes. Let $q_1 \leq \cdots \leq q_s$ and $p_1 \leq \cdots \leq p_t$ be primes such that either $s > t$ or $s = t$ and there exists $1 \leq i \leq t$ such that $q_i \neq p_i$, and

$$n = q_1 \cdots q_s = p_1 \cdots p_t$$

Fundamental Theorem of Arithmetic

Proof.

(Continued.) Now, this means that $p_1 \mid q_1 \cdots q_s$, so there exists $1 \leq k \leq s$ such that $p_1 = q_k$. Therefore $p_1 \geq q_1$. Similarily, $q_1 \mid p_1 \cdots p_t$, and so there exists $1 \leq k \leq t$ such that $q_1 = p_k$. Therefore $q_1 \geq p_1$ and $q_1 = p_1$. But this means that $q_2 \leq \cdots \leq q_s$ and $p_2 \leq \cdots \leq p_t$ are primes such that either $s > t$ or $s = t$ and there exists $2 \leq i \leq t$ such that $q_i \neq p_i$, and

$$n' = q_2 \cdots q_s = p_2 \cdots p_t$$

where $n' < n$. But this contradicts the leastness of n. $\square$

Euclidean Algorithm

Observe that the proof of Bézout's Lemma was an induction argument. This argument showed that if there exists a solution to the linear Diophantine equation $d = ax + by$ and $d \nmid a$, then there exists a solution to the linear Diophantine equation $r = ax + by$ where $0 < r < d$ and there exists $q \in \mathbb{Z}$ such that $a = qd + r$. We now turn to reverse engineering this induction argument to obtain an algorithm for solving linear Diophantine equations in two variables. However, before doing this, we will describe an algorithm for calculating the gcd of two integers.

Euclidean Algorithm

Definition

Let $a, b \in \mathbb{N}\backslash\{0\}$ with $b < a$. Recursively define

$$F_{a,b}(0) = a \ and \ F_{a,b}(1) = b$$

$$F_{a,b}(n+2) = \begin{cases} 0 & if \ F_{a,b}(n+1) = 0 \\ r & where \ (\exists q \in \mathbb{Z}) \left(\begin{array}{c} F_{a,b}(n) = qF_{a,b}(n+1) + r \\ \wedge (0 \leq r < F_{a,b}(n+1)) \end{array} \right) \\ & and \ F_{a,b}(n+1) \neq 0 \end{cases}$$

Note that this recursive definition makes sense because of the Division Algorithm. The following two properties follow immediately from the definition of $F_{a,b}$:

Lemma

Let $a, b, n \in \mathbb{N}\backslash\{0\}$ with $b < a$. If $F_{a,b}(n) \neq 0$, then $F_{a,b}(n+1) < F_{a,b}(n)$

Euclidean Algorithm

Lemma

Let $a, b, n \in \mathbb{N}\backslash\{0\}$ *with* $b < a$. *If* $F_{a,b}(n) = 0$, *then for all* $m \geq n$, $F_{a,b}(m) = 0$

Lemma

Let $a, b \in \mathbb{N}\backslash\{0\}$ *with* $b < a$. *There exists* $n \in \mathbb{N}$ *such* $F_{a,b}(n) = 0$

Proof.

Suppose, for a contradiction, that for all $n \in \mathbb{N}$, $F_{a,b}(n) \neq 0$. Let $r \neq 0$ be the least element in the range of $F_{a,b}$. Therefore there exists $m \in \mathbb{N}$ such that $F_{a,b}(m) = r$. But, since $F_{a,b}(m+1) < F_{a,b}(m) = r$, this is a contradiction. $\square$

Euclidean Algorithm

Lemma

Let $a, b \in \mathbb{N}\backslash\{0\}$ *with* $b < a$ *and let* $n \in \mathbb{N}$. *If* $F_{a,b}(n) \neq 0$, *then* $\gcd(a, b) = \gcd(F_{a,b}(n), F_{a,b}(n+1))$.

Proof.

By induction on n. The Lemma is clearly true for $n = 0$. Suppose that the Lemma is true for n. Assume that $F_{a,b}(n+1) \neq 0$. Therefore there exists $q \in \mathbb{Z}$ such that

$$F_{a,b}(n) = qF_{a,b}(n+1) + F_{a,b}(n+2)$$

Therefore, by an earlier result that was proved (before it was actually needed),

$$\gcd(F_{a,b}(n+1), F_{a,b}(n+2)) = \gcd(F_{a,b}(n), F_{a,b}(n+1))$$

And so, by the induction hypothesis,
$\gcd(F_{a,b}(n+1), F_{a,b}(n+2)) = \gcd(a, b)$. □

Euclidean Algorithm

Lemma

Let $a, b \in \mathbb{N}\backslash\{0\}$ with $b < a$. Let $n_0 \geq 2$ be least such that $F_{a,b}(n_0) = 0$. Then $\gcd(a, b) = F_{a,b}(n_0 - 1)$.

Proof.

Note that $F_{a,b}(0) \neq 0$ and $F_{a,b}(1) \neq 0$. Since n_0 is least, $F_{a,b}(n_0 - 1) \neq 0$ and

$$F_{a,b}(n_0 - 1) = \gcd(F_{a,b}(n_0), F_{a,b}(n_0 - 1)) = \gcd(a, b)$$

□

Let $a, b \in \mathbb{N}\backslash\{0\}$ with $b < a$. What does the function $F_{a,b}$ give us?

Euclidean Algorithm

It gives us a sequence $r_1, \ldots r_n \in \mathbb{N}$ with $r_n < \cdots < r_1$, for which there exists $q_1, \ldots, q_n, q_{n+1} \in \mathbb{Z}$ such that

$$\begin{aligned} a &= q_1 b + r_1 \\ b &= q_2 r_1 + r_2 \\ &\vdots \\ r_{n-2} &= q_n r_{n-1} + r_n \\ r_{n-1} &= q_{n+1} r_n \end{aligned}$$

where $r_n = \gcd(a, b)$.

$$\begin{aligned} \text{Eg. } 12378 &= 4 \cdot 3054 + 162 \\ 3054 &= 18 \cdot 162 + 138 \\ 162 &= 138 + 24 \\ 138 &= 5 \cdot 24 + 18 \\ 24 &= 18 + 6 \\ 18 &= 3 \cdot 6 \end{aligned}$$

Solving Diophantine Equations

Example

And the calculation on the previous slide shows that $\gcd(12378, 3054) = 6$

So, we an algorithm (a method) that allows us to compute $\gcd(a, b)$ for $a, b \in \mathbb{N}\backslash\{0\}$. Note that if $a, b \in \mathbb{Z}\backslash\{0\}$, then $\gcd(a, b) = \gcd(|a|, |b|)$.

Example

To find $\gcd(124, -16)$ *we compute* $\gcd(124, 16)$*:*

$$\begin{aligned} 124 &= 7 \cdot 16 + 12 \\ 16 &= 12 + 4 \\ 12 &= 3 \cdot 4 \end{aligned}$$

So, $\gcd(124, -16) = 4$

Solving Diophantine Equations

Note that in the process of computing $\gcd(a, b)$ for $a, b \in \mathbb{N}\backslash\{0\}$ we repeated applied the Division Algorithm. On the first application we obtain a solution to the Diophantine equation:

$$F_{a,b}(2) = ax + by$$

Since each successive value of $F_{a,b}$ is obtained from preceding values of $F_{a,b}$ by an application of the Divisio Algorithm, we are now in exactly the situation that we saw dealt with in Bézout's Lemma. And the proof of Bézout's Lemma shows us that all we need to do is rearrange the equations obtained in the process of executing the Euclidean algorithm to obtain a solution to the Diophantine equation:

$$ax + by = \gcd(a, b)$$

Solving Diophantine Equations

Example

In the calculation that $\gcd(12378, 3054) = 6$ *we saw that*

$$\begin{aligned}
6 &= 24 - 18 \\
&= 24 - (138 - 5 \cdot 24) \\
&= 6 \cdot 24 - 138 \\
&= 6 \cdot (162 - 138) - 138 \\
&= 6 \cdot 162 - 7 \cdot 138 \\
&= 6 \cdot 162 - 7 \cdot (3054 - 18 \cdot 162) \\
&= 132 \cdot 162 - 7 \cdot 3054 \\
&= 132 \cdot (12378 - 4 \cdot 3054) - 7 \cdot 3054 \\
&= 132 \cdot 12378 + (-535) \cdot 3054
\end{aligned}$$

So $(132, -535)$ *is a solution to the Diophantine equation:*

$$6 = 12378x + 3054y$$

Finding multiplicative inverses (mod n)

We now have the power to find multiplicative inverses in the groups $(\mathbb{Z}/n\mathbb{Z})^*$.

Example

$$124 = 13 \cdot 9 + 7$$
$$9 = 7 + 2$$
$$7 = 3 \cdot 2 + 1$$

So, $\gcd(124, 9) = 1$, *and* $[9]_{124} \in (\mathbb{Z}/124\mathbb{Z})^*$. *To find the inverse of* $[9]_{124}$ *in* $(\mathbb{Z}/124\mathbb{Z})^*$ *we need to solve the Diophantine equation:*

$$1 = 9x + 124y$$

Finding multiplicative inverses (mod n)

Example

$$
\begin{aligned}
1 &= 7 - 3 \cdot 2 \\
1 &= 7 - 3 \cdot (9 - 7) \\
1 &= 4 \cdot 7 - 3 \cdot 9 \\
1 &= 4 \cdot (124 - 13 \cdot 9) - 3 \cdot 9 \\
1 &= 4 \cdot 124 - 55 \cdot 9
\end{aligned}
$$

So $(-55, 4)$ *is a solution to* $1 = 9x + 124y$ *and* $[-55]_{124}$ *or* $[69]_{124}$ *is the inverse of* $[9]_{124}$ *in* $(\mathbb{Z}/124\mathbb{Z})^*$

What if we want to solve a linear Diophantine equation in the form: $d = ax + by$? We know that if $d = \gcd(a, b)$, then there exists a solution to this equation. And, if $d = 1$ and $\gcd(a, b) \neq 1$, then no solution exists.

Linear Diophantine Equations

Theorem

Let $a, b, c \in \mathbb{Z}$. There exists a solution to the linear Diophantine equation $ax + by = c$ if and only if $\gcd(a, b) \mid c$.

Proof.

($\Rightarrow$): Suppose that (x_0, y_0) is a solution to $ax + by = c$. Let $d = \gcd(a, b)$. Therefore, let $r, t \in \mathbb{Z}$ such that $a = dr$ and $b = dt$. So,

$$c = ax_0 + by_0 = drx_0 + dty_0 = d(rx_0 + ty_0)$$

which shows that $d \mid c$.

($\Leftarrow$): Let $d = \gcd(a, b)$ and assume that $d \mid c$. Using Bézout's Lemma, let (x_0, y_0) be a solution to $d = ax + by$. Let $s \in \mathbb{Z}$ be such that $c = ds$. Therefore

$$c = sd = s(ax_0 + by_0) = a(sx_0) + b(sy_0)$$

so (sx_0, sy_0) is a solution to $c = ax + by$. □

Linear Diophantine Equations

Theorem

Let $a, b, c \in \mathbb{Z}$ with $\gcd(a, b) \mid c$. Let (x_0, y_0) be a solution to $ax + by = c$. For all $t \in \mathbb{Z}$, (x_t, y_t) is a solution to $ax + by = c$ where

$$x_t = x_0 + \frac{b}{d}t \ and \ y_t = y_0 - \frac{a}{d}t$$

Moreover, if (x', y') is a solution to $ax + by = c$, then there exists a $t \in \mathbb{Z}$ such that $(x', y') = (x_t, y_t)$.

Proof.

Let $t \in \mathbb{Z}$ and let

$$x_t = x_0 + \frac{b}{d}t \text{ and } y_t = y_0 - \frac{a}{d}t$$

Linear Diophantine Equations

Proof.

(Continued.) Now,

$$ax_t + by_t = a\left(x_0 + \frac{b}{d}t\right) + b\left(y_0 - \frac{a}{d}t\right)$$

$$= ax_0 + \frac{ab}{d}t + by_0 - \frac{ab}{d}t = ax_0 + by_0 = c$$

which shows that (x_t, y_t) is a solution to $ax + by = c$.
We turn to showing that every solution can be written in this form. Let (x', y') be a solution to $ax + by = c$. Therefore

$$ax_0 + by_0 = c = ax' + by'$$

Therefore $a(x_0 - x') = b(y' - y_0)$. Let $d = \gcd(a, b)$, and let $r, s \in \mathbb{Z}$ such that $a = dr$ and $b = ds$. Note that we showed previously that, since $d = \gcd(a, b)$, this implies that $\gcd(r, s) = 1$.

Linear Diophantine Equations

Proof.

(Continued.) Dividing through by d we get:

$$r(x' - x_0) = s(y_0 - y')$$

And so $r \mid s(y_0 - y')$. Now, using Euclid's Lemmma and the fact that $\gcd(r, s) = 1$, we conclude that $r \mid (y_0 - y')$. Let $t \in \mathbb{Z}$ be such that $y_0 - y' = rt$. Therefore, since $r(x' - x_0) = s(y_0 - y')$,

$$x' - x_0 = st$$

This yields

$$x' = x_0 + st = x_0 + \frac{b}{d}t \text{ and } y' = y_0 - rt = y_0 - \frac{a}{d}t$$

□

Linear Diophantine Equations

Example

Consider the linear Diophantine equation: $1000 = 172x + 20y$.

$$\begin{aligned} 172 &= 8 \cdot 20 + 12 \\ 20 &= 12 + 8 \\ 12 &= 8 + 4 \\ 8 &= 2 \cdot 4 \end{aligned}$$

Since $\gcd(172, 20) = 4$ *and* $4 \mid 1000$, *this Diophantine equation has a solution. Now,*

$$\begin{aligned} 4 &= 12 - 8 \\ 4 &= 12 - (20 - 12) \\ 4 &= 2 \cdot 12 - 20 \\ 4 &= 2 \cdot (172 - 8 \cdot 20) - 20 \\ 4 &= 2 \cdot 172 - 17 \cdot 20 \end{aligned}$$

Linear Diophantine Equations

Example

So $(2, -17)$ *is a solution to* $4 = 172x + 20y$. *Therefore* $(500, -4250)$ *is a solution to* $1000 = 172x + 20y$. *Moreover, every solution* (x', y') *of* $1000 = 172x + 20y$ *is in the form*

$$x' = 500 + \frac{20}{4}t = 500 + 5t \ \ and \ y' = -4250 - \frac{172}{4}t = -4250 - 43t$$

where $t \in \mathbb{Z}$. *If, for example, we are interested in solutions* (x', y') *where* $x', y' > 0$, *then we need*

$$500 + 5t > 0 \ \ and \ \ -4250 - 43t > 0$$

This becomes $-100 < t < -(98 + \frac{36}{43})$, *which means that* $t = -99$ *is the only possibility. Therefore* $(5, 7)$ *is the unique positive solution to the Diophantine equation* $1000 = 172x + 20y$.

Linear Congruency Equations

A **linear congruence** is an equation in the form

$$a \cdot x \equiv b \pmod{n}$$

If $\gcd(a, n) = 1$, then $[a]_n \in (\mathbb{Z}/n\mathbb{Z})^*$. In this case we can find $[c]_n \in (\mathbb{Z}/n\mathbb{Z})^*$ such that

$$[ac]_n = [a]_n \otimes_n [c]_n = [1]_n$$

or, in other words, $ac \equiv 1 \pmod{n}$. It now follows, by multiplying through by b, that

$$abc \equiv b \pmod{n}$$

and so bc is a solution to the linear congruence $a \cdot x \equiv b \pmod{n}$. What happens when $\gcd(a, n) \neq 1$? For example, 4 is a solution to the linear congruence $6x \equiv 3 \pmod{21}$, and $\gcd(6, 21) = 3$

Linear Congruency Equations

Example

Suppose that we wish to solve the linear congruence

$$18x \equiv 30 \pmod{42}$$

Note that a solution to this linear congruence could be obtained from a solution to the Diophantine equation $30 = 18x + 42y$. *Now,*

$$42 = 2 \cdot 18 + 6$$
$$18 = 3 \cdot 6$$

so $\gcd(42, 18) = 6$ *and* $6 \mid 30$. *And,* $6 = 42 - 2 \cdot 18$. *So* $(-2, 1)$ *is a solution to* $6 = 18x + 42y$, *and* $(-10, 5)$ *is a solution to* $30 = 18x + 42y$. *So*

$$18 \cdot (-10) \equiv 30 \pmod{42} \text{ } or \text{ } 18 \cdot 32 \equiv 30 \pmod{42}$$

Linear Congruency Equations

Example

The solutions -10 *and* 32 *are equivalent* $(\mathtt{mod}\ 42)$*, however, using our characterisation of the solutions to linear Diophantine equations, note that:*

$$x' = -2 + \frac{42}{6} = 5 \ and \ y' = 1 - \frac{18}{6} = -2$$

are also solutions to $6 = 18x + 42y$*. Therefore* $(25, -10)$ *is also a solution to* $30 = 18x + 42y$*. And*

$$18 \cdot 25 \equiv 30 \ (\mathtt{mod}\ 42)$$

Similarily, $(-45, 20)$ *is a solution to* $30 = 18x + 42y$*. And so,* -45 *or* -3 *is such that*

$$18 \cdot (-3) \equiv 30 \ (\mathtt{mod}\ 42)$$

So, the linear congruence $18x \equiv 30 \ (\mathtt{mod}\ 42)$ *has multiple solutions that are not congruent* $(\mathtt{mod}\ 42)$*.*

Linear Congruency Equations

Theorem

Let $a, b \in \mathbb{Z}$ and let $n \in \mathbb{N}\backslash\{0\}$. The linear congruence equation

$$ax \equiv b \pmod{n}$$

has a solution if and only if $\gcd(a, n) \mid b$. *Moreover, if* $\gcd(a, n) \mid b$, *then the linear congruence equation has exactly* $\gcd(a, n)$ *solutions that are mutually incongruent* $\pmod{n}$.

Proof.

The linear congruence equation $ax \equiv b \pmod{n}$ has a solution if and only if there is a solution to the linear Diophantine equation $b = ax + ny$. And, as we showed previously, there is a solution to the linear Diophantine equation $b = ax + ny$ if and only if $\gcd(a, n) \mid b$.

Linear Congruency Equations

Proof.

(Continued.) Let $d = \gcd(a, n)$ and suppose that $d \mid b$. We now turn to showing that there are d solutions of the linear congruence equation $ax \equiv b \pmod{n}$ that are mutually incongruent. Let (x_0, y_0) be a solution to the Diophantine equation $b = ax + ny$. Therefore all solutions to the Diophantine equation $b = ax + ny$ are in the form

$$x_t = x_0 + \frac{n}{d}t \text{ and } y_t = y_0 - \frac{a}{d}t$$

where $t \in \mathbb{Z}$.

We claim that $x_0, \ldots, x_{d-1}$ are mutally incongruent $\pmod{n}$. Suppose, for a contradiction, that there exists $0 \leq t_1 < t_2 < d$ such that

$$x_0 + \frac{n}{d}t_1 \equiv x_0 + \frac{n}{d}t_2 \pmod{n}$$

But then $\frac{n}{d}t_1 \equiv \frac{n}{d}t_2 \pmod{n}$.

Linear Congruency Equations

Proof.

(Continued.) Let $r \in \mathbb{Z}$ be such that

$$\frac{n}{d}t_2 - \frac{n}{d}t_1 = nr$$

And, recalling that $\gcd(a, n) = d$ and in particular $d \mid n$, let $s \in \mathbb{Z}$ such that $n = ds$. Therefore

$$\frac{ds}{d}t_2 - \frac{ds}{d}t_1 = dsr$$

So $t_2 - t_1 = dr$ and $d \mid (t_2 - t_1)$, which is a contradiction. This shows that $x_0, \ldots, x_{d-1}$ are all incongruent $(\mathtt{mod}\ n)$.

Finally, we show that for all $t \in \mathbb{Z}$, the solution x_t is congruent to one of the solutions $x_0, \ldots, x_{d-1}$. Let $t \in \mathbb{Z}$. Using the Division Algorithm we can find $q \in \mathbb{Z}$ and $0 \leq r < d$ such that $t = qd + r$. □

Linear Congruency Equations

Proof.

(Continued.) Now,

$$x_t = x_0 + \frac{n}{d}t = x_0 + \frac{n}{d}(qd + r) = x_0 + \frac{n}{d}r + nq = x_r + nq$$

So, $n \mid (x_t - x_r)$ and $x_t \equiv x_r \pmod{n}$ where $0 \leq r < d$.
This completes the proof. □

Example

We saw previously that -3 *is a solution to the linear congruence equation* $18x \equiv 30 \pmod{42}$. *We now know that there are exactly six solutions to this equation that are mutally incongruent* $\pmod{42}$.

$$-3 + 7 = 4 \qquad -3 + 14 = 11$$
$$-3 + 21 = 18 \qquad -3 + 28 = 25$$
$$-3 + 35 = 32$$

System's of Congruences

Sunzi's problem expresses a system of congruences:

$$\begin{aligned} x &\equiv 2 \pmod 3 \\ x &\equiv 3 \pmod 5 \\ x &\equiv 2 \pmod 7 \end{aligned}$$

Theorem

(Chinese Remainder Theorem) Let $m_1, \ldots, m_n \in \mathbb{N}\backslash\{0\}$ *be pairwise relatively prime and let* $a_1, \ldots, a_n \in \mathbb{Z}$*. Then the system of congruences*

$$\begin{aligned} x &\equiv a_1 \pmod{m_1} \\ x &\equiv a_2 \pmod{m_2} \\ &\vdots \\ x &\equiv a_n \pmod{m_n} \end{aligned} \tag{5}$$

has a unique solution $\pmod m$ *where* $m = m_1 \cdots m_n$.

The Chinese Remainder Theorem

Proof.

We first prove the existence of a solution. For all $1 \leq k \leq n$, define

$$M_k = \frac{m}{m_k} = \prod_{i \neq k} m_i$$

Note that since $m_1, \ldots, m_n$ are pairwise relatively, it follows that for all $1 \leq k \leq n$, $\gcd(m_k, M_k) = 1$. Therefore for all $1 \leq k \leq n$, $[M_k]_{m_k} \in (\mathbb{Z}/m_k\mathbb{Z})^*$ and there exists $y_k \in \mathbb{Z}$ such that

$$[M_k y_k]_{m_k} = [M_k]_{m_k} \otimes_{m_k} [y_k]_{m_k} = [1]_{m_k} \text{ or } M_k y_k \equiv 1 \pmod{m_k}$$

$$\text{Let } x = \sum_{k=1}^{n} a_k M_k y_k$$

Since for all $1 \leq i, j \leq n$, if $i \neq j$, then $M_i \equiv 0 \pmod{m_j}$, it follows that x is a solution to (**??**).

The Chinese Remainder Theorem

Proof.

We now turn to showing uniqueness. Let $x, x' \in \mathbb{Z}$ be such that for all $1 \leq k \leq n$,

$$x \equiv a_k \equiv x' (\mathtt{mod}\ m_k)$$

We will show that x and x' must be congruent $(\mathtt{mod}\ m)$. Now, for all $1 \leq k \leq n$, $m_k \mid (x - x')$. An elementary induction argument applied to one of the consequences of Bézout's Lemma that we proved shows that since for all $1 \leq i, j \leq n$ with $i \neq j$, $\gcd(m_i, m_j) = 1$,

$$m = m_1 \cdots m_n \mid (x - x')$$

This shows that

$$x \equiv x' (\mathtt{mod}\ m)$$

□

The Chinese Remainder Theorem

Example

To address Sunzi's original question: Let $m_1 = 3$, $m_2 = 5$ *and* $m_3 = 7$, *and let* $a_1 = 2$, $a_2 = 3$ *and* $a_3 = 2$. *Let* $m = 3 \cdot 5 \cdot 7 = 105$ *and consider*

$$M_1 = \frac{m}{m_1} = 35 \ and \ M_2 = \frac{m}{m_2} = 21 \ and \ M_3 = \frac{m}{m_3} = 15$$

Note that $[M_1]_3 = [-1]_3$ *and* $[-1]_3$ *is its own inverse in* $(\mathbb{Z}/3\mathbb{Z})^*$, *so let* $y_1 = -1$. *Now,* $[M_2]_5 = [1]_5$, *so let* $y_2 = 1$. *And* $[M_3]_7 = [1]_7$, *so let* $y_3 = 1$. *Therefore, let*

$$x = -2 \cdot 35 + 3 \cdot 21 + 2 \cdot 15 = -70 + 63 + 30 = 23$$

So 23 *is the smallest natural number solution to Sunzi's problem. And* $[23]_{105}$ *is the unique congruence class of solutions.*

The Chinese Remainder Theorem

Note that the assumptions of the remainder theorem are required:

Example

There is no solution to the system

$$x \equiv 1 \pmod 4 \ and \ x \equiv 2 \pmod 6$$

Because if $x \equiv 1 \pmod 4$, *then* x *is odd, and if* $x \equiv 2 \pmod 6$, *then* x *is even.*

Example

Both 6 *and* 14 *satisfy*

$$x \equiv 2 \pmod 4 \ and \ x \equiv 6 \pmod 8$$

which shows that this system does not have a unique solution $\pmod{32}$.

The Chinese Remainder Theorem

Note that the uniqueness clause in the Remainder Theorem is important and useful. Using this result, we have a new method for solving linear congruences.

Example

Consider $43x \equiv 12 \pmod{56}$. *A solution to this problem can be obtained by solving the system of congruences:*

$$x \equiv 5 \pmod{7} \ and \ 3x \equiv 4 \pmod{8}$$

Recall that every element in the group $(\mathbb{Z}/8\mathbb{Z})^*$ *has order* 2. *So, we need to solve this system:*

$$x \equiv 5 \pmod{7} \ and \ x \equiv 4 \pmod{8}$$

Now, let $M_1 = 8$ *and* $M_2 = 7$. *And, let* $y_1 = 1$ *and* $y_2 = 7$. *Now, let*

$$x = 5 \cdot 8 + 4 \cdot 7 \cdot 7 = 236 \equiv 12 \pmod{56}$$

Wilson's Theorem

We conclude this section with a result dating from the 18^{th} century.

Theorem

(Wilson's Theorem) Let $p \in \mathbb{N}$ be prime. Then

$$(p-1)! \equiv -1 \pmod{p}$$

Proof.

It is clear that

$$(2-1)! = 1 \equiv 1 \equiv -1 \pmod{2} \text{ and } (3-1)! = 2 \equiv -1 \pmod{3}$$

Therefore, assume that $p > 3$. Let $a \in [p] \backslash \{0\}$. Now, $[a]_p \in (\mathbb{Z}/p\mathbb{Z})^*$, and there exists a unique $a^{-1} \in [p] \backslash \{0\}$ such that

$$[a^{-1}]_p \otimes_p [aa^{-1}]_p = [1]_p$$

We claim that $a = a^{-1}$ if and only if $a = 1$ or $a = p - 1$.

Wilson's Theorem

Proof.

Now,

$$1 \cdot 1 = 1 \equiv 1 \pmod{p} \text{ and } (p-1)^2 = p^2 - 2p + 1 \equiv 1 \pmod{p}$$

So, suppose that $a^2 \equiv 1 \pmod{p}$. This means that $p \mid (a^2 - 1)$ or $p \mid (a-1)(a+1)$. It now follows from Euclid's Lemma that $p \mid (a+1)$ or $p \mid (a-1)$. But this means that $a \equiv 1 \pmod{p}$ or $a \equiv -1 \pmod{p}$, so $a = 1$ or $a = p-1$. This shows that the only element of order 2 in $(\mathbb{Z}/p\mathbb{Z})^*$ is $[p-1]_p$.

Now, consider $2, 3, \ldots, p-2$. Each of these numbers represents different elements of the group $(\mathbb{Z}/p\mathbb{Z})^*$, and for each $2 \leq k \leq p-2$, there exists $2 \leq l \leq p-2$ with $l \neq k$ and such that $[l]_p$ is the inverse for $[k]_p$ in $(\mathbb{Z}/p\mathbb{Z})^*$. That is to say, every element of the list $2, 3, \ldots, p-2$ can be paired with another element of the list $2, 3, \ldots, p-2$ that is its inverse in the group $(\mathbb{Z}/p\mathbb{Z})^*$.

Wilson's Theorem

Proof.

It follows that

$$[2]_p \cdot [3]_p \cdots [p-2]_p = [1]_p$$

Multiplying by $[1]_p$ and $[p-1]_p$, we get

$$[1]_p \cdot [2]_p \cdots [p-2]_p \cdot [p-1]_p = [p-1]_p$$

Or, $(p-1)! \equiv p-1 \equiv -1 \pmod{p}$. □

Theorem

There are infinitely many composite numbers in the form $n! + 1$.

Proof.

Let $p \in \mathbb{N}$ be prime and let $n = p - 1$. It follows from Wilson's Theorem that $n! \equiv -1 \pmod{p}$. Therefore, $p \mid (n! + 1)$. The result now follows from the fact that there are infinitely many primes. □

Are there infinitely many primes in the form $n! + 1$? This is still an open question.

Algorithms

In the previous part we have encountered many different "methods" that can be used for performing calculations, such as the Euclidean algorithm, the procedure used for writing bijections as the composition of disjoint cylcles, etc.

In the present part, we will discuss how these abstract methods can be transformed into concrete *algorithms*, suitable for implementation on a computer. In doing so, we will view these algorithms as mathematical objects about which concrete mathematical questions can be asked, focusing particularly on the time (number of individual operations) that an algorithm needs to perform its task.

Let us first give an informal definition:

Definition

An algorithm is a finite sequence of precise instructions for performing a computation or for solving a problem.

A more formal definition follows.

Definition of an Algorithms

An algorithm is an effective method expressed as a finite list of well-defined instructions for calculating a function.

*Starting from an **initial state and initial input** (perhaps empty), the instructions describe a computation that, when executed, proceeds through a **finite** number of **well-defined** successive states, eventually producing **output** and terminating at a final ending state.*

The transition from one state to the next is not necessarily deterministic; some algorithms, known as randomized [sic] algorithms, incorporate random input.

Wikipedia contributors, "Algorithm," *Wikipedia, The Free Encyclopedia*, http://en.wikipedia.org/w/index.php?title=Algorithm&oldid=629026201 (accessed October 10, 2014).

Example: Finding the Maximum

Example

One possible algorithm for finding the maximum of a finite sequence of integers is as follows:

1. *Set the temporary maximum equal to the first integer in the sequence.*
2. *Compare the next integer in the sequence to the temporary maximum, and if it is larger than the temporary maximum, set the temporary maximum equal to this integer.*
3. *Repeat the previous step if there are more integers in the sequence.*
4. *Stop when there are no integers left in the sequence. The temporary maximum at this point is the largest integer in the sequence.*

Pseudocode

The description of the algorithm in Example **??** is quite cumbersome. In fact, most algorithms are intended to be realized as programs on computers. There are several programming languages that can implement such algorithms, such as Pascal or C++. It is assumed that you have at least a passing familiarity with such programming languages.

Instead of using a specific language, we will use express algorithms using **pseudocode**.

The advantage of pseudocode is that it is as succinct as a computer program but general enough to be universally understood.

Finding the Maximum

The algorithm that we described earlier for finding the maximal element in a list of integers can be expressed in pseudocode as follows:

Algorithm

(Maximal Element)

Input: *n integers,* $a_1, a_2, \ldots, a_n$
Output: $\max(a_1, \ldots, a_n)$

```
1 max ← a_1;
2 for i ← 2 to n do
3 |   if max < a_i then
4 |   |   max ← a_i;
5 |   end if
6 end for
7 return max
```

Properties of Algorithms

There are several properties that algorithms usually share. These include:

- **Input.** An algorithm has input values from a specified set;
- **Output.** For given input, the algorithm produces output values from a specified set;
- **Definiteness.** The steps of the algorithm are defined precisely;
- **Correctness.** For each input, the algorithm produces the correct output values;
- **Finiteness.** For given input, the algorithm produces output after a finite number of steps;
- **Effectiveness.** Each step of the algorithm can be performed exactly;
- **Generality.** The algorithm is generally applicable, not just for certain input values.

Properties of Algorithms

Example

Our algorithm that finds the maximum element in a list of integers has all of these properties:

- **Input.** *The input is a finite sequence of integers;*
- **Output.** *The output is the largest integer of the set;*
- **Definiteness.** *Only assignments, a finite loop and conditional statements occur; these are well-defined;*
- **Finiteness.** *The algorithm uses a finite number of steps because it terminates after all integers have been examined;*
- **Effectiveness.** *Each step of the algorithm can be performed exactly;*
- **Generality.** *The algorithm is generally applicable, because it can be used to find the maximum of any finite sequence of numbers;*
- **Correctness.** *The algorithm is correct, because it does return the maximum of the integers in the sequence. The proof of correctness is based on induction and requires a bit more background; we will study this in more detail and return to this example later.*

Iterative and Recursive Algorithms

We can generally divide algorithms into two types:

- **Iterative algorithms** repeat a key step until a problem is solved.
- **Recursive algorithms** solve a problem by reducing it to an instance of the same problem with smaller input.

An iterative algorithm to calculate the value of $n!$, $n \in \mathbb{N}\backslash\{0\}$, is as follows:

Algorithm

(Factorial – Iterative Version)

Input: *n, a positive integer*
Output: $n!$

```
1 factorial ← 1;
2 for i ← 1 to n do
3 |   factorial ← i · factorial ;
4 end for
5 return factorial
```

Recursive and Iterative Algorithms

A recursive algorithm to calculate the value of $n!$ is as follows:

Algorithm

(Factorial – Recursive Version)

Input: *n, a positive integer*
Output: $n!$

```
1 Function Factorial(n):
2     if n = 1 then
3         factorial ← 1
4     else
5         factorial ← n · Factorial(n − 1)
6     end if
7     return factorial
8 end
```

We will now study several examples of algorithms.

Search Algorithms

Search algorithms address the problem of finding an element x within a tuple $(a_1, \ldots, a_n)$ of elements, if it is present.

We consider x to be found if we know which $a_k = x$, i.e., if we know the value of k. The following algorithm is quite simple, comparing each element in order with x and terminating when a match is found.

Algorithm

(Linear Search)

Input: *x an integer and $a_1, a_2, \ldots, a_n$, n distinct integers*
Output: *i such that $a_i = x$, or 0 if no such i exists*

```
1 i ← 1;
2 while i ≤ n and x ≠ a_i do
3 |   i ← i + 1;
4 end while
5 if i ≤ n then location ← i ;
6 else location ← 0;
7 return (location)
```

Binary Search

If the tuple $(a_1, \ldots, a_n)$ consists of objects in ascending order (e.g., numbers sorted from smallest to largest) then another strategy is to successively split the list in half and only search with the sublists whose elements might contain x.

Example

To search for the number 7 in the list $(2, 3, 7, 8, 10, 11, 13, 16)$ *we first split the list into two smaller lists,* $(2, 3, 7, 8)$ *and* $(10, 11, 13, 16)$*. Since* $7 \not> 8$ *we do not search the second list. We next split the first list in two,* $(2, 3)$ *and* $(7, 8)$*. Since* $7 > 3$*, we search the second list, which we split in two,* (7) *and* (8)*. Since* $7 \not> 7$*, we now search the first list. Since it contains only one element, we make a comparison and find 7 in this list.*

Definition

The **floor function** *is the function that takes a real number x as its input and returns the largest integer, denoted $\lfloor x \rfloor$, that is less than or equal to x. The* **cieling function** *is the function that takes a real number x as its input and returns the smallest integer, denoted $\lceil x \rceil$, that is less than or equal to x.*

Binary Search

Algorithm

(Binary Search)

Input: x *an integer and* $a_1 < a_2 < \ldots < a_n$, n *distinct ordered integers*

Output: i *such that* $a_i = x$, *or 0 if no such* i *exists*

```
1  i ← 1;
2  j ← n ;
3  while i < j do
4  |   m ← ⌊(i + j)/2⌋;
5  |   if x > a_m then i ← m + 1;
6  |   else j ← m;
7  end while
8  if  x = a_i then location ← i;
9  else location ← 0;
10 return location
```

Sorting Algorithms

Another very important problem is the sorting of unsorted lists (tuples). We assume here that we are sorting numbers, but the following procedures are applicable whenever we have an ordering for a set of objects (e.g., lexicographic ordering for words).

We first introduce a very simple, but not usually very fast algorithm called the **bubble sort**. It essentially works by pairwise comparisons and transpositions.

Example

To put the list $(3, 2, 1, 4)$ *in ascending order, we first compare the first and the second element. Since* $3 > 2$*, we interchange the elements, yielding* $(2, 3, 1, 4)$*. Then we compare the second and the third element of this list. Since* $3 > 1$*, we interchange and get* $(2, 1, 3, 4)$*. Comparing the third and the fourth elements, we obtain* $3 \not> 4$*, so we do not interchange any elements. This completes the first pass.*

In the second pass, we again compare the first and the second elements and interchange them. The list is now in order.

Bubble Sort

Algorithm

(Bubble Sort)

Input: $a_1, \ldots, a_n$, *n unsorted elements*
Output: *all the* $a_i, 1 \leq i \leq n$ *in increasing order*

```
1 for i ← 1 to n − 1 do
2 |   for j ← 1 to n − i do
3 |   |   if a_j > a_{j+1} then interchange a_j and a_{j+1};
4 |   end for
5 end for
6 return (a_1, ..., a_n) in increasing order.
```

The name bubble sort is used because the largest values "bubble" to the top. The sorting procedure is illustrated by a video taken from

http://www.youtube.com/playlist?list=PLZh3kxyHrVp_AcOanN_jpuQbcMVdXbqei

More information on these videos can be found at

http://panthema.net/2013/sound-of-sorting/

Insertion Sort

Another algorithm is called **insertion sort**. This algorithm looks at the jth element of an n element list and inserts it in the correct position in the first $j-1$ elements.

Algorithm

(Insertion Sort)

Input: $a_1, \ldots, a_n$, n *unsorted elements*
Output: *all the* $a_i, 1 \leq i \leq n$ *in increasing order*

```
1 for j ← 2 to n do
2 |   i ← 1;
3 |   while a_j > a_i do i ← i + 1;
4 |   m ← a_j;
5 |   for k ← 0 to j − i − 1 do a_{j−k} ← a_{j−k−1};
6 |   a_i ← m
7 end for
8 return (a_1, ..., a_n) in increasing order
```

Optimisation Problems and Greedy Algorithms

One of the main interests in the study of algorithms is to find those that solve problems in the most efficient way possible. Problems seeking an optimal solution are known as **optimisation problems**.

In general, a problem is solved in different steps. E.g., the bubble sort algorithm compares two numbers and then interchanges them or leaves them as is. This is one step of the entire algorithm. Of course, taking the (locally) optimal course at every step does not automatically leads to a (globally) optimal solution.

Nevertheless, this is often the case and such algorithms are very important. Algorithms that find an optimal solution at every step are called **greedy algorithms**.

Optimisation of Change-Making

Example

The coins, of denomination less than 1$, that are issued in the United Kingdom are: 1p, 2p, 5p, 10p, 20p and 50p
Consider the problem of issuing change of a certain amount less than 1$with these coins. We wish to address the optimisation problem of whereby the change should be issued using the least number of coins possible.
For example, if 31p of change needs to be given then this could be accomplished with thirty-one 1p coins, or six 5p coins and one 1p coin, or two 5p coins, ten 2p coins and a 1p coin. The optimal solution would be: one 20p coin, one 10p coin and one 1p coin (three coins)

Greedy Change-Making

If n is the amount of change needed, then, at each stage, add the largest coin possible without exceding the total n until the total n is realised.

Algorithm

(Greedy Change-making)

Input: $c_1 > c_2 > \ldots > c_r$, *coin denominations and* n *a positive integer*

Output: M *a tuple of the* c_i $(1 \leq i \leq r)$, *with sum of all its elements being* n

```
1 M ← ∅;
2 for i ← 1 to r do
3 |   while n ≥ c_i do
4 |   |   add a coin with value c_i to M;
5 |   |   n ← n − c_i;
6 |   end while
7 end for
8 return M
```

Greedy Change-Making

This algorithm is greedy, because at each step it takes the greatest allowable denomination and adds it to the change. However, depending on the denominations of coins available, it is not necessarily optimal.
For example, suppose that the United Kingdom only issues 1p, 10p, 25p and 50p coins. And suppose that 30p needs to be issued. This algorithm would issue this change as one 25p and five 1p coins (six coins). But the optimal solution is three 10p coins.
This means that we need to prove that our greedy algorithm gives the optimal solution...

Lemma

If $0 \leq n < 100$, *then n pence in change using 1p, 2p, 5p, 10p, 20p and 50p coins with the fewest number coins possible has at most one 1p coin, two 2p coins, one 5p coin, one 10p coins, and two 20p coins. Moreover, the optimal change can not contain two 2p coins and a 1p coin, or two 20p coins and a 10p coin. The change in 1p, 2p, 5p, 10p and 20p coins cannot exceed 49p. The change in 1p, 2p, 5p and 10p coins can not exceed 19p. The change in 1p, 2p and 5p coins can not exceed 9p. And, the change in 1p and 2p coins can not exceed 4p.*

Greedy Change-Making

Proof.

Proof by contradiction. Let $0 \leq n < 100$. It should be clear that the optimal solution to giving n cents can not contain more than one 1p coin, two 2p coins, one 5p coin, one 10p coins, and two 20p coins. It also be clear that if two 2p coins and a 1p coin were present, then these could be replaced by a 5p coin. And if two 20p coins and a 10p coin are present then this could be replaced by a 50p coin.

Now, using these facts we see that largest amount of change that can be issued as part of an optimal solution in 20p, 10p, 5p, 2p and 1p coins is: two 20p coins, one 5p coin, and two 2p coins, which amounts to 49p. The largest amount of change that can be issued as part of an optimal solution in 10p, 5p, 2p and 1p coins is: one 10p, one 5p coin, and two 2p coins, which amounts to 19p. The largest amount of change that can be issued as part of an optimal solution in 5p, 2p and 1p coins is: one 5p coin, and two 2p coins, which amounts to 9p. The largest amount of change that can be issued as part of an optimal solution in 2p and 1p coins is: two 2p coins, which amounts to 4p. □

Greedy Change-Making

Theorem

Given the demoninations 1p, 2p, 5p, 10p, 20p and 50p the greedy algoritm produces change using the fewest coins possible.

Proof.

Let q be the number of 50p coins issued by the greedy algorithm and let q' be the number of 50p coins in the optimal solution. The greedy algorithm ensures that $q \geq q'$. But it can not be the case that $q' < q$, becasue the optimal solution can not contain more than 49p using 1p, 2p, 5p, 10p and 20p coins. This shows that $q = q'$. Once we have shown that the number of 50p coins in the output of the greedy algorithm is equal to the number of 50p coins in the optimal solution, we turn our attention to the 20p coins. The same arugment shows that the number of 20p coins must be the same, because the optimal solution can not contain more that 19p in the denominations 1p, 2p, 5p and 10p. Exactly the same argument also shows that the number of 10p, 5p, 2p and 1p coins must be the same. □

Computational Complexity

Implementing an algorithm generally requires resources, both of a physical and a temporal type. That is, if we want to run an algorithm such as the algorith we wrote for finding the maximum element of a list, we will need computer hardware, in particular memory, as well as the time to run the algorithm. Different algorithms may have different requirements; one algorithm may require less time but more hardware resources than another.

These requirements can be compared by considering the **space complexity** and the **time complexity** of a given algorithm. The memory requirements (space complexity) depend on the data structures used. We will not cover this aspect of computer science in our course, and instead restrict our analysis to the time complexity of algorithms.

Time complexity is generally analysed by considering the number of operations an algorithm requires for completion instead of actual time. This allows a comparison independent of specific computer characteristics (processor speeds, parallel vs. serial processing etc.)

Time Complexity

Example

Consider the time complexity of the algorithm for finding a maximal element in a list of integers. At each step, two comparisons are made:

- *one comparison to determine whether the end of the list has been reached and*
- *one comparison of the temporary maximum with the current element.*

If the list has n elements, these comparisons are performed $n-1$ times, with an additional comparison to exit the loop. Therefore, a total of $2n-1$ comparisons are made.

It is not essential how many computer operations are necessary to implement a single "comparison" or how long each comparison takes. What is important is that when the list size doubles, so does (essentially) the number of comparisons and hence the time needed for the algorithm. The factor 2 and the subtraction of 1 is not relevant in practice.

Measuring Time Complexity

We need to develop a formal system for analysing the time comlexity of algorithms.

- The time complexity of an algorithm A is measure relative some **basic operation** using a function $f : \mathbb{N} \longrightarrow \mathbb{N}$ thats input is a measure of the length of the input of the algorithm and output is a count of basic operations. Ideally this basic operation will run in a time that is independent of the input of the algorithm. With much more work we could define a theoretical model of a computer and thus define a basic opertation to be a single step on this abstract computer, but unfortunately we do not have time to do this theoretical work.
- A **worst-case** measure of the time complexity of an algorithm A with respect to a given basic operation is a function $f : \mathbb{N} \longrightarrow \mathbb{N}$ such that on any input of length n, then algorithm will use at most $f(n)$ basic operations.
- An **average-case** measure of the time complexity of an algorithm A with respect to a given basic operation is a function $f : \mathbb{N} \longrightarrow \mathbb{N}$ such that on any input of length n, then algorithm will use on average $f(n)$ basic operations.

Time Complexity

So, our previous analysis shows that the function $f : \mathbb{N} \longrightarrow \mathbb{N}$, defined by $f(n) = 2n - 1$, is a measure of the worst-case time complexity of our algorithm for finding the maximal element in a list of integers. The measure of length used in our analysis is the length of the list of integers given as input to the algorithm and the basic operations that we are counting are the numbers of comparisons of order.

Example

In general, performing average-case analysis is much harder than worst case analysis. Consider the Linear Search Algorithm that seeks a number x in a list $(a_1, \ldots, a_n)$ and assume that x appears in the list. If x occurs at the beginning of the list then as few as 3 comparisons are needed to locate x. An additional 2 comparisons are required for every place that x appears down the list. It follows that $f : \mathbb{N} \longrightarrow \mathbb{N}$ defined by

$$f(n) = \frac{\sum_{k=1}^{n}(2k+1)}{n} = n + 2$$

is a measure of the average-case time complexity of Linear Search.

The Landau Symbol O

We now need a means of classifying our measures of time complexity.

Definition

Let A be $\mathbb{R}$ or $\mathbb{N}$. Let $f : A \longrightarrow \mathbb{R}$ and $g : A \longrightarrow \mathbb{R}$. We say f is $O(g)$, pronouced "f is big-oh of g", if there exists $k, C \in \mathbb{N}$ such that for all $x \in A$ with $x > k$, $|f(x)| \leq C|g(x)|$. We call O the **Landau symbol** *big-oh.*

We have stated this definition for functions defined on both $\mathbb{R}$ and $\mathbb{N}$, however we will primarily be considering functions defined on $\mathbb{N}$ in this part of the course. That said, it is often useful to visualise functions defined on $\mathbb{N}$ as restriction of functions defined on $\mathbb{R}$ in order to prove big-oh results.

Example

- *Let $f : \mathbb{N} \longrightarrow \mathbb{N}$ be defined by $f(n) = n + n^2$ and $g : \mathbb{N} \longrightarrow \mathbb{N}$ be defined by $g(n) = n^2$. Now, for all $n \in \mathbb{N}$, $f(n) \leq 2n^2$. Therefore f is $O(g)$.*

The Landau Symbol O

We will start using the defining formula to abbrieviate functions $f : \mathbb{N} \longrightarrow \mathbb{R}$ (and $f : \mathbb{R} \longrightarrow \mathbb{R}$). For example, we will use n^n to abbrieviate the function $f : \mathbb{N} \longrightarrow \mathbb{N}$ defined by $f(n) = n^n$.

Example

$n!$ is $O(n^n)$ because for all $n \geq 1$,

$$n! = 1 \cdot 2 \cdots n \leq n \cdot n \cdots n = n^n$$

Taking the natural log of both sides of the above calculation and noting that $\ln(n^n) = n \ln(n)$ shows that $\ln(n!)$ is $O(n \ln(n))$.

Theorem

Let $f : \mathbb{N} \longrightarrow \mathbb{R}$ and $g : \mathbb{N} \longrightarrow \mathbb{R}$. If there exists $C \in \mathbb{R}$ with $C \geq 0$ such that

$$\lim_{n \to \infty} \frac{|f(n)|}{|g(n)|} = C$$

then f is $O(g)$.

The Landau Symbol O

Proof.

Let $C \in \mathbb{R}$ with $C \geq 0$ be such that

$$\lim_{n \to \infty} \frac{|f(n)|}{|g(n)|} = C$$

Therefore, for all $\epsilon > 0$, there exists $N \in \mathbb{N}$ such that for all $n > N$,

$$\left| \frac{|f(n)|}{|g(n)|} - C \right| < \epsilon$$

Choosing $\epsilon = 1$ and multiplying through by $|g(n)|$ we get $N \in \mathbb{N}$ such that for all $n > N$, $||f(n)| - C|g(n)|| < 1$. So, for all $n > N$, $|f(n)| < (1 + C)|g(n)|$. Since this implies that for all $n > N$, $|f(n)| < \lceil (1 + C) \rceil |g(n)|$, it follows that f is $O(g)$. □

Example

$(n + 1) \ln(n^2 + 1) + 3n^2$ *is* $O(n^2)$

The Landau Symbol O

Proof.

$$\lim_{n\to\infty}\left(\frac{(n+1)\ln(n^2+1)+3n^2}{n^2}\right)$$

$$=\lim_{n\to\infty}\left(\frac{(n+1)\ln(n^2+1)}{n^2}\right)+\lim_{n\to\infty}\left(\frac{3n^2}{n^2}\right)=0+3$$

□

The assertion that a function f is $O(g)$ says that asymtotically f grows no faster than a constant multiplied by g. We will use this asymtotic analysis of the the growth of measures of time complexity to classify the time complexity of algorithms. Note that the coarseness of this asymptotic analysis allows us to ignore things like exactly how long it takes to perform a basic operation (as long as this time is constant), and obscures hardware variations (e.g. one computer running twice as fast as another computer).

Big-Omega and Big-Theta

Note that the Landau symbol big-oh allows us to express upper bounds on the asymptotic growth of functions. We also need a means of expressing lower bounds on a function's asymptotic behaviour.

Definition

Let A be $\mathbb{R}$ or $\mathbb{N}$. Let $f : A \longrightarrow \mathbb{R}$ and $g : A \longrightarrow \mathbb{R}$. We say f is $\Omega(g)$, pronouced "f is big-omega of g", if g is $O(f)$. We say that f is $\Theta(g)$, pronounced "f is big-theta g" or "f is **order** *g", if f is $O(g)$ and f is $\Omega(g)$.*

Example

Let $f : \mathbb{N} \longrightarrow \mathbb{N}$ be defined by

$$f(n) = \sum_{k=1}^{n} k$$

Then $f(n) = \frac{n(n+1)}{2}$, so for all $n \in \mathbb{N}$, $n^2 \leq 2f(n)$ and $f(n) \leq n^2$. So f is order n^2.

Big-Omega and Big-Theta

Theorem

$\ln(n!)$ *is order* $n\ln(n)$

Proof.

We have already seen that $\ln(n!)$ is $O(n\ln(n))$, so it remains to show that $n\ln(n)$ is $O(\ln(n!))$. Let $n \in \mathbb{N}$ with $n \geq 2$. For all $1 \leq i \leq n$, $n(i-1) \geq i(i-1)$. Therefore, for all $1 \leq i \leq n$, $ni - i^2 + i \geq n$. Or, for all $1 \leq i \leq n$, $(n-(i-1))i \geq n$. So,

$$(n!)^2 = (n \cdot 1) \cdot ((n-1) \cdot 2) \cdots ((n-(n-1)) \cdot n) \geq n^n$$

Taking the natural log of both sides, we get $\ln(n^n) \leq \ln((n!)^2)$. So,

$$n\ln(n) \leq 2\ln(n!)$$

which shows that $n\ln(n)$ is $O(\ln(n!))$. □

The order of growth

Theorem

Let $n \in \mathbb{N}\backslash\{0\}$. If $f : \mathbb{R} \longrightarrow \mathbb{R}$ is a polynomial of degree n, then f is order x^n.

Proof.

Let $a_0, \ldots, a_n \in \mathbb{R}$ with $a_n \neq 0$ and suppose $f : \mathbb{R} \longrightarrow \mathbb{R}$ is defined by:

$$f(x) = \sum_{k=0}^{n} a_k x^k$$

For all $x \in \mathbb{R}$, we can use the triangle inequality to obtain

$$|f(x)| = \left| \sum_{k=0}^{n} a_k x^k \right| \leq \sum_{k=0}^{n} |a_k||x|^k$$

If $x > 1$, then

$$|f(x)| \leq x^n \sum_{k=0}^{n} \left(\frac{|a_k|}{x^{n-k}} \right) \leq x^n \left(\sum_{k=0}^{n} |a_k| \right)$$

The order of growth

Proof.

(Continued.) By choosing $C \in \mathbb{N}$ with $C \geq \sum_{k=0}^{n} |a_k|$, we can see that f is $O(x^n)$.

Conversely, it is clear that for all $\epsilon > 0$, there exists $D \in \mathbb{R}$ with $D > 0$ such that for all $x > D$,

$$\left| \sum_{k=0}^{n} a_k x^{k-n} \right| \geq |a_n| - \epsilon$$

So, choosing $\epsilon = \frac{|a_n|}{2}$, we get $D \in \mathbb{R}$ with $D > 0$ such that for all $x > D$,

$$\left| \sum_{k=0}^{n} a_k x^{k-n} \right| \geq \frac{|a_n|}{2} \text{ and } |f(x)| = x^n \left| \sum_{k=0}^{n} a_k x^{k-n} \right| \geq \frac{|a_n|}{2} x^n$$

So, if $C \in \mathbb{N}$ is such that $C \geq \frac{2}{|a_n|}$, then $|x^n| \leq C|f(x)|$, which shows that x^n is $O(f)$.

The order of growth

However, we do get some distinction:

Theorem

Let $p, q \in \mathbb{R}$ with $0 < p < q$. Then n^q is not $O(n^p)$.

Proof.

Suppose, for a contradiction, that there exists $M, C \in \mathbb{N}$ such that for all $n > M$, $n^q \leq Cn^p$. Let $D \in \mathbb{N}$ be such that $D > M$ and $D^{q-p} > C$. So $D^q > CD^p$, which is our desired contradiction. □

Theorem

n is not $O(\ln(n))$

Proof.

Suppose, for a contradiction, that there exists $M, C \in \mathbb{N}$ such that for all $n > M$, $n \leq C \ln(n)$. But now for all $n > \sqrt{M}$,
$n^2 \leq C \ln(n^2) = 2C \ln(n) \leq 2Cn$, which would show that n^2 is $O(n)$. □

Time Complexity of Bubble Sort

Example

Consider the complexity of the Bubble Sort Algorithm. We define a function $f : \mathbb{N} \longrightarrow \mathbb{N}$ *that on input* n *counts the worst-case number of comparisons needed sort a list of length* n*. During the* i^{th} *pass,* $n - i$ *comparisons are made, and the whole algorithm runs through* $n - 1$ *passes. This yields a total of*

$$\sum_{i=1}^{n-1}(n-i) = n(n-1) - \sum_{i=1}^{n-1} i = \frac{n(n-1)}{2}$$

comparisons. We also need to count the number of comparisons that are built into the two ***for loops*** *is our algorithm. At the start of each new loop a comparison needs to be made to see if the loop should be terminated or not. This adds*

$$\sum_{i=1}^{n-1}(n-i) = \frac{n(n-1)}{2}$$

comparisons. Therefore $f(n) = n(n-1)$*, and so* f *is order* n^2*.*

Time Complexity of Bubble Sort

- Note that this analysis has ignored the "swaps" that the Bubble Sort Algorithm makes. We are making the assumption that the time cost of making the swaps is small compared to the time required to make the comparisons. Note that with this assumption our worst-case time complexity analysis becomes the same as the equivalent average-case case time complexity analysis because the Bubble Sort Algorithm uses the same number of comparisons for any two inputs of the same length.
- In the preceding analysis we accounted for the comparisons that are incorporated into the **for loops**. In almost all practical applications, the number of operations needed to detect if a loop has ended is not larger than the number of operations used within the loop. Therefore, iin most cases, it is fine to neglect these operations when determining the complexity of an algorithm.

Time Complexity of Bubble Sort

- An unfortunate feature of our time complexity analysis of Bubble Sort is that our choice of basic operation (order comparison of two integers) does not necessarily run in an amount of time that is independant of the input. That is, comparing two large numbers may be slower than comparing two small numbers. In order for our analysis to be sensible it needs to be the case that this difference does not significantly change the asymtotic growth of the function that measures the complexity of the algorithm we are considering. Nevertheless, it is always important to state clearly what the basic operation is in any time complexity analysis so that one can take this into account when interpreting the analysis.

Complexity Categories

In general, we distinguish between the following complexities for algorithms (in increasing order of complexity):

Complexity	Terminology
$\Theta(1)$	constant complexity
$\Theta(\log_2(n))$	logarithmic complexity
$\Theta(n)$	linear complexity
$\Theta(n \log_2(n))$	$n \log(n)$ complexity
$\Theta(n^b)$, with $b > 0$	polynomial complexity
$\Theta(b^{n^\epsilon})$, with $b > 1$ and $0 < \epsilon < 1$	sub-exponential complexity
$\Theta(b^n)$, with $b > 1$	exponential complexity
$\Theta(n!)$	factorial complexity

Since we are working with functions $f : \mathbb{N} \longrightarrow \mathbb{N}$ it is often more convienient to use $\log_2$ or $\log_{10}$ instead of ln.

Complexity Categories

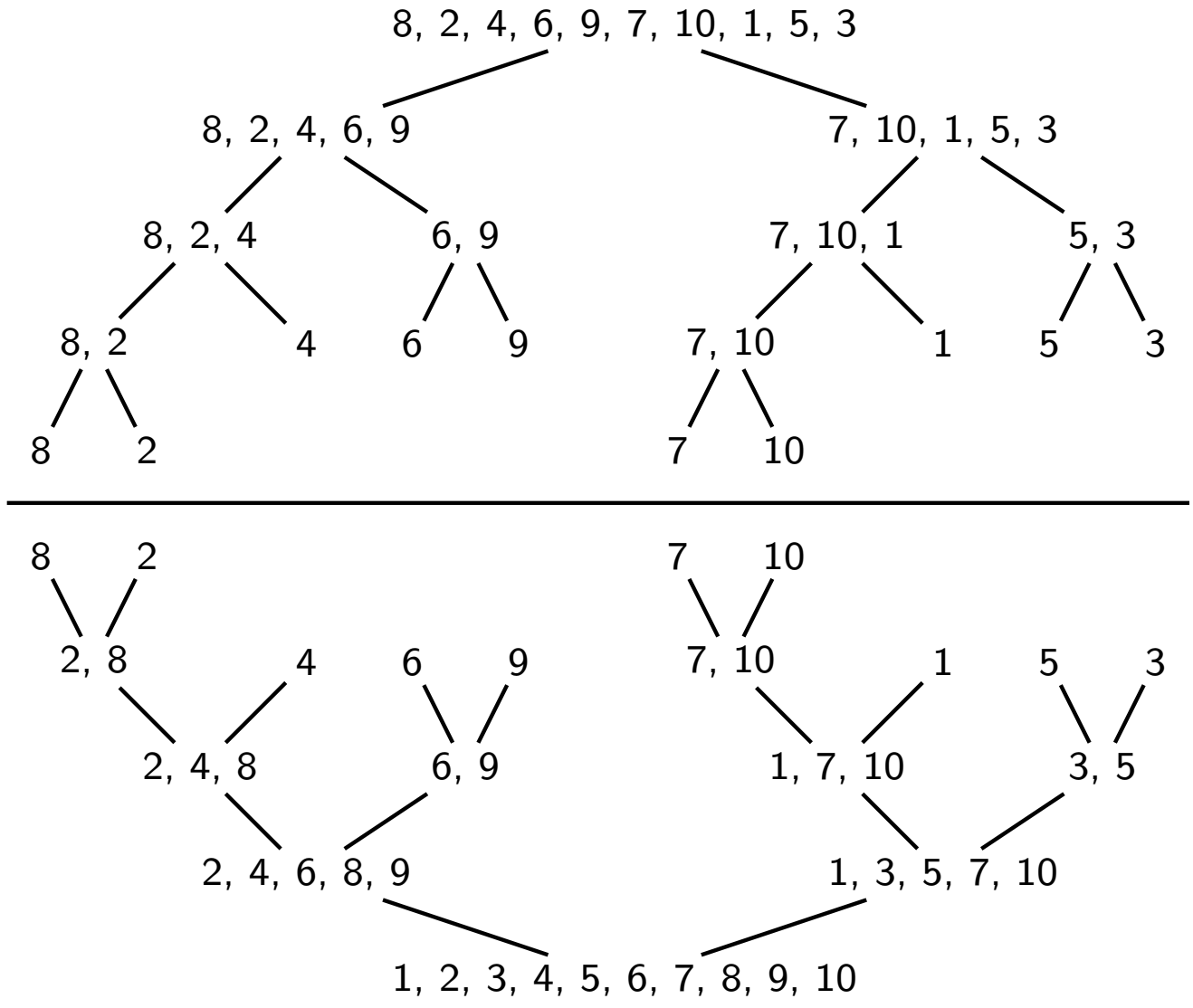
By classifing complexity in terms of the asymtotic growth of the functions that measure the required operations as a function of the length of the input, we create categories that are robust in the sense that they ignore variations in computer specifications and even change in specifications as technology improves. However, inevitably, this categorisation my not adequately take into account the behaviour of an algorithm in reality. For example:

- By only taking into account asymptotic behaviour we ignore the fact that an algorithm might dominate a very fast growing function on all inputs that are practical, and only grow more slowly asymtotically.
- If the time complexity measure f of an algorithm is $O(n^k)$ for some $k > 1$ (polynomial time or less), then the problem that the algorithm solves is thought to be **tractable** or "solvable by a computer". However, if f is order $n^{10^{82}}$, then this is clearly not the case.

Merge Sort

An efficient sorting algorithm that can be formulated recursively is the *merge sort*. It works by successively splitting a list in half until only lists of length 1 remain. These are then successively joined to ordered lists.

We illustrate by showing how merge sort orders the list 8, 2, 4, 6, 9, 7, 10, 1, 5, 3 at right.



Merge Sort

The recursive pseudocode for merge sort is quite simple; it depends on having an algorithm for merging two ordered lists into a single ordered list:

Algorithm

(Merge Sort)

Input: $L = a_1, \ldots, a_n$
Output: L, *sorted into elements in nondecreasing oder*

```
1 Function MergeSort(L):
2     if n > 1 then
3         m ← ⌊n/2⌋;
4         L1 ← a1, ..., am;
5         L2 ← am+1, ..., an;
6         L ← Merge(MergeSort(L1), MergeSort(L2))
7     end if
8     return L
9 end
```

Merge Sort

Algorithm

(Merge)

Input: L_1, L_2, *two sorted lists*
Output: L *a merged list of* L_1 *and* L_2, *with elements in increasing order*

```
1  Function Merge(L1, L2):
2      L ← empty list;
3      while L1 and L2 are both nonempty do
4          remove smaller of first element of L1 and L2 from the list it
             is in and put it at the right end of L;
5          if removal of this element makes one list empty then
6              remove all elements from the other list and append to L
7          end if
8      end while
9      return L
10 end
```

Complexity of Merge Sort

We begin by focussing our attention on the complexity of the algorithm used to merge to sorted lists.

Lemma

Two sorted lists with m and n elements, respectively, can be merged into a sorted list using the Merge algorithm using no more than $n + m - 1$ comparisons.

Proof.

A worst-case analysis of the Merge algorithm reveals that the least efficient way for it to run is for neither list to become empty until there is only one element in the other list. Each at each stage when both the lists are nonempty one comparison is made. Therefore, in the worst-case scenario, $n + m - 1$ comparisons are made. □

Complexity of Merge Sort

Theorem

The number of comparisons needed to Merge Sort a list with n elements is $O(n \log_2(n))$.

Proof.

To simplify things we will assume that $n = 2^m$. If this is not the case, then the following argument can be modified to yield the same result.
At the first stage of the splitting procedure, the list is split into two sublists with 2^{m-1} elements each. This procedure is repeated, and after k splittings, there are 2^k sublists with 2^{m-k} elements each. After m steps, there are 2^m lists of length 1.
At the beginning of the merging process, 2^m lists are merged into 2^{m-1} lists of length 2, requiring 2^{m-1} comparisons. At the k^{th} stage of the merging process $2^{m-(k-1)}$ lists each with 2^{k-1} elements are merged into 2^{m-k} lists. This is 2^{m-k} merges, and each of these merges requires at most $2^{k-1} + 2^{k-1} - 1 = 2^k - 1$ comparisons.

Complexity of Merge Sort

Proof.

(Continued.) So, the total number of comparisons needed is:

$$\sum_{k=1}^{m} 2^{m-k}(2^k - 1) = \sum_{k=1}^{m} 2^m + \sum_{k=1}^{m} 2^{m-k} = m2^m + \sum_{k=1}^{m} 2^{k-1}$$

Recalling the sum of a geometric series, we get a worst-case count of the total number of comparisons needed to be
$m2^m - (2^m - 1) = n\log_2(n) - n + 1$.
It follows that a function $f : \mathbb{N} \longrightarrow \mathbb{N}$ that counts the worst-case number of comparisons needed by Merge Sort as a function of the length of the input list is defined by $n\log_2(n) - n + 1$. This function is clearly $O(n\log_2(n))$. □

Recall that the time complexity of Bubble Sort was order n^2. This result indicates that Merge Sort is more efficient than Bubble Sort. It turns out that Merge sort is optimal!

Recurrence Relations

In order to understand the complexity of recursively defined algorithms we need gain a better understanding of recursively defined functions from $\mathbb{N}$ to $\mathbb{N}$ (or $\mathbb{N}$ to $\mathbb{R}$). We will now turn our attention to examining these functions, which, in this context, we call **recurrence relations**.

Definition

Let $f : \mathbb{N} \times \mathbb{R}^k \longrightarrow \mathbb{R}$ *and let* $a_0, \ldots, a_{k-1} \in \mathbb{R}$. *A function* $g : \mathbb{N} \longrightarrow \mathbb{R}$ *that satisfies:*

$$\begin{aligned} g(n) &= a_n && \text{for all } 0 \leq n < k \\ g(n) &= f(n, a_{n-k}, \ldots, a_{n-1}) && \text{for all } n \geq k \end{aligned}$$

is said to satisfy the **recurrence relation** *defined by* f *with* **initial conditions** $a_0, \ldots, a_{k-1}$.

It should be clear that a function satisfying a recurrence relation is just a function that recursively defined by the rule f. Most of the examples we will consider in this course will be functions $g : \mathbb{N} \longrightarrow \mathbb{N}$ that satisfy a recurrence relation.

Recurrence Relations

It is convenient to think of and write functions $g : \mathbb{N} \longrightarrow \mathbb{R}$ that satisfy a recurrence relation as sequences (g_n) (after all, that is what a function with domain $\mathbb{N}$ is).

Example

- *The Fibonacci sequence (f_n) satisfies the recurrence relation*

$$f_n = f_{n-1} + f_{n-2}$$

with initial conditions $f_0 = 1$ and $f_1 = 1$.

- *Let (g_n) be the sequence where g_n is the number of way to bracket the product $x_0 \cdot x_1 \cdots x_n$ to specify the order of multiplication. Then (g_n) satisfies the recurrence relation*

$$C_n = \sum_{k=0}^{n-1} C_k C_{n-k-1}$$

with initial conditions $C_0 = 1$ and $C_1 = 1$.

Recurrence Relations

Theorem

Let $f : \mathbb{N} \times \mathbb{R}^k \longrightarrow \mathbb{R}$ and let $a_0, \ldots, a_{k-1} \in \mathbb{R}$. Then there exists a unique $g : \mathbb{N} \longrightarrow \mathbb{R}$ that satisfies the recurrence relation define by f with initial conditions $a_0, \ldots, a_{k-1}$.

Proof.

We prove by induction on $m \geq k$ that there exists a unique $g_m : [m] \longrightarrow \mathbb{R}$ that satisfies

$$\begin{aligned} g(n) &= a_n && \text{for all } 0 \leq n < k \\ g(n) &= f(n, a_{n-k}, \ldots, a_{n-1}) && \text{for all } n \geq k < m \end{aligned} \tag{6}$$

It is clear that (**??**) uniquely specifies $g_k : [k] \longrightarrow \mathbb{R}$. Asumme that there exists a unique $g_m : [m] \longrightarrow \mathbb{R}$ satisfying (**??**). Therefore, since f is a function, $g_{m+1} : [m+1] \longrightarrow \mathbb{R}$ defined by

$$g_{m+1} = g_m \cup \{(m, f(m, a_{m-k}, \ldots, a_{m-1}))\}$$

is the unique function with domain $[m+1]$ that satisfies (**??**).

Recurrence Relations

Proof.

It now follows that the functions $g_m : [m] \longrightarrow \mathbb{R}$ satisfying (**??**) are unique and by letting

$$g = \bigcup_{m \geq k} g_m$$

we obtain the unique function satisfying the recurrence relation defined by f with initial conditions $a_0, \ldots, a_{k-1}$. □

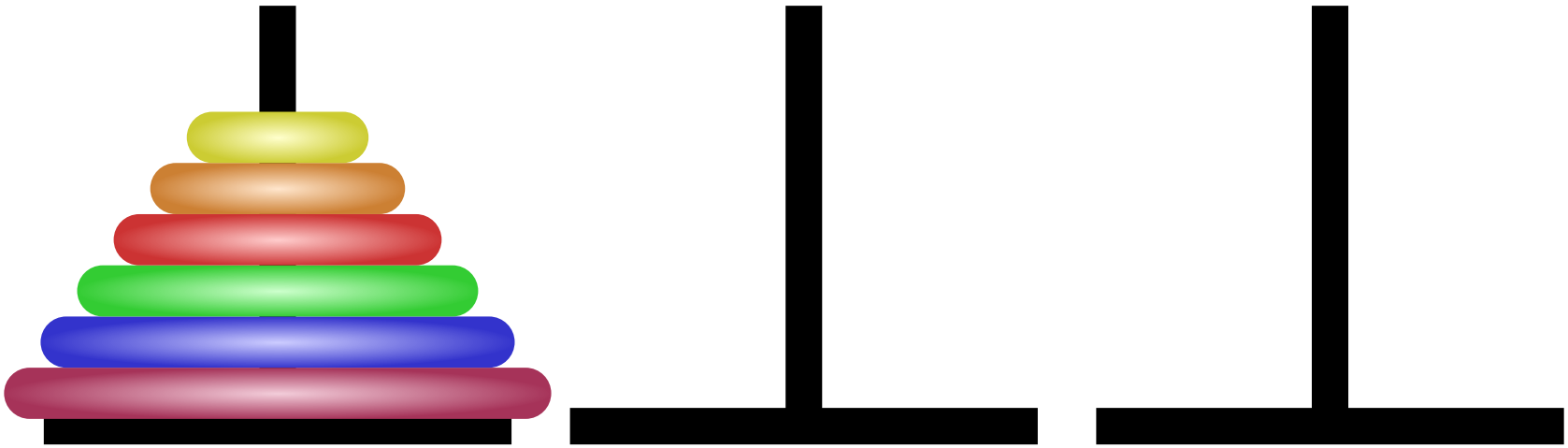
We now turn finding explicit descriptions of functions (sequences) that satisfy recurrence relations.

Tower of Hanoi

Example

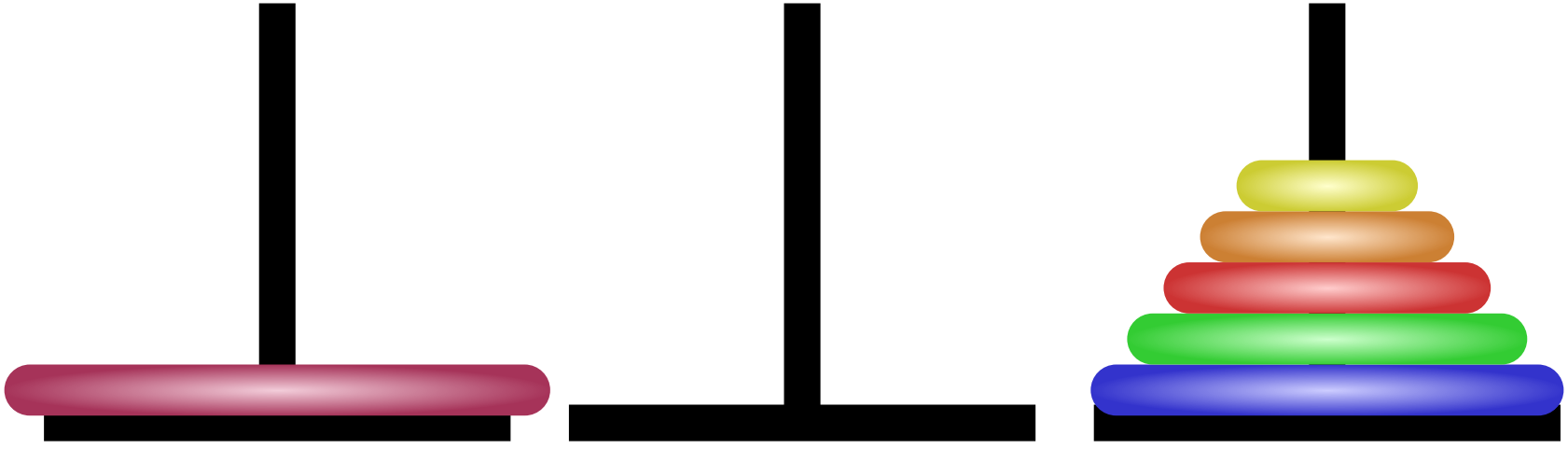
The tower of Hanoi is a famous problem: n concentric disks of increasing radius are placed upon one of three pegs. The goal is to move the stack of disks from Peg 1 to Peg 2, but in such a way that

- *only one disk is moved at a time and*
- *a larger disk is never placed on top of a smaller disk.*



Tower of Hanoi

Our goal is to find a recurrence relation for the number H_n of moves required to move a stack of n disks from Peg 1 to Peg 2. Suppose we need H_{n-1} moves to transfer the stack of $n-1$ disks to Peg 3:



We can then transfer the largest disk to Peg 2, and then again use H_{n-1} moves to transfer the other disks to Peg 2 on top of the largest disk. Thus,

$$H_n = 2H_{n-1} + 1, \qquad n \geq 2, \qquad H_1 = 1.$$

Tower of Hanoi

Simple calculations yield:

$$H_2 = 2 + 1 = 2^2 - 1$$
$$H_3 = 2(2 + 1) + 1 = 2^2 + 2 + 1 = 2^3 - 1$$
$$H_4 = 2(2^2 + 2 + 1) + 1 = 2^3 + 2^2 + 2 + 1 = 2^4 - 1$$

Theorem

$H_n = 2^n - 1$

Proof.

We prove this by induction. Note that $H_1 = 2^1 - 1$. Assume that $H_n = 2^n - 1$. Therefore

$$H_{n+1} = 2H_n + 1 = 2(2^n - 1) + 1 = 2^{n+1} - 1$$

and the result follows. □

Fibonacci Sequence

The Fibonacci Sequence (f_n) can be represented as a system of linear equations.

$$f_1 = f_1$$
$$f_2 = f_0 + f_1$$

And, in general:

$$f_{n+1} = f_{n+1}$$
$$f_{n+2} = f_n + f_{n+1}$$

So, letting

$$A = \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix} \text{ and } \vec{f}_n = \begin{pmatrix} f_{n+1} \\ f_n \end{pmatrix}$$

we have for all $n \in \mathbb{N}$, $\vec{f}_{n+1} = A\vec{f}_n$. Therefore $\vec{f}_n = A^n\vec{f}_0$.

Fibonacci Sequence

Now, $\mathbf{det}(A - \alpha I) = -\alpha(1-\alpha) - 1 = \alpha^2 - \alpha - 1$, which has roots:

$$\alpha_1 = \frac{1+\sqrt{5}}{2} \text{ and } \alpha_2 = \frac{1-\sqrt{5}}{2}$$

These are the **eigenvalues** of A. The corresponding **eigenvectors** are

$$\vec{v}_1 = \begin{pmatrix} \alpha_1 \\ 1 \end{pmatrix} \text{ and } \vec{v}_2 = \begin{pmatrix} \alpha_2 \\ 1 \end{pmatrix}$$

We all know that $\vec{v}_1$ and $\vec{v}_2$ are linearly independent and so

$$P = (\vec{v}_1 \ \vec{v}_2) = \begin{pmatrix} \alpha_1 & \alpha_2 \\ 1 & 1 \end{pmatrix} \text{ is invertible with}$$

$$P^{-1} = \frac{1}{\alpha_1 - \alpha_2} \begin{pmatrix} 1 & -\alpha_2 \\ -1 & \alpha_1 \end{pmatrix}$$

Fibonacci Sequence

Moreover, since $\vec{v}_1$ and $\vec{v}_2$ are eigenvectors, it follows that $AP = PD$ where

$$D = \begin{pmatrix} \alpha_1 & 0 \\ 0 & \alpha_2 \end{pmatrix}$$

So, $A = PDP^{-1}$ and $A^n = PD^nP^{-1}$. Now,

$$D^n = \begin{pmatrix} \alpha_1^n & 0 \\ 0 & \alpha_2^n \end{pmatrix}$$

So, $\vec{f}_n = PD^nP^{-1}\vec{f}_0$, and $\vec{f}_0 = (1,1)^T$, so

$$\begin{pmatrix} f_{n+1} \\ f_n \end{pmatrix} = \frac{1}{\alpha_1 - \alpha_2} \begin{pmatrix} \alpha_1^{n+1}(1-\alpha_2) + \alpha_2^{n+1}(\alpha_1 - 1) \\ \alpha_1^n(1-\alpha_2) + \alpha_2^n(\alpha_1 - 1) \end{pmatrix}$$

In other words for all $n \in \mathbb{N}$,

$$f_n = \frac{1-\alpha_2}{\alpha_1 - \alpha_2}\alpha_1^n + \frac{\alpha_1 - 1}{\alpha_1 - \alpha_2}\alpha_2^n$$

Recurrence Relations

Definition

Let $f : \mathbb{N} \times \mathbb{R}^k \longrightarrow \mathbb{R}$ and let $a_0, \ldots, a_{k-1} \in \mathbb{R}$. Consider the recurrence relation defined by f with initial conditions $a_0, \ldots, a_{k-1}$. If f is in the form

$$f(n) = c_1 a_{n-1} + \cdots c_k a_{n-k} + f'(n)$$

then we say that the recurrence relation defined by f is a **linear recurrence relation** *with* **degree** *k. If f' is the zero sequence, then we say that the recurrence relation defined by f is* **homogeneous**, *otherwise we say that it is* **inhomogeneous**.

Example

- *The Fibonacci Sequence is a homogeneous linear recurrence relation.*
- *The Tower of Hanoi Puzzle is solved by an inhomogeneous linear recurrence relation*
- *The recurrence relation that counts the number of way to bracket the multiplication of $n + 1$ distinct numbers is not linear*

Linear Homogeneous Recurrence Relations

Suppose that (a_n) satisfies the homogeneous linear recurrence relation

$$a_n = c_1 a_{n-1} + \cdots + c_k a_{n-k}$$

We can write this recurrence relation as a matrix A. What does A look like? The sequence (a_n) can then be written as k-dimensional vectors

$$\vec{a}_{n-k} = (a_{n-1}, \ldots, a_{n-k})^T$$

We now have that $\vec{a}_m = A^m \vec{a}_0$, and so a solution to the recursion relation can be obtained by finding explicit form for the exponents of the matrix A. If A is diagonalisable, i.e. if there exists an invertible matrix P and a diagonal matrix D with diagonal entries $\alpha_1, \ldots, \alpha_k$ such that

$$PDP^{-1} = A$$

then $A^m = PD^m P^{-1}$. In this case, each element of the sequence (a_n) will be in the form

$$a_n = q_1 \alpha_1^n + \cdots + q_k \alpha_k^n$$

Linear Homogeneous Recurrence Relations

Theorem

Let (a_n) and (b_n) satisfy the homogeneous linear recurrence relation

$$x_n = c_1 x_{n-1} + \cdots + c_k x_{n-k} \tag{7}$$

*Then for all $A, B \in \mathbb{C}$, the sequence $(Aa_n + Bb_n)$ also satisfies (**??**).*

Proof.

$$\begin{aligned} Aa_n + Bb_n &= A(c_1 a_{n-1} + \cdots + c_k a_{n-k}) + B(c_1 b_{n-1} + \cdots + c_k b_{n-k}) \\ &= c_1(Aa_{n-1} + Ba_{n-1}) + \cdots + c_k(Aa_{n-k} + Bb_{n-k}) \end{aligned}$$

□

These considerations motivate us to seek solutions in the form (a_n) with $a_n = \alpha^n$ to the homogeneous linear recurrence relation

$$x_n = c_1 x_{n-1} + \cdots + c_k x_{n-k}$$

Linear Homogeneous Recurrence Relations

If $\alpha \in \mathbb{C}$ and the sequence (a_n) defined by $a_n = \alpha^n$ satisfies the homogeneous linear recurrence relation

$$x_n = c_1 x_{n-1} + \cdots + c_k x_{n-k} \tag{8}$$

Then $\alpha^n = c_1\alpha^{n-1} + \cdots + c_k\alpha^{n-k}$. So, if $\alpha \neq 0$, then α is a root of the polynomial

$$\lambda^k - c_1\lambda^{k-1} - \cdots - c_k \tag{9}$$

We call (**??**) the **characteristic polynomial** of the recurrence relation (**??**) and we have:

Theorem

*If $\alpha_1, \ldots, \alpha_k$ are roots of the characteristic polynomial of the linear recurrence relation (**??**) then for all $A_1, \ldots, A_k \in \mathbb{C}$, the sequence (a_n) defined by*

$$a_n = A_1\alpha_1^n + \cdots + A_k\alpha_k^n$$

*satisfies (**??**).*

Linear Homogeneous Recurrence Relations

Observe, that the theorem on the previous slide makes no mention of initial conditions. This raises the question: Given a list of initial conditions and a linear homogeneous recurrence relation can we always find a sequence that satisfies the recurrence relation with the prescribed initial conditions?

Lemma

Let $\alpha_1, \ldots, \alpha_k$ be distinct roots of the polynomial

$$\lambda^k - c_1\lambda^{k-1} - \cdots - c_k$$

Then the $k \times k$ matrix

$$M = \begin{pmatrix} 1 & 1 & \cdots & 1 \\ \alpha_1 & \alpha_2 & \cdots & \alpha_k \\ \alpha_1^2 & \alpha_2^2 & \cdots & \alpha_k^2 \\ \vdots & & & \vdots \\ \alpha_1^{k-1} & \alpha_2^{k-1} & \cdots & \alpha_k^{k-1} \end{pmatrix}$$

is invertible.

Linear Homogeneous Recurrence Relations

Proof.

It follows from elementary linear algebra that it is sufficient to show that M^T is invertible. We will do this by showing that the system $M^T\vec{v} = \vec{0}$ has only the trivial solution. Suppose that $\vec{v} = (d_1, \ldots, d_k)^T$ is a solution to $M^T\vec{v} = \vec{0}$. It follows that for all $1 \leq i \leq k$,

$$d_1 + d_2\alpha_i + \cdots + d_k\alpha_i^{k-1} = 0$$

Therefore, for all $1 \leq i \leq k$, α_i is a root of the polynomial

$$d_1 + d_2x + \cdots + d_kx^{k-1}$$

But, a nontrivial polynomial of degree $\leq k-1$ can have no more than $k-1$ distinct roots, so $d_1 = \cdots = d_k = 0$. □

The matrix M that we just showed is invertible is known as the **Vandermonde matrix**.

Linear Homogeneous Recurrence Relations

Theorem

Let $a_0, \ldots, a_{k-1} \in \mathbb{C}$. Let $\alpha_1, \ldots, \alpha_k$ be k distinct roots of the characteristic polymial of the recurrence relation

$$x_n = c_1 x_{n-1} + \cdots + c_k x_{n-k} \tag{10}$$

Then there exists a sequence (a_n) in the form

$$a_n = q_1 \alpha_1^n + \cdots + q_k \alpha_k^n$$

*that satisfies (**??**) with initial conditions $a_0, \ldots, a_{k-1}$*

Proof.

We have already seen that a sequence (a_n) in the form

$$a_n = q_1 \alpha_1^n + \cdots + q_k \alpha_k^n$$

satisfies (**??**). Therefore we need to show that such a sequence can satisfy the prescribed initial conditions.

Linear Homogeneous Recurrence Relations

Proof.

(Continued.) The initial conditions would be satisfied if we could choose $q_1, \ldots, q_k$ such that for all $0 \leq n < k$,

$$a_n = q_1\alpha_1^n + \cdots + q_k\alpha_k^n$$

Letting $A = (a_0, \ldots, a_{k-1})^T$ and $Q = (q_1, \ldots, q_k)^T$, this system of equations can be written as $A = MQ$ where

$$M = \begin{pmatrix} 1 & 1 & \cdots & 1 \\ \alpha_1 & \alpha_2 & \cdots & \alpha_k \\ \alpha_1^2 & \alpha_2^2 & \cdots & \alpha_k^2 \\ \vdots & & & \vdots \\ \alpha_1^{k-1} & \alpha_2^{k-1} & \cdots & \alpha_k^{k-1} \end{pmatrix}$$

Since M is invertible, $Q = M^{-1}A$ yields values for $q_1, \ldots, q_k$ that ensures that the sequence (a_n) satisfies the prescribed initial conditions. □

Linear Homogeneous Recurrence Relations

This gives us the tools to solve homogeneous linear recurrence relations when the roots of the characteristic polynomial are all distinct.

Example

Consider a sequence (a_n) *such that*

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

with initial conditions $a_0 = 2$, $a_1 = 5$ *and* $a_2 = 15$. *This recurrence relation has characteristic polynomial*

$$\lambda^3 - 6\lambda^2 + 11\lambda - 6$$

Observing that $1^3 - 6 \cdot 1^2 + 11 \cdot 1 - 6 = 0$, *and doing some long division reveals that*

$$\lambda^3 - 6\lambda^2 + 11\lambda - 6 = (\lambda - 1)(\lambda - 2)(\lambda - 3)$$

Linear Homogeneous Recurrence Relations

Example

And so, the characteristic equation has three distinct roots: $\alpha_1 = 1$, $\alpha_2 = 2$ *and* $\alpha_3 = 3$. *Therefore* (a_n) *has the form*

$$a_n = q_1 + q_2 2^n + q_3 3^n$$

Letting $A = (2, 5, 15)^T$ *and* $Q = (q_1, q_2, q_3)^T$, *we need to solve the system* $MQ = A$ *where*

$$M = \begin{pmatrix} 1 & 1 & 1 \\ 1 & 2 & 3 \\ 1 & 4 & 9 \end{pmatrix}$$

A few row reductions yields

$$\left(\begin{array}{ccc|c} 1 & 1 & 1 & 2 \\ 0 & 1 & 0 & -1 \\ 0 & 0 & 2 & 4 \end{array}\right)$$

And so $a_n = 1 - 2^n + 2 \cdot 3^n$.

Linear Homogeneous Recurrence Relations

When the characteristic polynomial of the linear recurrence relation being considerred has repeated roots then the situation becomes rather more complicated. However, the following general result holds:

Theorem

Let $a_0, \ldots, a_{k-1} \in \mathbb{C}$. Let $\alpha_1, \ldots, \alpha_t$ be roots of the characteristic polymial of the recurrence relation

$$x_n = c_1 x_{n-1} + \cdots + c_k x_{n-k} \tag{11}$$

with multiplicities $m_1, \ldots, m_t$, respectively. Then there exists a sequence (a_n) in the form

$$a_n = Q_1 \alpha_1^n + \cdots + Q_t \alpha_t^n$$

$$with\ Q_i = \sum_{j=0}^{m_i - 1} q_{i,j} n^j\ \ for\ 1 \leq i \leq t$$

*that satisfies (**??**) with initial conditions $a_0, \ldots, a_{k-1}$*

Linear Homogeneous Recurrence Relations

Unfortunately we do not have time to prove this result.

Example

Consider a sequence (a_n) *such that*

$$a_n = -3a_{n-1} - 3a_{n-2} - a_{n-3}$$

with initial conditions $a_0 = 1$, $a_1 = -2$ *and* $a_2 = -1$. *The characteristic polynomial of this recurrence relation is*

$$\lambda^3 + 3\lambda^2 + 3\lambda + 1$$

Using the Binomial Theorem, we recognise this polynomial as $(\lambda + 1)^3$, *and so* -1 *is a repeated of this characteristic polynomial. It follows that* (a_n) *is of the form*

$$a_n = (q_1 + q_2 n + q_3 n^2) \cdot (-1)^n$$

The fact that $a_0 = 1$ *immediately shows that* $q_1 = 1$.

Linear Homogeneous Recurrence Relations

Example

We also get the equations:

$$-2 = -1 - q_2 - q_3 \ and \ -1 = 1 + 2q_2 + 4q_3$$

Adding twice the first equation to the second yields

$$-5 = -1 + 2q_3 \ or \ q_3 = -2$$

And so $q_2 = 3$. *Therefore* (a_n) *is the sequence*

$$a_n = (-1)^n(1 + 3n - 2n^2)$$

We next turn our attention to inhomogeneous linear recurrence relations. The situation with inhomogeneous recurrence relations is analogous to the situation that you have encounterred when dealing with nonhomogeneous ordinary differential equations.

Inhomogeneous Linear Recurrence Relations

Suppose that the sequences (a_n) and (b_n) both satisfy the recurrence relation

$$x_n = c_1 x_{n-1} + \cdots + c_k x_{n-k} + f'(n) \tag{12}$$

$$\text{So } a_n - b_n = c_1(a_{n-1} - b_{n-1}) + \cdots + c_k(a_{n-k} - b_{n-k})$$

And $(a_n - b_n)$ satisfies the recurrence relation

$$x_n = c_1 x_{n-1} + \cdots + c_k x_{n-k} \tag{13}$$

Theorem

*Let (a_n) satisfy the recurrence relation (**??**). If (b_n) satisfies the recurrence relation (**??**) then (b_n) is of the form*

$$b_n = c_n + a_n$$

*where (c_n) satisfies the recurrence relation (**??**).*

Inhomogeneous Linear Recurrence Relations

This means that by finding a single sequence (a_n) satisfying

$$x_n = c_1 x_{n-1} + \cdots + c_k x_{n-k} + f'(n) \tag{14}$$

we can determine a sequence (b_n) satisfying (**??**) with any prescribed initial conditions.

Example

Consider a sequence (a_n) *such that*

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

with $a_0 = 3$. *Solutions to the associated homogeneous recurrence relation are sequences* (b_n) *with* $b_n = q_1 3^n$. *Therefore we need to seek a particular solution to the inhomogeneous recurrence relation. Since* $f'(n) = 2n$ *is a polynomial of degree* 1 *we guess that a particular solution* (p_n) *can be found with*

$$p_n = cn + d$$

Inhomogeneous Linear Recurrence Relations

Example

This would require that

$$cn + d = 3(c(n-1) + d) + 2n$$

So

$$(2c + 2)n + 2d - 3c = 0$$

This means that $c = -1$ *and* $d = -\frac{3}{2}$. *So* (a_n) *is of the form*

$$a_n = q_1 3^n - n - \frac{3}{2}$$

Since $a_0 = 3$, $q_1 = \frac{9}{2}$, *and so*

$$a_n = \frac{9}{2} \cdot 3^n - n - \frac{3}{2}$$

Inhomogeneous Linear Recurrence Relations

Example

Consider a sequence (a_n) *such that*

$$a_n = 5a_{n-1} - 6a_{n-2} + 7^n$$

with $a_0 = \frac{9}{20}$ *and* $a_1 = \frac{43}{20}$. *The characteristic polynomial of the associated homogeneous recurrence relation is*

$$\lambda^2 - 5\lambda + 6 = (\lambda - 2)(\lambda - 3)$$

with roots $\alpha_1 = 2$ *and* $\alpha_2 = 3$. *So, solutions to the homogeneous recurrence relation are sequences* (b_n) *with* $b_n = q_1 2^n + q_2 3^n$. *Since* $f'(n) = 7^n$ *we guess that a particular solution* (p_n) *can be found with* $p_n = c7^n$. *This requires that*

$$c7^n = 5c7^{n-1} - 6c7^{n-2} + 7^n$$

Or, $20c = 49$. *Which yields* $c = \frac{49}{20}$.

Inhomogeneous Linear Recurrence Relations

Example

This means that (a_n) *is of the form*

$$a_n = q_1 2^n + q_2 3^n + \frac{49}{20} \cdot 7^n$$

The initial conditions yield equations

$$\frac{9}{20} = q_1 + q_2 + \frac{49}{20} \; and \; \frac{43}{20} = 2q_1 + 3q_2 + \frac{343}{20}$$

So, $q_1 = -(q_2 + 2)$ *and* $-15 = -2(q_2 + 2) + 3q_2$*. Which yields* $q_2 = -11$ *and* $q_1 = 9$*. So* (a_n) *is of the form*

$$a_n = 9 \cdot 2^n - 11 \cdot 3^n + \frac{49}{20}$$

In both of these examples we had to guess a form for the particular solution of the inhomogeneous recurrence relations. It turns out that these forms work in general.

Inhomogeneous Linear Recurrence Relations

Theorem

Let $c_1, \ldots, c_k \in \mathbb{R}$ and consider the inhomogenoeous recurrence relation

$$x_n = c_1 x_{n-1} + \cdots + c_k x_{n-k} + f'(n) \ \textit{with} \ f'(n) = \left(\sum_{i=0}^{t} b_i n^i \right) s^n \qquad (15)$$

*Then (**??**) has a particular solution in the form*

$$n^m \left(\sum_{i=0}^{t} q_i n^i \right) s^n$$

*where $m = 0$ if s is not a root of the characteristic polynomial of the homogeneous recurrence relation associated with (**??**), and if s is a root of the characteristic polynomial of the homogeneous recurrence relation associated with (**??**), then m is the multiplicity of that root.*

Inhomogeneous Linear Recurrence Relations

Example

Consider a sequence (a_n) *such that*

$$a_n = 6a_{n-1} - 9a_{n-2} + 3^n$$

with $a_0 = 0$ *and* $a_1 = 1$. *The characteristic polynomial of the associated homogeneous recurrence relation is*

$$\lambda^2 - 6\lambda + 9 = (\lambda - 3)^2$$

that clearly has root 3 *with multiplicity* 2. *Therefore if* (b_n) *satisfies that associated homogeneous recurrence relation then* $b_n = 3^n \cdot (q_1 + q_2 n)$. *We seek a particular sequence* (p_n) *satisfying the inhomogeneous recurrence relation in the form* $p_n = cn^2 3^n$. *This requires that*

$$cn^2 3^n = 6c(n-1)^2 3^{n-1} - 9c(n-2)^2 3^{n-2} + 3^n$$

Inhomogeneous Linear Recurrence Relations

Example

So $c = \frac{1}{2}$ *and* (a_n) *is in the form*

$$a_n = 3^n \cdot (q_1 + q_2 n) + \frac{1}{2} \cdot n^2 \cdot 3^n$$

The fact that $a_0 = 0$, *yields* $q_1 = 0$. *The fact that* $a_1 = 1$, *yields* $1 = 3q_2 - \frac{3}{2}$. *So* $q_2 = \frac{5}{6}$. *Therefore the sequence* (a_n) *satisfies*

$$a_n = \frac{5}{6} \cdot 3^n \cdot n - \frac{1}{2} \cdot n^2 \cdot 3^n$$

Divide-and-Conquer Algorithms

The principle of "divide and conquer", (originating from **divide et impera**, divide and rule, in Latin) goes back to the ancient Romans who used political, economic and military means to break up alliance of opponents or concentrations of power so that each part was less powerful than themselves.

In computer algorithms, *divide-and-conquer* algorithms are those that break up a problem into several smaller instances and tackle each instance separately. They are similar to recursive algorithms, but "multi-branched", i.e., referring to several instances of themselves, not just one, with smaller input.

We are interested in analysing the time complexity of divide-and-conquer algorithms. It turns out that recurrence relations are well-suited to describing the number of steps needed for such an algorithm to complete and that we can find quite general formulas for big-Oh estimates for many classes of algorithms.

Divide-and-Conquer Algorithms

Example

(i) *The Binary Search Algorithm reduces the problem of locating an element of a list to locating it in a list of half the length. Two comparisons are needed to do this, one to determine which half-list to search, the other to determine whether any terms remain in the list. Therefore, the number of comparisons needed to search a list of length n is given by*

$$f(n) = f(n/2) + 2.$$

Since only one list of length $n/2$ is searched, this is actually a recursive algorithm, a special case of a divide-and-conquer algorithm.

(ii) *A divide-and-conquer algorithm to find the minimum and maximum of a list of length $n = 2k$ might split the list into two equally sized sublists, find the minimum and maximum in each sublist and compare the values of the two sublists. The number of comparisons needed is then given by*

$$f(n) = 2f(n/2) + 2.$$

Divide-and-Conquer Algorithms

Example

(iii) *The Merge Sort Algorithm reduces the problem of sorting a list with n elements to that of sorting two lists with $n/2$ elements. It then uses fewer than n comparisons to merge the two sorted lists. Therefore, it uses fewer than $M(n)$ comparisons, where*

$$M(n) = 2M(n/2) + n.$$

Definition

Let $f : A \longrightarrow B$ be a function with $A, B \subseteq \mathbb{R}$. We say that f is **increasing** *if for all $x, y \in A$, if $x < y$, then $f(x) \leq f(y)$. We say that f is* **strictly increasing** *if for all $x, y \in A$, if $x < y$, then $f(x) < f(y)$. We say that f is* **decreasing** *if for all $x, y \in A$, if $x < y$, then $f(y) \leq f(x)$. We say that f is* **strictly decreasing** *if for all $x, y \in A$, if $x < y$, then $f(y) < f(x)$.*

Time Complexity of Divide-and-Conquer Algorithms

We first need an elementary Lemma.

Lemma

Let $a, b, c \in \mathbb{R}$ with $a, b, c > 0$.

$$a^{\log_b(c)} = c^{\log_b(a)}$$

Proof.

$\log_c(a^{\log_b(c)}) = \log_b(c) \cdot \log_c(a)$ and

$$\log_c(a) = \frac{\log_b(a)}{\log_b(c)}$$

So, $\log_c(a^{\log_b(c)}) = \log_b(a)$. Raising c to the power of both sides yields

$$a^{\log_b(c)} = c^{\log_b(a)}$$

Time Complexity of Divide-and-Conquer Algorithms

Theorem

Let $a, c \in \mathbb{R}$ *with* $a \geq 1$ *and let* $b \in \mathbb{N}$ *with* $b > 1$. *Let* $f : \mathbb{N} \longrightarrow \mathbb{R}$ *be an increasing function that satisfies the recurrence relation*

$$f(n) = a \cdot f\left(\frac{n}{b}\right) + c$$

whenever $b \mid n$. *Then*

$$f(n) \; is \begin{cases} O(n^{\log_b(a)}) & if \; a > 1 \\ O(\log_b(n)) & if \; a = 1 \end{cases}$$

Moreover, when $n = b^k$, *where* $k \in \mathbb{N}\backslash\{0\}$, *and* $a > 1$,

$$f(n) = c_1 n^{\log_b(a)} + c_2 \; where \; c_1 = f(1) + \frac{c}{a-1} \; and \; c_2 = -\frac{c}{a-1}$$

Time Complexity of Divide-and-Conquer Algorithms

Proof.

Let $k \in \mathbb{N}\backslash\{0\}$ and let $n = b^k$. Observe that

$$\begin{aligned} f(n) &= a \cdot f\left(\frac{n}{b}\right) + c \\ &= a^2 \cdot f\left(\frac{n}{b^2}\right) + ac + c \\ &\vdots \\ &= a^k \cdot f(1) + \sum_{i=0}^{k-1} a^i c = a^k \cdot f(1) + c\left(\sum_{i=0}^{k-1} a^i\right) \end{aligned}$$

So, if $a = 1$, then $f(n) = f(1) + ck$ or $f(n) = f(1) + c\log_b(n)$. Now, if n is not a power of b, then there exists $k \in \mathbb{N}$ such that $b^k < n < b^{k+1}$. And, since f is increasing,

$$f(n) \leq f(b^{k+1}) = f(1) + c(k+1) = (f(1) + c) + c\log_b(n)$$

Time Complexity of Divide-and-Conquer Algorithms

Proof.

(Continued.) In both cases (when n is a power of b, and when n is not) $f(n)$ is $O(\log_b(n))$. This completes the proof of the theorem when $a = 1$. We now deal with the case where $a > 1$. Assume, firstly, that $n = b^k$ where $k \in \mathbb{N}\backslash\{0\}$. Now, recall that

$$\sum_{i=0}^{k-1} a^i = \frac{(a^k - 1)}{(a - 1)}$$

So,

$$\begin{aligned} f(n) &= a^k f(1) + c\frac{(a^k - 1)}{(a - 1)} \\ &= a^k \left(f(1) + \frac{c}{(a - 1)} \right) - \frac{c}{(a - 1)} \end{aligned}$$

Time Complexity of Divide-and-Conquer Algorithms

Proof.

(Continued.) Now, $a^k = a^{\log_b(n)} = n^{\log_b(a)}$, so

$$f(n) = c_1 n^{\log_b(a)} + c_2 \text{ where } c_1 = f(1) + \frac{c}{a-1} \text{ and } c_2 = -\frac{c}{a-1}$$

So, this proves the second part of the theorem, and the first part when $n = b^k$. Finally, suppose that n is not a power of b. Therefore, there exists $k \in \mathbb{N}$ such that $b^k < n < b^{k+1}$. And, since f is increasing,

$$f(n) \leq f(b^{k+1}) = c_1 a^{k+1} + c_2 = c_1 a a^k + c_2$$

$$\text{So } f(n) \leq (c_1 a) a^{\log_b(n)} + c_2 = (c_1 a) n^{\log_b(a)} + c_2$$

which shows that f is $O(n^{\log_b(a)})$. This completes the proof. □

Time Complexity of Divide-and-Conquer Algorithms

Example

- *Suppose that* $f : \mathbb{N} \longrightarrow \mathbb{N}$ *satisfies* $f(n) = 5f\left(\frac{n}{2}\right) + 3$ *when* $2 \mid n$ *and* $f(1) = 7$. *The proof of the preceding theorem shows that when* $n = 2^k$,

$$f(n) = 5^k \left(7 + \frac{3}{4}\right) - \frac{3}{4}$$

  Moreover, if f *is increasing then* f *is* $O(n^{\log_2(5)})$

- *We have a mentioned already that a function* $f : \mathbb{N} \longrightarrow \mathbb{N}$ *that counts the worst-case number of comparisons used by the Binary Search Algorithm satisfies the recurrence relation*

$$f(n) = f\left(\frac{n}{2}\right) + 2$$

  Therefore the worst-case time complexity of Binary Search is $O(\log_2(n))$.

The Master Theorem

Theorem

(Master Theorem) Let $c, d \in \mathbb{R}$ *with* $c > 0$ *and* $d \geq 0$ *and let* $a, b \in \mathbb{N}$ *with* $a \geq 1$ *and* $b > 1$*. Let* f *be an increasing function that satisfies the recurrence relation*

$$f(n) = a \cdot f\left(\frac{n}{b}\right) + cn^d$$

whenever there exists $k \in \mathbb{N}\backslash\{0\}$ *such that* $n = b^k$*. Then*

$$f(n) \ is \begin{cases} O(n^d) & if\ a < b^d \\ O(n^d \log_b(n)) & if\ a = b^d \\ O(n^{\log_b(a)}) & if\ a > b^d \end{cases}$$

The Master Theorem

Example

We have mentioned that a function $f : \mathbb{N} \longrightarrow \mathbb{N}$ *that counts the worst-case number of comparisons used by the Merge Sort Algorithm satisfies the recurrence relation*

$$f(n) = 2 \cdot f\left(\frac{n}{2}\right) + n$$

Therefore the Master Theorem tells us that f *is* $O(n \log_2(n))$*, which agrees with our earlier analysis.*

The Closest-Pair Problem

I now wish to describe an interesting example of a divide-and-conquer algorithm. The problem that we wish to solve is the following...

Problem: Given n points $a_1 = (x_1, y_1), \ldots, a_n = (x_n, y_n)$ on the Cartesian plane, we wish to find the two closest points according to the Euclidean distance function

$$d(a_i, a_j) = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$$

Note that to check every pair of points would require order $\binom{n}{2}$ operations, which is order n^2.

I wish to describe for you a more efficient divdide-conquer algorithm that is due to Michael Samos.

The structure of this algorithm is very similar to the structure of Merge Sort. The algorithm first recursively divides the problem in half, and then shows that these smaller instances of the problem can be "merged" to find a solution to the original problem.

The Closest-Pair Problem: Division Step

At the beginning of the algorithm the points $a_1 = (x_1, y_1), \ldots, a_n = (x_n, y_n)$ are ordered first according to their x-coordinated, and then again according to their y-coordinated using the Merge Sort Algorithm. We have already seen that this process requires $O(n \log_2(n))$ comparisons. These two sortings only need to be done once. Once completed the sorted lists can be reused throughout the algorithm.

We assume that $n = 2^k$ for simplicity. The recursive division step of the algorithm uses the ordering of the x-coordinates of the points to divide the plane using a vertical line into two regions that each contain an equal number of points. If any points fall on the vertical dividing line, then these points are divided between the two regions. This division using vertical lines is then recursively repeated until the plane is dived into 2^{k-1} regions each containing 2 points.

The distance between any two points in the same region can now be computed using $O(n)$ computations.

The Closest-Pair Problem: "Merge" Step

The recursive division step of the algorithm thus reduces the original problem to the following "merge problem".

Merge Problem: Given two regions $\mathcal{O}_L$ and $\mathcal{O}_R$ divided by a vertical dividing line l that each contain m points and such that the closest two points in $\mathcal{O}_L$ are known and known to be distance d_L apart and the closest two points in $\mathcal{O}_R$ are known and known to be distance d_R apart, we need to find the closest two points in the region $\mathcal{O}_L \cup \mathcal{O}_R$.

Obviously, the distance between the closest two points in $\mathcal{O}_L \cup \mathcal{O}_R$ can be obatined by comparing the distance beween points that are either both in $\mathcal{O}_L$, both in $\mathcal{O}_R$, or such that one point is in $\mathcal{O}_L$ and the other point is in $\mathcal{O}_R$. We have already compared pairs that are both in $\mathcal{O}_L$ or both in $\mathcal{O}_R$, so we need to compare the distance between pairs with one point in $\mathcal{O}_L$ and the other point in $\mathcal{O}_R$ to see if we can beat the distance $\min(d_L, d_R)$.

The Closest-Pair Problem: "Merge" Step

Let $d = \min(d_L, d_R)$. We will examine points in $\mathcal{O}_L$ to see if any of these points are within d of a point in $\mathcal{O}_R$. Obviously we only need to consider the points in $\mathcal{O}_L$ that are within distance d of the dividing line l, but it could be the case that all of the points in $\mathcal{O}_L$ are close to the line l so this consideration does not help a worst-case complexity analysis. The key observation is that for any point $\mathcal{O}_L$, there is a fixed number (not dependent on m) number of points in $\mathcal{O}_R$ that we need to compare it to.

Form a subset A of $\mathcal{O}_L$ of points that are within d of l that are ordered buy their y-coordinated. This can be done using the already sorted points using $O(m)$ comparisons. Given any point p in A we only need to examine points on the other side of l that lie in the $d \times 2d$ strip closest to p. This $d \times 2d$ strip lies entirely in $\mathcal{O}_R$ and be divided into eight squares of size $\frac{d}{2} \times \frac{d}{2}$. Each of these square can only contain at most one point. Otherewise, there would be two points in $\mathcal{O}_R$ that are closer than d.

The Closest-Pair Problem: "Merge" Step

This means that a function $f : \mathbb{N} \longrightarrow \mathbb{N}$ that counts the worst-case number of steps required to find the closest pair of points amongst an input of of n sorted points will satisfy the recurrence relation:

$$f(n) = 2 \cdot f\left(\frac{n}{2}\right) + cn$$

Therefore, by the Master Theorem, it follows that f is $O(n \log_2(n))$.

Generating Functions

Definition

Let $f : \mathbb{N} \longrightarrow \mathbb{R}$ (so f is a sequence of real numbers). The **generating function** *for f is the formal power series*

$$G(x) = \sum_{n=0}^{\infty} f(n)x^k$$

Note that we can talk about the generating function of a finite sequence by making f eventually 0.

Example

- *The generating function for the sequence* $1, 1, 1, 1, 1, 1$ *is*

$$1 + x + x^2 + x^3 + x^4 + x^5 = \frac{(x^6 - 1)}{(x - 1)}$$

Generating Functions

Example

- *The Binomial Theorem shows that the generating function for the sequence*

$$\binom{n}{0}, \binom{n}{1}, \ldots, \binom{n}{n} \text{ is } G(x) = (1+x)^n$$

- *If* $|x| < 1$, *then*

$$G(x) = \frac{1}{1-x} = \sum_{n=0}^{\infty} x^n$$

and so $G(x)$ *is the generating function for the sequence* $(1, 1, 1, \ldots)$

- *Similarily, if* $|x| < \frac{1}{a}$, *then*

$$G(x) = \frac{1}{1-ax} = \sum_{n=0}^{\infty} a^n x^n$$

and so $G(x)$ *is the generating function for the sequence* $(1, a, a^2, \ldots)$

The Cauchy Product

Since generating function are power series, we need to recall some tools from Calculus that allow us to manipulate convergent series.

Definition

Let (a_n) and (b_n) be sequences. The **convolution** *of (a_n) and (b_n), writen $(a_n) * (b_n)$, is the sequence (c_n) such that for all $n \in \mathbb{N}$,*

$$c_n = \sum_{i=0}^{n} a_i b_{n-i}$$

Theorem

Let $\sum a_n$ and $\sum b_n$ be series that both converge absolutely. Then $\sum c_n$ where $(c_n) = (a_n) * (b_n)$, called the **Cauchy product**, *also converges absolutely. Moreover,*

$$\sum_{n=0}^{\infty} c_n = \left(\sum_{n=0}^{\infty} a_n \right) \left(\sum_{n=0}^{\infty} b_n \right)$$

The Cauchy Product

Example

Consider $G(x) = \frac{1}{(1-x)^2}$. *Therefore*

$$G(x) = \left(\sum_{n=0}^{\infty} x^n\right)\left(\sum_{n=0}^{\infty} x^n\right) = \sum_{n=0}^{\infty}\sum_{k=0}^{n} x^n = \sum_{n=0}^{\infty}(n+1)x^n$$

and so $G(x)$ *is the generating function for the sequence* $(1, 2, 3, \ldots)$

Theorem

For all $n \in \mathbb{N}\backslash\{0\}$ *and* $k \in \mathbb{N}$ *with* $k \leq n$,

$$\sum_{i=0}^{k} \binom{n}{i}\binom{n}{k-i} = \binom{2n}{k}$$

This is result is known as **Vandermonde's convolution formula** *and was known to the Chinese mathematician Zhu Shijie who lived around AD1300.*

The Cauchy Product

Proof.

The Binomial Theorem tells us that a generating function for the sequence

$$\binom{2n}{0}, \binom{2n}{1}, \ldots, \binom{2n}{2n}$$

is given by $G(x) = (1+x)^{2n}$. Now, observe that

$$G(x) = (1+x)^n(1+x)^n = \left(\sum_{i=0}^{n} \binom{n}{i} x^i\right)\left(\sum_{j=0}^{n} \binom{n}{j} x^j\right)$$

$$= \sum_{k=0}^{n}\sum_{i=0}^{k} \binom{n}{i}\binom{n}{k-i} x^k$$

Comparing the coefficients of x^k shows that for all $k \leq n$,
$\sum_{i=0}^{k} \binom{n}{i}\binom{n}{k-i} = \binom{2n}{k}$

Generating Functions

This shows that generating functions are a powerful combinatorial tool. Here are some more examples:

Example

Find the number of solutions of $a_0 + a_1 + a_2 = 17$ *if* $2 \leq a_0 \leq 5$, $3 \leq a_1 \leq 6$ *and* $4 \leq a_2 \leq 7$. *The number of solutions to this equation satisfying the given constraints is equal to the coefficient of* x^{17} *in the expression*

$$\left(\sum_{n=2}^{5} x^n\right)\left(\sum_{n=3}^{6} x^n\right)\left(\sum_{n=4}^{7} x^n\right) = x^9\left(\sum_{n=0}^{3} x^n\right)\left(\sum_{n=0}^{3} x^n\right)\left(\sum_{n=0}^{3} x^n\right)$$

This must be 3, *since the coefficient of* x^8 *in* $(1 + x + x^2 + x^3)^3$ *is* 3.

Example

Consider the sequence (a_n) *that satisfies the recurrence relation*

$$a_n = 8a_{n-1} + 10^{n-1}$$

with $a_0 = 1$. *Let* $G(x)$ *be the generating function for* (a_n).

Recurrence Relations Again

Example

So

$$G(x) = \sum_{n=0}^{\infty} a_n x^n$$

Multiplying the recurrence relation by x^n yields:

$$a_n x^n = 8a_{n-1}x^n + 10^{n-1}x^n$$

Therefore

$$\begin{aligned} G(x) - 1 &= \sum_{n=1}^{\infty} a_n x^n = \sum_{n=1}^{\infty} (8a_{n-1}x^n + 10^{n-1}x^n) \\ &= 8x \left(\sum_{n=1}^{\infty} a_{n-1}x^{n-1} \right) + x \left(\sum_{n=1}^{\infty} 10^{n-1}x^{n-1} \right) \\ &= 8x \left(\sum_{n=0}^{\infty} a_n x^n \right) + x \left(\sum_{n=0}^{\infty} 10^n x^n \right) = 8xG(x) + \frac{x}{1-10x} \end{aligned}$$

Recurrence Relations Again

Example

This says that $(1-8x)G(x) = \frac{1-9x}{1-10x}$ *or*

$$G(x) = \frac{1-9x}{(1-8x)(1-10x)}$$

Expanding using partial fractions yields

$$G(x) = \frac{1}{2}\left(\frac{1}{1-8x} + \frac{1}{1-10x}\right)$$

Which can be written as the power series:

$$G(x) = \frac{1}{2}\left(\sum_{n=0}^{\infty} 8^n x^n + \sum_{n=0}^{\infty} 10^n x^n\right)$$

Recalling that $G(x)$ *was the generating function for* (a_n) *yields* $a_n = \frac{1}{2}\left(8^n + 10^n\right)$

The Binomial Series

Definition

Let $u \in \mathbb{R}$ (or $\mathbb{C}$). The **generalised binomial coefficient** $\binom{u}{k}$ *is defined by:*

$$\binom{u}{0} = 1 \ and \ \binom{u}{k} = \frac{u(u-1)\cdots(u-k+1)}{k!} \ for \ k \geq 1$$

Recall the following result from Calculus:

Theorem

Let $u, x \in \mathbb{R}$ with $|x| < 1$.

$$(1+x)^u = \sum_{n=0}^{\infty} \binom{u}{n} x^n$$

where the series on the right converges absolutely.

The Binomial Series

Example

$$\sqrt{1+x} = \sum_{n=0}^{\infty} \binom{\frac{1}{2}}{n} x^n$$

Now, $\binom{\frac{1}{2}}{0} x^0 = 1$, *and for all* $n \geq 1$,

$$\begin{aligned}
\binom{\frac{1}{2}}{n} &= \frac{\frac{1}{2}(\frac{1}{2}-1)\cdots(\frac{1}{2}-(n-1))}{n!} \\
&= (-1)^{n-1}\frac{\frac{1}{2}(1-\frac{1}{2})\cdots(n-1-\frac{1}{2})}{n!} \\
&= \frac{(-1)^{n-1}}{2^n}\frac{(2-1)(4-1)\cdots(2(n-1)-1)}{n!} \\
&= \frac{(-1)^{n-1}}{2^n}\frac{3\cdot 5\cdots(2n-3)}{n!}
\end{aligned}$$

The Binomial Series

Example

(Continued.) So, for all $n \geq 1$,

$$\begin{aligned}\binom{\frac{1}{2}}{n} &= \frac{(-1)^{n-1}}{2^n} \frac{1}{2^{n-1}(n-1)!} \frac{(2n-2)!}{n!} \\ &= \frac{(-1)^{n-1}}{2^n} \frac{1}{2^{n-1}n} \binom{2n-2}{n-1} \\ &= -2\frac{(-1)^n}{4^n} \frac{1}{n} \binom{2n-2}{n-1}\end{aligned}$$

Therefore, for all $x \in \mathbb{R}$ with $|x| < 1$,

$$\sqrt{1+x} = 1 - 2\sum_{n=1}^{\infty} \frac{(-1)^n}{4^n} \binom{2n-2}{n-1} \frac{x^n}{n}$$

The number of ways to bracket the multiplication of $n+1$ terms

Recall that we were able to count the number of ways of bracketing $n+1$ terms in order to specify the order of multiplication by defining a sequence (C_n) that satisfies the recurrence relation

$$C_{n+1} = \sum_{k=0}^{n} C_k C_{n-k}$$

with the initial conditions $C_0 = 1$ and $C_1 = 1$. We now turn our attention to finding an expression for this non-linear recurrence relation. Let $G(x)$ be the generating function for (C_n). I.e.

$$G(x) = \sum_{n=0}^{\infty} C_n x^n$$

The number of ways to bracket the multiplication of $n+1$ terms

Now,

$$G(x)^2 = \left(\sum_{i=0}^{\infty} C_i x^n\right)\left(\sum_{j=0}^{\infty} C_j x^n\right) = \sum_{n=0}^{\infty}\sum_{k=0}^{n} C_k C_{n-k} x^n$$

$$= \sum_{n=0}^{\infty} C_{n+1} x^n = \frac{1}{x}\left(G(x) - 1\right)$$

This means that $xG(x)^2 - G(x) + 1 = 0$, and so

$$G(x) = \frac{1 \pm \sqrt{1-4x}}{2x}$$

Let's consider the power series expansion of $G(x) = \frac{1-\sqrt{1-4x}}{2x}$. Noting that

$$\frac{1-\sqrt{1-4x}}{2x} = \frac{2}{1+\sqrt{1-4x}}$$

Shows that $G(x)$ has a power series expansion at $x = 0$.

The number of ways to bracket the multiplication of $n+1$ terms

$$G(x) = \frac{1}{2x}(1-\sqrt{1-4x}) = \frac{1}{2x}\cdot 2\sum_{n=1}^{\infty}\frac{(-1)^n}{4^n}\binom{2n-2}{n-1}\frac{(-4)^n x^n}{n}$$

$$= \frac{1}{x}\cdot\sum_{n=1}^{\infty}\binom{2n-2}{n-1}\frac{x^n}{n} = \sum_{n=0}^{\infty}\binom{2n}{n}\frac{x^n}{n+1}$$

If we had instead tried to "formally" expand $\frac{1+\sqrt{1-4x}}{2x}$, we would have seen that the coefficients of the resulting series would have been negative. It follows that

$$G(x) = \sum_{n=0}^{\infty} C_n x^n = \sum_{n=0}^{\infty}\binom{2n}{n}\frac{x^n}{n+1}$$

And, for all $n \in \mathbb{N}$, $C_n = \frac{1}{n+1}\binom{2n}{n}$. These sequence (C_n) is called the **Catalan numbers**.

Graphs

Graphs are abstract mathematical objects that can be used to represent networks and relationships. A graph consists of a set of nodes or vertices, and a set of edges each of which connect two of the vertices.

Definition

We say that $G = (V, E)$ *is a* **graph** *if* $V \neq \emptyset$ *is a set and* $E \subseteq \mathcal{P}_2(V)$*. The elements* V *are called the* **vertices** *of* G *and the elements of* E *are called the edges of* G*. If* G *is a graph then we will write* $V(G)$ *for the vertices of* G *and* $E(G)$ *for the edges.*

There are several generalisations of graphs that are also important.

Definition

We say that $G = (V, E)$ *is a* **directed graph** *or* **digraph** *if* $V \neq \emptyset$ *is a set and* $E \subseteq V \times V$*.*

Definition

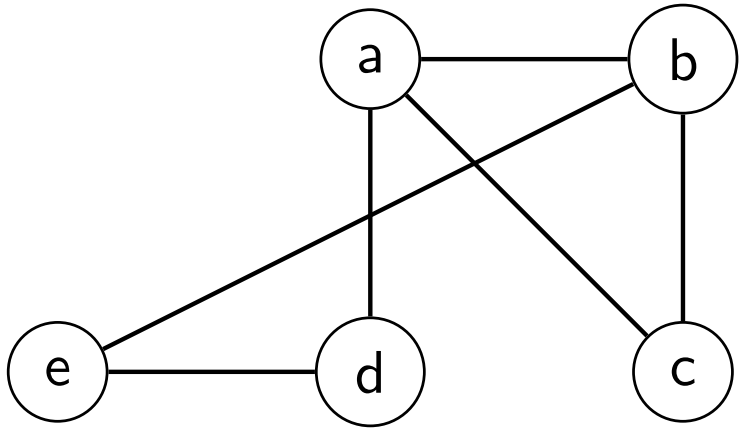
We say that $G = (V, E, \psi)$ *is a* **multigraph** *if* $V \neq \emptyset$ *and* E *are sets and* $\psi : E \longrightarrow \mathcal{P}_2(V) \cup \mathcal{P}_1(V)$*. We say that* $G = (V, E, \psi)$ *is a* **directed multigraph** *if* $V \neq \emptyset$ *and* E *are sets and* $\psi : E \longrightarrow V \times V$*.*

Drawing Graphs

We usually represent a graph by drawing the vertices as points and the edges as lines joining the vertices. For example, the graph $G = (V, E)$ with

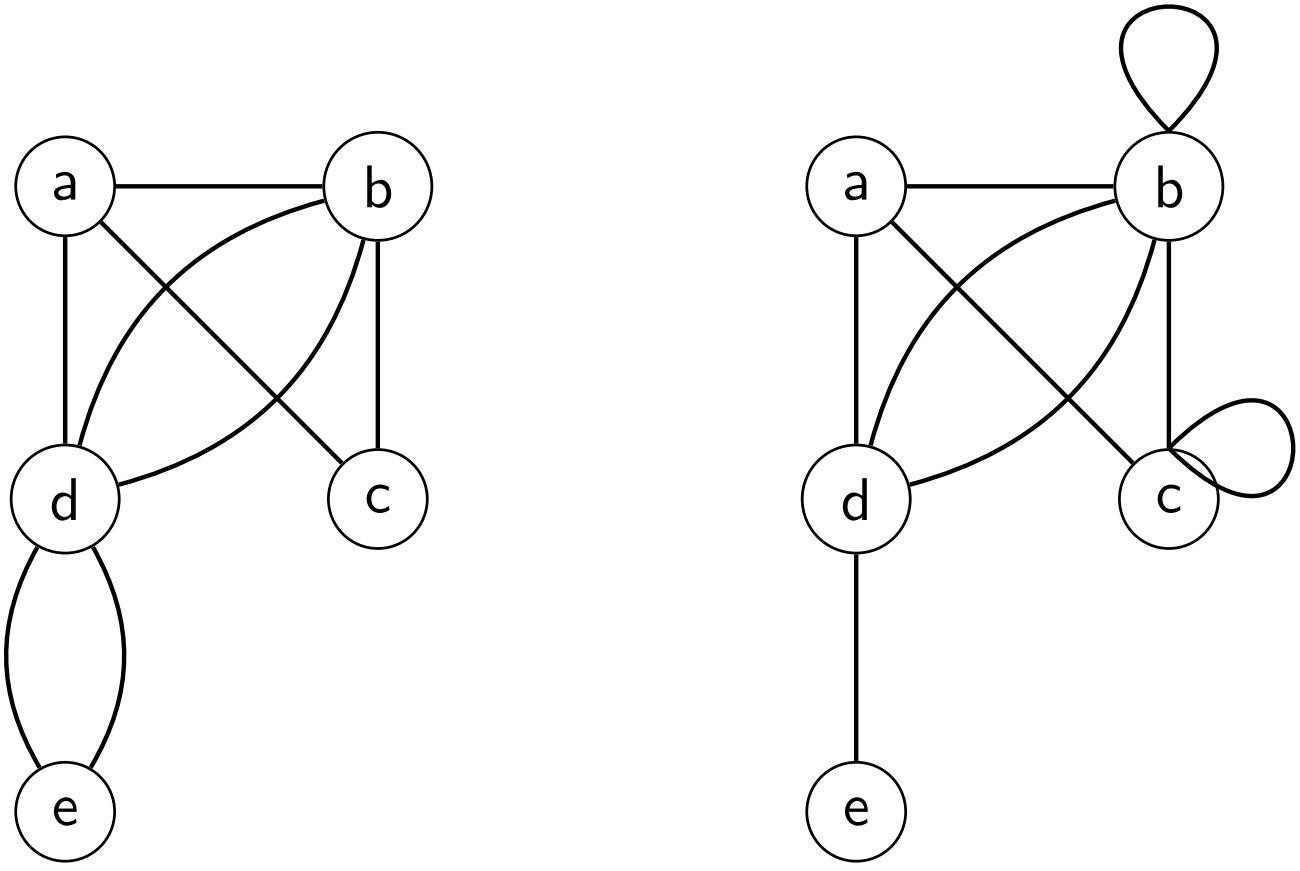
$$V = \{a, b, c, d, e\},$$
$$E = \{\{a, b\}, \{b, c\}, \{a, c\}, \{a, d\}, \{d, e\}, \{b, e\}\}$$

can be represented as



Drawing Multigraphs

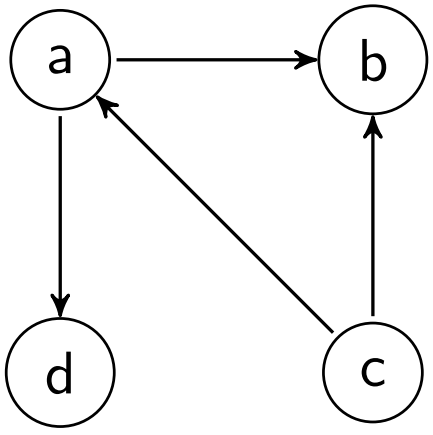
Below are drawings of multigraphs:

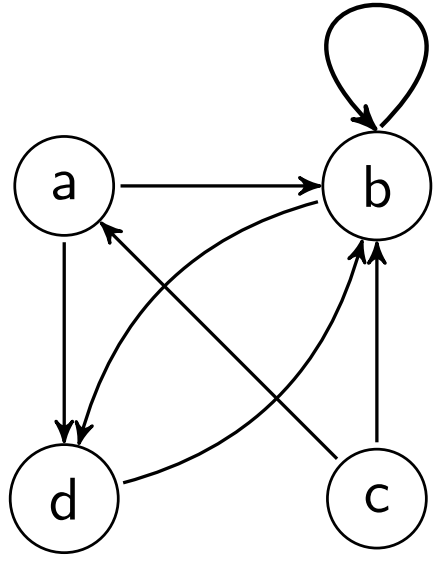


The graph on the left is $G = (V, E, \psi)$ where $V = \{a, b, c, d, e\}$, $E = [8]$ and $\psi = \left\{ \begin{array}{l} (0, \{d, e\}), (1, \{a, b\}), (2, \{b, c\}), (3, \{a, c\}), \\ (4, \{b, d\}), (5, \{b, d\}), (6, \{a, d\}), (7, \{d, e\}) \end{array} \right\}$.

Drawing Directed Graphs

Two directed graphs:





The graph on the left is $G = (V, E)$ where $V = \{a, b, c, d\}$ and $E = \{(a, b), (c, b), (c, a), (a, d)\}$. What is the graph on the right?

Graphs

Definition

Let $G = (V, E)$ be a graph or a directed graph. If V is finite, then we say that G is a **finite graph**. *If G is a finite graph, then the* **order** *of G, sometimes denoted ν_G, is $|V|$ and the* **size** *of G, sometimes denoted ε_G, is $|E|$.*

Definition

Let $n \in \mathbb{N}$ with $n \geq 3$. The **cycle** *of order n is $C_n = (V, E)$, where $V = \mathbb{Z}/n\mathbb{Z}$ and $E = \{\{[i]_n, [i+1]_n\} \mid i \in [n]\}$.*

Definition

Let $n \in \mathbb{N}$ with $n \geq 1$. The **complete graph** *of order n is the graph $K_n = (V, E)$ where $V = [n]$ and $E = \mathcal{P}_2([n])$.*

Definition

Let $n \in \mathbb{N}$ with $n \geq 3$. The **wheel** *W_n is the graph (V, E) where*

$$V = \mathbb{Z}/n\mathbb{Z} \cup \{a\} \text{ and } E = \{\{[i]_n, [i+1]_n\} \mid i \in [n]\} \cup \{\{a, [i]_n\} \mid i \in [n]\}$$

Finite Graphs

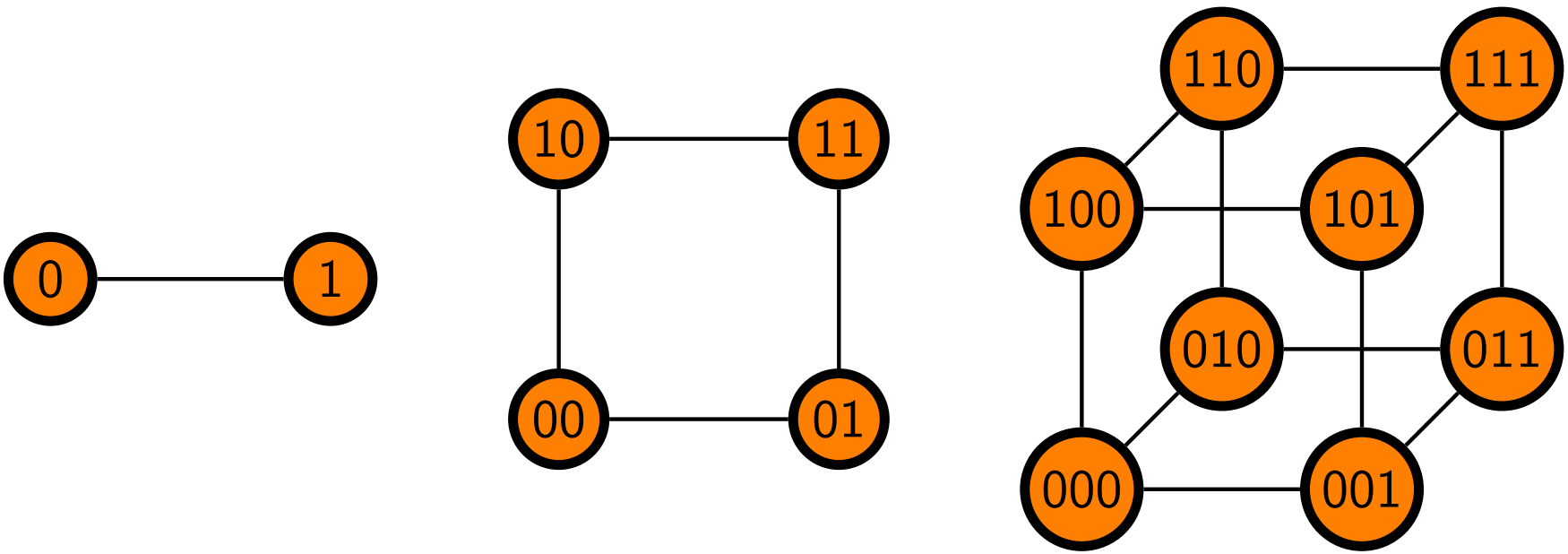
Definition

Let $n \in \mathbb{N}$ with $n \geq 1$. The **hypercube** *of dimension n, denoted Q_n, is the graph such that*

$$V(Q_n) = \{f \mid f : [n] \longrightarrow \{0,1\}\} \text{ and}$$

$$E(Q_n) = \{\{f,g\} \in \mathcal{P}_2(V(Q_n)) \mid f \text{ and } g \text{ differ on exactly one input}\}$$

The cubes Q_1, Q_2, Q_3:



Subgraphs and Isomorphisms

Definition

Let $G = (V_G, E_G)$ and $H = (V_H, E_H)$ be graphs. We say that H is **subgraph** *of G, an sometimes write $H \subseteq G$ (sorry!), if $V_H \subseteq V_G$ and $E_H \subseteq E_G$. If H is a subgraph of G and $V_G = V_H$, then we say H* **spans** *G (and call H a* **spanning subgraph** *of G). We say that H is an* **induced subgraph** *of G if $V_H \subseteq V_G$ and $E_H = E_G \cap (\mathcal{P}_2(V_H))$, and in this case we say that H is* **induced** *by V_H.*

Definition

Let $G = (V_G, E_G)$ and $H = (V_H, E_H)$ be graphs. We say that G and H are **isomorphic**, *and write $G \cong H$, if there exists a bijection $f : V_G \longrightarrow V_H$ such that for all $x, y \in V_G$,*

$$\{x, y\} \in E_G \text{ if and only if } \{f(x), f(y)\} \in E_H$$

Note that since graphs are much simpler structures than groups or orders it should be easier to recognise when small finite graphs are isomorphic.

Subgraphs and Isomorphisms

Since two isomorphic graphs are structurally the same, it is common practise to use "... is a subgraph of..." as an abbreviation for "... is isomorphic to a subgraph of...". This is similar to what we were doing when we were talking about cyclic groups... We called anything that was isomorphic to a cyclic group of order n "the cyclic group of order n".

Definition

Let $G = (V, E)$ be a graph. Let $A \subseteq V$. We write $G[A]$ for the unique subgraph of G that is induced by A. And we write $G - A$ for $G[V \setminus A]$. We will write $G - v$ instead of $G - \{v\}$ when $v \in V$.
Let $F \subseteq E$. We will write $G[F]$ for the spanning subgraph of G with edges F ((V, F)). We will write $G - F$ for $G[E \setminus F]$. If $e \in E$, then we will write $G - e$ instead of $G - \{e\}$.

Example

- *If $m \leq n$, then K_m is a subgraph of K_n. In fact if $A \subseteq [n]$ with $|A| = m$, then $K_n[A] \cong K_m$.*

Subgraphs

Example

- *Let* $F = \{\{0,1\},\{1,2\},\{2,3\},\{3,0\}\}$. $K_4[F] \cong C_4$, *so* C_4 *is isomorphic to a spanning subgraph of* K_4.
- *Let* $n \geq 3$. $W_n[\mathbb{Z}/n\mathbb{Z}] = W_n - a$ *and this graph is isomorphic to* C_n. *Therefore for all* $n \geq 3$, C_n *is an induced subgraph of* W_n.
- *Let* $F = \{\{0,1\},\{1,2\},\{2,0\}\}$. *Note that* $K_4[F]$ *is not isomorphic to* C_3, *because* $K_4[F]$ *has* 4 *vertices and* C_3 *only has* 3. *However,* $K[F] - 3$ *is isomorphic to* C_3.

Definition

Let $G = (V, E)$ *be a finite graph and let* $v \in V$. *The* **neighbourhood** *of* v *is the set*

$$N_G(v) = \{u \in V \mid \{u, v\} \in E\}$$

Degrees

Definition

Let $G = (V, E)$ be a finite graph. The **degree function (or sequence)** *is the function $d_G : V \longrightarrow \mathbb{N}$ defined by: for all $v \in V$,*

$$d_G(v) = |N_G(v)|$$

The value $d_G(v)$ is called the **degree** *of v. If $v \in V$ is such that $d_G(v) = 0$, then v is said to be* **isolated**. *If $v \in V$ is such that $d_G(v) = 1$, then v is called a* **leaf**. *The* **minimum degree** *of G, denoted $\delta(G)$, and the* **maximum degree** *of G, denoted $\Delta(G)$ are defined, respectively, by*

$$\delta(G) = \min\{d_G(v) \mid v \in V\} \ and \ \Delta(G) = \max\{d_G(v) \mid v \in V\}$$

Example

- *Let $n \geq 3$. The graph W_n has one vertex with degree n, and n vertices with degree 3.*
- *Let $n \geq 1$. The graph K_n has n vertices with degree $n - 1$.*

Degrees

The following result is due to Euler (1736).

Theorem

(Handshaking Theorem) If $G = (V, E)$ is a graph, then

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

and the number of vertices with odd degree is even.

Proof.

Every edge has two ends and so contributes a value of two to the sum of the degrees. □

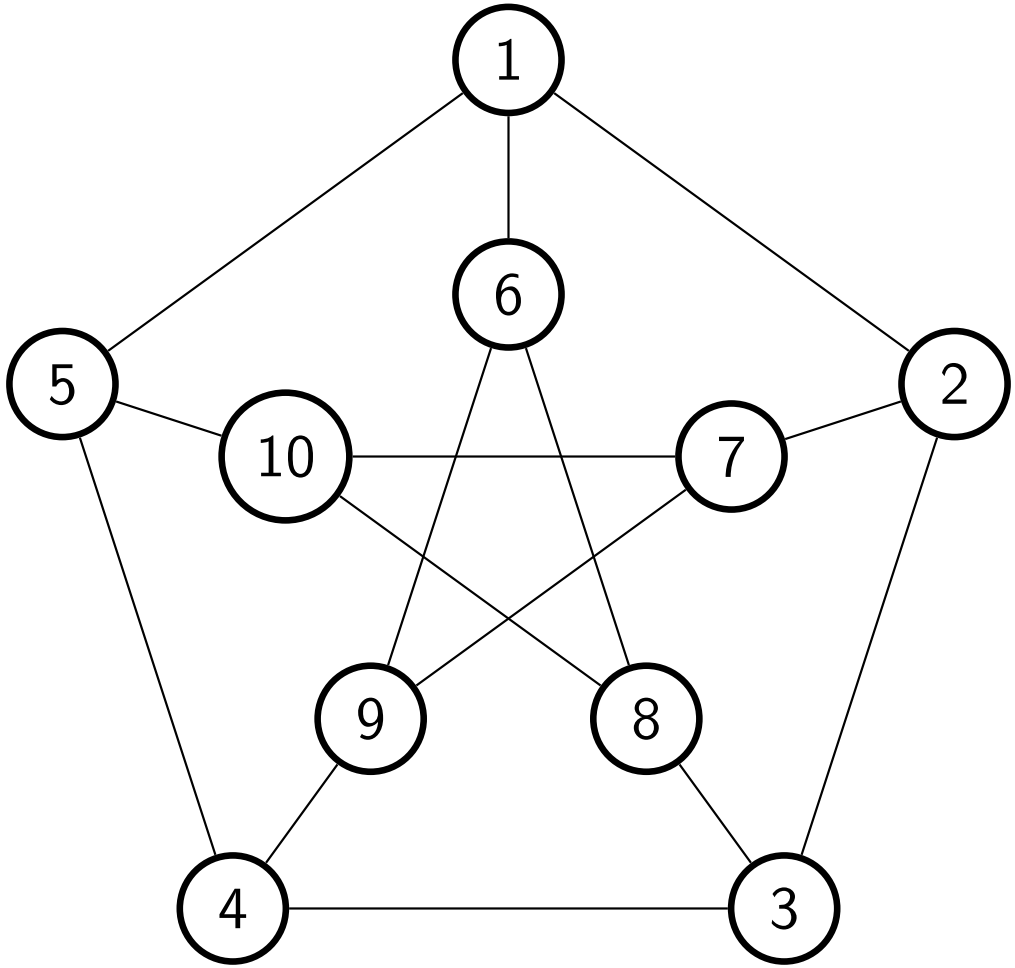
Definition

*Let $G = (V, E)$ be a graph and let $n \in \mathbb{N}$. We say that G is n-***regular** *if for all $v \in V$, $d_G(v) = n$. We may also simply say that G is* **regular** *if there exists $n \in \mathbb{N}$ such that G is n-regular.*

Degree sequences

Example

The **Petersen Graph** *is a* 3*-regular graph of order* 10*:*



Degree Sequences and Switches

Consider a graph $G = (V, E)$ of order n. The degree sequence of this graph (the function d_G) is a sequence of n natural numbers. Since there is no ordering on the vertices of G, we may as well consider this function to be a non-increasing list of natural numbers. The first question about graphs that we will address is: which sequences $d_1 \geq \cdots \geq d_n$ can be degree sequences for graphs?

Definition

Let $d_1, \ldots, d_n \in \mathbb{N}$ with $d_1 \geq \ldots \geq d_n$. We say that the sequence $d_1 \geq \ldots \geq d_n$ is **graphical** *if there exists a graph $G = (V, E)$ with $V = \{v_1, \ldots, v_n\}$ such that for all $1 \leq i \leq n$, $d_G(v_i) = d_i$.*

Lemma

If $d_1, \ldots, d_n \in \mathbb{N}$ with $d_1 \geq \ldots \geq d_n$ and $d_1 > n - 1$, then $d_1 \geq \ldots \geq d_n$ is not graphical.

Proof.

A vertex in a graph of order n can be joined to at most $n - 1$ other vertices. □

Switches

Definition

Let $G = (V_G, E_G)$ be a graph. We say that $H = (V_H, E_H)$ is a 2-switch of G if there exists distinct $x, y, w, z \in V_G$ such that $\{x, y\}, \{w, z\} \in E_G$ and $\{x, w\}, \{y, z\} \notin E_G$, and $V_H = V_G$ and

$$E_H = (E_G \backslash \{\{x, y\}, \{w, z\}\}) \cup \{\{x, w\}, \{y, z\}\}$$

We write $G \xrightarrow{2s} H$ if there exists a finite sequence of graphs $G = H_0, \ldots, H_n = H$ such that for all $0 \leq i < n$, H_{i+1} is a 2-switch of H_i.

Note that the relation $G \xrightarrow{2s} H$ is commutative, because if H is a 2-switch of G, then G is a 2-switch of H.

Example

Consider $G = (V, E_G)$ and $H = (V, E_H)$ where $V = [6]$ and $E_G = \mathcal{P}_2([3]) \cup \mathcal{P}_2(\{3, 4, 5\})$ and $E_H = \{\{0, 1\}, \{1, 2\}, \{2, 3\}, \{3, 4\}, \{4, 5\}, \{5, 0\}\}$ then H is a 2-switch of G. Moreover, $H \cong C_6$.

Switches and Degree Sequences

Lemma

Let $d_1, \ldots, d_n \in \mathbb{N}$ with $d_1 \geq \ldots \geq d_n$. Let $G = (V, E)$ be a graph such that $V = \{v_1, \ldots, v_n\}$ and for all $1 \leq i \leq n$, $d_G(v_i) = d_i$. Then there exists a graph G' such that $G \xrightarrow{2s} G'$ and $N_{G'}(v_1) = \{v_2, \ldots v_{d_1+1}\}$.

Proof.

Let $d = \Delta(G) = d_1$. Suppose that $N_G(v_1) \neq \{v_2, \ldots v_{d+1}\}$. Let $2 \leq i \leq d$ be such that $\{v_1, v_i\} \notin E$. Therefore, since $d_G(v_1) = d$, there exists $j > d + 1$ such that $\{v_1, v_j\} \in E$. Now, $d_G(v_i) = d_i \geq d_j = d_G(v_j)$ and v_j is connected to v_1. Therefore v_j is connected to $d_j - 1$ vertices other than v_1, and v_i is connected to $d_i > d_j - 1$ many other vertices. Therefore, there exists v_k such that $\{v_i, v_k\} \in E$ and $\{v_j, v_k\} \notin E$. We can now perform a 2-switch on G using the vertices v_1, v_i, v_j, v_k to obtain a graph H'. This 2-switch adds the edges $\{v_1, v_i\}$ and $\{v_j, v_k\}$, and removes the edges $\{v_i, v_k\}$ and $\{v_1, v_j\}$. Therefore $N_H(v_1) = N_G(v_1) \cup \{v_i\}$.
If $N_H(v_1) \neq \{v_2, \ldots, v_{d+1}\}$, then this process can be repeated until the desired result is obtained. □

Graphs with the same degree sequence

Theorem

(Berge (1973)) Let G and H be graphs with $V(G) = V(H) = V$. For all $v \in V$, $d_G(v) = d_H(v)$, if and only if $G \xrightarrow{2s} H$.

Proof.

$\Leftarrow$: Let $G = H_0, \ldots, H_n = H$ be a sequence such that for all $0 \leq i < n$, H_{i+1} is a 2-switch of H_i. The fact that for all $v \in V$, $d_G(v) = d_H(v)$, follows immediately from the fact that 2-switches preserve the degree sequence.

$\Rightarrow$: We will prove the cnverse by induction on the order of G and H. Suppose that the result holds for graphs of order n. Let $V = \{v_1, \ldots, v_{n+1}\}$ and let $d_1, \ldots, d_{n+1} \in \mathbb{N}$ with $d_1 \geq \ldots \geq d_{n+1}$. Let G and H be graphs such that for all $1 \leq i \leq n+1$, $d_G(v_i) = d_H(v_i) = d_i$ (note that this is exactly the same as saying that G and H have the same degree sequence). Using the previous lemma, we can find $G \xrightarrow{2s} G'$ and $H \xrightarrow{2s} H'$ such that $N_{G'}(v_1) = N_{H'}(v_1)$.

Graphs with the same degree sequence

Proof.

(Continued.) Therefore, the degree sequence of $G' - v_1$ and $H' - v_1$ are identical: $d_2 - 1, \ldots, d_{d_1+1} - 1, d_{d_1+2}, \ldots, d_{n+1}$. Therefore, by the induction hypothesis, $G' - v_1 \xrightarrow{2s} H' - v_1$. It should be clear that having the additional vertex v_1 in $G' - v_1$ and $H' - v_1$ will not change the 2-switches that take $G' - v_1$ to $H' - v_1$, and so $G' \xrightarrow{2s} H'$. Therefore, by reversing the 2-switches that take H to H', we get $G \xrightarrow{2s} H$. The result now follows by induction. □

The next result yields a recursive algorithm that can be used to decide whether a degree sequence is graphical or not.

Theorem

Let $n \geq 2$. Let $d_1, \ldots, d_n \in \mathbb{N}$ with $d_1 \geq \ldots \geq d_n$. Then $d_1 \geq \ldots \geq d_n$ is graphical if and and only if the sequence

$$d_2 - 1, \ldots, d_{d_1+1} - 1, d_{d_1+2}, \ldots, d_n$$

ordered in non-increasing order is graphical.

Graphical Degree Sequences

Proof.

$\Leftarrow$: Let $G = (V, E)$ be a graph with $V = \{v_2, \ldots, v_n\}$ such that for all $2 \leq i \leq d_1 + 1$, $d_G(v_i) = d_i - 1$, and for all $d_1 + 2 \leq i \leq n$, $d_G(v_i) = d_i$. Define a new graph $H = (V', E')$ such that $V' = V \cup \{v_1\}$ and

$$E' = E \cup \{\{v_1, v_i\} \mid 2 \leq i \leq d_1 + 1\}$$

Therefore, for all $1 \leq i \leq n$, $d_H(v_i) = d_i$, which shows that $d_1 \geq \ldots \geq d_n$ is graphical.

$\Rightarrow$: Conversely, let $G = (V, E)$ be a graph with $V = \{v_1, \ldots, v_n\}$ such that for all $1 \leq i \leq n$, $d_G(v_i) = d_i$. Using the previous two results we can find a graph G' with the same vertex set and degree sequence as G such that

$$N_{G'}(v_1) = \{v_2, \ldots, v_{d_1+1}\}$$

Now, $V(G' - v_1) = \{v_2, \ldots, v_n\}$ and $G' - v_1$ satisfies: for all $2 \leq i \leq d_1 + 1$, $d_{G'-v_1}(v_i) = d_i - 1$, and for all $d_1 + 2 \leq i \leq n$, $d_{G'-v_1}(v_i) = d_i$, showing the desired sequence is graphical. $\square$

Graphical Degree Sequences

Example

The sequence $5 \geq 5 \geq 5 \geq 5 \geq 4 \geq 3 \geq 1$ *is graphical, if and only if the sequence* $4 \geq 4 \geq 4 \geq 3 \geq 2 \geq 1$ *is graphical, if and only if the sequence* $3 \geq 3 \geq 2 \geq 1 \geq 1$ *is graphical, if and only if the sequence* $2 \geq 1 \geq 0 \geq 1$, *or* $2 \geq 1 \geq 1 \geq 0$ *is graphical, if and only if* $0 \geq 0 \geq 0$ *is graphical. And,* $0 \geq 0 \geq 0$ *is clearly graphical, so* $5 \geq 5 \geq 5 \geq 5 \geq 4 \geq 3 \geq 1$ *is graphical.*

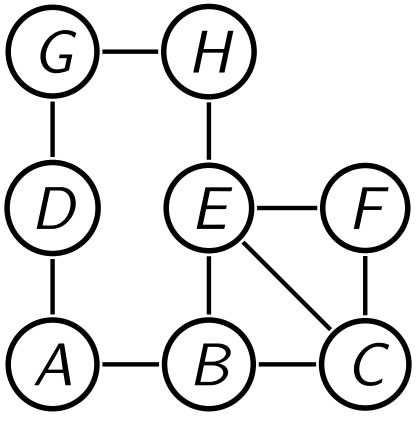
This algorithm can easily be reverse engineered to produce a graph with a given degree sequence.

Definition

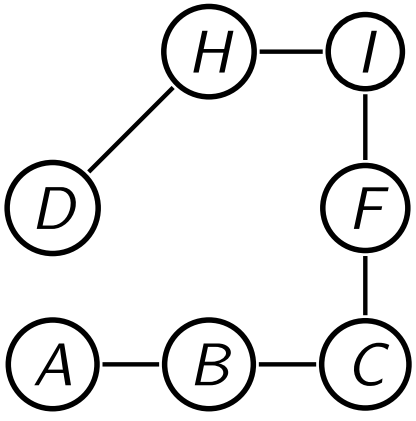
Let $G_1 = (V_1, E_1)$ *and* $G_2 = (V_2, E_2)$ *be graphs. Then the* **union** *of* G_1 *and* G_2, *denoted* $G_1 \cup G_2$, *is the graph with vertices* $V_1 \cup V_2$ *and edges* $E_1 \cup E_2$.

Unions of Graphs

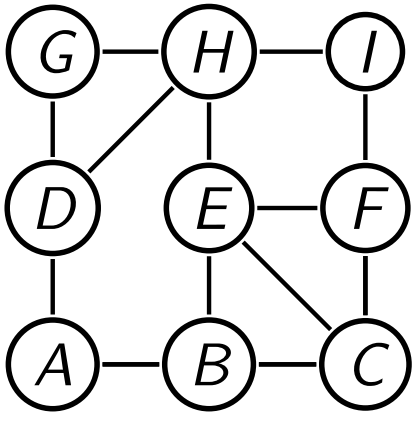
Example



G_1



G_2



$G_1 \cup G_2$

Adjacency Tables

We will now introduce several ways of representing finite graphs. A simple way to do so is using an **adjacency table**, which lists all the vertices of the graph and the vertices adjacent to them.

Example

Consider $G = (V, E)$ *with* $V = \{a, b, c, d, e\}$ *and*

$$E = \{\{a, b\}, \{a, c\}, \{a, e\}, \{c, d\}, \{c, e\}, \{d, e\}\}$$

Vertex	*Adjacent Vertices*
a	$b,\ c,\ e$
b	a
c	$a,\ d,\ e$
d	$c,\ e$
e	$a,\ c,\ d$

Adjacency Tables

A similar table can be used for directed graphs.

Example

Consider $G = (V, E)$ *with* $V = \{a, b, c, d, e\}$ *and*

$$E = \left\{ \begin{array}{c} (a,b),(a,c),(a,d),(a,e),(b,b),(b,d), \\ (c,a),(c,c),(c,e),(e,b),(e,c),(e,d) \end{array} \right\}$$

Initial Vertex	*Terminal Vertices*
a	$b,\ c,\ d,\ e$
b	$b,\ d$
c	$a,\ c,\ e$
d	
e	$b,\ c,\ d$

Adjacency Matrices

Instead of tables, matrices can also be used to describe graphs. Assume that $G = (V, E)$, where $V = \{v_1, \ldots, v_n\}$ has been ordered in some way. We then define the **adjacency matrix**, A_G, by

$$A_G = (a_{ij})_{i,j=1}^{n} \text{ where } a_{ij} = \begin{cases} 1 & \text{if } \{v_i, v_j\} \in E \\ 0 & \text{if } \{v_i, v_j\} \notin E \end{cases}$$

Note that the adjacency matrix is always symmetric, i.e., $a_{ij} = a_{ji}$.

Example

Consider $G = (V, E)$ *with* $V = \{v_1, v_2, v_3, v_4, v_5\}$ *and*

$$E = \{\{v_1, v_2\}, \{v_1, v_3\}, \{v_1, v_5\}, \{v_3, v_4\}, \{v_3, v_5\}, \{v_4, v_5\}\}$$

$$A_G = \begin{pmatrix} 0 & 1 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 & 0 \end{pmatrix}$$

Adjacency Matrices

In the case of graphs, adjacency matrices are zero-one matrices with zero trace. However, adjacency matrices can also be used to describe, in which case the entries a_{ij} are equal to the multiplicity of the edge $\{v_i, v_j\}$. The adjacency matrix remains symmetric for any type of undirected graph.

An adjacency matrix A_G for a directed graph $G = (V, E)$ with ordered vertices $V = \{v_1, \ldots, v_n\}$ is defined by

$$A_G = (a_{ij})_{i,j=1}^{n} \text{ where } a_{ij} = \begin{cases} 1 & \text{if } (v_i, v_j) \in E \\ 0 & \text{if } (v_i, v_j) \notin E \end{cases}$$

Clearly, it does not need to be symmetric.

Example

The directed graph $G = (V, E)$ with $V = \{v_1, v_2, v_3\}$ and $E = \{(v_1, v_2), (v_3, v_2), (v_3, v_1)\}$ has adjacency matrix:

$$A_G = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 0 \\ 1 & 1 & 0 \end{pmatrix}$$

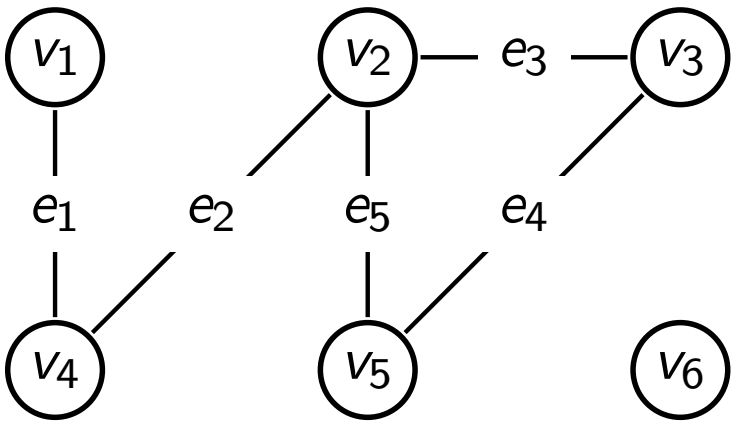
Incidence Matrices

Graphs can also be represented using **incidence matrices**. If $G = (V, E)$ is a graph, and $V = \{v_1, \ldots, v_n\}$ and $E = \{e_1, \ldots, e_m\}$ have been ordered in some way, we define $M_G = (m_{ij})$ by

$$m_{ij} = \begin{cases} 1 & \text{if } v_i \in e_j \\ 0 & \text{otherwise} \end{cases}$$

Example

The following is a graph and its incidence matrix:



$$M_G = \begin{array}{c} \\ v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \\ v_6 \end{array} \begin{array}{c} \begin{array}{ccccc} e_1 & e_2 & e_3 & e_4 & e_5 \end{array} \\ \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix} \end{array}$$

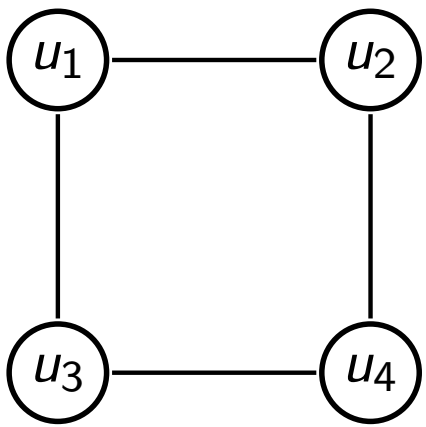
Isomorphic Graphs

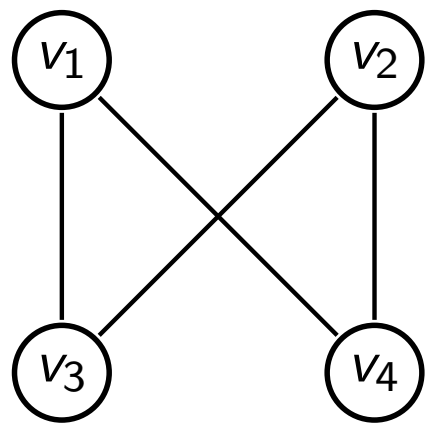
Theorem

Two graphs are isomorphic if and only if they have a common adjacency matrix. Moreover, two isomorphic graphs have exactly the same set of adjacency matrices.

Example

Let $U = \{u_1, u_2, u_3, u_4\}$ *and* $V = \{v_1, v_2, v_3, v_4\}$*, and let* $G = (U, E)$ *and* $H = (V, F)$ *be the two graphs depicted below:*





The bijection $f = \{(u_1, v_1), (u_2, v_4), (u_3, v_3), (u_4, v_2)\}$ *witnesses the fact that these two graps are isomorphic.*

Isomorphic Graphs

Example

(Continued.) Using the isomorphism f to transfer the ordering of U to V we obtain identical adjacency matrices:

$$A_G = \begin{array}{c} \\ u_1 \\ u_2 \\ u_3 \\ u_4 \end{array} \begin{array}{c} \begin{array}{cccc} u_1 & u_2 & u_3 & u_4 \end{array} \\ \begin{pmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{pmatrix} \end{array} = \begin{array}{c} \\ v_1 \\ v_4 \\ v_3 \\ v_2 \end{array} \begin{array}{c} \begin{array}{cccc} v_1 & v_4 & v_3 & v_2 \end{array} \\ \begin{pmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{pmatrix} \end{array} = A_H$$

However, if we order V as v_1, v_2, v_3, v_4 we get:

$$A_H = \begin{array}{c} \\ v_1 \\ v_2 \\ v_3 \\ v_4 \end{array} \begin{array}{c} \begin{array}{cccc} v_1 & v_2 & v_3 & v_4 \end{array} \\ \begin{pmatrix} 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 \end{pmatrix} \end{array}$$

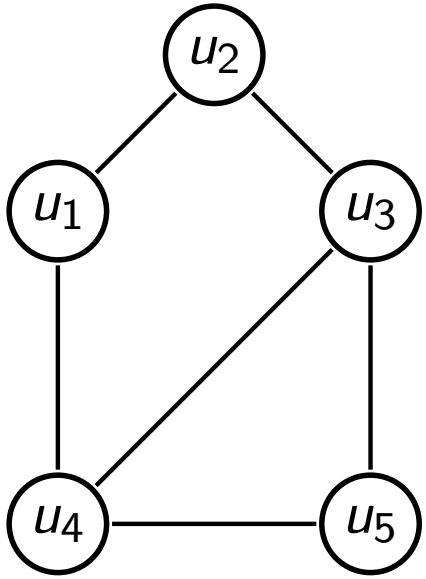
Graph Invariants

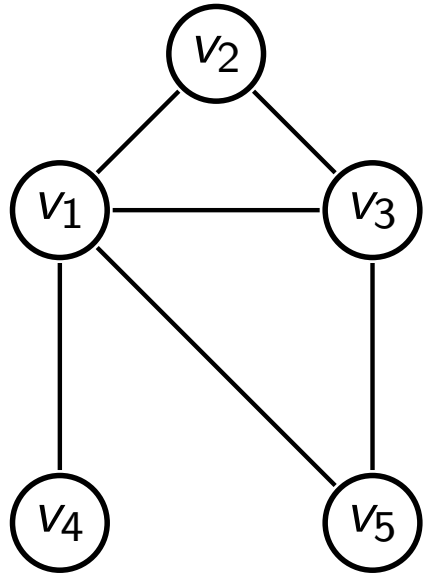
Properties of graphs that are preserved by graph isomorphisms are called **graph invariants**. The following are graph invariants:

- The number of edges of a graph,
- The total number of vertices in a graph,
- For all $k \in \mathbb{N}$, the total number of vertices of degree k

Example

The two graphs below are not isomorphic because one has a vertex of degree 4*, and the other does not:*





Invariance of Subgraphs

Let $G = (U, E)$ and $H = (V, F)$ be graphs. Suppose that the bijection $\phi : U \longrightarrow V$ witnesses the fact that $G \cong H$.

Firstly, note that if $u \in U$ is a vertex of degree k in G, then $\phi(u)$ must be a vertex of degree k in H (Prove this!)

Secondly, if $G' = (U', E')$ is a subgraph of G, then the map ϕ restricts to G' to show that H has a subgraph that is isomorphic to G'. I.e. $H' = (\phi\text{“}U', F')$ where

$$F' = \{\{\phi(u), \phi(v)\} \mid \{u, v\} \in E'\}$$

is isomorphic to G' and H' is a subgraph of H. This shows that isomorphic graphs must have exactly the same subgraphs.

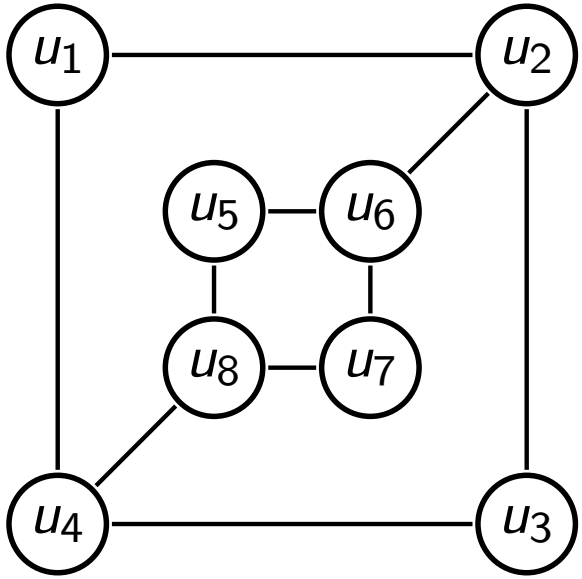
Lemma

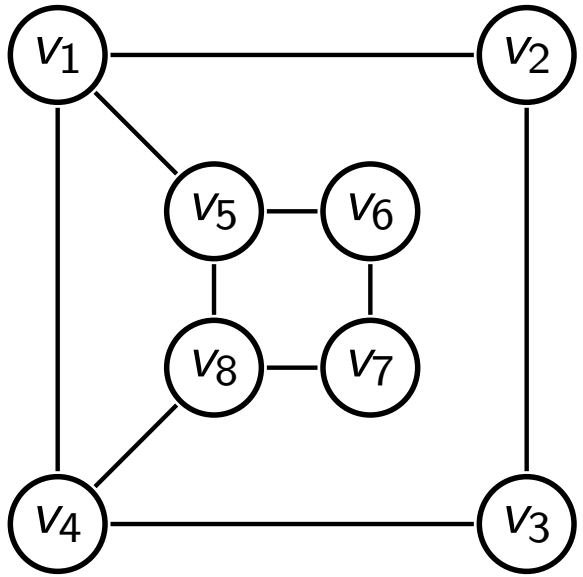
Let $k \in \mathbb{N}$. If two graphs are isomorphic, then all subgraphs consisting of vertices of degree k must be isomorphic.

Isomorphic Graphs

Example

Consider the two graphs:



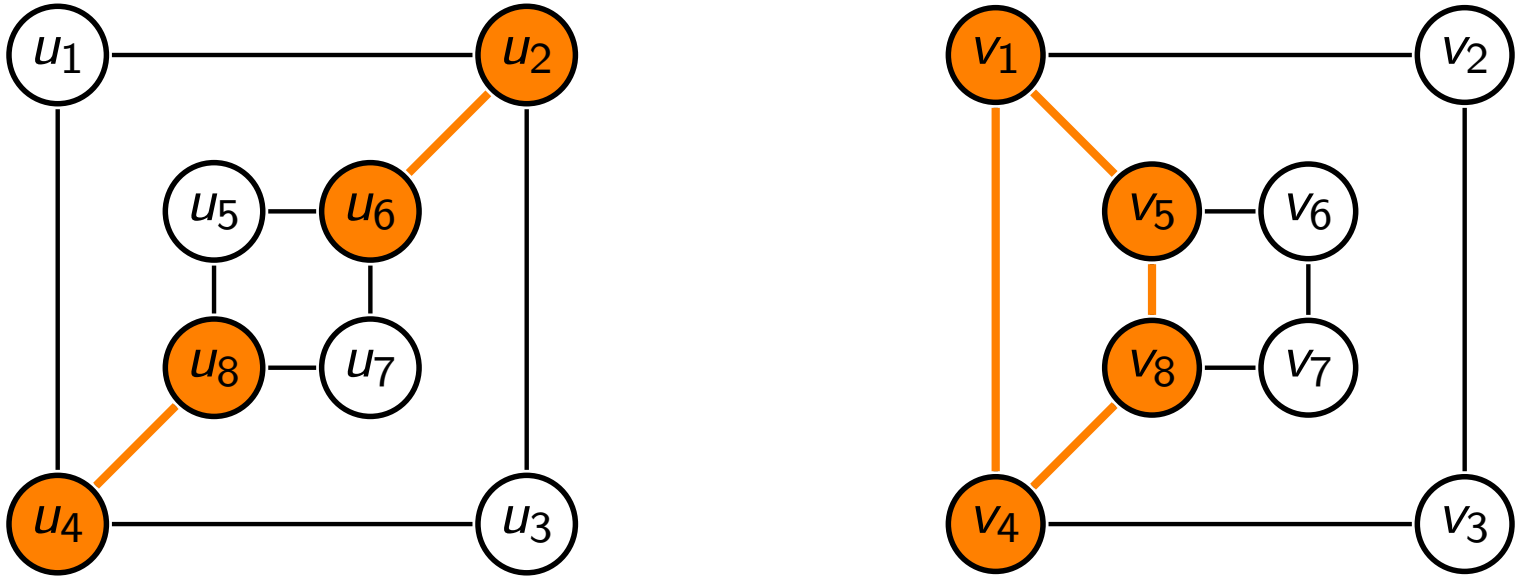


For all $k \in \mathbb{N}$, these graphs contain the same number of vertices of degree k.

Isomorphic Graphs

Example

(Continued.) The subgraphs consisting of vertices of degree 3 and their common edges are marked in orange:

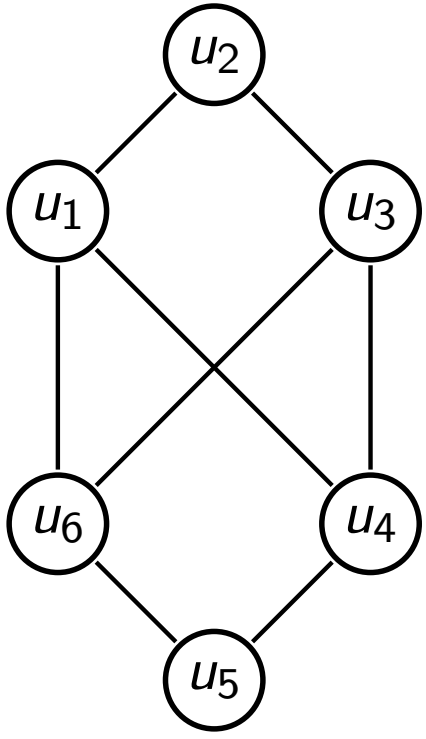


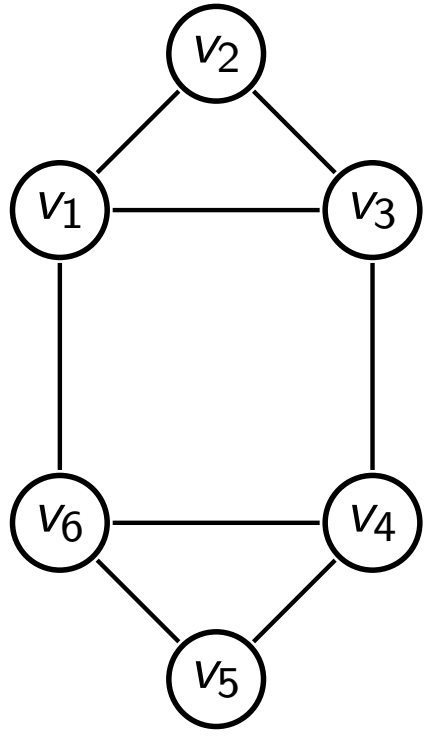
Therefore, these graphs are not isomorphic.

Isomorphic Graphs

Example

Consider the two graphs below:





The graph on the right has K_3 as a subgraph and the graph on the left does not, so these two graphs are not isomorphic.

Walks

Definition

Let $G = (V, E)$ be a graph. Let $k \in \mathbb{N}$ and let $u, v \in V$. We say that $W : [k+1] \longrightarrow V$ is a **walk** *of* **length** *k from u to v in G, and write $W : u \xrightarrow{k} v$, if $W(0) = u$ and $W(k) = v$, and for all $i \in [k]$, $\{W(i), W(i+1)\} \in E$.*

We might write $W : u \xrightarrow{*} v$ to indicate that W is a walk from u to v of some, unspecified, length. We may also write $W : u \to u_1 \to \cdots \to u_{k-1} \to v$ when we want to be more specific about the walk. We might also write $W = e_0 \cdots e_{k-1}$ when we want to specify the edges of a walk.

Definition

Let $G = (V, E)$ be a graph and let $u, v \in V$. Let $W : [k+1] \longrightarrow V$ be a walk of length k from u to v in G. We say that W is **closed** *if $u = v$. If W is injective, then we say that W is a* **path**. *We say that W is a* **cycle** *if $u = v$ and for all $1 \leq i, j \leq k-1$, $W(i) \neq W(j)$. If $k = 0$, then we call W the* **trivial path**.

Walks

Theorem

Let $G = (V, E)$ be a graph with $V = \{v_1, \ldots, v_n\}$ and adjacency matrix $A_G = (a_{ij})$. Let $k \in \mathbb{N}\backslash\{0\}$ and let $1 \leq i, j \leq n$. The number of walks of length k from v_i to v_j is equal to the $(i, j)^{th}$ entry of the matrix A_G^k.

Proof.

We prove this result by induction on k. The definition of the adjacency matrix $A_G = (a_{ij})$ ensures that the result holds when $k = 1$.
Assume that the result holds for k. Let $B = (b_{ij}) = A_G^k$. Therefore $A_G^{k+1} = BA = (c_{ij})$ and the $(i, j)^{\text{th}}$ entry of A_G^{k+1} is

$$c_{ij} = \sum_{l=1}^{n} b_{il} a_{lj}$$

By the induction hypothesis, for all $1 \leq l \leq n$, b_{il} is the number of walks of length k between v_i and v_l. This value is added to c_{ij} if and only if $\{v_l, v_j\} \in E$. Therefore c_{ij} counts the number of walks of length $k + 1$ between v_i and v_j. This completes the proof. □

Connected Components

Let $G = (V, E)$ be a graph. Let $u, v, w \in V$. If $W : u \xrightarrow{k} v$, then $W^{\leftarrow} : [k+1] \longrightarrow V$ defined by: for all $i \in [k+1]$, $W^{\leftarrow}(i) = W(k-i)$ is walk of length k from v to u. If $W_1 : u \xrightarrow{k_1} v$ and $W_2 : v \xrightarrow{k_2} w$, then $W_1 W_2 : [k_1 + k_2 + 1] \longrightarrow V$ defined by: for all $i \in [k_1 + k_2 + 1]$,

$$W_1 W_2(i) = \begin{cases} W_1(i) & \text{if } i < k_1 + 1 \\ W_2(i - k_1) & \text{if } i \geq k_1 + 1 \end{cases}$$

These two observations, combined with the fact that the trivial path show that for all $u \in V$, there exists $W : u \xrightarrow{*} u$, show that the relation $\sim \subseteq V \times V$ defined by: for all $u, v \in V$,

$$(u, v) \in \sim \text{ if and only if there exists } W : u \xrightarrow{*} v$$

is an equivalence relation. Therefore

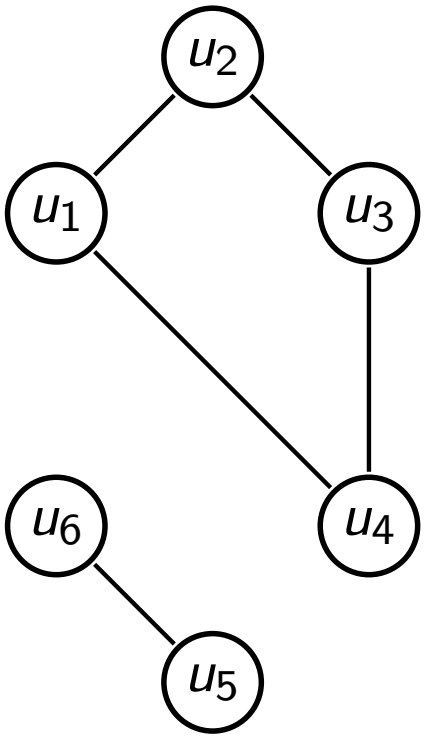
$$C(G) = \{[u]_\sim \mid u \in V\}$$

is a partition of G. The cardinality of $C(G)$ is the **number of connected components of** G. If $|C(G)| = 1$, then we say that G is **connected**.

Connected Components

Example

Consider the graph $G = (U, E)$ *shown below:*



$$C(G) = \{\{u_1, u_2, u_3, u_4\}, \{u_5, u_6\}\}$$

G is not connected because $|C(G)| = 2$. Alternatively, one could observed that there is no W : $u_1 \xrightarrow{*} u_5$, and so u_1 and u_5 must be in distinct $\sim$-equivalence classes.

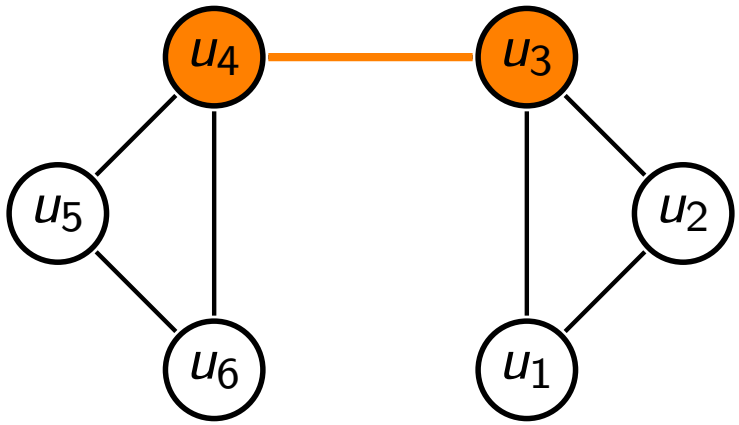
Definition

Let $G = (V, E)$ be a graph. We say that $v \in V$ is a **cut vertex** *of G if $G - v$ has more connected components than G. We say that $e \in E$ is a* **cut edge** *of G if $G - e$ has more connected components than G.*

Cut Vertices

Example

In the graph below, the cut edges and cut vertices have been marked in orange:



Euler Trails and Tours

We now turn to addressing the question of whether or not there exists a walk in a graph that traverses an edge exactly once. This problem was first considerred by Euler who addressed the question of whether or not one (in 1735) could tour the city of Königsberg and in doing so pass over each of its seven bridges exactly once.

Definition

Let $G = (V, E)$ be a graph. A walk specified by its edges $W = e_1 \cdots e_n$ is a **trail** *if for all $1 \leq i < j \leq n$, $e_i \neq e_j$. A trail $W = e_1 \cdots e_n$ is an* **Euler trail** *if $E = \{e_1, \ldots, e_n\}$. An Euler trail $W = e_1 \cdots e_n$ is an* **Euler tour** *if there exists $u \in V$ such that $W : u \xrightarrow{n} u$. If G has an Euler tour, then we say that G is* **Eulerian**.

Example

$$W = \left\{ \begin{array}{c} (0,0),(1,1),(2,2),(3,3),(4,4) \\ (5,0),(6,2),(7,4),(8,1),(9,3),(10,0) \end{array} \right\}$$

is an Euler tour of K_5, so K_5 is Eulerian.

Euler Trails and Tours

Theorem

Let $G = (V, E)$ be a connected graph. Then G is Eulerian if and only if every vertex of G has even degree.

Proof.

$\Rightarrow$: Let $W : u \xrightarrow{*} u$ be an Euler tour. Let $v \in V$ with $v \neq u$. Every instance of W passing through v corresponds to a distinct edge through which W enters v, and a distinct edge through which W leaves v. Therefore, each passes through v contributes 2 to the degree of v, and, since each edge of G appears in W, these passes must account for the entire degree of v. Therefore the degree of v is even. It is also true that each pass through u while the walk is 'in progress' must also contribute 2 to the degree of u. Since W starts and ends at u, the degree of u must also be even.

$\Leftarrow$: Assume that the degree of every vertex in G is even. Let $u \in G$. Note that since G is connected, $d_G(u) \neq 0$. We first claim that there exists a trail $W : u \xrightarrow{*} u$.

Euler Trails and Tours

Proof.

(Continued.) This is true because for each vertex $v \neq u$ you move to via an edge, you can always leave that vertex by a distinct edge, because the degree of v is even. This means that you can start building W by leaving u and continue to leave the vertices you reach by distinct edges until you reach u again. Therefore, let $W' = e_1 \cdots e_k$ be a trail $W' : u \xrightarrow{*} u$ that contains the most number of edges. We will argue that W' is an Euler trail. Let $w_0, \ldots, w_l$ be the vertices that are in the domain of W'. Let

$$H = G - \{e_1, \ldots, e_k\}$$

If there exists $0 \leq i \leq l$ such that the degree of w_i in H is nonzero, then the degree of w_i must be even in H. If the degree of w_i is nonzero in H, then we could find a trial $U : w_i \xrightarrow{*} w_i$ with $U = f_1 \cdots f_m$ in H. If e_j and e_{j+1} are two edges incident with w_i in W' then we could extend W' to $W'' = e_1 \cdot e_j f_1 \cdots f_m e_{j+1} \cdots e_k$, which would contradict the maximality of W'. Therefore $w_1, \ldots, w_l$ have degree zero in H.

Euler Trails and Tours

Proof.

(Continued.) Now, we argue that $\{w_1, \ldots, w_l\} = V$. Suppose that $x \in V$ with $x \notin \{w_1, \ldots, w_l\}$. Since G is connected, there exists a walk $U' : w_1 \xrightarrow{*} x$ with $U' = p_1 \cdots p_s$. At least one of the edges $p_1, \ldots, p_s$ must connect a vertex in $\{w_1, \ldots, w_l\}$ with a vertex in $V \backslash \{w_1, \ldots, w_l\}$. But this edge cannot be in the walk W', which contradicts the fact that every vertex in $\{w_1, \ldots, w_l\}$ has degree 0 in H. This shows that W' must be an Euler tour. □

Example

- *K_n is Eulerian exactly when $n \geq 3$ is odd*
- *W_n is never Eulerian*

Note that this proof yields an algorithm for finding an Euler tour in a graph $G = (V, E)$ in which every vertex has even degree.

Euler Trails and Tours

The algorithm can be described as follows: Let $G = (V, E)$ be a connected graph such that every vertex of G has even degree.

- Starting at any vertex $u \in V$, and moving along any edge $e \in E$ with $u \in e$, find a trail $W : u \xrightarrow{*} u$ in G with $W = e_1 \cdots e_k$ and $e_1 = e$.
- Let $H = G - \{e_1, \ldots, e_k\}$. If the edge set of H is nonempty, then repeat the previous step until all of the edges have been removed.
- Compose the trails found to produce an Euler tour in the graph G.

Note that this algorithm makes it clear that if $G = (V, E)$ is a connected graph such that every vertex of G has even degree, then for all $e \in E$, there exists an Euler tour of G that begins by moving along the edge e. In fact, if $W' : u \xrightarrow{*} v$ is any walk in G, then there exists an Euler tour of G that begins with W'.
This result also allows us to characterise the existence of an Euler trail.

Theorem

Let $G = (V, E)$ be a connected graph. G has an Euler trail if and only if G has at most two vertices of odd degree.

Euler Trails and Tours

Proof.

$\Rightarrow$: Let $u, v \in V$ and let $W : u \xrightarrow{*} v$ be an Euler trail of G. Note that since G is connected, the domain of W must be V. Since W must enter and leave every $w \in V \backslash \{u, v\}$ by a distinct edge, it follows that for all $w \in V \backslash \{u, v\}$, w has even degree. Therefore, at worst, u and v in G have odd degree.

$\Leftarrow$: Suppose that G has at most two vertices of odd degree. It follows from the handshaking theorem that G either has 0 vertices of odd degree or 2 vertices of odd degree. If G has 0 vertices of odd degree, then the previous theorem shows that G has an Euler tour. So, suppose that there are $u, v \in V$ such that $u \neq v$, and u and v have odd degree. Let $a \notin V$, and let $V' = V \cup \{a\}$. Let $e_1 = \{u, a\}$ and $e_2 = \{a, v\}$, and let $E' = E \cup \{e_1, e_2\}$. Let $G' = (V', E')$. Now, G' is a connected graph such that every vertex of G' has even degree. This means that we can find an Euler tour $W = e_1 e_2 e_3 \cdots e_k$ of G'. Removing the edges e_1 and e_2 from W yields an Euler trail of G. $\square$

Hamilton Paths and Cycles

Euler tours are walks that traverse every edge in a graph. We now address the problem of finding walks that visit each vertex exactly once.

Definition

Let $G = (V, E)$ be a graph. Let $W : u \xrightarrow{*} v$ be a walk. We say that W is a **Hamilton path** *if W is a path and $\operatorname{ran} W = V$. We say W is a* **Hamilton cycle** *if W is a cycle and $\operatorname{ran} W = V$. We say that G is* **Hamiltonian** *if G has a Hamilton cycle.*

Example

- *$W = \{(0,0),(1,1),(2,2),(3,3),(4,4)\}$ is a Hamilton path in K_5*
- *$W = \{(0,0),(1,1),(2,2),(3,3),(4,0)\}$ is a Hamilton cycle in K_4*
- *In fact, for all $n \geq 3$, K_n is Hamiltonian*

Hamilton Paths and Cycles

Note that if G is a graph and $W : u_1 \to \cdots \to u_k \to u_1$ is a Hamilton cycle, then for all $1 \leq i \leq k$,

$$W' : u_i \to u_{i+1} \to \cdots \to u_k \to u_1 \to \cdots u_{i-1} \to u_i$$

is also a Hamilton cycle. Therefore, if G is Hamiltonian, then G has a Hamilton cycle starting at any vertex.

Lemma

Let $G = (V, E)$ be a graph. If G is Hamiltonian, then for all $S \subseteq V$ with $S \neq \emptyset$, the number of connected components of $G - S$ is less than or equal to $|S|$.

Proof.

Let $S \subseteq V$ with $S \neq \emptyset$ and let $u \in S$. Let $W : u \xrightarrow{*} u$ be a Hamilton cycle in G. If $G - S$ has 1 connected component, then the result holds, so assume that $G - S$ has k connected components with $k > 1$. Let $G_1, \ldots, G_k$ be the connected components of $G - S$. Note that W must pass through all of the components.

Hamilton Paths and Cycles

Proof.

(Continued.) For all $1 \leq i \leq k$, let u_i be the last vertex in the path of W that belongs to G_i. For all $1 \leq i \leq k$, let v_i be the vertex joined to u_i by the edge traversed by W. It follows that $v_i \in S$. Moreover, since W is a cycle, for all $1 \leq i < j \leq k$, $v_i \neq v_j$. Therefore, $|S| \geq k$. □

Theorem

(Ore 1962) Let $G = (V, E)$ be a graph with $|V| \geq 3$. Let $u, v \in V$ with

$$d_G(u) + d_G(v) \geq |V|$$

Then G is Hamiltonian if and only if

$$G' = (V, E') \ where \ E' = E \cup \{\{u, v\}\}$$

is Hamiltonian.

Hamilton Paths and Cycles

Proof.

$\Rightarrow$: This direction is immediate because adding an edge will not make a graph not Hamiltonian.

$\Leftarrow$: $G' = (V, E')$ with $E' = E \cup \{\{u, v\}\}$. Let $W : u \xrightarrow{*} u$ be a Hamilton cycle in G'. If W does not traverse the edge $\{u, v\}$, then there is nothing to prove. So, assume that W traverses the edge $\{u, v\}$, and without loss of generality we can assume that $\{u, v\}$ is the last edge traversed by W. I.e. there exists a Hamilton path $W' : u \xrightarrow{*} v$ given by

$$W' : u \to v_1 \to \cdots \to v_k \to v$$

We can find $1 < i \leq k$ such that $\{u, v_i\} \in E$ and $\{v_{i-1}, v\} \in E$, otherwise $d_G(v) < |V| - d_G(u)$. But now

$$u \to v_1 \to \cdots \to v_{i-1} \to v \to v_k \to v_{k-1} \to \cdots \to v_i \to u$$

is a Hamilton cycle of G. $\square$

The Closure of a Graph

Definition

Let $G = (V, E)$ be a finite graph. Define the **closure** *of G, denoted $\mathrm{cl}(G)$, by $\mathrm{cl}(G) = (V, E^+)$ where*

$$E^+ = \{\{u, v\} \in \mathcal{P}_(V) \mid d_G(u) + d_G(v) \geq |V|\}$$

Theorem

Let $G = (V, E)$ be a finite graph with $|V| \geq 3$.

(i) *G is Hamiltonian if and only if $\mathrm{cl}(G)$ is Hamiltonian*

(ii) *If $\mathrm{cl}(G)$ is $K_{|V|}$, then G is Hamiltonian*

Proof.

Firstly note that (ii) follows immediately from (i) and the fact that for all $n \geq 3$, K_n is Hamiltonian. So, we turn to proving (i). Let $\mathrm{cl}(G) = (V, E^+)$. Note that since $\mathrm{cl}(G)$ spans G and $E \subseteq E^+$, it is clear that if G is Hamiltonian, then so is $\mathrm{cl}(G)$.

The Closure of a Graph

Proof.

(Continued.) Suppose that $\mathrm{cl}(G)$ is Hamiltonian. Let $e_1, \ldots, e_n$ be the edges in $E^+ \backslash E$. I.e. The new edges in $\mathrm{cl}(G)$. Define $E_0 = E$ and $G_0 = (V, E_0)$, and for all $1 \leq k \leq n$, $E_{k+1} = E_k \cup \{e_k\}$ and $G_{k+1} = (V, E_{k+1})$. We claim that for all $0 \leq k \leq n$, G_k is Hamiltonian if and only if G_{k+1} is Hamiltonian. However, this follows from Ore's Theorem and the fact that if $e_k = \{u, v\}$, then $d_G(u) + d_G(v) \geq |V|$. And so, $d_{G_k}(u) + d_{G_k}(v) \geq |V|$. Therefore, since $\mathrm{cl}(G) = G_{n+1}$, G is Hamiltonian. This proves the theorem. □

Corollary

Let $G = (V, E)$ be a finite graph with $|V| \geq 3$. If for all $u, v \in V$ with $u \neq v$,

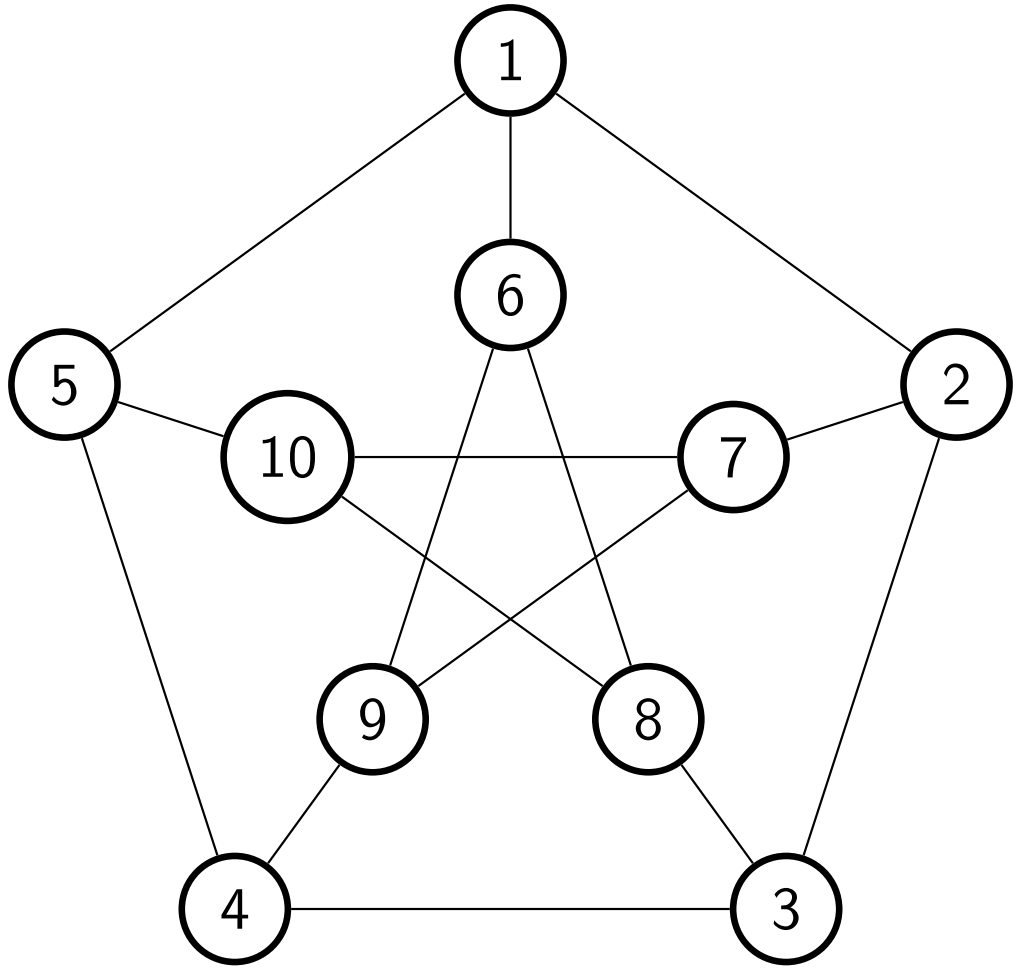
$$d_G(u) + d_G(v) \geq |V|$$

then G is Hamiltonian. In particular, if $\delta(G) \geq \frac{1}{2}|V|$, then G is Hamiltonian.

Proof.

The Petersen Graph

Recall the Petersen graph:



This graph has the property that if there is a path of length 2 between two distinct vertices u and v, then there is no edge between u and v.

The Petersen Graph

Moreover, if there is a path of length 3 between any two distinct vertices u and v, then there is no edge between u and v. In other words, the Petersen graph has no subgraphs that are isomorphic to C_3 or C_4.

Theorem

The Petersen graph is not Hamiltonian

Proof.

Let $G = (V, E)$ be the Petersen graph with $V = \{v_0, \ldots, v_9\}$. Suppose that there exists a Hamilton cycle W. I.e. W is a cycle of length 10. Note that $|E| = 15$, and so there must be 5 edges that do not appear in W. No two of these additional 5 edges can meet at the same vertex, because the Petersen graph is 3-regular.

The Petersen Graph

Proof.
(Continued.) Think of these 5 edges as chords through a cycle of order 10. None of these chords can join two vertices that are distance $1, 2$ or 3 apart on the 10 cycle, otherwise we would have a copy of C_4. Not all of these chords can be join to an opposite vertices, otherwise we would get a copy of C_4. This means there is at least one chord that joins two vertices that are 4 apart on the 10 cycle. Convince yourself that when this happens there is no way of drawing the other chords that doesn't create a copy of C_4. $\square$

Colourings

Definition

Let $G = (V, E)$ be a graph and let A be a set. A function $f : V \longrightarrow A$ is called a **vertex colouring** *of G. A function $f : E \longrightarrow A$ is called an* **edge colouring** *of G. If $f : E \longrightarrow A$ is an edge colouring and $A \subseteq \mathbb{R}$ then f is also called a* **weight function** *or a* **distance function**.

Definition

Let $G = (V, E)$ be a connected graph equipped with a weight function $\alpha : E \longrightarrow \mathbb{R}$. For all walks $W : [k+1] \longrightarrow V$, define

$$\alpha(W) = \sum_{i=0}^{k-1} \alpha(\{W(i), W(i+1)\})$$

For all $u, v \in V$, define

$$d_G^{\alpha}(u, v) = \min\{\alpha(W) \mid W : u \xrightarrow{*} v\}$$

Shortest Path Problem

Shortest Path Problem: Let $G = (V, E)$ be a connected graph equipped with a weight function $\alpha : E \longrightarrow \mathbb{N}$ and let $u, v \in G$. What is the value of $d_G^{\alpha}(u, v)$?

We will now present an algorithm, due to Edsger Dijkstra, that solves this problem. Let $G = (V, E)$ be a connected graph equipped with a weight function $\alpha : E \longrightarrow \mathbb{N}$, and let $a \in V$. The algorithm proceeds by defining two sequences $S_0 \subseteq \ldots \subseteq S_n$ and $L_0, \ldots, L_n$, such that for all $0 \leq i \leq n$, $S_i \subseteq V$ and $L_i : V \longrightarrow \mathbb{N} \cup \{\infty\}$. Let $S_0 = \emptyset$, and define L_0 by: $L_0(a) = 0$ and for all $v \in V$ with $v \neq a$, $L_0(v) = \infty$.
At the $(k+1)^{\text{th}}$ stage of the algorithm define S_{k+1} by adding the vertex u_k with smallest value according to L_k. Define L_{k+1} by updating the values of L_k for vertices that are not in S_{k+1}: If $v \notin S_{k+1}$, define

$$L_{k+1} = \begin{cases} L_k(v) & \text{if } \{u_k, v\} \notin E \\ \min(L_k(v), L_k(u_k) + \alpha(\{u, v\})) & \text{if } \{u_k, v\} \in E \end{cases}$$

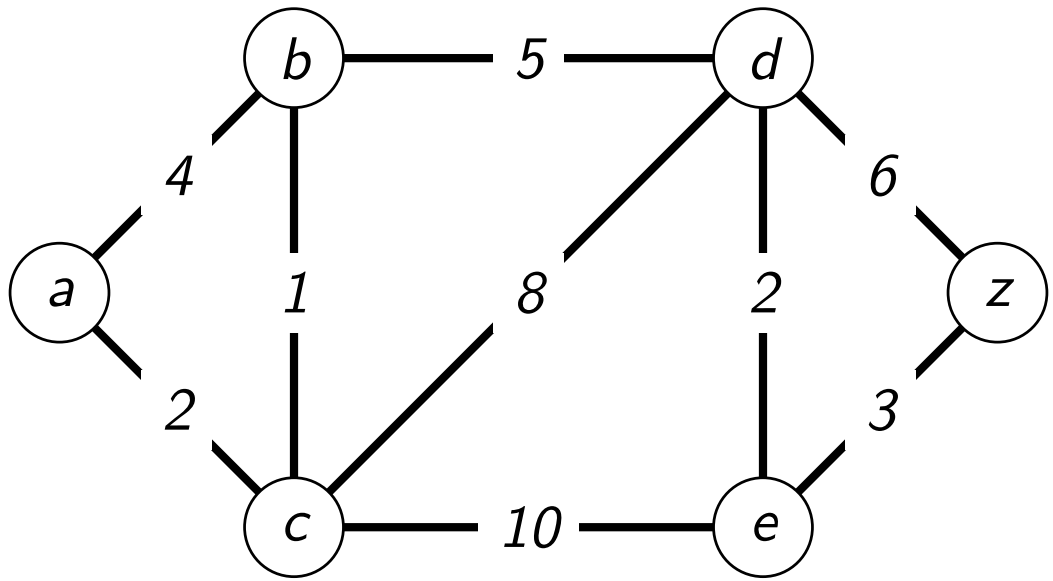
The algorithm proceeds until $S_{k+1} = V$.

Dijkstra's Algorithm

The idea behind this algorithm is that at each stage k, the function L_k yields the shortest path between a and any vertex that only passes through the vertices in S_k.

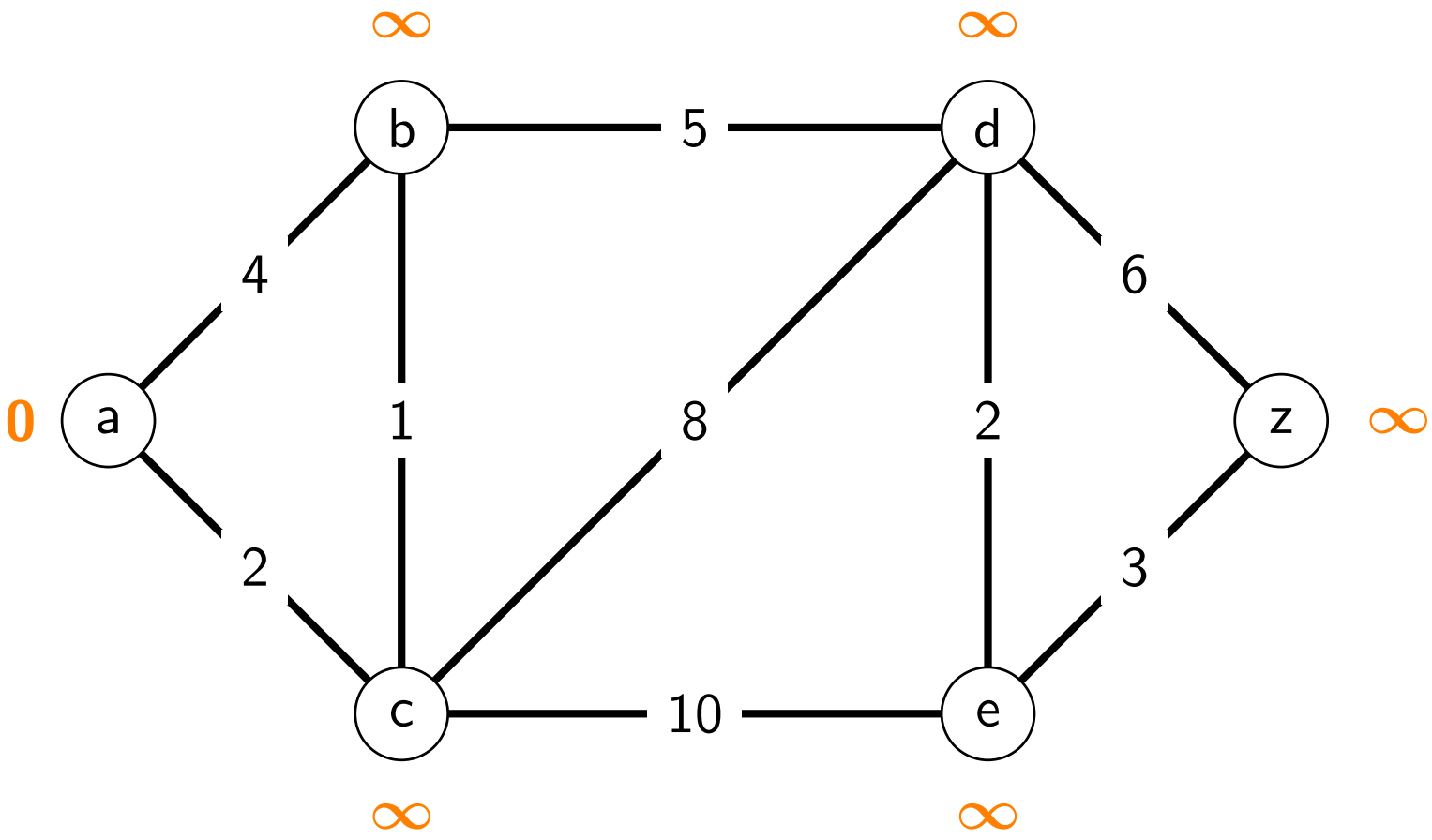
Example

We will find the shortest path from a to z in the graph given below:



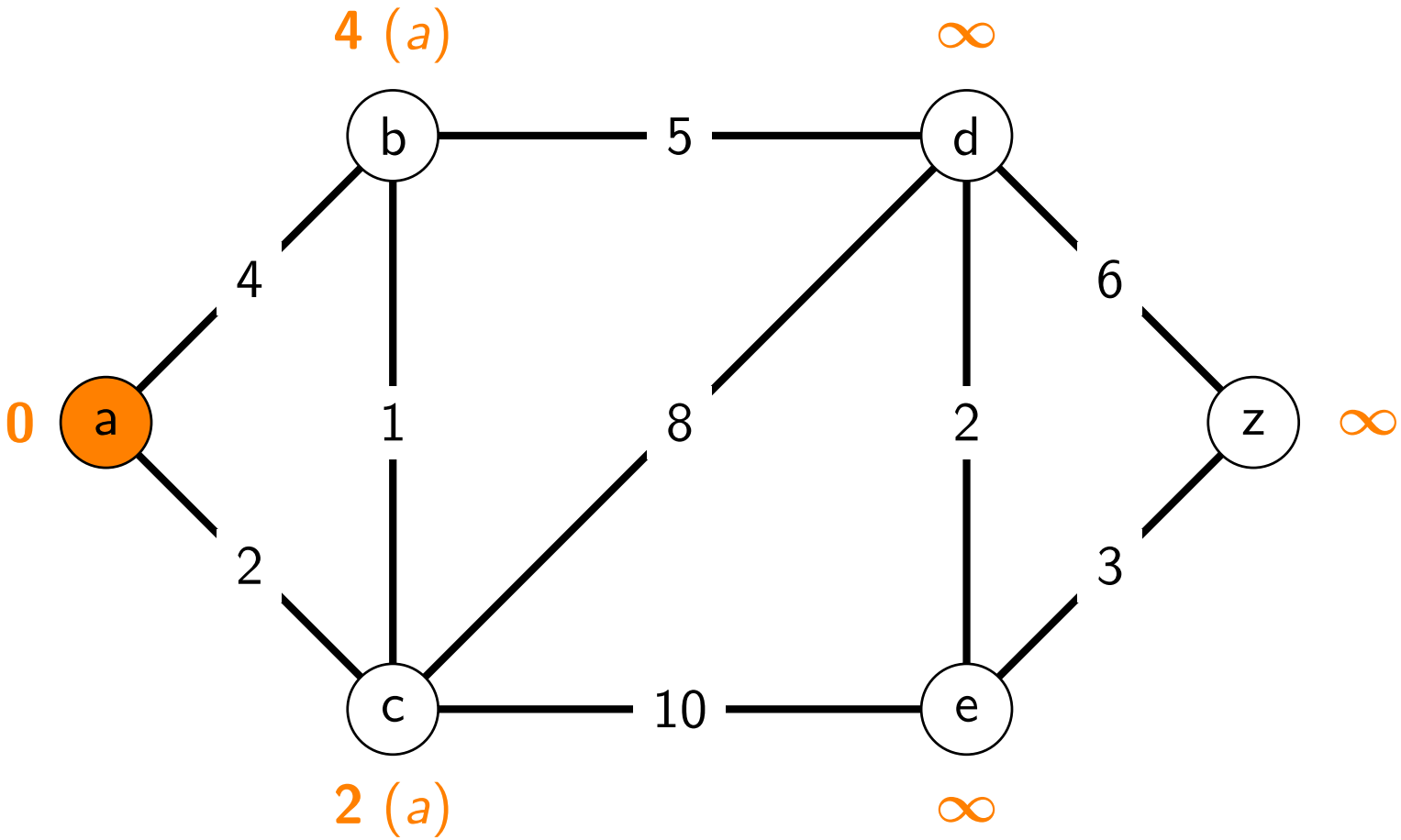
Dijkstra's Algorithm - 0^{th} Iteration

$S_0 = \emptyset$.



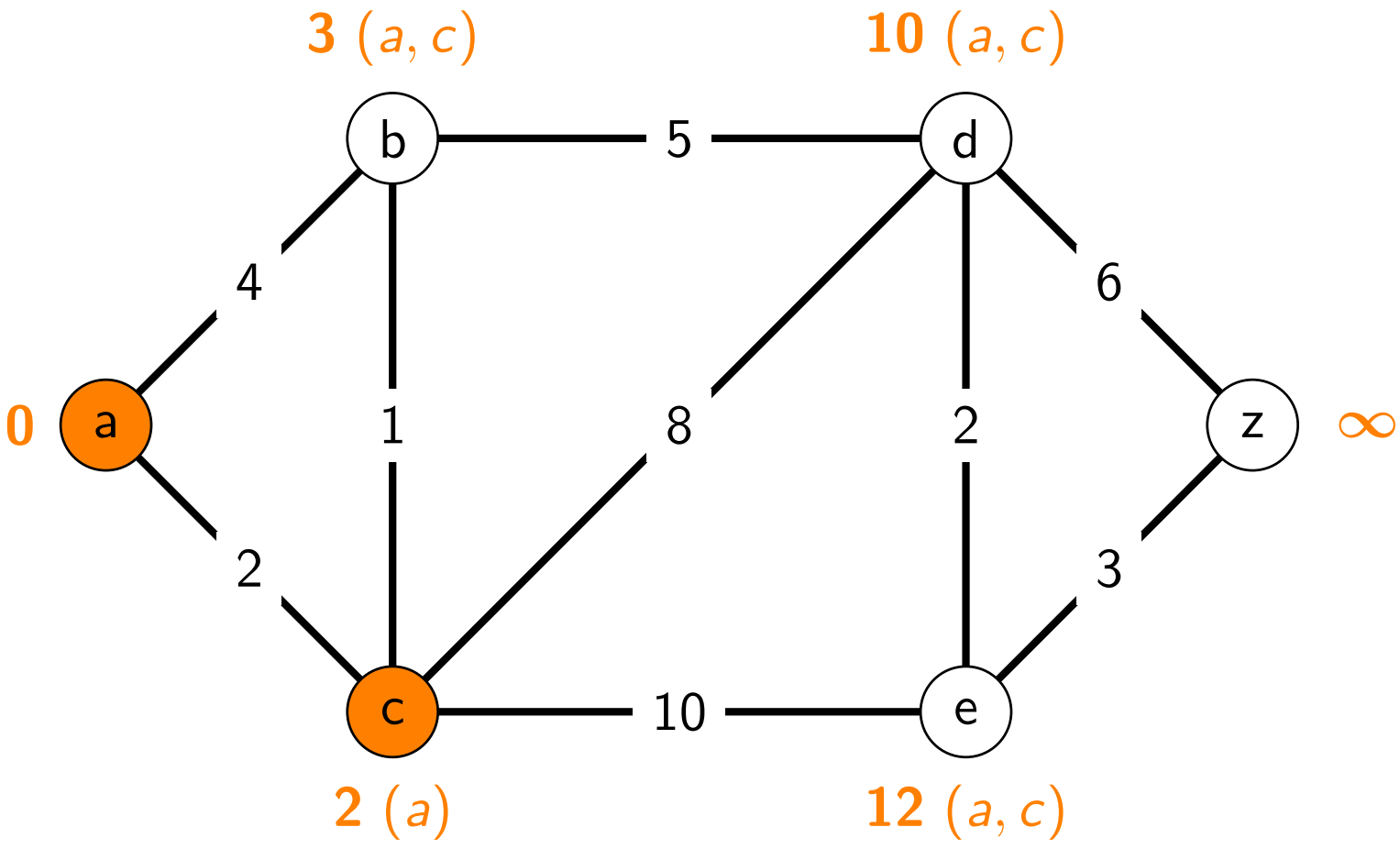
Dijkstra's Algorithm - 1st Iteration

$S_1 = \{a\}$.



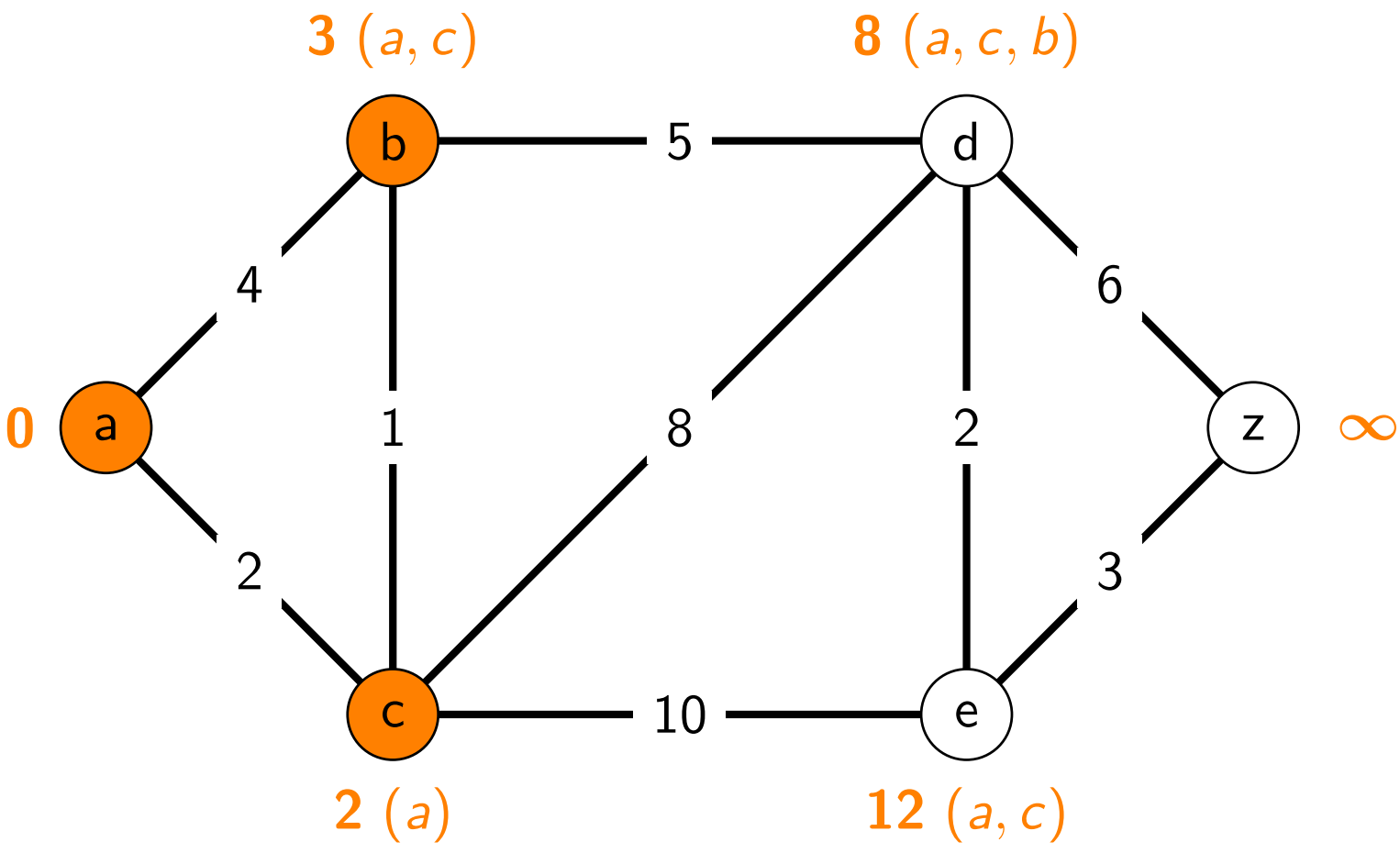
Dijkstra's Algorithm - 2nd Iteration

$S_2 = \{a, c\}$.



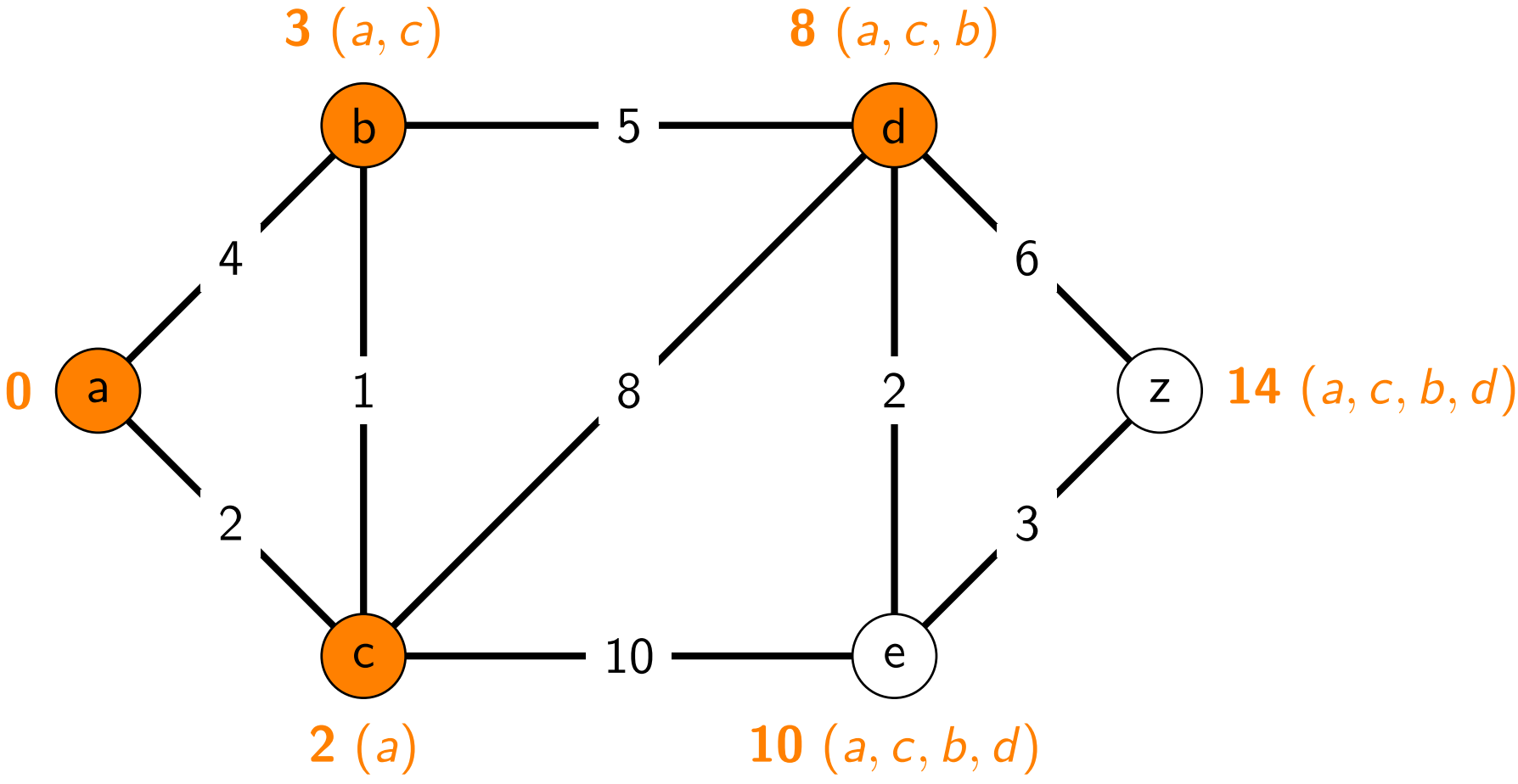
Dijkstra's Algorithm - 3$^{\text{th}}$ Iteration

$S_3 = \{a, b, c\}$.



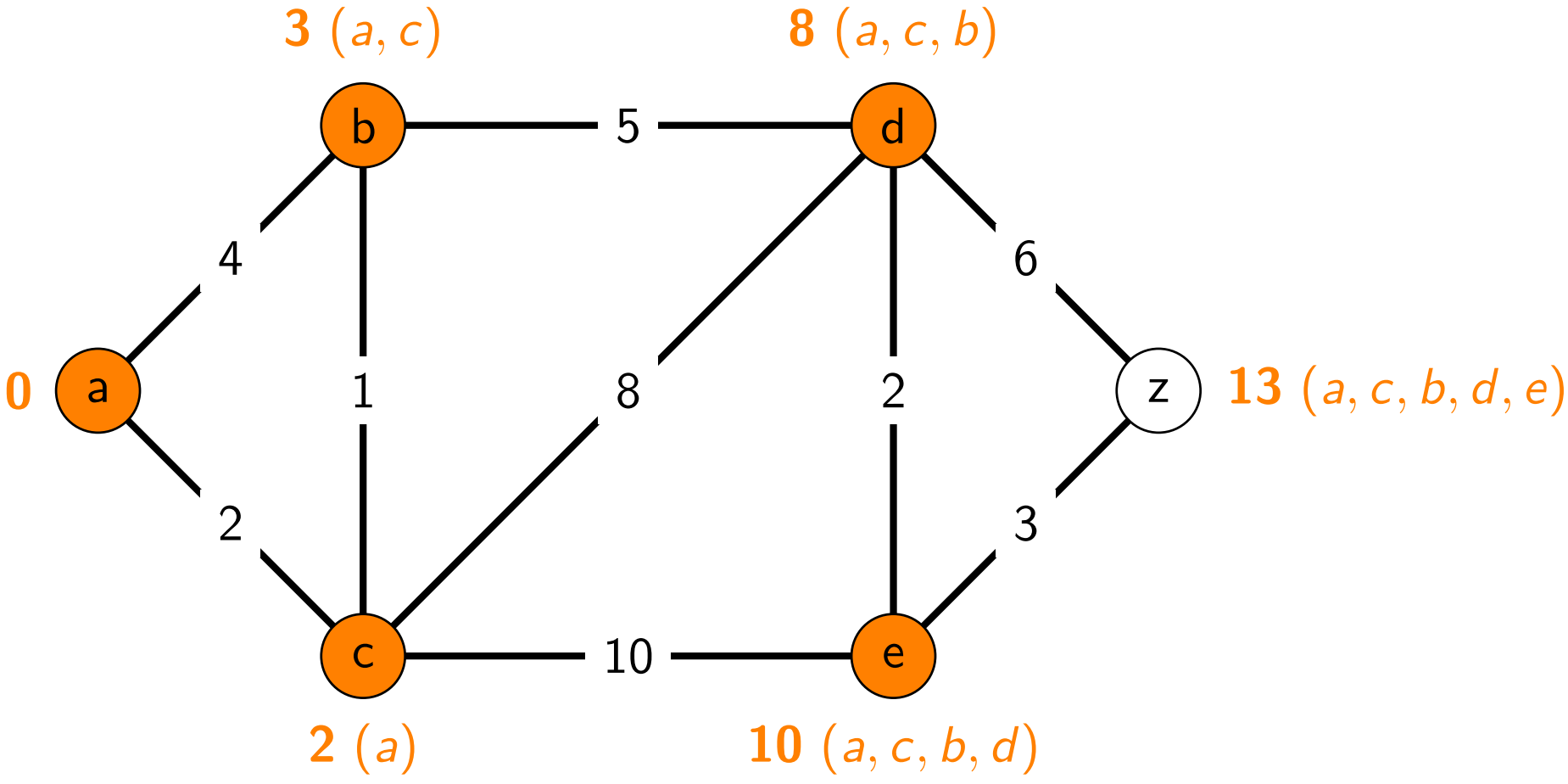
Dijkstra's Algorithm - 4^{th} Iteration

$S_4 = \{a, b, c, d\}$.



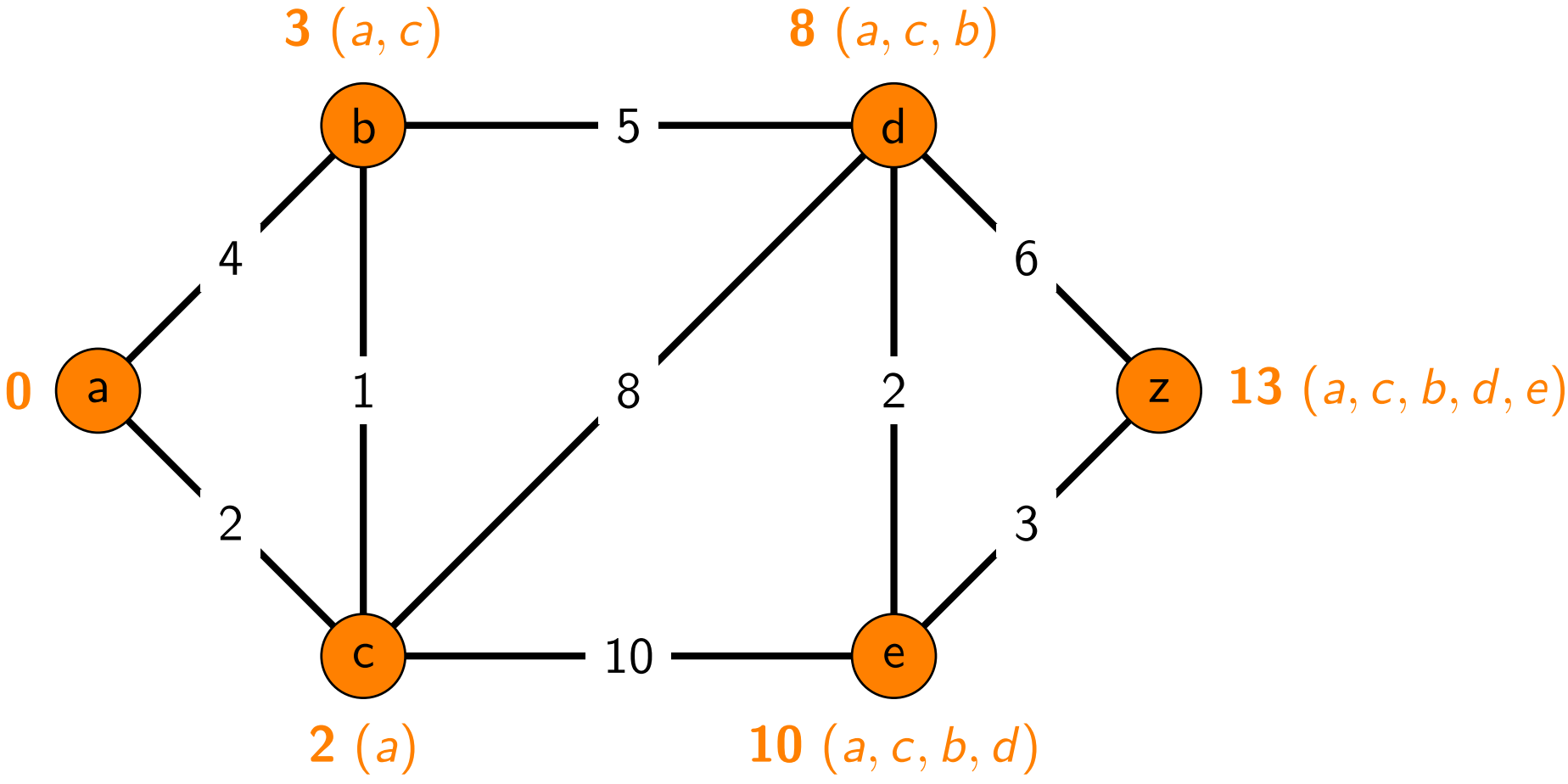
Dijkstra's Algorithm - 5^{th} Iteration

$S_5 = \{a, b, c, d, e\}$.



Dijkstra's Algorithm - 6^{th} Iteration

$S_5 = \{a, b, c, d, e, z\}$.



Dijkstra's Algorithm

To establish that Dijkstra's algorithm actually gives shortest path from a to z, we will prove the following result:

Theorem

Let $G = (V, E)$ be a connected graph equipped with a weight function $\alpha : E \longrightarrow \mathbb{N}$, and let $a \in V$. Let $\emptyset = S_0 \subseteq \cdots \subseteq S_n = V$ and $L_0, \ldots L_n$ be obtained by Dijkstra's Algorithm. Then for all $0 \leq k \leq n$,

(i) *For all $v \in S_k$, $L_k(v)$ is the shortest path for a to v*

(ii) *For all $v \notin S_k$, $L_k(v)$ is the shortest path for a to v that passes through vertices (besides v itself) that are in S_k*

Proof.

We prove these statements by induction in $k \in \mathbb{N}$. For $k = 0$, $S_0 = \emptyset$, so (i) is vacuously true. Furthermore, there is no path from a to a vertex other than a using vertices in S_0, so, by definition, its length is ∞. It follows that (i) and (ii) are true for $k = 0$.

Dijkstra's Algorithm

Proof.

(Continued.) Assume that (i) and (ii) are true for $k \in \mathbb{N}$. The vertices in S_k are hence labeled with the length of the shortest path from a by L_k. Let $u \in S_{k+1}$ but $u \notin S_k$, so u is the vertex added in the kth iteration. Thus, u yields the smallest value of L_k of all vertices not in S_k.

It is clear (from (ii)) that u is labeled by L_k with the length of a shortest path containing only vertices in S_k. We claim that $L_k(u)$ is actually the length of a shortest path without regard to where the path's vertices lie. We show this by contradiction: Suppose that there exists a path of length less than $L_k(u)$ from a to u containing a vertex not in S_k. Let v be the first vertex along this path which is not in S_k. Then $L_k(v) < L_k(u)$ so v should have been added to S_k instead of u, giving a contradiction. Hence the label of u is the shortest path from a to u. This implies (i) with k replaced by $k + 1$.

Dijkstra's Algorithm

Proof.

(Continued.) Let $v \notin S_{k+1}$. A shortest path from a to v containing only elements of S_{k+1} either contains u or it does not. If it does not contain u, then $L_{k+1}(v) = L_k(v)$ is the length of the shortest path from a to v since we assumed that (ii) is true for k. If it does contain u, then by the induction hypothesis, the path is made up of a shortest path from a to u followed by the edge from u to v. But then the length of this path is just given by the value assigned to $L_{k+1}(v)$ by Dijkstra's Algorithm. Thus, (ii) is true with k replaced by $k + 1$. □

Corollary

Dijktsra's algorithm finds the length of a shortest path between two vertices in a connected weighted graph.

Theorem

Dijkstra's algorithm uses $O(n^2)$ operations (additions and comparisons) to find the length of a shortest path between two vertices in a connected weighted graph consisting of n vertices.

Complete p-partite Graphs

Ramsey Theory is a rich and active branch of combinatorics that studies unavoidable patterns that appear when structures, such a graphs, are large enough.

Definition

Let V be a set and let $p \in \mathbb{N}$. Let $f : V \longrightarrow [p]$ (think of f as a partition of V). The **complete p-partite graph** *defined by f is the graph $G = (V, E)$ where*

$$E = \{\{u, v\} \in \mathcal{P}_2(V) \mid f(u) \neq f(v)\}$$

Let $p, n \in \mathbb{N}$ with $p \geq 3$. My goal is to build a complete $(p-1)$-partite graph of order n that does not contain an isomorphic copy of K_p. Using the division algorithm we can find $t, r \in \mathbb{N}$ with $r < p - 1$ such that

$$n = t(p-1) + r$$

Complete p-partite Graphs

Let $V = [n]$. Therefore V can be partitioned into sets $X_1, \ldots, X_{p-1}$ such that

(i) for all $1 \leq q \leq r$, $|X_q| = t + 1$

(ii) for all $r + 1 \leq q \leq p - 1$, $|X_q| = t$

Define $f : V \longrightarrow [p-1]$ by: for all $x \in V$ and for all $q \in [p-1]$,

$$f(x) = q \text{ if and only if } x \in X_q$$

Let $G = (V, E)$ be the complete $(p-1)$-partite defined by f. It is clear that G does not have any subgraph that is isomorphic to K_p. Let $T(n, p)$ be the number of edges in this complete $(p-1)$-partite graph. So,

$$T(n,p) = \frac{p-2}{2(p-1)} n^2 - \frac{r}{2}\left(1 - \frac{r}{p-1}\right)$$

Turán's Theorem

Theorem

(Turán 1941) Let $n, p \in \mathbb{N}$ with $p \geq 3$. Let $G = (V, E)$ be a graph of order n. If $|E| > T(n, p)$, then G has a subgraph that is isomorphic to K_p.

Proof.

Let $t, r \in \mathbb{N}$ be such that

$$n = t(p-1) + r$$

We prove the theorem by induction on t. If $t = 0$, then

$$T(n, p) = \binom{n}{2}$$

which is the size of K_n and so there is nothing to prove. Suppose that $t \geq 1$ and the result holds for graphs with order that has quotient $t-1$ when divided by $p-1$. Let $G = (V, E)$ be a graph of maximal size such that G has no subgraph which is isomorphoric to K_p.

Turán's Theorem

Proof.

(Continued.) Now, G must have a subgraph that is isomorphic to K_{p-1}, because adding one edge to G yields a graph that is isomorphic to K_p. Therefore, there exists $A \subseteq V$ with $|A| = p - 1$ such that $G[A] \cong K_{p-1}$. For all $v \in V \backslash A$, there is at most $p - 2$ $u \in A$ such that $\{u, v\} \in E$, otherwise $G[A \cup \{v\}]$ would be isomorphic to K_p. Moreover, $G - A$ does not contain a subgraph isomorphic to K_p, and the order of $G - A$ is

$$(t - 1)(p - 1) + r$$

Therefore, by the induction hypothesis, the size of $G - A$ is at most $T(n - p + 1, p)$. So, accounting for the $\binom{p-1}{2}$ edges is $G[A]$ and the $(n - p + 1)(p - 2)$ edges between $G[A]$ and $G - A$, and doing some calculations, yields

$$|E| \leq T(n - p + 1, p) + \binom{p - 1}{2} + (n - p + 1)(p - 2) = T(n, p)$$

The result now follows by induction. □