Contents

1 Review of Propositional Logic

Task: Recall enough propositional logic to see how it matches up with set theory.

Definition: A proposition is any declarative sentence that is either true or false.

1.1 Connectives

		Connectives	Notation in Maths
and	$\wedge$		
or	$\vee$	"Inclusive or"	
not	$\neg$	Sometimes denoted $\sim$	
implies	$\rightarrow$	if/then; called implication	$\Rightarrow$
if and only if	$\leftrightarrow$	Called equivalence	$\Leftrightarrow$

1.1.1 Truth Table of the Connectives

Let P, Q be propositions:

P	Q	P∧Q
F	F	F
F	T	F
T	F	F
T	T	T

P	Q	P∨Q
F	F	F
F	T	T
T	F	T
T	T	T

P	¬P
F	T
T	F

P	Q	P→Q
F	F	T
F	T	T
T	F	F
T	T	T

P	Q	P↔Q
F	F	T
F	T	F
T	F	F
T	T	T

Priority of the Connectives

Highest to Lowest: $\neg, \wedge, \vee, \rightarrow, \leftrightarrow$

1.2 Important Tautologies

$$\begin{array}{rcl}
(P \rightarrow Q) & \leftrightarrow & (\neg P \vee Q) \\
(P \leftrightarrow Q) & \leftrightarrow & [(P \rightarrow Q) \wedge (Q \rightarrow P)] \\
\neg(P \wedge Q) & \leftrightarrow & (\neg P \vee \neg Q) \\
\neg(P \vee Q) & \leftrightarrow & (\neg P \wedge \neg Q)
\end{array} \quad \left.\begin{array}{l} \\ \\ \end{array}\right\} \text{De Morgan Laws}$$

As a result, $\neg$ and $\vee$ together can be used to represent all of $\neg, \wedge, \vee, \rightarrow, \leftrightarrow$.

Less obvious: One connective called the sheffer stroke P|Q (which stands for "not both P and Q" or "P nand Q") can be used to represent all of $\neg$, $\wedge$, $\vee$, $\rightarrow$, $\leftrightarrow$ since $\neg$P $\leftrightarrow$ P|P and P $\vee$ Q $\leftrightarrow$ (P|P) | (Q|Q).

Recall if P→Q is a given implication, Q→P is called the converse or P→Q. $\neg$Q → $\neg$P.

1.3 Indirect Arguments/Proofs by Contradiction/Reductio as absurdum

Based on the tautology (P→Q) ↔ (¬Q → ¬P)

Example: Famous argument that $\sqrt{2}$ is irrational.

Proof:

Suppose $\sqrt{2}$ is rational, then it can be expressed is fraction form $\frac{a}{b}$. Let us **assume** that our fraction is in the lowest term, **i.e.** their only common divisor is 1.

Then,

$$\sqrt{2} = \frac{a}{b}$$

Squaring both sides, we have

$$2 = \frac{a^2}{b^2}$$

Multiplying both sides by B^2 yields

$$2b^2 = a^2$$

Since a^2b^2, we can conclude that a^2 is even because whatever the value of b^2 has to be multiplied by 2. If a^2 is even, then a is also even. Since a is even, no matter what the value of a is, we can always find an integer that if we divide a by 2, it is equal to that integer. If we let that integer be k, then $\frac{a}{b} = k$ which means that $a = 2k$.

Substituting the value of $2k$ to a, we have $2b^2 = (2k)^2$ which means that $2b^2 = 4k^2$. dividing both sides by 2 we have $b^2 = 2k^2$. That means that the value b^2 is even, since whatever the value of k you have to multiply it by 2. Again, is b^2 is even, then b is even.

This implies that both a and b are even, which means that both the numerator and the denominator of our fraction are divisible by 2. This contradicts our **assumption** that $\frac{a}{b}$ has no common divisor except 1. Since we found a contradiction, our assumption is, therefore, false. Hence the theorem is true.

qed

2 Predicate logic and Quantifiers

Task: Understand enough predicate logic to make sense of quantified statements.

In predicate logic, propositions depend on variable x, y, z, so their truth value may change depending on which values these variables assume: $P(x), Q(x, y), R(x, y, z)$

2.1 Introduce quantifiers

2.1.1 $\exists$ existential quantifier

Syntax: $\exists x P(x)$

Definition: $\exists x P(x)$ is true if $P(x)$ is true or some value of x; it is false otherwise.

2.1.2 $\forall$ universal quantifier

Syntax: $\forall x P(x)$

Definition: $\forall x P(x)$ is true if $P(x)$ is true for all allowable values of x. It is false otherwise.

2.1.3 $\exists!$ for one and only one

Syntax: $\exists! x P(x)$

Definition: $\exists! x P(x)$ is true if $P(x)$ is true for exactly one value of x and false for all often values of x; otherwise, $\exists! x P(x)$ is false.

2.2 Alternation of Quantifiers

$\forall x \exists y \forall z \quad P(x, y, z)$
NB: The order <u>cannot</u> be exchanged as it might modify the truth values of the statement (think of examples with two quantifiers).

2.3 Negation of Quantifiers

$$\neg(\exists x P(x)) \quad \leftrightarrow \quad \forall x \neg P(x)$$
$$\neg(\forall x P(x)) \quad \leftrightarrow \quad \exists x \neg P(x)$$

3 Set Theory

Task: Understand enough set theory to make sense of other mathematical objects in abstract algebra, graph theory, etc. Set theory started around 1870's $\rightarrow$ late development in mathematics but now taught early in one's maths education due to Bourbaki school.

Definition: A set is a collection of objects. $x \in A$ means the element X is in the set A (**i.e.** belongs to A).

Examples:

1. All students in a class.
2. $\mathbb{N}$ the set of natural numbers starting at 0.

   $\mathbb{N}$ is defined via the following two axioms:

(a) $0 \in \mathbb{N}$

(b) if $x \in \mathbb{N}$ then $x+1 \in \mathbb{N}$ $(x \in \mathbb{N} \to X + A \in \mathbb{N})$

3. $\mathbb{R}$ set of real numbers also introduced axiomatically

   $\mathbb{R}$ the set of real numbers.

   (a) Additive closure: $\forall x, y \exists z (x + y = z)$
   (b) Multiplicative closure: $\forall x, y, \exists z (x \times y = z)$
   (c) Additive associativity: $x + (y + z) = (x + y) + z$
   (d) Multiplicative associativity: $x \times (y \times z) = (x \times y) \times z$
   (e) Additive commutativity: $x + y = y + x$
   (f) Multiplicative commutativity: $x \times y = y \times x$
   (g) Distributivity: $x \times (y + z) = (x \times y) + (x \times z)$ and $(y + z) \times x = (y \times x) + (z \times x)$
   (h) Additive identity: There is a number, denoted 0, such that or all $x, x + 0 = x$
   (i) Multiplicative identity: There is a number, denoted 1, such that for all $x, x \times 1 = 1 \times x = x$
   (j) Additive inverses: For every x there is a number, denoted $-x$, such that $x + (-x) = 0$
   (k) Multiplicative inverses: For every nonzero x there is a number, denoted x, such that $x \times x^{-1} = x^{-1} \times x = 1$
   (l) $0 \neq 1$
   (m) Irreflexivity of $<$:$\sim (x < x)$
   (n) Transitivity of ¡: If $x < y$ and $y < z$, then $x < z$
   (o) Trichotomy: Either $x < y, y < x$, or $x = y$
   (p) If $x < y$, then $x + y < y + z$
   (q) If $x < y$ and $0 < z$, then $x \times z < y \times z$ and $z \times x < z \times y$
   (r) Completeness: If a nonempty set of real numbers has an upper bound, then it has a *least* upper bound.

4. $\emptyset$ is the empty set (The set with no elements).

Definition: Let A, B be sets. A=b if and only if all elements of A are elements of B and all elements of B are elements of A,
i.e. $A = b \leftrightarrow [\forall x (x \in A \to x \in B)] \cap [\forall y (y \in B \to y \in A)]$

3.1 Two Ways to Describe Sets

1. The enumeration/roster method: list all elements of the set.
   NB: order is irrelevant.
   $A = \{0, 1, 2, 3, 4, 5\} = \{5, 0, 2, 3, 1, 4\}$

2. The formulaic/set builder method: give a formula that generates all elements of the set.
   $A = \{x \in \mathbb{N} \mid 0 \leq x \wedge x \leq 5\} = \{0, 1, 2, 3, 4, 5\} = \{x \in \mathbb{N} : 0 \leq x \wedge x \leq 5\}$

Using $\mathbb{N}$ and the set-builder method, we can define:

$\mathbb{Z} = \{m - n \mid \forall m, n \in \mathbb{N}\}$

$n = 0$ in any natural numbers $\Rightarrow$ we generate all of $\mathbb{N}$

$m = 0$ in any natural number $\Rightarrow$ we generator all negative integers

$\mathbb{Q} = \{\frac{p}{q} \mid p, q \in \mathbb{Z} \wedge q \neq 0\}$

Definition: A set A is called finite if it has a finite number of elements; otherwise it is called infinite.

4 Set Operations

Task: Understand how to represent sets by Venn diagrams. Understand set union, intersection, complement and difference.

Definition: Let A, B be sets. A is a <u>subset</u> of B. If all elements of A are elements of B, **i.e.** $\forall x(x \in A \to x \in B)$. We denote that A is a subset of B by $A \subseteq B$

Example: $\mathbb{N} \subseteq \mathbb{Z}$

Definition: Let A, B be sets. A is a <u>proper</u> subset of B if $A \subseteq B \wedge A \neq B$, **i.e.** $A \subseteq B \wedge \exists x \in B s.t. x \notin A$.
A proper subset is always a subset, but a subset is not always a proper subset.

Notation: $A \subset B$

Example: $\mathbb{N} \subset \mathbb{Z}$ since $\exists -1 \in \mathbb{N}$
NB: $\forall A$ a set $\emptyset \subseteq A$

Recall: $B \subseteq C$ means $\forall x(x \in B \to x \in C)$, but $\emptyset$ has no elements so in $\emptyset \subseteq A$ the quantifier $\forall$ operates on a domain with no elements. Clearly, we need to give meaning to $\exists$ and $\forall$ on empty sets.

<u>Boolean Convention</u>

$\left.\begin{array}{l}\forall \text{ is true on the empty set}\\ \exists \text{ is false on the empty set}\end{array}\right\}$ Consistent with common sense

Definition: Let A, B be two sets. The <u>union</u> $A \cup B = \{x \mid x \in A \vee x \in B\}$

Definition: Let A, B be two sets. The <u>intersection</u> $A \cap B = \{x \mid x \in A \wedge x \in B\}$

Definition: Let A, B be sets. A and B are called <u>disjoint</u> is $A \cap B = \emptyset$

Definition Let A, B be two sets. $A - B = A \backslash B = \{a \mid x \in A \wedge x \notin B\}$

Examples:

$A = \{1, 2, 5\}$ $\quad$ $B = \{1, 3, 6\}$

$A \cup B = \{1, 2, 3, 5, 6\}$ $\quad$ $A \cap B = \{1\}$

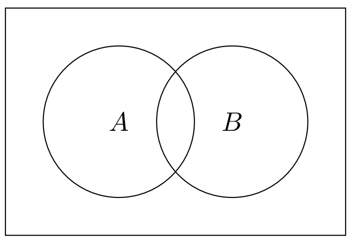
$A \backslash B = \{2, 5\}$ $\quad$ $B \backslash A = \{3, 6\}$

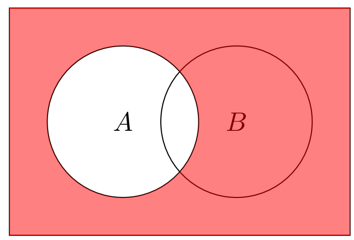
Definition: Let A, U be sets s.t. $A \subseteq U$. The complement of A in U = $U \backslash A = A^C = \{x \mid x \in U \wedge x \notin A\}$

Remark: The notation A^C is unambiguous only if the universe U is clearly defined or understood.

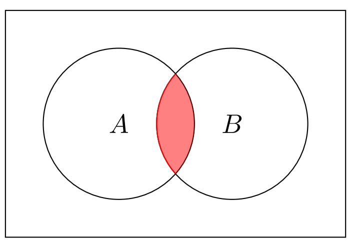
4.1 Venn Diagrams

Schematic representation of set operations.

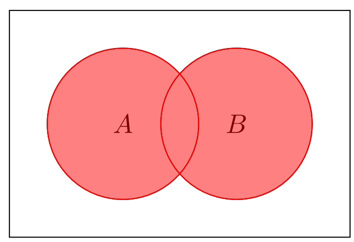




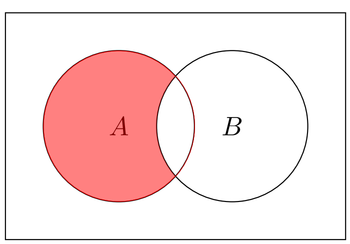
A^C



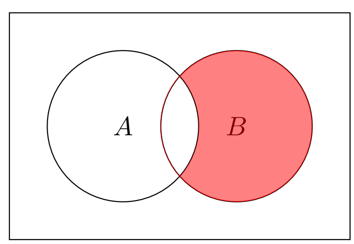
$A \cap B$



$A \cup B$



$A \backslash B$



$B \backslash A$

4.2 Properties of Set Operations

Correspondence between Logic and Set Theory

Logical Connective	Set operation
$\wedge$	intersection $\cap$
$\vee$	union $\cup$
$\neg$	complement $(\ )^C$

As a result, various properties of set operations become obvious:

- Commutativity
  - $A \cap B = B \cap A$
  - $A \cup B = B \cup A$
- Associativity
  - $(A \cup B) \cup C = A \cup (B \cup C)$
  - $(A \cap B) \cap C = A \cap (B \cap C)$
- Distributivity
  - $A \cap (B \cup C) = (A \cap B) \cup (A \cap C)$
  - $A \cup (B \cap C) = (A \cup B) \cap (A \cup B)$
- De Morgan Laws in Set Theory
  - $(A \cap B)^C = A^C \cup B^C$
  - $(A \cup B)^C = A^C \cap B^C$
- Involutivity of the Complement
  - $(A^C)^C = A$

NB: An involution is a map such that applying it twice gives the identity. Familiar examples: reflecting across the x-axis, the y-axis, or the origin in the plane.

- Transitivity of Inclusion
  - $A \subseteq B \wedge B \subseteq C \rightarrow A \subseteq C$
- Criterion for proving equality of sets
  - $A = B \leftrightarrow A \subseteq C \wedge B \subseteq A$
- Criterion for proving non-equality of sets
  - $A \neq B \leftrightarrow (A \backslash B) \cup (B \backslash A) \neq 0$

4.3 Example Proof in Set Theory

Proposition: $\forall A, B$ sets. $(A \cap B) \cup (A \backslash B) = A$

Proof: Use the criterion for proving equality of sets from above, **i.e.** inclusion in both directions.

Show $(A \cap B) \cup (A \backslash B) \subseteq A$: $\forall x \in (A \cap B) \cup (A \backslash B), x \in (A \cap B)$ or $x \in A \backslash B$. If $x \in (A \cap B)$ then clearly $x \in A$ as $A \cap B \subseteq A$ by definition. If $x \in A \backslash B$, then by definition $x \in A$ and $x \notin B$ so definitely $x \in A$. In both cases, $x \in A$ as needed.

Show $A \subseteq (A \cap B) \cup (A \backslash B)$: $\forall x \in A$, we have two possibilities, namely $x \in B$ or $x \notin B$. If $x \in B$, then $x \in A$ and $x \in B$, so $x \in A \cap B$. If $x \notin B$, then $x \in A$ and $x \notin B$, so $x \in A \backslash B$. In both cases, $x \in (A \cap B)$ or $x \in (A \backslash B)$ so $x \in (A \cap B) \cup (A \backslash B)$ as needed.

qed

5 The Power Set

Task: Understand what the power set of a set A is.

Definition: Let A be a set. The power set of A denoted $P(A)$ is the collection of all the subsets of A.

Recall: $\emptyset \subseteq A$. It is also clear from the definition of a subset that $A \subseteq A$.

Examples:

1. $A = \{0, 1\}$
   $P(A) = \{\emptyset, \{0\}, \{1\}, \{0, 1\}\}$
2. $A = \{a, b, c\}$
   $P(A) = \{\emptyset, \{a\}, \{b\}, \{c\}, \{a, b\}, \{a, c\}, \{b, c\}, \{a, b, c\}\}$
3. $A = \emptyset$
   $P(A) = \{\emptyset\}$
   $P(P(A)) = \{\emptyset, \{\emptyset\}\}$

NB: $\emptyset$ and $\{\emptyset\}$ are different objects. $\emptyset$ has no elements, whereas $\{\emptyset\}$ has one element.

Remark: $P(A)$ and A are viewed as living in separate worlds to avoid phenomena like Russell' paradox.

Q: If A has n elements, how many elements does $P(A)$ have?

A: 2^n

Theorem: Let A be a set with n elements, then $P(A)$ contains 2^n elements.

Proof: Based on the on/off switch idea.

$\forall x \in A$, we have two choices: either we include x in the subset or we don't (on vs off switch). A has n elements $\Rightarrow$ we have 2^n subsets of A.

qed

Alternate Proof: Using mathematical induction.

NB: It is an axiom of set theory (in the ZFC standard system) that every set has a power set, which implies no set consisting of all possible sets could limit, else what would its power set be?

6 Cartesian Products

Task: Understand sets like $\mathbb{R}^1$ in a more theoretical way.

Recall from Calculus:

$$\mathbb{R} = \mathbb{R}^1 \ni x$$

$$\mathbb{R} \times \mathbb{R} = \mathbb{R}^2 \ni (x_1, x_1)$$

$$\vdots$$

$$\underbrace{\mathbb{R} \times \mathbb{R}}_{\text{n times}} = \mathbb{R}^n \ni (x_1, x_2, \ldots, x_n)$$

These are examples of Cartesian products.

Definition: Let A, B be sets. The Cartesian product denoted by $A \times B$ consists of all ordered pairs (x, y) $s.t.$ $x \in A \wedge y \in B$, **i.e.** $A \times B = \{(x, y) \mid x \in A \wedge y \in B\}$

Further Examples:

1. $A = \{1, 3, 7\}$
   $B = \{1, 5\}$
   $A \times B = \{(1, 1), (1, 5), (3, 1), (3, 5), (7, 1), (7, 5)\}$
   NB: The order in which elements in a pair matters: $(7, 1)$ is different from $(1, 7)$. This is why we call (x, y) an ordered pair.
2. $A = \{(x, y) \in \mathbb{R}^2 \mid x^2 + y^2 = 1\} \leftarrow$ circle of radius 1
   $B = \{z \in \mathbb{R} \mid -2 \leq z \leq 2\} = \{-2, 2\} \leftarrow$ closed interval
   $A \times B \leftarrow$ cylinder of radius 1 and height 4

6.1 Cardinality (number of elements) in a Cartesian product

If A has n elements and B has p elements, $A \times B$ has np elements.

Example:

1. $\#(A) = 3 \qquad A = \{1, 3, 7\}$
   $\#(B) = 2 \qquad B = \{1, 5\}$
   $\#(A \times B) = 3 \times 2 = 6$
2. Both A and B are infinite sets, so $A \times B$ is infinite as well.

Remark: We can define Cartesian products of any length, **e.g.** $A \times A \times B \times A$, $B \times A \times B \times A \times B$, etc. If all sets are finite, the number of elements is the product of the numbers of elements of each factor. If $\#(A) = 3$ and $\#(B) = 2$ as above, $\#(A \times B \times A) = 3 \times 3 \times 3 = 27$ and $\#(B \times A \times B) = 2 \times 3 \times 2 = 12$.

7 Relations

Task: Define subsets of Cartesian products with certain properties. Understand the predicates " = " (equality) and other predicates in predicate logic in a more abstract light.

Start with $x = y$. The elements x is some notation R to y (equality in this case). We can also denote it as xRy or $(x, y) \in E$

Let x, y in $\mathbb{R}$, then $E = \{(x, x) \mid x \in \mathbb{R}\} \subset \mathbb{R} \times \mathbb{R}$.

The "diagonal" in $\mathbb{R} \times \mathbb{R}$ gives exactly the elements equal to each other.

More generally:

Definition: Let A, B be sets. A subset of the Cartesian product $A \times B$ is called a relations between A and B. A subset of the Cartesian product $A \times A$ is called a relations on A.

Remark: Note how general this definition is. To make it useful for understanding predicates, we will need to introduce key properties relations can satisfy.

Example: $A = \{1, 3, 7\} \qquad B = \{1, 2, 5\}$

We can define a relation S on $A \times B$ by $S = \{(1, 1), (1, 5), (3, 2)\}$. This means $1S1$, $1S5$ and $3S2$ and no other ordered pairs in $A \times B$ satisfy S.

Remark: The relations we defined involve 2 elements, so they are often called <u>binary relations</u> in the literature.

8 Equivalence Relations

Task: Define the most useful kind of relation.

Definition: A relation R on a set A is called

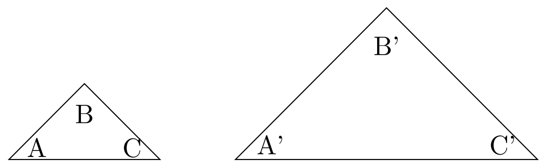
1. <u>reflexive</u> iff (if and only if) $\forall x \in A, xRx$
2. <u>symmetric</u> iff $\forall x, y \in A, xRy \to yRx$
3. <u>transitive</u> iff $\forall x, y, z \in A, xRy \wedge yRz \to xRz$

An equivalence relation on A is a relation that is reflexive, symmetric and transitive.

Notation: Instead of xRy, an equivalence relation is often denoted by $x \equiv y$ or $x \sim y$.

Examples:

1. "=" equality is an equivalence relation.
   (a) $x = x$ reflexive
   (b) $x = y \Rightarrow y = x$ symmetric
   (c) $x = y \wedge y = z \Rightarrow x = z$ transitive
2. $A = \mathbb{N}$
   $x \equiv y \bmod 3$ is an equivalence relation. $x \equiv y \bmod 3$ means $x - y = 3m$ for some $m \in \mathbb{Z}$, **i.e.** x and y have the same remainder when divided by 3. The set of all possible remainders is $\{0, 1, 2\}$
   NB: In correct logic notation, $x \equiv y \bmod 23$ if $\exists m \in \mathbb{Z} s.t. x - y = 3m$
   (a) $x \equiv x \bmod 3$ since $x - x = 0 = 3 \times 0 \to$ reflexive
   (b) $x \equiv y \bmod 3 \Rightarrow y \equiv x \bmod 3$ because $x \equiv y \bmod 3$ means $x - y = 3m$ for some $m \in \mathbb{Z} \Rightarrow y - x = -3m = 3 \times (-m) \Rightarrow y \equiv x \bmod 3 \to$ symmetric
   (c) Assume $x \equiv y \bmod 3$ and $y \equiv z \bmod 3$
   $x \equiv y \bmod 3 \Rightarrow \exists m \in \mathbb{Z}$ s.t. $x - y = 3m \Rightarrow y = x - 3m$
   $y \equiv z \bmod 3 \Rightarrow \exists p \in \mathbb{Z}$ s.t. $y - z = 3p \Rightarrow y = z + 3p$
   Therefore, $x - 3m = z + 3p \Leftrightarrow x - z = 3p + 3m = 3(p + m)$
   Since $p, m \in \mathbb{Z}, p + m \in \mathbb{Z} \Rightarrow x \equiv z \bmod 3 \to$ transitive.
3. Let $f : A \to A$ be any function on a non empty set A. We define the relation $R = \{(x, y) \mid f(x) = f(y)\}$
   (a) $\forall x \in A, f(x) = f(x) \Rightarrow (x, x) \in R \to reflexive$
   (b) If $(x, y) \in R$, then $f(x) = f(y) \Rightarrow f(y) = f(x)$, **i.e.** $(y, x) \in R \to$ symmetric
   (c) If $(x, y) \in R$ and $(y, z) \in R$, then $f(x) = f(y)$ and $f(y) = f(z)$, which by the transitivity of equality implies $f(x) = f(z)$, **i.e.** $(x, z) \in R$ as needed, so R is transitive as well.
   $f(x)$ can be $e^x, sin\, x, (x)$, etc.



4. Let λ be the set of all triangles in the plane. $ABC \sim A'B'C'$ if ABC and $A'B'C'$ are similar triangles, **i.e.** have equal angles.
   (a) $\forall ABC \in \lambda, ABC \sim ABC$ so $\sim$ is reflexive
   (b) $ABC \sim A'B'C' \Rightarrow A'B'C' \sim ABC$ so $\sim$ is symmetric
   (c) $ABC \sim A'B'C'$ and $A'B'C' \sim A''B''C'' \Rightarrow ABC \sim A''B''C''$, so $\sim$ is transitive

Clearly (a), (b), (c) use the fact that equality of angles is an equivalence relation.

Exercise: For various predicates you’ve encountered, check whether reflexive, symmetric or transitive. Examples of predicates include $\neq, <, >, \leq, \geq, \subseteq$

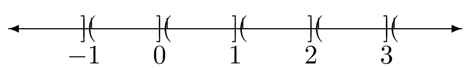
9 Equivalence Relations and Partitions

Task: Understand how equivalence relations divide sets.

Definition: Let A be a set. A partition of A is a collection of non empty sets, any two of which are disjoint such that their union is A, **i.e.** $\lambda = \{A_\alpha \mid \alpha \in I\}$ s.t. $\forall \alpha, \alpha' \in I$ satisfy $\alpha \neq \alpha', A_\alpha \cap A_{\alpha'} \neq \emptyset$ and $\bigcup_{\alpha \in I} A_\alpha = A$

Here I is an indexing act (may be infinite). $\underset{\alpha\in I}{A_\alpha}$ is the union of all the A_α’s (possibly an infinite union)

Example $\{(n, n+1) \mid n \in \mathbb{Z}\}$ is a partition of $\mathbb{R}$



$\bigcup_{n\in\mathbb{Z}} (n, n+1] = \mathbb{R}$

$(n, n+1] \cap (m, m+1] = \emptyset$ if $n \neq m$

Definition: If R is an equivalence relations on a set A and $x \in A$, the equivalence class of x denoted $[x]_R$ is the set $\{y \mid xRy\}$. The collection of all equivalence classes is called A modulo R and denoted A/R.

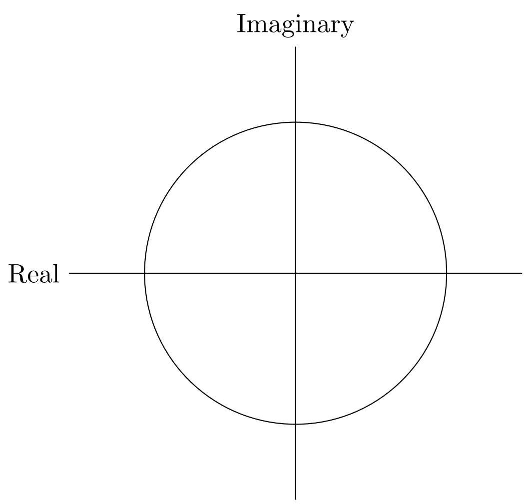
Examples:

1. $A = \mathbb{N}$ $\qquad x \equiv y \text{ mod } 3$
   We have the equivalence classes $[0]_R, [1]_R$ and $[2]_R$ given by the then possible remainders under division by 3.
   $[0]_R = \{0, 3, 6, 9, \dots\}$
   $[1]_R = \{1, 4, 7, 10, \dots\}$
   $[2]_R = \{2, 5, 8, 11, \dots\}$
   Clearly $[0]_R \cup [1]_R \cup [2]_R = \mathbb{N}$ and they are mutually disjoint $\Rightarrow R$ gives a partition of $\mathbb{N}$.

2. $ABC \sim A'B'C'$
   $[ABC] = \{$The set of all triangles with angles of magnitude $\measuredangle ABC, \measuredangle BAC, \measuredangle ACB\}$
   The union over the set of all $[ABC]$ is the set of all triangles and $[ABC] \cap [A'B'C'] = \emptyset$ if $ABC \nsim A'B'C'$ since it means these triangles have at least one angle that if difference.

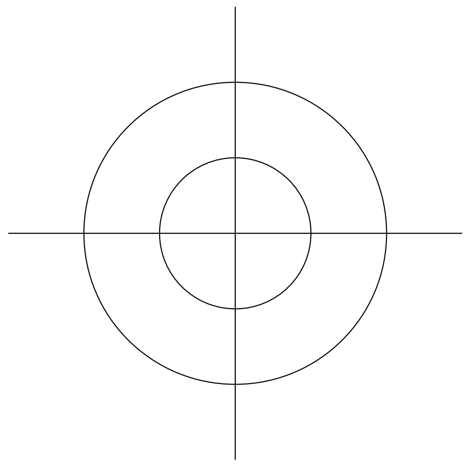
3. $A = \mathbb{C}$ $\qquad x \cap y$ if $|x| = |y|$ $\qquad$ equivalence relation
   $[x] = \{y \in \mathbb{C} \mid |x| = |y|\} = [r]$ for $r \in [0, +\infty) \wedge (r \geq 0)$

circle of radius $|x|$



$\bigcup\limits_{r \in [0,+\infty)} [r] = \mathbb{C}$
$[r_1] \cap [r_2] \neq \emptyset$ if $r_1 \neq r_2$ since two distinct circles in $\mathbb{C} \simeq \mathbb{R}^2$ with empty intersection.

circles $r_1 \wedge r_2$



Theorem: For any equivalence relation R on a set A, its equivalence classes form a partition of A, **i.e.**

1. $\forall x \in A, \exists y \in A$ s.t. $x \in [y]$ (every element of A sits somewhere)
2. $xRy \Leftrightarrow [x] = [y]$ (all elements related by R belong to the same equivalence class)
3. $\neg(xRy) \Leftrightarrow [x] \cap [y] = \emptyset$ (if two elements are not related by R, the they belong to disjoint equivalence classes)

Proof:

1. Trivial. Let $y = x$. $x \in [x]$ because R is an equivalence relation. Hence reflexive, so xRx holds.
2. We will prove $xRy \Leftrightarrow [x] \subseteq [y]$ and $[y] \subseteq [x]$
$\Rightarrow$ Fix $x \in A, [x] = \{z \in A \mid xRz\} \Rightarrow \forall y \in A$ s.t. $xRy, y \in [x]$. Furthermore, $[y] = \{w \in A \mid yRw\}$
$\Rightarrow \forall w \in [u], yRw$ but $xRy \Rightarrow xRw$ by transitivity. Therefore, $w \in [x]$. We have shown $[y] \subseteq [x]$.
Since R is an equivalence relation, it is also symmetric. **i.e.** $xRy \Leftrightarrow yRx$. So by the same argument with x and y swapped $yRx \Rightarrow [x] \subseteq [y]$. Thus $xRy \Rightarrow [x] = [y]$.
$\Rightarrow [x] = [y] \Rightarrow y \in [x]$ but $[x] = \{y \in A \mid xRy\}$
3. $\Rightarrow$ We will prove the contrapositive. Assume $[x] \cap [y] \neq \emptyset \Rightarrow \exists z \in [x] \cap [y]. z \in [x]$ means xRy, whereas $z \in [y]$ means $yRx \Leftrightarrow zRy$ by symmetric of R. We thus have xRz and $zRy \Rightarrow xRy$ by transitivity of $R. xRy$ contradicts $\neg(xRy)$ so indeed $\neg(xRy) \Rightarrow [x] \cap [y] = \emptyset$
$\Leftarrow$ Once again we use the contrapositive.
Assume $\neg(\neg(xRy)) \Leftrightarrow xRy$. By part (b) $xRy \Rightarrow [x] = [y] \Rightarrow [x] \cap$

$[y] \neq \emptyset$ since $x \in [x]$ and $y \in [y]$, **i.e.** These equivalence classes are non empty. We have obtained the needed contradiction.

qed

Q: What partition does "=" impose on $\mathbb{R}$?

A: $[x] = \{x\}$ since $E = \{(x, x) \mid x \in \mathbb{R}\}$ the diagonal.
The one element equivalence class is the smallest equivalence class possible (by definition, an equivalence class cannot be empty as it contains x itself). We call such a partition the finest possible partition.

Remark: The theorem above shows how every equivalence relations partitions a set. It turns out every partition of a set can be used to define an equivalence relation: xRy is x nd y belong to the same subset of the partition (check this is indeed an equivalence relations!). Therefore, there is a 1-1 correspondence between partitions and equivalence relations: to each equivalence relation there corresponds a partition and vice versa.

10 Partial Orders

Task: Understand another type of relation with special properties.

Definition: Let A be a set. A relation R on A is called anti-symmetric if $\forall x, y \in A$ s.t. $xRy \wedge yRx$, then $x = y$.

Definition: A partial order is a relation on a set A that is reflexive, anti-symmetric, and transitive.

Examples:

1. $A = \mathbb{R}$ $\quad\quad$ $\leq$ "less than or equal to" is a partial order
   (a) $\forall x \in \mathbb{R} x \leq x \rightarrow$ reflexive
   (b) $\forall x, y \in \mathbb{R}$ s.t. $x \leq y \wedge y \leq x \implies x = y \rightarrow$ anti-symmetric
   (c) $\forall x, y, z \in \mathbb{R}$ s.t. $x \leq y \wedge y \leq z \implies x \leq z \rightarrow$ transitive
   Same conclusion if $A = \mathbb{Z} \vee A\mathbb{N}$
2. A is a set. Consider $P(A)$, the power set of A. The relation $\subseteq$ "being a subset of" is a partial order.
   (a) $\forall B \in P(A), B \subseteq B \rightarrow$ reflexive.
   (b) $\forall B, C \in P(A), B \subseteq C \wedge C \subseteq B \implies B = C$ (recall the criterion for proving equality of sets) $\rightarrow$ anti-symmetric
   (c) $\forall B, C, D \in P(A)$ s.t. $B \subseteq C \wedge C \subseteq D \implies B \subseteq D \rightarrow$ transitive

The most important example of a partial order is example (2) "being a subset of".

Q: Why is "being a subset of" a partial order as opposed to a total order?

A: There might exist products B, C of A s.t. neither $B \subseteq C$ nor $C \subseteq B$ holds, **i.e.** where $B \wedge C$ are not related via inclusion.

11 Functions

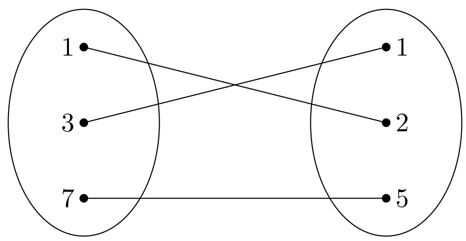
Task: Define a function rigorously and make sense of terminology associated to functions.

Definition: Let A, B be sets. A function $f : A \to B$ is a rule that assigns to <u>every</u> element of A <u>one and only one</u> elements of B, **i.e.** $\forall x \in A \exists! y \in B$ s.t. $f(x) = y$. A is called the <u>domain</u> of f and B is called the <u>codomain</u>.

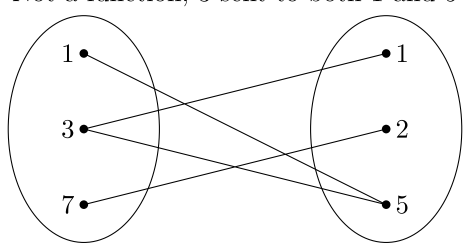
Examples:

1. $A = \{1, 3, 7\}$
   $B = \{1, 2, 5\}$

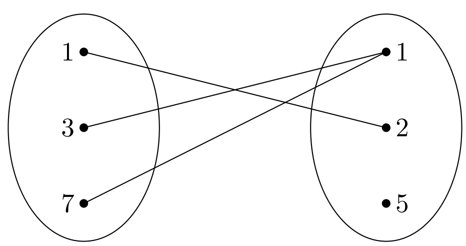
Is a function.



Not a function; 3 sent to both 1 and 5



Is a function.



2. $A = B = \mathbb{R}$ $\qquad$ $F : \mathbb{R} \to \mathbb{R}$ given by $f(x) = x$ is called the identity function.

Definition: Let A, B be sets and let $f : A \to B$ be a function. The <u>range</u> of f denoted by $f(A)$ is the subset of B defined by $f(A) = \{y \in B \mid \exists x \in A$ s.t. $f(x) = y\}$.

Definition: Let A be a set. A <u>Boolean function</u> on A is a function $F : A \to \{T, F\}$ which has A as its domain and the set of truth values $\{T, F\}$ as is codomain. $f : A \to \{T, F\}$ thus assigns truth values to the elements of A.

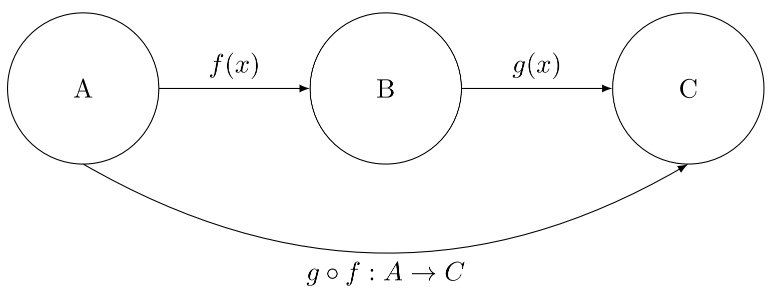
Function are often represented by graphs. If $f : A \to B$ is a function, the <u>graph</u> of f denoted $\Gamma(f)$ is the subset of the Cartesian product $A \times B$ given by $\{(x, f(x)) \mid x \in A\}$.

Q: Is it possible to obtain every subset of $A \times B$ as the graph of some function?

A: No! For $f : A \to B$ to be a function $\forall x \in A \quad \exists! y \in B$ s.t. $f(x) = y$, so for $\Gamma \subseteq A \times B$ to be the graph of some function, Γ must satisfy that $\forall x \in A \quad \exists! y \in B$ s.t. $(x, y) \in \Gamma$. Then we can define f by letting $y = f(x)$.

12 Composition of Functions

Task: Understand the natural operation that allows us to combine functions.



Example:

$f : \mathbb{R} \to \mathbb{R} \qquad f(x) = 2x$
$g : \mathbb{R} \to \mathbb{R} \qquad cos\ x$
$g \circ f(x) = g(f(x)) = g(2x = cos(2x))$
$f \circ g(x) = f(g(x)) = f(cos\ x) = 2(cos\ x) = 2cos\ x$

13 Inverting Functions

Task: Figure out which properties a function has to satisfy so that its action can be undone, **i.e.** when we can define an inverse to the original function.

Given $f : A \to B$, want $f^{-1} : B \to A$ s.t. $f^{-1} \circ f : A \to A$ is the identity $f^{-1} \circ f(x) = f^{-1}(f(x)) = x$
$A \xrightarrow{f} B \xrightarrow{f^{-1}} A$
It turns out f has to satisfy two properties for f^{-1} to exist.

1. Injective
2. Surjective

Definition: A function $f : A \to B$ is called injective or an injection (sometimes called one to one) if $f(x) = f(y) \Rightarrow x = y$

Examples:
$\sin x : [0, \frac{\pi}{2}] \to \mathbb{R}$ is injective
$\sin x : \mathbb{R} \to \mathbb{R}$ is not injective because $\sin x = \sin \pi = 0$

Definition: A function $f : A \to B$ is called surjective or a surjection (sometimes called onto) if $\forall z \in B \exists x \in A$ s.t. $\overline{f(x) = z}$.

Remark: f assigns a value to each element of A by its definition as a function, but it is not required to cover all of B. f is surjective if its range is all of B.

Examples:
$\sin x : \mathbb{R} \to [-1, 1]$ is surjective
$\sin x : \mathbb{R} \to \mathbb{R}$ is not surjective since $\nexists x \in \mathbb{R}$ s.t. $\sin x = 2$. We know $|\sin x| \leq 1 \forall x \in \mathbb{R}$

Definition: A function $f : A \to B$ is called bijective or a bijection if f is both injective and surjective.

Example: $f : \mathbb{R} \to \mathbb{R} \qquad f(x) = 2x + 1$ is bijective.

- Check injectivity $f(x_1) = f(x_2) \Rightarrow 2x_1 + 1 = 2x_2 + 1 \Leftrightarrow 2x_1 = 2x_2 \Leftrightarrow x_1 = x_2$ as needed.
- Check surjectivity $\forall z \in \mathbb{R}. f(x) = z$ means $2x + 1 = z$.
  Solve for x: $2x = z - 1 \Rightarrow x = \frac{z-1}{2} \in \mathbb{R} \Rightarrow f$ is surjective.

Remark: All bijective functions have inverses because we can define the inverse of a bijection and it will be a function:

- Surjectivity ensures f^{-1} assigns an element to every element of B (its domain).
- Injectivity ensures f^{-1} assigns to each elements of B one and only one elements of A.

Conclusion: $f : A \to B$ bijective $\Rightarrow f^{-1}$ exists, **i.e.** f^{-1} is a function. It turns out (reverse the arguments above) that f^{-1} exists $\Rightarrow f : A \to B$ is bijective.
Altogether we get the following theorem:

Theorem: Let $f : A \to B$ be a function. f^{-1} exists $\Leftrightarrow f : A \to B$ is bijective.

Q: How do we find the inverse function f^{-1} given $f : A \to B$?

A: If $f(x) = y$, solve for x as a function of y since $f^{-1}(f(x)) = f^{-1}(y) = x$ s $f^{-1} \circ f$ is the identity.

Example: $f(x) = 2x + 1 = y$. Solve for x in terms of y.
$f : \mathbb{R} \to \mathbb{R}$
$2x = y - 1 \qquad x = \frac{y-1}{2}$

14 Functions Defined on Finite Sets

Task: Derive conclusions about a function given the number of elements of the domain and codomain if finite; understand the pigeonhole principle.

Proposition: Let A, B be sets and let $f : A \to B$ be a function. Assume A is finite. Then f is injective $\Leftrightarrow$ $f(A)$ has the same number of elements as A.

Proof:

A is finite so we can write it as $A = \{a_1, a_2, \ldots, a_p\}$ for some p. Then $f(A) = \{f(a_1), f(a_2), \ldots, f(a_p)\} \subseteq B$. A priori, some $f(a_i)$ might be the same as some $f(a_j)$. However, f injective $\Leftrightarrow f(a_i) \neq f(a_j)$ whenever $i \neq j \Leftrightarrow f(A)$ has example p elements just like A.

qed

Corollary 1 Let A, B be finite sets such that $\#(A) = \#(B)$. Let $f : A \to B$ be a function. f is injective $\Leftrightarrow$ f is bijective.

Proof:

$\Rightarrow$ Suppose $f : A \to B$ is injective. Since A is finite, by the previous proposition, $f(A)$ has the same number of elements as A, but $f(A) \subseteq B$ and B has the same number of elements as $A \Rightarrow \#(A) = \#(f(A)) = \#(B)$, which means $f(A) = B$, **i.e.** f is also surjective $\Rightarrow$ f is bijective.

$\Leftarrow$ f is bijective $\Leftarrow$ f is injective.

qed

Corollary 2 (The Pigeonhole Principle) Let A, B be finite sets. If $\#(B) < \#(A)$, and let $f : A \to B$ be a function. $\exists a, a' \in A$ where $a \neq a'.f(a) = f(a')$

Remark: The name pigeonhole principle is due to Paul Erdös and Richard Rado. Before is was known as the principle of the drawers of Dirichlet. It has a simple statement, but it's a very powerful result in both mathematics and computer science.

Proof: Since $f(A) \subseteq B$ and $\#(B) < \#(A), f(A)$ cannot hve as many elements as A, so by the proposition, f cannot be injective, **i.e.** $\exists a, a' \in A$ where $a \neq a'$ (**i.e.** distinct elements) s.t. $f(a) = f(a')$

qed

Examples:

1. You have 8 friends. At least two of them were born the same day of the week. #(days of the week) $= 7 < 8$.
2. A family of five fives each other presents for Christmas. There are 12 presents under the tree. We conclude at least one person for three presents or more.
3. In a list of 30 words in English, at least two will begin with the same letter. #(Letter in the English alphabet) = 26 ¡ 30.

14.1 Behaviour of Functions on Infinite Sets

Let A be a set and $f : A \to A$ be a function. If A is finite , the corollary 1 tells us f injective $\Leftrightarrow$ f bijective. What if A is not finite?

14.1.1 Hilbert's Hotel problem (jazzier name: Hilbert's paradox of the Grand Hotel)

A fully occupied hotel with infinitely many rooms can always accommodate an additional guest as follows: The person in Room 1 removes to Room 2. The person in Room 2 moves to Room 3 and so on, **i.e.** if the rooms at $x_1, x_2, x_3 \ldots$ define the function $f(x_1) = x_2, f(x_2) = x_3, \ldots, f(x_m) = x_{m+1}$.

Claim: As defined f is injective but not surjective (hence not bijective!). Let $H = \{x_1, x_2, \ldots\}$ the hotel consisting of infinitely many rooms. $f : H \to H$ is given by $f(x_n) = f(x_{n+1})$. $f(H) = H \backslash \{x_1\}$. We cn use this idea to prove:

Proposition: A set A is finite $\Leftrightarrow$ $\forall f : A \to A$ an injective function is also bijective.

Proof: If the set X is finite then it follows immediately that every injective function $f : X \to X$ is bijective.
Suppose that the set X is infinite. Then there exists some infinite sequence $x_1, x_2, x_3, \ldots$ of distinct elements of X (where an element of X occurs at most once in this list). Then there exists a function $f : X \to X$ defined such that $f(x_n) = x_{n+1}$ for all positive integers of n, and $f(x) = x$ for all elements of x of X. If x is not a member of the infinite sequence $x_1, x_2, x_3, \ldots$ then the only elements of X that gets mapped to x is the element x itself; if $x = x_n$, where $x > 1$, then the only element of X gets mapped to x. It follows that the function f is injective. However it is not surjective, since x_1 does not belong to the range of the function. This function f is thus an example of a function from the set X to itself which is injective but not bijective.

15 Mathematical Induction

Task: Understand how to construct a proof using mathematical induction.

$\mathbb{N} = \{0, 1, 2, \dots\}$ set of natural numbers.
Recall that $\mathbb{N}$ is constructed using 2 axioms:

1. $0 \in \mathbb{N}$
2. If $n \in \mathbb{N}$, then $n + 1 \in \mathbb{N}$

Remarks:

1. This is exactly the process of counting.
2. If we start at 1, then we construct $\mathbb{N}* = \{1, 2, 3, 4, \dots\} = \mathbb{N} \backslash \{0\}$

via the axioms

1. $1 \in \mathbb{N}*$
2. if $n \in \mathbb{N}*$, then $n + 1 \in \mathbb{N}*$

$\mathbb{N}$ or $\mathbb{N}*$ is used for mathematical induction.

15.1 Mathematical Induction Consists of Two Steps:

Step 1 Prove statements $P(1)$ called <u>the base case</u>.

Step 2 For any n, assume $P(n)$ and prove $P(n+1)$. This is called <u>the inductive step</u>. In other words, step 2 proves the statement $\forall n P(n) \rightarrow P(n + 1)$

Remark: Step 2 is not just an implication but infinitely many! In logic notation, we have:

Step 1 $P(1)$

Step 2 $\forall n(P(n) \rightarrow P(n + 1))$
Therefore, $\forall n P(n)$
Let's see how the argument proceeds:

1. $P(1)$ Step 1 (base case)
2. $P(1) \rightarrow P(2)$ by Step 2 with $n = 1$
3. $P(2)$ by 1 & 2
4. $P(2) \rightarrow P(3)$ by Step 2 with $n = 2$
5. $P(3)$ by 3 & 4
6. $P(3) \rightarrow P(4)$ by Step 2 with $n = 3$
7. $P(4)$ by 5 & 6

$\vdots$

8. $P(n)$ for any n.
   This is like a row of dominos: knocking over the first one in a row makes all the others fall. Another idea is climbing a ladder.

Examples:

1. Prove $1+3+5+\cdots+(2n-1)=n^2$ by induction.

   Base Case: Verify statement for $n=1$

   When $n=1$, $2n-1=2\times 1-1=1^2$

   Inductive Step: Assume $P(n)$, **i.e.** $1+3+5+\cdots+(2n-1)=n^2$ and seek to prove $P(n+1)$, **i.e.** the statement $1+3+5\cdots+(2n-1+2(n+1)-1)=(n+1)^2$

   We start with LHS: $1+\underbrace{3+5+\cdots+(2n-1)}_{n^2}+(2(n+1)-1)=$

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

2. Prove $1+2+3++n=\frac{n(n+1)}{2}$ by induction.

   Base Case: Verify statement for $n=1$

   When $n=1, 1=\frac{1\times(1+1)}{2}=\frac{1\times 2}{2}=1$

   Inductive Step: Assume $P(n)$, **i.e.** $1+2+3+\cdots+n=\frac{n\times(n+1)}{2}$ and seek to prove $1+2+3+\cdots+n=\frac{(n+1)(n+2)}{2}$

   $\underbrace{1+2+3+\cdots+n}_{\frac{n(n+1)}{2}}+n+1=\frac{n(n+1)}{2}+n+1=(n+1)(\frac{n}{2}+1)=$

   $(n+1)\frac{n+2}{2}=\frac{(n+1)(n+2)}{2}$ as needed.

Remarks:

1. For some argument by induction, it might be necessary to assume not just $P(n)$ at the inductive step but also $P(1), P(2), \ldots, P(n-1)$. This is called strong induction.

   Base Case: Prove $P(1)$

   Inductive Step: Assume $P(a), P(2), \ldots, P(n)$ and prove $P(n+1)$.

   An example of result requiring the use of strong induction is the Fundamental Theorem of Arithmetic: $\forall n \in \mathbb{N}, n \geq 2, n$ can be expressed as a product of one or more prime numbers.

2. One has to be careful with argument involving induction. Here is an illustration why:

   Polya's argument that all horses are the same colour:

   Base Case: $P(1)$ There is only one horse, sot hat has a colour.

   Inductive Step Assume any n horses are the same colour.

   Consider a group of $n+1$ horses. Exclude the first horse and look at the other n. All of these are the same colour by our assumption. Now exclude the last horse. The remaining n horses are the same colour by our assumption. Therefore, the first horse, the horses in the middle, and the last horse are all of the same colour. We have established the inductive step.

Q: Where does the argument fail?

A: For $n = 2, P(2)$ is false because there are no middle horses to compare to.

The Grand Hotel Cigar Mystery

Recall Hilbert's hotel - the grand Hotel. Suppose that the Grand Hotel does not allow smoking and no cigars may be taken into the hotel. In spite of the rules, the guest in Room 1 goes to Room 2 to get a cigar. The guest in Room 2 goes to Room 3 to get 2 cigars (one for him and one for the person in room 1), etc. In other words, guest in Room N goes to Room N+1 to get N cigars. They will each get back to their rooms, smoke one cigar, and give the result to the person in Room N-2.

Q: Where is the fallacy?

A: This is an induction argument without a base case. No cigars are allowed in the hotel so no guests have cigars. An induction cannot get off the ground without a base case.

16 Abstract Algebra

Task: Understand binary operators, semigroups, monoids, and groups as well as their properties.

16.1 Binary Operations

Definition: Let A be a set. A binary operation $*$ on A is an operation applied to any two elements $x, y \in A$ that yields on elements $x * y$ in A. In other words, $*$ us a binary operation on A if $\forall x, y \in A, x * y \in A$.

Examples:

1. $\mathbb{R}, +$ addition on $\mathbb{R} : \forall x, y \in \mathbb{R}, x + y \in \mathbb{R}$
2. $\mathbb{R}, -$ subtraction on $\mathbb{R} : \forall x, y \in \mathbb{R}, x - y \in \mathbb{R}$
3. $\mathbb{R}, \times$ multiplication on $\mathbb{R} : \forall x, y \in \mathbb{R}, x \times y \in \mathbb{R}$
4. $\mathbb{R}, /$, division on $\mathbb{R}$ is NOT a binary operation because $\forall x \in \mathbb{R} \exists o \in \mathbb{R}$ s.t. $\frac{x}{o}$ is undefined (not an element of $\mathbb{R}$)

Definition: A binary operation $*$ on a set A is called commutative if $\forall x, y \in A, x * y = y * x$

Examples:

1. $\mathbb{R}, +$ is commutative since $\forall x, y \in \mathbb{R}, x + y = y + x$

2. $\mathbb{R}, \times$ is commutative since $\forall x, y \in \mathbb{R}, x \times y = y \times x$
3. $\mathbb{R}, -$ is not commutative since $\forall x, y \in \mathbb{R}, x - y \neq y - x$ in general. $x - y = y - x$ only if $x = y$
4. Let Mn be the set of n by n matrices with entries in $\mathbb{R}$ and let $*$ be matrix multiplication. $\forall A, B \in Mn, A * B \in Mn$, so $*$ is a binary operation, but $AB \neq BA$ in general. Therefore $*$ is not commutative.

Definition: A binary operation $*$ on a set A is called associative if $\forall x, y, z, (x * y) * z = x * (y * z)$

Examples:

1. $\mathbb{R}, +$ is associative since $\forall x, y, z \in \mathbb{R}, (x + y) + z = x + (y + z)$
2. $\mathbb{R}, \times$ is associative since $\forall x, y, z \in \mathbb{R}, (x \times y) \times z = x \times (y \times z)$
3. Intersection $\cap$ on sets is associative since $\forall A, B, C$ sets $(A \cap B) \cap C = A \cap (B \cap C)$.
4. Union $\cup$ on sets is associative since $\forall A, B, C$ sets $(A \cup B) \cup C = A \cup (B \cup C)$
5. $\mathbb{R}, -$ is not associative since $(1-3)-5 = -2-5 = -7$ but $1-(3-5) = 1 - (-2) = 1 + 2 = 3$

Remark: When we are dealing with associative binary operations we can drop the parentheses, **i.e.** $(x * y) * z$ can be written $x * y * z$.

16.2 Semigroups

Definition: A semigroup is a set endowed with an associative binary operation. We denote the semigroup $(A, *)$

Examples:

1. $(\mathbb{R}, +)$ and $(\mathbb{R}, -)$ are semigroups.
2. Let A be a set and let $P(A)$ be its power set. $(P(A), \cap)$ and $(P(A), \cup)$ are both semigorups.
3. $(Mn, *)$, the set of $n \times n$ matrices with entries in $\mathbb{R}$ with the operation of matrix multiplication (which is associative $\rightarrow$ a bit gory to prove) forms a semigroup.

   Since $*$ is associative on a semigroup, we can define a^n :
   $a^1 = a$
   $a^2 = a * a$
   $a^3 = a * a * a$
   $\vdots$
   Recursively, $a^1 = 1$ and $a^n = a * a^{n-1}, \forall n > 1$

NB: In $(\mathbb{R}, \times), \forall a \in \mathbb{R}, a^n = \underbrace{a \times a \times \cdots \times a}_{n\ times}$, whereas in $(\mathbb{R}, +), \forall a \in \mathbb{R}, a^n = \underbrace{a + a + \cdots + a}_{n\ times} = na$. Be careful what $*$ stands for!

Theorem: Let $(A, *)$ be a semigroup. $\forall a \in A, a^m * a^n = a^{m+n}, \forall m, n \in \mathbb{N}*$.

Proof: By induction on m.

Base Case: $m = 1$ $\qquad a^1 * a^n = a * a^n = a * 1 + n$

Inductive Step: Assume the result is true for $m = p$, **i.e.** $a^p * a^n = a^{p+n}$ and seek to prove that $a^{p+1} * a^n = a^{p+1+n}$

$a^p + 1 * a^n = (a * a^p) * a^n = a * (a^p * a * n) = a * a^{p+n} = a^{p+1+n}$

Theorem: Let$(A, *)$ be a semigroup. $\forall a \in A, (a^m)^n = a^{mn}, \forall m, n \in \mathbb{N}*$

Proof: By induction on n.

Base Case: $n = 1$ $\qquad (a^m)^1 = a^m = a^{m \times 1}$

Inductive Step: Assume the result if true for $n = p$, **i.e.** $(a^m)^p = a^{mp}$ and seek to prove that $(a^m)^{p+1} = a^{m(p+1)}$

$(a^m)^{p+1} = (a^m)^p * a^m = a^{mp} * a^m = a^{mp+m} = a^{m(p+1)}$

16.2.1 General Associative Law

Let $(A, *)$ be a semigroup. $\forall a_1, \ldots, a_s \in A, a_1 * a_2 * \cdots * a_s$ has the same value regardless of how the product is bracketed.

Proof Use associativity of $*$.

qed

NB: Unless $(A, *)$ has a commutative binary operation, $a_1 * a_2 * \cdots * a_s$ does depend on the <u>ORDER</u> in which the $a_j's$ appearin $a_1 * a_2 * \cdots * a_s$

16.2.2 Identity Elements

Definition: Let $(A, *)$ be a semigroup. An element $e \in A$ is called an identity element for the binary operation $*$ if $e * x = x * e = x, \forall x \in A$.

Examples:

1. $(\mathbb{R}, +)$ has 0 as the identity element.
2. $(\mathbb{R}, \times)$ has 1 as the identity element.
3. Given a set $A, (P(A), \cup)$ has $\emptyset$ (the empty set) as its identity elements, whereas $(P(A), \cap)$ has A as its identity element.
4. $(Mn, *)$ has In, the identity matrix as its identity element.

Theorem A binary operation on a set cannot have more than one identity elements, **i.e.** if an identity element exists, then it is unique.

Proof: Assume not (proof by contradiction). Let e and e' both be identity elements for a binary operation on a set A. $e = e * e' = e'$

qed

16.3 Monoids

Definition: A <u>monoid</u> is a set A endowed with an associative binary operation $*$ that has an identity element e. In other words, a monoid is a semigroup $(A, *)$ where $*$ has an identity element e.

Definition: A monoid $(A, *)$ is called <u>commutative</u> (or <u>Abelian</u>) if the binary operation $*$ is commutative.

Example:

1. $(\mathbb{R}, +)$ is a commutative monoid with $e = o$.
2. $(\mathbb{R}, \times)$ is a commutative monoid with $e = 1$.
3. Given a set A, $(P(A), \cup)$ is a commutative monoid with $e = \emptyset$.
4. $(M, n*)$ is a monoid since $e = In$, but it is not commutative since matrix multiplication is not commutative.
5. $(\mathbb{N}, +)$ is a commutative monoid with $e = o$, whereas $(\mathbb{N}*, +)$ is merely a semigroup (recall $\mathbb{N}* = \mathbb{N}\backslash\{0\}$)

Theorem: Let $(A, *)$ be a monoid and let $a \in A$. Then $a^m * a^n = a^{m+n}, \forall m, n \in \mathbb{N}$

Remark: Recall that we proved this theorem for semigroups if $m, n \in \mathbb{N}*$. We now need to extend that result.

Proof: A monoid is a semigroup $\implies \forall a \in A, a^m * a^n = a^{m+n}$ whenever $m, n \in \mathbb{N}*$, **i.e.** $m > 0$ and $n > 0$. Now let $m = 0$. $a^m * a^n = a^0 * a^n = e * a^n = a^n = a^{0+n}$
If $n = 0, a^m * a^n = a^m * a^0 = a^m * e = a^m = a^{m+0}$

qed

Theorem: Let $A, *)$ be a monoid, $\forall a \in A \forall m, n \in \mathbb{N}, (a^m)^n = a^{mn}$

Remark: Once again, we had this result for semigroups when $m > 0$ and $n > 0$

Proof: By the remark, we only need to prove the result when $m = 0$ or $n = 0$. If $m = 0, (a^0)^n = (e)^n = e = a^0 = a^{0\times n}$. If $n = 0$ then $(a^m)^0 = e = a^0 = a^{0\times m}$

17 Inverses

Task: Understand what an inverse is and what formal properties it satisfies.

Definition: Let $(A, *)$ be a monoid with identity element e and let $a \in A$. An element y of A is called the <u>inverse</u> of x if $x * y = y * x = e$. If an element $a \in A$ has an inverse, then a is called <u>invertible</u>.

Examples:

1. $(\mathbb{R}, +)$ has identity element 0. $\forall x \in \mathbb{R}, (-x)$ is the inverse of x since $x + (-x) = (-x) = x = 0$.
2. $(\mathbb{R}, \times)$ has identity element 1. $x \in \mathbb{R}$ is invertible only if $x \neq 0$. If $x \neq 0$, the inverse of x is $\frac{1}{x}$ since $x \times \frac{1}{x} = \frac{1}{x} \times x = 1$.
3. $(Mn, *)$ the identity element is In. $A \in Mn$ is invertible if $det(A) \neq 0$. A^{-1} the inverse is exactly the one you computed in linear algebra. If $det(a) = 0, A$ is <u>NOT</u> invertible.
4. Given a set $A, (P(A), \cup)$ has $\emptyset$ as its identity element of all the elements of $P(A)$ only $\emptyset$ is invertible and has itself asits inverse: $\emptyset \cup \emptyset = \emptyset \cup \emptyset = \emptyset$

Theorem: Let $(A, *)$ be a monoid. If $a \in A$ has an inverse, then that inverse is unique.

Proof: By contradiction: Assume not, then $\exists a \in A$ s.t. both b and c in A are its inverses, **i.e.** $a * b = b * a = e$, the identity element of $(A, *)$ and $a * c = c * a = e$ and $b \neq c$, then $b = b * e = b * (a * c) = (b * a) * c = e * c = c$.

qed

Since every invertible element of a for $(A, *)$ a monoid has a unique inverse, we can denote the inverse by the more standard notation a^{-1}.

Next, we need to understand inverses of elements obtained via the binary operation:

Theorem: Let $(A, *)$ be a monoid and let a, b be invertible elements of A. $a * b$ is also invertible and $a * b^{-1} = b^{-1} * a^{-1}$.

Remark: You might remember this formula from linear algebra when you looked at the inverse of a product of matrices AB.

Proof: Let e be the identity element of $(A, *) a * a^{-1} = a^{-1} * a = e$ and $b * b^{-1} = b^{-1} * b = e$. We would like to show $b^{-1} * a * -1$ is the inverse of $a * b$ by computing $(a * b) * (b^{-1} * a^{-1})$ and $(b^{-1} * a^{-1}) * (a * b)$ and showing both are e.

$(a * b) * (a^{-1} * b^{-1}) = a * (b * b^{-1}) * a^{-1} = a * e * a^{-1} = a * a^{-1} = e$

$(b^{-1} * a^{-1}) * (a * b) = b^{-1} * (a^{-1} * a) * b = b^{-1} * e * b = (b^{-1} * e) * b = b^{-1} * b = e$

Thus $b^{-1} * a^{-1}$ satisfies the conditions needed for it to be the inverse of $a * b$.Since an inverse of unique, $a * b$ is invertible and $b^{-1} * a^{-1}$.

Theorem: Let $(A, *)$ be a monoid, and let $a, b \in A$. Let $x \in A$ be invertible. $a = b * x \Leftrightarrow b = a * x^{-1}$. Similarly, $a = x * b \Leftrightarrow b = x^{-1} * a$

Proof: Let e be the identity element of $(A, *)$.

First $a = b * x \Leftrightarrow b = a * x^{-1}$:

$\Rightarrow$ Assume $a = b * x.a * x^{-1} = (b * x) * x^{-1} = b * x * x^{-1} = b * e = b$ as needed.

$\Leftarrow$ Assume $b = a * x^{-1}$. Then $b * x = (a * x^{-1}) = a * (x^{-1} * x) = a * e = 1$ as needed.

Apply the same type of argument to show $a = x * b \Leftrightarrow b = x^{-1} * a$.

qed

Let $(A, *)$ be a monoid. We can now make sense of a^n for $n \in \mathbb{Z}, n < -$ for every $n \in A$ invertible. Since n is a negative integer, $\exists p \in \mathbb{N}$ s.t. $n = -1$. Set $a^n = a^{-p} = (a^p)^{-1}$.

Theorem: Let $(A, *)$ be a monoid and let $a \in A$ be invertible. Then $a^n * a^m = a^{m+n} \quad \forall m, n \in \mathbb{Z}$.

Proof: When $m \geq 0 \wedge n \geq n$ we have already proven this result. The rest of the proof splits into cases.

Case 1: $m = n \vee n = 0$

If $m = 0, n \in \mathbb{Z}, a^m * a^n = a^0 * a^n = e * a^n = a^n = a^{0+n}$ as needed.

If $m \in \mathbb{Z}, n = 0, a^m * a^n = a^m * a^0 = a^m * e = a^m = a^{m+0}$ as needed.

Case 2: $m < 0 \wedge n < 0$

$m < 0 \Rightarrow \exists p \in \mathbb{N}\ s.t.\ p = -m.\ n < 0 \Rightarrow \exists q \in \mathbb{N}\ s.t.\ q = -n.$

$a^m = a^{-p} = (a^p)^{-1} \wedge a^n = a^{-q} = (a^q)^{-1}$

$a^m * a^n = (a^p)^{-1} * (a^q)^{-1} = (a^q * a^p)^{-1} = (a^{p+q})^{-1} = a^{-(p+q)} = a^{-q-p} = a^{m+n} = a^{n+m}$

Case 3: $m \wedge n$ have opposite signs.

Without loss of generality, assume $m < 0 \wedge n > 0$ (the case $m > 0 \wedge n < 0$ is handled by the same argument). Since $m < 0, \exists p \in \mathbb{N}\ s.t.\ p = -m$. This case splits into two subcases:

Case 3.1: $m + n \geq 0$

Set $q = m + n$. Then $a^{m+n} = a^q = e * a^q = (a^p)^{-1} * a^p * a^q = (a^p)^{-1} * a^{p+q} = a^{-p} * a^{p+q} = a^m * a^{-m+m+n} = a^m * a^n$

Case 3.2: $m + n < 0$

Set $q = -(m+n) = -m-n \in \mathbb{N}^*$. Then $a^{m+n} = a^{-q} = (a^q)^{-1} * e = (a^q)^{-1} * (a^{-n} * a^n) = (a^q)^{-1} * (a^n)^{-1} * a^n = (a^n * a^q)^{-1} * a^n = (a^{n+p})^{-1} * a^n = (a^{n-m-n})^{-1} * a^n = (a^{-m})^{-1} * a^n = (a^p)^{-1} * a^n = a^m * a^n$

Theorem: Let $(A, *)$ be a monoid, and let a be an invertible element of A. $\forall m, n \in \mathbb{Z}, (a^m)^n = a^{mn}$.

Proof: We consider 3 cases:

Case 1: $n > 0$, **i.e.** $n \in \mathbb{N}^*$. $m \in \mathbb{Z}$ with no additional restrictions we proceed by induction on m.

Base Case: $n = 1$ $\quad (a^m)^1 = a^m = a^{m\times 1}$

Inductive Step: We assume $(a^m)^n = a^{mn}$ and seek to prove $(a^m)^{n+1} = a^{m(n+1)}$. Start with $(a^m)^{n+1} = (a^m)^n * (a^m)^1 = a^{mn} * a^m = a^{mn+m} = a^{m(n+1)}$

Case 2: $n = 0$; no restriction on $m \in \mathbb{Z}$

$(a^m)^n = (a^m)^0 = e = a^0 = a^{m\times 0} = a^{mn}$

Case 3: $n < 0$; no restriction on $m \in \mathbb{Z}$.

Since $n < 0, \exists p \in \mathbb{N}\, s.t.\ p = -n$. By case 1, $(a^m)^p = a^{mp}$

$(a^m)^n = (a^m)^{-p} = ((a^m)^p)^{-1} = (a^{mp})^{-1} = a^{-mp} =^{m(-p)}= a^{mn}$

17.1 Groups

A motion formally defined in the 1870's even though theorems about groups proven as early as a century before that.

Definition: A group is a monoid in which every element is invertible. In other words, a group is a set A endowed with a binary operation $*$ satisfying the following properties:

1. $*$ is associative, **i.e.** $\forall x, y, z \in A, (x * y) * z = x * (y * z)$
2. There exists an identity element $e \in A$, **i.e.** $\exists e \in A\, s.t. \forall a \in A, a * e = e * a = a$
3. Every element of A in invertible, **i.e.** $\forall a \in A \exists a^{-1} \in A\ s.t.\ a * a^{-1} = a^{-1} * a = e$

Notation for Groups: $(A, *) \vee (\underbrace{A}_{set}, \underbrace{*}_{operation}, \underbrace{e}_{identity})$

Remark: Closure under the operation $*$ is implicit in the definition **i.e.** $\forall a, b \in A, a * b \in A$

Definition: A group $(A, *, e)$ is called commutative or Abelian if its operation $*$ is commutative.

Examples:

1. $(\mathbb{R}, +, 0)$ is an Abelian group.

   $-x$ is the inverse of $x, \forall x \in \mathbb{R}$
2. $(\mathbb{Q}^*, \times, 1)$ $\quad \mathbb{Q}^* = \mathbb{Q}^* \backslash \{0\}$ $\quad (\mathbb{Q}^*, \times, 1)$ is Abelian

   $\forall q \in \mathbb{Q}^*, q^{-1} = \frac{1}{q}$ is the inverse.
3. $(\mathbb{R}^3, +, 0)$ vectors in $\mathbb{R}^3$ with vector addition forms an Abelian group.

   $(x, y, z) + (x', y', z') = (x + x', y + y', z + z')$ vector addition.

   $0 = (0, 0, 0)$ is the identity. $\quad (-x, -y, -z) = -(x, y, z)$ is the inverse of (x, y, z).

4. $(\tilde{\text{M}}m, *, In)$ $n \times n$ invertible matrices with real coefficients under matrix multiplication with In as the identity elements forms a group which is NOT Abelian.
5. Set $A = \mathbb{Z}$ and recall the equivalence relation $x \equiv y \bmod 3$ **i.e.** $x \wedge y$ have te same remainder under the division by 3. Recall that $\mathbb{Z}/N = \{0, 1, 2\}$, **i.e.** the set of equivalence classes under the partition determined by this equivalence relation. We denote $\mathbb{Z}/N\ \{0, 1, 2\} = \mathbb{Z}_3$

   Consider $(\mathbb{Z}_3, \oplus_3, 0)$ where $\oplus_3$ is the operation of addition modulo 3, **i.e.** $1 + 0 = 1, 1 + 1 = 2, 1 + 2 = 3 \equiv 0 \bmod 3$.

Claim: $(\mathbb{Z}_3, \oplus_3, 0)$ is an Abelian group.

Proof of Claim: Associativity of $\oplus_3$ follows from the associativity of $+$, addition of $\mathbb{Z}$. Clearly, 0 is the identity (don't forget 0 stands for all elements with remainder 0 under division by 3, **i.e.** $\{0, 3, -3, 6, -3, \dots\}$). To compute inverses recall that $a \oplus_3 a^{-1} = 0, 0$ is the inverse of 0 because $0 + 0 = 0$. 2 is he inverse of 1 because $1 + 2 = 3 \equiv 0 \bmod 3$, and 1 is the inverse of 2 because $2 + 1 = 3 \equiv 0 \bmod 3$.

More generally, consider the equivalence relation on $\mathbb{Z}$ given by $x \equiv y \bmod n$ for $n \geq 1.\mathbb{Z}/N = \{0, 1, \dots, n-1\} = \mathbb{Z}_n$. All possible remainders under division by n are the equivalence classes. Let $\oplus_n$ be addition mod n. By the same argument as above, $(\mathbb{Z}_n, \oplus_n, 0)$ is an Abelian group.

Q: What is we consider multiplication mod n, **i.e.** $\otimes_n$. Is $(\mathbb{Z}_n, \otimes_n, 1)$ a group?

A: No! $(\mathbb{Z}_n, \otimes_n, 1)$ is not even a monoid because $1 \otimes_n 0 = 0 \otimes_n 1 = 0$, so 1 is not an identity element for $\otimes_n$ on $\mathbb{Z}_n$.

Q: Can this be fixed?

A: Troubleshoot how to get rid of 0.

Consider $\mathbb{Z}_n^* = \mathbb{Z}_n \backslash \{0\} = \{1, 2, \dots, n-1\}$ all non-zero elements in $\mathbb{Z}_n^*$. This eliminates 0 as an element, but can 0 arise any other way from the binary operation? It turns out the answer depends on n. If n is not prime, say $n = 6$, we get two divisors, **i.e.** elements that yield 0 when multiplied by precisely the factors of n, for $n = 6$, $\mathbb{Z}_6^* = \{1, 2, 3, 4, 5\}$ but $2 \otimes_6 3 = \overline{6 \equiv 0 \bmod 6}$, so $2 \wedge 3$ are two divisors.

Claim: If n is prime, then $(\mathbb{Z}_n^*, \otimes_n, 1)$ is an Abelian group.

Used in cryptology $\rightarrow$ you will see next semester.

As an example, let us look at the multiplication table for $\mathbb{Z}_5^*$ to see the inverse of various elements: $\mathbb{Z}_5^* = \mathbb{Z}_5 \backslash \{0\} = \{0, 1, 2, 3, 4, \} \backslash \{0\} - \{1, 2, 3, 4\}$

	1	2	3	4
1	1	2	3	4
2	2	4	1	2
3	3	1	4	2
4	4	3	2	1

$1^{-1} = 1 \quad 1 \otimes_5 1 = 1$

$2^{-1} = 3 \quad 2 \otimes_5 3 = 6 \equiv 1 \bmod 5$

$3^{-1} = 2 \quad 3 \otimes_5 2 = 6 \equiv 1 \bmod 5$

$4^{-1} = 1 \quad 4 \otimes_5 4 = 16 \equiv 1 \bmod 5$

The fact that $\mathbb{Z}_n^*, \otimes_n, 1$ is Abelian follows from the commutativity of multiplication on $\mathbb{Z}$.

6. Let $(A, *, e)$ be any group and let $a \in A$.
   Consider $A' = \{a^m \mid m \in \mathbb{Z}\}$ all powers of a. It turns out $(A', *, e)$ is a group called the cyclic group determined by a. $(A', *, e)$ is Abelian regardless of whether the original group was Abelian or not because of the theorem we proved on powers of a: $\forall m, n \in \mathbb{Z} \qquad a^m * a^n = a^{m+n} = a^{n+m} = a^n * a^m$.
   Cyclic groups come in two flavours: finite (A' is a finite set) and infinite (A' is an infinite set).
   For example, let $(A, *, e) = (\mathbb{Q}^*, , \times, 1)$
   If $a = -1 \qquad A' = \{(-1)^m \mid m \in \mathbb{Z}\} = \{-1, 1\}$ is finite.
   If $a = 2 \qquad A' = \{2^m \mid m \in \mathbb{Z}\} = \{1, 2, \frac{1}{2}, 4, \frac{1}{4}, \dots\}$ is infinite.

18 Homomorphisms and Isomorphisms

Task: Understand the most natural functions between objects in abstract algebra such as semigroups, monoids or groups.

Definition: Let $(A, *)$ and $(B, *)$ ve vitg semigroups, monoids or groups. A function $f : A \to B$ is called a homomorphism if $f(x * y) = f(x) * f(y)\, \forall x, y \in A$. In other words, if f is a function that respects (behaves well with respect) to the binary operation.

Examples:

1. Consider $(\mathbb{Z}, +, 0)$ and $(\mathbb{R}^*, \times, 1)$.
   Pick $a \in \mathbb{R}^*$, then $f(n) = a^n$ is a homomorphism between $(\mathbb{Z}, +, 0)$ and $(\mathbb{R}^*, \times, 1)$ because $(\mathbb{R}^*, \times, 1)$ is a group, and we proved for groups that $a^{m+n} = f(m+n) = a^m * a^n = f(m) * f(n) \qquad \forall m, n \in \mathbb{Z}$.
2. More generally, $\forall a \in A$ invertible, where $(A, *)$ is a monoid with identity element e, $f(n) = a^m$ gives a homomorphism between $(\mathbb{Z}, +, 0)$ and $(A', *, e)*$, where as before $A' = \{a^m \mid m \in \mathbb{Z}\} \subset A$.
   We get even better behaviour if we require $f : A \to B$ to be bijective.

Definition: Let $(A, *)$ and $(B, *)$ both be semigroups, monoids or groups. A function $f : A \to B$ is called an isomorphism if $f : A \to B$ is both bijective AND a homomorphism.

Examples:

1. Let $A' = \{2^m \mid m \in \mathbb{Z}\} = \{1, 2, \frac{1}{2}, 4, \frac{1}{4}, \dots\}$
   $f(m) = 2^m$ from $(\mathbb{Z}, +, 0)$ to $(A', \times, 1)$ is an isomorphism since $2^m \neq 2^n$ if $m \neq n$.
2. Let $A' = \{(-1)^m \mid m \in \mathbb{Z}\} = \{-1, 1\}$
   $f(m) = (-1)^m$ from $(\mathbb{Z}, +, 0)$ to $(A', \times, 1)$ is NOT an isomorphism since it's not injective $(-1)^2 = (-1)^4 = 1$.

Theorem: Let $(A, *)$ and $(B, *)$ both be semigroups, monoids or groups. The inverse $f^{-1} : B \to A$ of any isomorphism $f : A \to B$ from A to B is itself an isomorphism.

Proof: If $f : A \to B$ is an isomorphism $\Rightarrow f : A \to B$ is bijective $\Rightarrow f^{-1} : B \to A$ is bijective (proven when we discussed functions).

To show $f^{-1}B \to A$ is a homomorphism, let $u, v \in B$. $\exists x, y \in A$ s.t. $x = f^{-1}(u)$ and $y = f^{-1}(v)$, but then $u = f(x)$ and $v = f(y)$.

Since $f : A \to B$ is a homomorphism, $f(x * y) = f(x) * f(y) = u * v$. Then $f^{-1}(u * v) = f^{-1}(f(x * y)) = x * y = f^{-1}(u) * f^{-1}(v)$ as needed.

qed

Definition: Let $(A, *)$ and $(B, *)$ both be semigroups, monoids or groups. If $\exists f : A \to B$ an isomorphism betwen A and B, then $(A, *)$ and $(B, *)$ are said to be isomorphic.

Remark: "Ismorphic" comes from "iso" same + "morphé" form same abstract algebra structure on both $(A, *)$ and $(B, *)$ given to you in two different guises. As the French would say: "Même Marie, autre chapeau" same Mary, different hat.

19 Formal Languages

Task: Use what we learned about structures in abstract algebra in order to make sense of formal languages and grammars.
Let A be a finite set. When studying formal languages, we call A an <u>alphabet</u> and the elements of A <u>letters</u>.

Examples:

1. $A = \{0, 1\}$ binary digits
2. $A = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$ decimal digits
3. $A =$ letters of the English alphabet

Definition: $\forall n \in \mathbb{N}^*$, we define a <u>word</u> of length n in the alphabet A as being any string of the form $a_1, a_2, \ldots, a_n \, s.t. \, a_i \in A \quad \forall i, 1 \leq i \leq n$. Let A^n be the set of all words of length n over the alphabet A.

Remark: There is a one-to-one correspondence between the string $a_1 a_2 \ldots a_n$ and the ordered n-tuple $(a_1, a_2, \ldots, a_n) \in A^n = \underbrace{A \times \cdots \times A}_{n \, times}$ the Cartesian product of n copies of A.

Definition: Let $A^+ = \bigcup\limits_{n=1}^{\infty} A^n = A^1 \cup A^2 \cup A^3 \cup \ldots.$ A^+ is the set of all words of positive length over the alphabet A.

Examples:

1. $A = \{0, 1\}, A^+$ is the set of all binary strings of finite length that is at least on, **i.e.** $0, 1, 01, 10, 00, 11$, etc.
2. If $A =$ letters of the English alphabet, then A^+ consists of all non-empty strings of finite lengths of letters from the English alphabet. It is useful to also have the empty for ε in our set of strings. ε has length 0. Define $A^0 = \{\varepsilon\}$ and then adjoin the empty word ε to A^+. We get $A^* = \{\varepsilon\} \cup A^+ = A^0 \cup \bigcup_{n=1}^{\infty} A^n = \bigcup_{n=0}^{\infty} A^n$.

Notation: We denote the length of a word w by $\mid w \mid$. Next introduce an operation on A^*.

Definition: Let A be a finite set and let $w_1 \wedge w_2$ be words in A^*. $w_1 = a_1 a_2 \ldots a_m \wedge w_2 = b_1 b_2 \ldots b_n$. The <u>concatenation</u> of $w_1 \wedge w_2$ is the word $w_1 \circ w_2$, where $w_1 \circ w_2 = a_1 a_2 \ldots a_m b_1 b_2 \ldots b_n$. Sometimes $w_1 \circ w_2$ is denoted as just $w_1 w_2$. Note that $\mid w_1 \circ w_2 \mid = \mid w_1 \mid + \mid w_2 \mid$.
Concatenation of words is:

1. associative
2. <u>NOT</u> commutative is A has more than one element.

Proof of (1): Let $w_1, w_2, w_3 \in A^*. w_1 = a_1 a_2 \ldots a_m$ for some $m \in \mathbb{N}, w_2 = b_1 b_2 \ldots b_n$ for some $m \in \mathbb{N}$ and $w_3 = c_1 c_2 \ldots c_p$ for some $p \in \mathbb{N}$. $w_1 \circ w_2) \circ w_3 = w_1 \circ (w_2 \circ w_3) = a_1 a_2 \ldots a_m b_1 b_2 \ldots b_n c_1 c_2 \ldots c_p$.

qed

Proof of (2): Since A has at least two elements, $\exists a, b \in A$ s.t. $a \neq b$.

$a \circ b = ab \neq ba = b \circ a.$

qed

A^* is closed under the operation of concatenation $\Rightarrow$ concatenation is a binary operation of A^* as $\forall w_1, w_2 \in A^*, w_1 \circ w_2 \in A^*$.

Theorem Let A be a finite set. $(A^*, \circ)$ is a monoid with identity element ε.

Proof: Concatenation $\circ$ is an associative binary operation on A^* as we showed above. Moreover, $\forall w \in A^*, \varepsilon \circ w = w \circ \varepsilon = w$, so ε is the identity element of A^*.

qed

Definition: Let A be a finite set. A <u>language</u> over A is a subset of A^*. A language L over A is called a <u>formal language</u> if $\exists$ a finite set of rules of algorithm that generates exactly L, **i.e.** all words that belong to L and no other words.

Theorem: Let A be a finite set.

1. If L_1 and L_2 are languages over $A, L_1 \cup L_2$ is a language over A.
2. If L_1 and L_2 are languages over $A, L_1 \cap L_2$ is a language over A.
3. If L_1 and L_2 are languages over A, the concatenation of $L_1 \wedge L_2$ given by $L_1 \circ L_2 = \{w_1 \circ w_2 \in A^* \mid w_1 \in L_1 \wedge w_2 \in L_2\}$ is a language over A.
4. Let L be a language over A. Define $L^1 = L$ inductively for and $n \geq 1 \quad L^n = L \circ L^{n-1}$. L^n is a language over A. Furthermore, $L^* = \{\varepsilon\} \cup L^1 \cup L^2 \cup L^3 \cup \cdots = \bigcup\limits_{n=0}^{\infty} L^n$ is a language over A.

Proof: By definition, a language over A is a subset of A^*. Therefore, if $L_1 \subseteq A^* \wedge L_2 \subseteq A^*$, then $L_1 \cup L_2 \subseteq A^* \wedge L_1 \cap L_2 \subseteq A^*$. $\forall w_1 \circ w_2 \in L_1 \circ L_2, w_1 \circ w_2 \in A^*$ becomes $w_1 \in A^n$ for some n and $w_2 \in A^m$ for some m so $w_1 \circ w_2 \in A^{m+n} \subseteq A^* = \bigcup\limits_{n=1}^{\infty} A^n$.

Applying the same reasoning inductively, we see that $L \subset A^* \Rightarrow L^* \subseteq A^*$ as $L^m \subseteq A^* \, \forall n \geq 0$.

qed

Remark: This theorem gives us a theoretic way of building languages, but we need a practical way. The practical way of building a language is through the notion of a grammar.

Definition: A (formal) grammar is a set of production rules for strings in a language.
To generate a language we use:

1. A the set, which is the alphabet of the language.
2. A start symbol $\langle s \rangle$
3. A set of production rules.

Example: $A = \{0, 1\}$; start symbol $\langle s \rangle$; 2 production rules given by:

1. $\langle s \rangle \to 0\langle s \rangle 1$
2. $\langle s \rangle \to 01$

Let's see what we generate: via rule 2 $\quad \langle s \rangle \to 01$, so we get $\langle s \rangle \Rightarrow 01$
Via rule 1 $\quad \langle s \rangle \to 0\langle s \rangle 1$, then via rule 2, $0\langle s \rangle 1 \to 0011$. We write the process as $\langle s \rangle \to 0\langle s \rangle 1 \Rightarrow 0011$.
Via rule 1, $\langle s \rangle \to 0\langle s \rangle 1$, then via rule 1 again $0\langle s \rangle 1 \to 00\langle s \rangle 11$, then via rule 2, $00\langle s \rangle 11 \to 000111$.
We got $\langle s \rangle \Rightarrow 0\langle s \rangle 1 \Rightarrow 00\langle s \rangle 11 \Rightarrow 000111$.
The language L we generated thus consists of all strings of the form $0^m 1^m$ (m 0's followed by m 1's) for all $m \geq 1, m \in \mathbb{N}$
We saw 2 types of strings that appeared in this process of generating L:

1. terminals, **i.e.** the elements of A

2. nonterminals, **i.e.** strings that don't consist solely of 0's and 1's such as $\langle s\rangle$, $0\langle s\rangle 1$, $00\langle s\rangle 11$, etc.

The production rules then have the form:

nonterminal $\to$ word over the alphabet V = terminals, non-terminals
$\langle T\rangle \to$ w

In our notation, the set of nonterminals if $V\backslash A$, so $\langle T\rangle \in V\backslash A \wedge w \in V^* = \bigcup_{n=0}^{\infty} V^n$. To the production rule $\langle T\rangle \to w$.
We can associate the ordered pair $(\langle T\rangle, w) \in (V\backslash A) \times V^*$, so the set of production rules, which we will denote by P, is a subset of the Cartesian product $(V\backslash A) \times V^*$.
Grammers come in two flavours:

1. Context-free grammars where we can replace any occurrence of $\langle T\rangle$ by w if $\langle T\rangle \to w$ is one of our production rules.
2. Context-sensitive grammars only certain replacements of $\langle T\rangle$ by w are allowed, which are governed by the syntax of our language L.

The example we have was of a context free grammar. We can now finally define context free grammars.

Definition: A context free grammar $(V, A, \langle s\rangle, P)$ consists of a finite set V, a subset A of V, an element $\langle s\rangle$ of $V\backslash A$, and a finite subset P of the Cartesian product $V\backslash A \times V^*$.

Notation: $(\underbrace{V}_{set\,of\,terminals\,and\,non\,terminals}, \underbrace{A}_{set\,of\,terminals}, \underbrace{\langle s\rangle}_{start\,symbol}, \underbrace{P}_{set\,of\,production\,rules})$

Example: $A = \{0, 1\}$; start symbol $\langle s\rangle$; 3 production rules given by:

1. $\langle s\rangle \to 0\langle s\rangle 1$
2. $\langle s\rangle \to 01$
3. $\langle s\rangle \to 0011$

We notice here that the word 0011 can be generated in 2 ways in this context free grammar:

By rule 3, $\langle s\rangle \to 0011$ so $\langle s\rangle \Rightarrow 0011$
$\vee$
By rule 1, $\langle s\rangle \to 0\langle s\rangle 1$ and by rule 2, $0\langle s\rangle 1 \to 0011$. Therefore, $\langle s\rangle \Rightarrow 0\langle s\rangle 1 \Rightarrow 0011$.

Definition: A grammar is called ambiguous if it generates the same string in more than one way.
Obviously, we prefer to have unambiguous grammars, else we waste computer operations.
Next, we need to spell out how words relate to each other in the production of our language via the grammar:

Definition: Let w' and w'' be words over the alphabet V = terminals, nonterminals. We say that <u>w' directly yields w''</u> if $\exists$ words $u \wedge v$ over the alphabet V and a production rule $\langle T \rangle \rightarrow w$ of the grammar s.t. $w' = u\langle T \rangle \wedge w'' = uwv$, where either or both of the words u and v may be the empty word.

In other words, w' directly yields $w'' \Leftrightarrow \exists$ production rule $\langle T \rangle \rightarrow w$ in the grammar s.t. w'' may be obtained from w' by replacing a simple occurrence of the nonterminal $\langle T \rangle$ within the word w' by the word w.

Notation: w' directly yields w'' is denoted by $w' \Rightarrow w''$

Definition: Let $w' \wedge w''$ be words over the alphabet V. We say that w' yields w'' if either $w' = w''$ or else $\exists$ words $w_0, w_1, \ldots w_n$ over the alphabet V s.t. $w_0 = w', w_n = w'', w_{i-1} \Rightarrow w_i$ for all $i, 1 \leq i \leq n$. In other words, $w_0 \Rightarrow w_1 \Rightarrow w_2 \Rightarrow \cdots \Rightarrow w_n - 1 \Rightarrow w_n$

Notation: w' yields w'' is denotes by $w' \stackrel{*}{\Rightarrow} w''$.

Definition: Let $(V, A, \langle s \rangle, P)$ be a context free grammar. The <u>language</u> generated by this grammar is the subset L or A^* defined by $L = \{w \in A^* \mid \langle s \rangle \stackrel{*}{\Rightarrow} w\}$

In other words, th language L generated by a context free grammar $(V, A, \langle s \rangle, P)$ consists of the set of all finite strings consisting entirely of terminals that may be obtained from the start symbol $\langle s \rangle$ by applying a finite sequence of production rules of the grammar where the application of one production rule causes <u>one and only one nonterminal</u> to be replaced by the string in V^* corresponding of the right hand side of the production rule.

19.1 Phrase Structure Grammars

Definition: A phrase structure grammar $(V, A, \langle s \rangle, P)$ consists of a finite set V, a subset A of V, an element $\langle s \rangle$ of $V \backslash A$, and a finite subset P of $(V^* \backslash A^*) \times V^*$

In a context free grammar, the set of production rules $P \subset (V \backslash A) \times V^*$.

In a phrase structure grammar, $P \subset (V^* \backslash A^*) \times V^*$. In other words, a production rule in a phrase structure grammar $r \rightarrow w$ has a left hand side n that may contain more than one nonterminal. It is required to contain <u>at least one</u> nonterminal.

For example, if $A = \{0, 1\}$ and $\langle s \rangle$ is the start symbol in a phrase grammar grammar, $0\langle s \rangle 0 \langle s \rangle 0 \rightarrow 00010$ would be an acceptable production rule in a phrase structure grammar but not in a context free grammar.

The notions $w' \Rightarrow w''$ (w' directly yields w'') and $w' \stackrel{*}{\Rightarrow} w''$ (w' yields w'') are defined the same way as for context free grammars except that our production rules may, of course, be more general as we saw in the example above.

Definition: Let $(V, A, \langle s \rangle, P)$ be a phrase structure grammar. The language generated by this grammar is the subset L or A^* defined by $L\{w \in A^* \mid \langle s \rangle \stackrel{*}{\Rightarrow} w\}$

Remark: The term phrase structure grammars was introduced by Noam Chomsky.

Definition: A language L generated by a context-free grammar is called a context-free language.

We now want to understand a particularly important subclass of context free languages called regular languages.

20 Regular Languages

Task: Understand when a language is regular and how regular languages are produced. Understand basics of automata theory.

History: The term regular languages was introduced by Stephen Kleene in 1951. A more descriptive name is finite-state languages as we will see that a language is regular $\Leftrightarrow$ it can be recognised by a finite state acceptor, which is a type of finite state machine.
The definition of a regular language is very abstract, though. First, describe what operations the collection of regular languages is closed under: Let A be a finite set, and let A^* be the set of all words over the alphabet A. The regular language over the alphabet A constitutes the smallest collection C of subsets of A^* satisfying that:

1. All finite subsets of A^* belong to C.
2. C is closed under the Kleene star operation (if $M \subseteq A^*$ is inside C, **i.e.** $M \subseteq C$, then $M^* \subseteq C$)
3. C is closed under concatenation (if $M \subseteq A^*, N \subseteq A^*$ satisfy that $M \subseteq C \wedge N \subseteq C$, then $M \circ N \subseteq C$)
4. C is closed under union (if $M \subseteq A^* \wedge N \subseteq A^*$ satisfy that $M \subseteq C \wedge N \subseteq C$, then $M \cup N \subseteq C$)

Definition: Let A be a finite set, and let A^* be the set of words over the alphabet A. A subset L of A^* is called a regular language over the alphabet A if $L = L_m$ for some finite sequence $L_1, L_2, \ldots, L_m$ of subsets of A^* with the property that $\forall i, 1 \leq i \leq m, L_i$ satisfies one of the following:

1. L_i is a finite set
2. $L_i = L_j^*$ for some $j, 1 \leq j \leq i$ (the Kleene star operation applied to one of the previous $L_j's$)
3. $L_i = L_j \circ L_k$ for some j, k such that $1 \leq j, k < i$ (L_i is a concatenation of previous L_j's)

4. $L_i = L_j \cup L_k$ for some j, k such that $1 \leq k, j < i$ (L_i is a union of previous L_j's)

Example 1: Let $A = \{0,1\}$. Let $L = \{0^m 1^n \mid m, n \in \mathbb{N} \quad m \geq 0, n \geq 0\}$
L is a regular language. Note that L consists of all strings of first 0's, then 1's or the empty string ε. $0^m 1^n$ stands for m 0's followed by n 1's, **i.e.** $0^m \circ 1^n$. Let us examine $L' = \{0^m \mid m \in \mathbb{N}, m \geq 0\}$ and $L'' = \{1^n \mid n \in \mathbb{N}, n \geq 0\}$

Q: Can we obtain them via operatons listed among $1 - 4$?

A: Yes! Let $M = \{0\} \qquad M \subseteq A \subseteq A^*$ and $M^* = L^1 = \{0^m \mid m \in \mathbb{N} \quad m \geq 0\}$. Let $N = \{1\} \quad N \subseteq A \subseteq A^*$ and $N^* = L'' = \{1^n \mid n \in \mathbb{N}, n \geq 0\}$. In other words, we can do $L_1 = \{0\}$, $L_2 = \{1\}$, $L_3 = L_1^*$, $l_4 = L_2^*$, $L_5 = L_4 \circ L_5 = L$. Therefore, L is a regular language.

Example 2 Let $A = \{0,1\}$. Let $L = \{0^m 1^m \mid m \in \mathbb{N}, m \geq 1\}$. L is the language we used as an example earlier. It turns out L is <u>NOT</u> regular. This language consists of strings of 0's followed by an equal number of strings of 1's. For a machine to decide that the string $0^m 1^m$ is inside the language is must store the number of 1's, as it examines the number of 0's or vice versa. The number of strings of the type $0^m 1^m$ is not finite, however, so a finite-state machine cannot recognise this language. Heuristically, regular languages correspond to problems that can be solved with finite memory, **i.e.** we only need to remember one of finitely many things. By contrast, nonregular languages correspond to problems that cannot be solved with finite memory.

Theorem: The collection of regular languages C is also closed under the following two operations:

1. Intersection, **i.e.** if L', L'' are regular languages (**i.e.** $L' \cup L'' \in C$) then their intersection $L' \cap L''$ is a regular language.
2. Complement, **i.e.** if L is a regular language (**i.e.** $L \in C$), then $A * \backslash L$ is a regular language ($A^* \backslash L \in C$).

Remark: These two properties did not come into the definition of a regular language, but they are true and often quite useful.

20.1 Finite State Acceptors and Automata Theory

Definition: An <u>automation</u> is a mathematical model of a computing device. Plural of automation is <u>automata</u>.

Basic idea: Reason about computability without having to worry about the complexity of actual implementation.
It is most reasonable to consider at the beginning just finite states automata, **i.e.** machines with a finite number of internal states. The data

entered discretely, and each datum causes the machine to either remain in the same internal state or else make the transition to some other state determined solely by 2 pieces of information:

1. The current state
2. The input datum

In other words, if S is the finite set of all possible states of our finite state machine, then the transition mapping t that tells us how the internal state of the machine changes on inputting a datum will depend on the current state $s \in S$ and the imput datum a, **i.e.** the machine will enter a (potentially) new state $s' = t(s, a)$.

Want to use finite state machines to recognise languages over some alphabet A. Let L be our language.
Since our finite state machine accepts (**i.e.** returns yes to) w if $w \in L$,

Input	Output
Word $w = a_1 \dots a_n, a_i \in A \forall i$	Yes if $w \in L$
	No if $w \notin L$

we call our machine a finite state acceptor. We want to give a rigorous definition of a finite state acceptor. To check $w = a_1 \dots a_n$, we input each a_i starting with a_1 and trace how the internal state of the machine changes. S is our set of states of the machine (a finite set). The transition mapping t takes the pair(s, a) and returns the new state $s' = t(s, a)$ (where $s \in S \wedge a \in A$) that the machine has reached so $t : S \times A \to S$.
Some elements and subets of S are important to understand:

1. The initial state $i \in S$ where the machine starts
2. The subset $F \subseteq S$ of finishing states

It turns out that knowing S, F, i, t, A satisfies a finite state acceptor completely.

Definition: A finite state acceptor (S, A, i, t, F) consists of a finite set S of states, a finite set A that is the input alphabet, a starting state $i \in S$, a transition mapping $t : S \times A \to S$, and a set F of finishing states, where $F \subseteq S$.

Definition: Let (S, A, i, t, F) be a finite state acceptor, and let A^* denote the set of words over the input alphabet A. A word $a_1, a_2 \dots a_n$ of length n over the alphabet A is said to be recognised or accepted by the finite state acceptor if $\exists s_0, s_1, \dots, s_n \in S$ states s.t. $s_0 = i$ (the initial state), $s_n \in F$, and $s_i = t(s_{i-1}, a_i) \forall i \quad 1 \leq i \leq n$.

Definition: Let (S, A, i, t, F) be a finite state acceptor. A language L over the alphabet A is said to be recognised or accepted by the finite state

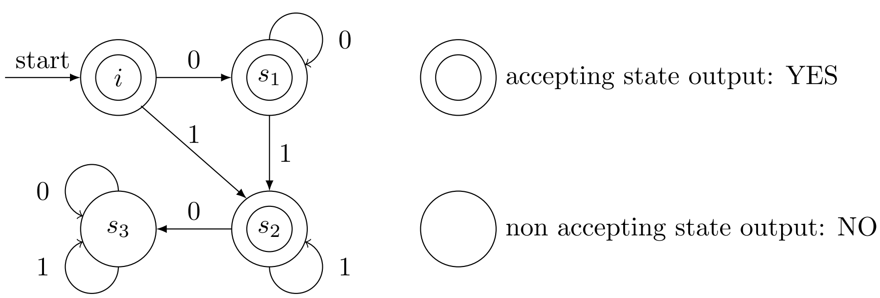
acceptor.
In the definition of a finite state acceptor, t is the transition mapping, which may or may not be a function (hence the careful terminology). This is because finite state acceptors come in 2 flavours:

1. Deterministic: every state has exactly one transition for each possible input, **i.e.** $\forall (s,a) \in S \times A \; \exists! \; t(s,a) \in S$. In other words, the transition mapping is a function.
2. Non-deterministic: an input can lead to one, more than one or no transition for a given state. Some $(s,a) \in S \times A$ might be assigned to more than one element of S, **i.e.** the transition mapping is not a function.

Surprisingly $\exists$ algorithm that transforms a non deterministic (thought more complex one) using the powerset construction.
As a result, we have the following theorem:

Theorem: A language L over som alphabet A is a regular language $\Leftrightarrow$ L is recognised by a deterministic finite state acceptor with input alphabet $A \Leftrightarrow L$ is recognised by a nondeterministic finite state acceptor with input alphabet A.

Example: Build a deterministic finite state acceptor for the regular language $L = \{0^m 1^n \mid m, n \in \mathbb{N}, m \geq 0, n \geq 0\}$



Accepting states in this examples: i, s_1, s_2
Non accepting states: s_3
Start states: i
Here $S = \{i, s_1, s_2, s_3\}$ $\quad F = \{i, s_1, s_2\}$ $\quad A = \{0,1\}$ $\quad t : S \times A \to S$ $\quad t(i,0) = s_1$ $\quad t(i,1) = s_2$ $\quad t(s_1,0) = s_1$ $\quad t(s_1,1) = s_2$ $\quad t(s_2,0) = s_3$ $\quad t(s_2,1) = s_2$
Let's process some strings:

String	ε (empty string)
State (i)	i
Output	YES

String	0	0	1	1	1
State i	s_1	s_1	s_2	s_2	s_2
Output	YES				

String	1	1
State i	s_2	s_2
Output	YES	

String	1
State i	s_2
Output	YES

String	0	1	0	1
State i	s_1	s_2	s_3	s_3
Output	NO			

Now that we really understand what a finite state acceptor is, we can develop a criterion for recognised regular languages called the Myhill-Nerode theorem based on an equivalence relation we can set up on words in our language over the alphabet A.

Definition: Let $x, y \in L$, a language over the alphabet A. We call x and y equivalent over L denoted by $x \equiv_L y$ if $\forall w \in A^*, xw \in L \Leftrightarrow yw \in L$.

Note: xw means the concatenation $x \circ w$, and yw is the concatenation $y \circ w$.

Idea: If $x \equiv_L y$, then x and y place our finite state acceptor into the same state s.

Notation: Let L/N be the set of equivalence classes determined by the equivalence relation $\equiv_L$.

The Myhill-Nerode Theorem: Let L be a language over the alphabet A. If the set L/N of equivalence classes in L is infinite, then L is not a regular language.

Sketch of Proof: All element of one equivalence class in L/N place our automation into the same state s. Elements of distinct equivalence classes place the automation into distinct state, **i.e.** if $[x], [y] \in L/N$ and $[x] \neq [y]$, then all elements of $[x]$ place the automation into some state s, while all elements of $[y]$ place the automation into some state s', with $s \neq s' \Rightarrow$ an automation that can recognise L has as many states at the number of equivalence classes in L/N, but L/N is NOT finite $\Rightarrow$ L cannot be recognised by a finite state automation $\Rightarrow$ L is not regular by the theorem above.

qed

21 Regular Grammars

Task: Understand what is the form of the production rules of a grammar that generates a regular language.

Recall that a context-free grammar is given by $(V, A, \langle s \rangle, P)$ where every production rule $\langle T \rangle \to w$ in P causes one and only one nonterminal to be replaced by a string in V^*.

Definition: A context-free grammar $(V, A, \langle s \rangle, P)$ is called a regular grammar is every production rule in P is of one of the three forms:

(i) $\langle A\rangle \to \mathrm{b}\langle B\rangle$

(ii) $\langle A\rangle \to \mathrm{b}$

(iii) $\langle A\rangle \to \varepsilon$

where $\langle A\rangle$ and $\langle B\rangle$ are nonterminals, b is a terminal, and ε is the empty word. A regular grammar is said to be in normal form if all its production rules are of types (i) and (iii).

Remark: In the literature, you often see this definition labelled left-regular grammar as opposed to right-regular grammar, where the production rules of types 1 have the form $\langle A\rangle \to \langle B\rangle\mathrm{b}$, (**i.e.** the terminal is one the right of the nonterminal). This distinction is not really important as long as we stick to one type throughout since both left regular grammars and right regular grammars generate regular languages.

Lemma: Any language generated by a regular grammar may be generated by a regular grammar in normal form.

Proof: Let $\langle A\rangle \to$b be a rule of type (ii). Replace it by two rules: $\langle A\rangle \to$b$\langle F\rangle$ and $\langle F\rangle \to \varepsilon$, where $\langle F\rangle$ s a new nonterminal. Add $\langle F\rangle$ to the set V. We do the same for every rule of type (ii) obtaining a bigger set V, but now our production rules are only of type (i) and (iii) and we are generating the same language.

qed

Example: Recall the regular language $L = \{0^m 1^n \mid m, n \in \mathbb{N}, m \geq 0, n \geq 0\}$. We can generate it from the regular grammar in normal gorm given by production rules:

1. $\langle s\rangle \to 0\langle A\rangle$
2. $\langle A\rangle \to 0\langle A\rangle$
3. $\langle A\rangle \to \varepsilon$
4. $\langle s\rangle \to \varepsilon$
5. $\langle A\rangle \to 1\langle B\rangle$
6. $\langle B\rangle \to 1\langle B\rangle$
7. $\langle s\rangle \to 1\langle B\rangle$
8. $\langle B\rangle \to \varepsilon$

Rules (1), (2), (5), (6), (7) are of type (i), where rules (3), (4) and (8) are of types (iii).
(1) and (3) gives 0. (1), (2) applied $m-1$ times and (3) gives 0^m for $m \geq 2$.
(7) and (8) give 1. (7), (6) applied $n-1$ times and (8) give 1^n for $n \geq 2$.
(1), (5) and (8) give 01. (1), (5), (6) applied $n-1$ times and (8) gives 01^n for $n \geq 2$.

(1), (2) applied $m-1$ times, (5) and (8) gives 0^m1 for $m \geq 2$.
(1), (2) applied $m-1$ times, (5), (6) applied $n-1$ times, and (8) gives 0^m1^n for $m \geq 2, n \geq 2$.
Rule (4) gives the empty word $\varepsilon = 0^01^0$.

Q: Why does a regular grammar yield a regular language, **i.e.** one recognised by a finite state acceptor?

A: Not obvious from the definition, <u>but</u> we can construct the finite state acceptor from the regular grammar as follows: our regular grammar is given by $(V, A, \langle s\rangle, P)$. <u>Want</u> a finite state acceptor (S, A, i, t, F). Immediately, we see the alphabet A is the same and $i = \langle s\rangle$. This gives us the idea of associating to every nonterminal symbol in $B\backslash A$ a state. $\langle s\rangle \in V\backslash A$, so that's good. Next we ask:

Q: Is it sufficient for $S = V\backslash A$?

A: No! Our set F of finishing/accepting states should be nonempty. So we add an element $\{f\}$ to $V\backslash A$, where our acceptor will end up when a word in our language. Thus, $S = (V\backslash A) \cup \{f\}$ and $F\{f\}$. $F \subseteq S$ as needed.

Q: How do we define t?

A: Use the production rules in P! For every rule of type (i), which is of the form $\langle A\rangle \to \mathrm{b}\langle B\rangle$ set $t(\langle A\rangle,\mathrm{b}) = \langle B\rangle$. This works out well because our nonterminals $\langle A\rangle$ and $\langle B\rangle$ are states of the acceptor and the terminal $b \in A$ for t takes an element of $S \times A$ to an element of S as needed. Now look at production rules of type (ii), $\langle A\rangle \to b$ and of types (iii), $\langle A\rangle \to \varepsilon$. Those are applied when we <u>finish</u> constructing a word w in our language L, **i.e.** at the very last step, so our acceptor should end up in the finishing state f whenever a production rule of type (ii) or (iii) is applied. Write a production rule of type (ii) or (iii) as $\langle A\rangle \to w$, then we can set $t(\langle A\rangle, w) = f$. We have finished constructing t as well. Technically, $t : S \times (A \cup \{\varepsilon\}) \to S$ instead of $t : S \times A \to S$, but we can easily fix the transition function t by combining the last two transitions for each accepted word.

Remark: The same general principals as we used above allow us to go from a finite state acceptor to a regular grammar. This gives us the following theorem:

Theorem: A language L is regular $\Leftrightarrow$ L is recognised by a finite state acceptor $\Leftrightarrow$ L is generated by a regular grammar.

21.1 Applications of Formal Languages and Grammars as well as Automata Theory

1. Compiler architecture uses context-free grammars

2. Parsers - recognise if commands comply with the syntax of a language
3. Pattern matching and data mining - guess the language from a given set of words (applied in CS, etc)
4. Natural language processing - example in David Wilkins' notes pp40-44
5. Checking proofs by computers/automatic theorem proving - simpler example of this kind in David Wilkins' notes pp45-57 that pertains to propositional logic
6. Theory of regular expressions (related to the definition of a regular language that we gave) enables
   (a) grep/awk/sed in Unix
   (b) More efficient coding (avoiding unnecessary detoirs in your code)
7. Biology - John Conway's game of life is a cellular automation
8. Modelling of AI characters in games uses the finite state automation idea. Our character can choose among different behaviours based on stimuli - like a finite state automation reacting to input
9. Strategy and tactics in games - teach the opposition to recognise certain patterns, then suddenly change them to gain an advantage and score - used in football, fencing, etc.
10. Learning a sport/a numerical instrument/a new field or subject - split the information into blocks and learn how to combine them into meaningful patterns - uses notions from context-sensitive grammars.
11. Finite state automata and probability - chaos theory, financial mathematics.

etc...

22 Graph Theory

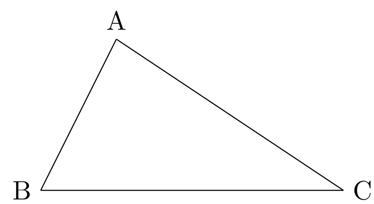
Task: Introduce terminology related to graphs; understand different types of graphs; learn how to put together arguments involving graphs.
An undirected graph consists of:

1. A finite set of points V called vertices
2. A finite set E of edges joining two distinct vertices of the graph.

Understand the meaning of an edge better: Let V be the set of vertices. Consider $P(V)$, the power set of V. Let $V_2 \subseteq P(V)$ consist of all subsets of V containing exactly 2 points, **i.e.** $V_2 = \{A \in P(V) \mid \#(A) = 2\}$
Identify each element in V_2 with the edge joining the two points. In other words, if $\{a, b\} \in V_2$, then we can let ab be the edge corresponding to $\{1, b\}$.

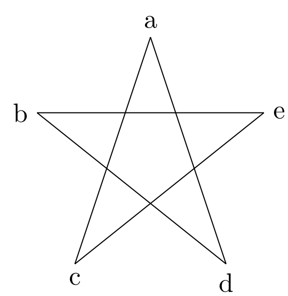
Examples:

1. A traingle is an undirected graph.
   $V = \{A, B, C\}$



   3 possible 2 element subsets of V: $\{A, B\} \to AB$
   $\{A, C\} \to AC$
   $\{B, C\} \to BC$
   $E = \{AB, AC, BC\}$

2. A pentagram is an example of an undirected graph.
   $V = a, b, c, d, e$



   $E = \{ac, ad, be, ce, bd\}$

Convention: The set of vertices cannot be empty, **i.e.** $V \neq 0$.

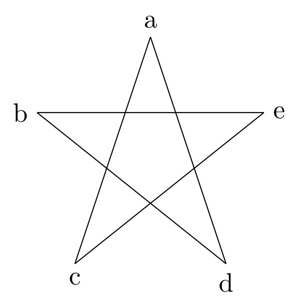
Q: If $V \neq \emptyset$, what is the simplest possible undirected graph?

A: A graph consisting of a single point, **i.e.** with one vertex and two edges.

Definition: A graph is called trivial if it consists of one vertex and zero edges.
Next, study how vertices and edges relate to each other.

Definition: If v is a vertex of some graph, if e is an edge of that graph, and it $e = vv'$ for v' another vertex, then the vertex v is called incident to the edge e and the edge e is called incident to the vertex v.

Example:
b is incident to edges be and bd
be is incident to vertices b and e

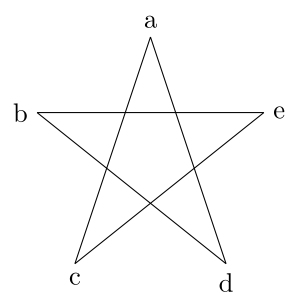


Definition: Let (V, E) be an undirected graph. Two vertices $A, B \in V \quad A \neq B$ are called adjacent if $\exists$ edge $AB \in E$.
We represent the incidence relations among the vertices V and edges E of an undirected graph via:

1. An incidence table
2. An incidence matrix

Legend:
1 an incidence relation holds
0 an incidene relation does not hold

From the pentagram:
$V = \{a, b, c, d, e\}$
$E = \{ac, ad, be, bd, ce\}$



The indidence table is:

	ac	ad	be	bd	ce
a	1	1	0	0	0
b	0	0	1	1	0
c	1	0	0	0	1
d	0	1	0	1	0
e	0	0	1	0	1

Correspondingly, the incidence matrix is:

$$\begin{pmatrix} 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 \end{pmatrix}$$

Note that for the incidence matrix to make sense, we need to know that vertices were considered in the order $\{a, b, c, d, e\}$ and edges in the order $\{ac, ad, be, bd, ce\}$. If we shuffle either set, the incidence matrix chances. With this in mind, we can now rigorously define the incidence matrix:

Definition: Let (V, E) be an undirected graph with m vertices and n edges. Let vertices be ordered as $v_1, v_2, \ldots, v_m$, and let the edges be ordered $e_1, e_2, \ldots, e_n$. The incidence matrix for such a graph is given by $\begin{pmatrix} a_{11} & a_{12} & \ldots & a_{1n} \\ a_{21} & a_{22} & \ldots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \ldots & a_{mn} \end{pmatrix}$, where the entry a_{ij} in row i and column j has the value 1 if the i^{th} vertex is incident to the j^{th} edge and has value 0 otherwise.

Similarly, we can define the adjacency table and the adjacency matrix of a graph:

Definition: Let (V, E) be an undirected graph with m vertices, and let these vertices be ordered as $v_1, v_2, \ldots, v_m$. The adjacency matrix for this graph is given by $\begin{pmatrix} b_{11} & b_{12} & \dots & b_{1m} \\ b_{21} & b_{22} & \dots & b_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ b_{m1} & b_{m2} & \dots & b_{mm} \end{pmatrix}$ where $b_{ij} = 1$ if v_i and v_j are adjacent to each other and $b_{ij} = 0$ if v_i and v_j are not adjacent to each other.

Remark: "Being adjacent to" is a symmetric relation on he set of vertices V, so the adjacency matrix is symmetric, **i.e.** $b_{ij} = b_{ji} \quad \forall i, j \quad 1 \leq i, j \leq m$. It is not reflexive so all the entries on the diagonal are zero.

22.1 Complete graphs

Definition: A graph (V, E) is called complete if $\forall v, v' \in V$ s.t. $v \neq v'$, the edge $vv' \in E$. In other words, a complete grapg has the highest number of edges possible given its number of vertices.

Examples:

1. The triangle is a complete graph.
2. The pentagram is not a complete graph.

Notation: A complete graph with n vertices is denoted by K_n.

Q: How does the adjacency matrix of a complete graph look like?

A: All entrices are 1 except on the diagonal, where they are all two.

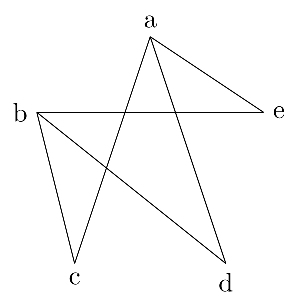
Definition: A graph (V, E) is called bipartite is $\exists$ subsets V_1 and V_2 s.t.

1. $V_1 \cup V_2 = V$
2. $V_1 \cap V_2 = \emptyset$
3. Every edge in E is of the form vw with $v \in V_1$ and $w \in V_2$.

A bipartite graph is called a complete bipartite graph if $\forall v \in V_1 \quad \forall w \in V_2 \quad \exists vw \in E$.

Notation: A complete bipartite graph where the set V_1 has p elements and the set V_2 has q elements is denoted by $K_{p,q}$.

Example:
$V_1 = \{a, b\}$
$V_2 = \{c, d, e\}$
$V = \{a, b, c, d, e\}$
$E = \{ac, ad, ae, bc, bd, be\}$
is a complete bipartite graph.



Next, relate graph to each other via functions with special properties.

23 Isomorphism of Graphs

Definition: An isomorphism between two graphs (V, E) and (V', E') is a bijective function $\varphi : V \to V'$ satisfying that $\forall a, b \in V$ with $a \neq b$ the edge $ab \in E \Leftrightarrow$ the edge $\varphi(a)\varphi(b) \in E'$.

Recall: A function $\varphi : V \to V'$ is bijective $\Leftrightarrow$ it has an inverse $\varphi^{-1} : V' \to V'$. The bijection $\varphi V \to V'$ that gives the isomorphism between (V, E) and $(V'E')$ thus sets up the following:

1. A 1-1 correspondence of the vertices V of (V, E) with the vertices V' of $(V', E') \to$ comes from $\varphi : V \to V'$ being bijective.
2. A 1-1 correspondence of the edges E of (V, E) with the edges E' of $(V', E') \to$ comes from the additional property in the definition of an isomorphism that $\forall a, b \in V$ with $a \neq b, ab \in E \Leftrightarrow \varphi(a)\varphi(b) \in E'$.

Remark: Just like an isomorphism of groups discussed earlier in the course, an ismorphism of graphs means (V, E) and (V', E') have the same "iso" from "morphé". "Being isomorphic" is an equivalence relations, so we get classes of graphs that have the same form as our equivalence classes.

Definition: If there exists an isomorphism $\varphi : V \to V'$ between two graphs (V, E) and (V', E'), then (V, E) and (V', E') are called isomorphic.

24 Subgraphs

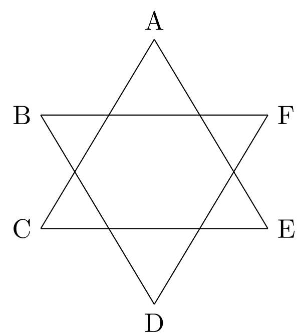
Task: Understand subobjects of a graph.

Definition: Let (V, E) and (V', E') be graphs. The graph (V', E') is called a subgraph of (V, E) if $V' \subseteq V$ and $E' \subseteq E$, **i.e.** if (V', E') consists of a subset V' of the vertices of (V, E) and a subset E' of edges (V, E) between vertices in V'.

Example: Star of David on the flag of Israel

$V = \{a, b, c, d, e, f\}$

$E = \{ac, ce, ae, bf, fd, bd\}$



2 triangle subgraphs of the star of David:

$V' = \{a, c, e\}$ $\quad E' = \{ac, ce, ae\}$

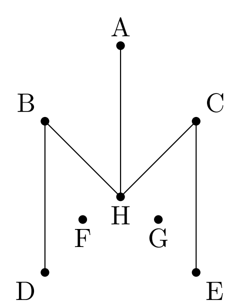
$V'' = \{b, f, d\}$ $\quad E'' = \{bf, fd, bd\}$

25 Vertex Degrees

Task: Use numbers to understand incidence relationships.

Definition: Let (V, E) be a graph. The degree $\deg v$ of a vertex $v \in V$ is defined as the number of edges of the graph that are incidence to v, **i.e.** the number of edge with v as one of their endpoints.

Example:



def $f = \deg g = 0$

$\deg d = \deg e = \deg a = 1$

$\deg b = \deg c = 2$

deg $h = 3$

Definition: A vertex of degree 0 is called an isolated vertex.

Definition: A vertex of degree 1 is called a pendent vertex.

Theorem: Let (V, E) be a graph. Then $\sum_{v \in V}$deg $v = 2\#(E)$, where $\sum_{v \in V}$deg v is the sum of the degrees of all the vertices of the graph, and $\#(E)$ is the number of edges of the graph.

Proof: $\sum_{v \in V}$deg v is the sum of all the entries in the adjacency matrix. Every edge $vv' \in E$ contributes 2 to the sum $\sum_{v \in V}$deg v, 1 for the vertex v and 1 for the vertex v' $\Rightarrow$ each edge must be counted twice, so $\sum_{v \in V}$deg $v = 2\#(E)$.

qed

Corollary: $\sum_{v \in V}$deg v is an even integer.

Proof: Since $\sum_{v \in V}$deg $v = 2\#(E)$, and $\#(E) \in \mathbb{N}$, the result follows.

qed

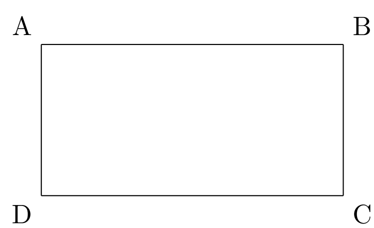
Corollary: In any graph, the number of vertices of odd degrees must be even.

Proof: Assume not, then $\sum_{v \in V}$deg v is an odd integer as $odd + even = odd \Rightarrow\Leftarrow$ to the previous corollary.

qed

Definition: A graph is called k-regular for some non-negative integer k if every vertex of the graph has degree equal to k.

Example: A rectangle is 2-regular.
deg $a =$ deg $b =$ deg $c =$ deg $d = 2$.



Definition: A graph (V, E) is called regular is $\exists k \in \mathbb{N}$ s.t. (V, E) is k-regular.

Corollary: Let (V, E) be a k-regular graph. Then $k\#(V) = 2\#(E)$ where $\#(V)$ is the number of vertices and $\#(E)$ is the number of edges.

Proof: By the theorem, $\sum_{v\in V}$deg $v = 2\#(E)$, but (V,E) is k-regular $\Rightarrow$ deg $v = k \forall v \in V$. Therefore $\sum_{v\in V}$deg $v = \#(V).k = 2\#(E)$.

qed

Example: Consider a complete graph (V,E) with n vertices. (V,E) is $(n-1)$-regular because every vertex is adjacent to all the remaining $(n-1)$ vertices.

Corollary: A complete bipartite graph $k_{p,q}$ is regular $\Leftrightarrow p = q$

Proof: Recall that $V = V_1 \cup V_2 \qquad V_1 \cap V_2 = \emptyset$ for a bipartite graph.
$\Leftarrow$ is $p = q, \forall v \in V_1$ satisfies that deg $v = p = q$ and $\forall v \in V_2$ satisfies that def $v = p = q$ since the graph is complete $\Rightarrow (V,E)$ is p-regular.
$\Rightarrow$ (V,E) is regular $\Rightarrow \forall v \in V_1$ and $\forall v' \in V_2$, deg $v =$ deg v', but (V,E) is complete $\Rightarrow v$ is adjacent to all vertices in V_2, **i.e.** def $v = \#(V_1)$ and v' is adjacent to all vertices in V_1, **i.e.** deg $v' = \#(V_2) \rightarrow \#(V_1) = \#(V_2)$

26 Walks, trails and paths

Task: Make rigorous the notion of traversing parts of a graph in order to understand its structure better.

Definition: Let (V,E) be a graph. A walk $v_0v_1v_2\dots v_n$ of length n in the graph from vertex a to vertex b is determined by a finite sequence $v_0, v_1, v_1, \dots, v_n$ of vertices of the graph s.t. $v_0 = a, v_n = b$ and $v_{i-1}v_i$ is an edge of the graph for $i = 1, 2, \dots, n$.

Definition: A walk $v_0v_1v_2\dots v_n$ in a graph is said to traverse the edges $v_{i-1}v_i$ and to pass through the vertices $v_0, v_1, \dots, v_n$. Length of walk = # of edges traversed $\rightarrow$ the smallest possible number is two edges. As a result, we have the following definition:

Definition: A walk that consists of a single vertex $v \in V$ and has length two is called trivial.

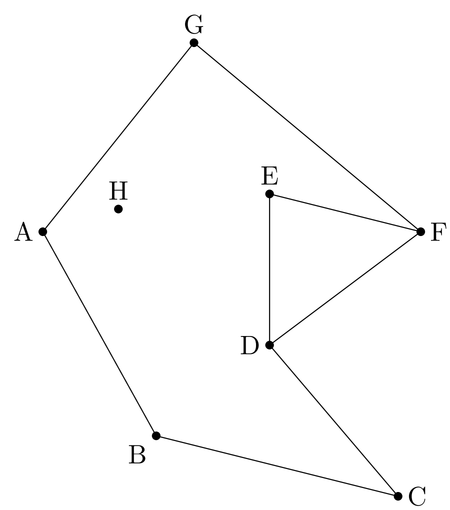
Definition: Let (V,E) be a graph. A trail $v_0v_1v_2\dots v_n$ of length n in the graph from some vertex a to some vertex b is a walk of length n from a to b with the property that edges $v_{i-1}v_i$ are distinct for $i - 1, 2, \dots, n$. In other words, a trail is a walk in the graph, which traverses edges of the graph at most once.

Definition: Let (V,E) be a graph. A path $v_0v_1v_2\dots v_n$ of length n in the graph from some vertex a to some vertex b is a walk of length n from a to b with the property that vertices $v_0, v_1 \dots v_n$ are distinct. In other words, a path in a graph is a walk in the graph, which passes through the vertices of the graph at most once.

Definition: A walk, trail or path is a graph is called trivial if it is a walk of length two consist of a single vertex $v \in V$; otherwise, the walk, trail, or path is called non-trivial.

Example:

1. h is a trivial walk/trail/path
2. $defd$ is a trail, but not a path because we pass through the vertex d twice.
3. def is a path
4. $gfdefdc$ is a walk but not a trail of a path

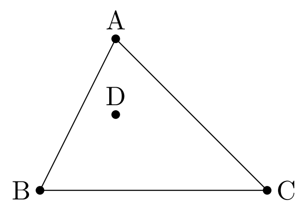


27 Connected Graphs

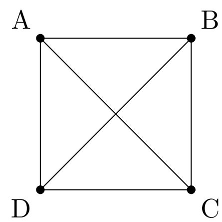
Task: Use the ideas above related to traversing parts of a graph in order to define a particularly important category of graphs.

Definition: An undirected graph (V, E) is called connected if $\forall u, v \in V$ vertices, $\exists$ path in the graph from u to v.

Examples: 1. Is not connected as d is not connected to any other vertex.



2. Is connected. $\exists$ path between any two of the vertices.



Theorem: Let (V, E) be a undirected graph, and let $u, v \in V.\exists$ path between u and v in the graph $\Leftrightarrow \exists$ walk in the graph between u and v.

Proof: $\Rightarrow$ trivial: A path is a walk.
$\Leftarrow \exists$ walk between u and v. Choose the walk of least length between u and v, (**i.e.** $\nexists$ a walk of lower length than this one) and prove it is a path. Let this walk be $a_0a_1 \dots a_n$ with $a_0 = u$ and $a_n = v$. Assume $\exists j, k$ with $a \leq j, k \leq n$ s.t. $j < k$ and $a_j = a_k$, but then $a_0a_1 \dots a_ja_k + 1 \dots a_n$ would be a walk from u to v of strictly smaller length than $a_0a_1 \dots a_n$. $\Rightarrow\Leftarrow$ as we chose $a_0a_1 \dots a_n$ to be of minimal length $\Rightarrow a_i \neq a_j \forall i, j$ s.t. $0 \leq i, j \leq n \Rightarrow a_0a_1 \dots a_n$ is a path between u and v.

qed

Corollary: An undirected grapg (V, E) is connected $\Leftrightarrow \forall u, v \in V \exists$ walk in the graph between u and v.

28 Components of a graph

Task: Divide a graph into subgraphs that are isolated from each other.

Let (V, E) be an undirected graph. We define a relation $\sim$ on the set of vertices V, where $a, b \in V$ satisfy $a \sim b$ iff $\exists$ walk in the graph from a to b.

Lemma: Let (V, E) be an undirected graph. The relation $a \sim b$ or $a, b \in V$, which holds iff $\exists$ walk in the graph between a and b is an equivalence relation.

Proof: We must show $\sim$ is reflexive, symmetric, and transitive.

Reflexive: $\forall v \in V, v \sim v$ since the trivial walk is a walk from v to itself.

Symmetric: If $a \sim b$ for $a, b \in V$, then $\exists$ walk $v_0 v_1 \ldots v_n$ where $v_0 = a$ and $v_n = b$. This walk can be reversed to $v_n v_{n-1} \ldots v_1 v_0$, which now goes from $v_n = b$ to $v_0 = a$. Therefore, $b \sim b$ as needed.

Transitive: If $a \sim b$ and $b \sim c$, for $a, b, c \in V$, there $\exists$ walk $av_1v_2 \ldots v_{n-1}b$ from a to b and $\exists$ walk $bw_1w_2 \ldots w_{m-1}c$ from b to c. We put these two walks together (concatenate them) to yield the walk $av_1v_2 \ldots v_{n-1}bw_1w_2 \ldots w_{m-1}c$ from a to c. Therefore $a \sim c$.

qed

The equivalence relation $\sim$ on V partitions it into dsjoint subsets $v_1, v_2, \ldots v_p$, where

1. $v_1 \cup v_2 \cup \cdots \cup v_p = V$
2. $v_i \cap v_j = \emptyset$ if $i \neq j$
3. Two vertices $a, b \in v_i \Leftrightarrow a \sim b$, **i.e.** $\exists$ walk in (V, E) from a to b

Note that an edge is a walk of length 1, so if $a, b \in V$ satisfy that $\exists ab \in E$, then a and b belong to the same v_i. As a result, we can partition the set of edges as follows:

$$E_i\{ab \in E \mid a, b \in v_i\}$$

Clearly, $E_1 \cup E_2 \cup \cdots \cup E_p = E \wedge E_i \cap E_j = \emptyset$ is $i \neq j$. Furthermore, $(V_1, E_1), (V_2, E_2), \ldots, (V_p, E_p)$ are subgraphs of (V, E), and these subgraphs are disjoint since $v_i \cap v_j = \emptyset$ and $E_i \cap E_j = \emptyset$ is $i \neq j$. The subgraphs (V_i, E_i) are called the <u>components</u> (or <u>connected components</u>) of the graph (V, E).

Lemma: The vertices and edges of any walk in an undirected graph are all contained in a single component of that graph.

Proof: Let $v_0 v_1 \ldots v_n$ be a walk in a graph (V, E), then $v_0 v_1 \ldots v_r$ is a walk in $(V, E) \forall r \quad 1 \leq r \leq n \Rightarrow v_0 \sim v_r \forall r \quad 1 \leq r \leq n \Rightarrow v_r$ belongs to the same component of the graph as v_0. The same is true for all the edges $v_{i-1}v_i$ for $1 \leq i \leq n$

qed

Lemma: Each component of an undirected graph is connected.

Proof: Let (V, E) be a graph and let (V_1, E_i) be any component of (V, E).$\forall u, v \in V_i$, by definition $\exists$ walk in (V, E) between u and v. By previous lemma, however, all vertices and edges of this walk are in $(V_i, E_i) \Rightarrow$ the walk between u and v is a walk in (V_i, E_i), but this assertion is true $\forall u, v \in V_i \Rightarrow (V_i, E_i)$ is connected.

qed

28.1 Moral of the story

Any undirected graph can be represented as a disjoint union of connected subgraphs, namely its components $\Rightarrow$ the study of undirected graphs reduces to the study of connected graphs, as components don't share either vertices or edges.

29 Circuits

Task: Use closed walks to understand the structure of graphs better.

Definition: Let (V, E) be a graph. A walk $v_0 v_1 \dots v_n$ in (V, E) is called closed if $v_0 = v_n$, **i.e.** if it starts and edges at the same vertex.

Definition: Let (V, E) be a graph. A circuit is a nontrivial closed trail in (V, E), **i.e.** a closed walk with no repeated edges passing through at least two vertices.

Definition: A circuit is called simple if the vertices $v_0, v_1, v_2, \dots v_{n-1}$ are distinct.

NB: This is the strangest condition regarding vertices tha we can impose ince $v_0 = v_n$.

Alternative terminology: Some authors use cycle to denote a simple circuit, which for ohers cycle denotes a circuit regardless of whether it is simple or not.

Q: When does a graph have simple circuits?

A: We can give 2 criteria for the existence of simple circuits:

1. Every vertex has degres ≥ 2.
2. $\forall u, v \in V$ s.t. $\exists$ 2 distinct paths from u to v.

Theorem: If (V, E) has no isolated or pendant vertices, **i.e.** $\forall v \in V$ deg $v \geq 2$, then (V, E) contains at least one simple circuit.

Proof: Consider all paths (V, E). The maximum length of a path is $\#(V) - 1$ since a path of length p passed through $p + 1$ vertices. Take a path $v_0 v_1 \dots v_m$ is (V, E) of maximum length, **i.e.** any other path in (V, E) has length $\leq m =$ length of $v_0 v_1 \dots v_m$. Now consider the vertex v_m. deg $v_m \geq 2$ by assumption. We know v_{m-1} is adjacent to v_m since the edge $v_{m-1} v_m$ is part of the path $v_0 v_1 \dots v_m$, but deg $v_m \geq 2$ means $\exists w \in V$ s.t. $w w_m \in E$. If $w \neq v_i$ for $0 \leq i \leq m - 2$, then $v_0 v_1 \dots v_m w$ is a path in (V, E) longer than $v_0 v_1 \dots v_m \Rightarrow\Leftarrow$ to the act that $v_0 v_1 \dots v_m$ was chosen of maximal length. Therefore, $w = v_i$ for some $0 \leq i \leq m - 2$, but then $v_i v_{i+1} \dots v_m v_i$ is a simple circuit in the graph.

qed

Theorem: Let (V, E) be an undirected graph and let $u, v \in V$ be vertices s.t. $u \neq v$ and $\exists$ at least two distinct paths in (V, E) from u to v. Then the graph contains at least one simple circuit.

Proof: Let $a_0 a_1 a_2 \dots a_m$ and $b_0 b_1 \dots b_n$ be the two distinct paths in the graph between u and v, **i.e.** $a_0 = b_0 = u$ and $a_m = b_m = v$. Let $m \leq n$. Since the paths are distinct $\exists i$ with $0 \leq i \leq m$ s.t. $a_1 \neq b_i$. Choose the smallest i for which $a_i \neq b_i$, **i.e.** $a_0 = b_0, a_1 = b_1, \dots, a_{i-1} = b_{i-1}$, but $a_i \neq b_i$. We have thus eliminated the redundencies at the beginning of the paths. We now need to eliminate redundencies at the other end of the paths. We know $a_m = b_m$ so $a_j \in \{b_m \mid i-1 < k \leq n\}$ is certainly satisfied for $j = m$, but we want to choose the <u>smallest</u> index for which this condition is satisfied. Let this index be $p \Rightarrow a_p \in \{b_k \mid i-1 < k \leq n\}$, **i.e.** $a_p = b_s$ for some s s.t. $i-1 < s \leq n$. Since p is the smallest index satisfying $a_p \in \{b_k \mid i-1 < k \leq n\}, a_i, a_{i+1}, \dots, a_{p-1} \notin \{b_k \mid i-1 < k \leq n\} \Rightarrow \underbrace{a_{i-1} a_i \dots a_p}_{\textit{indices running in increasing order}} (= b_s) \underbrace{b_{s-1} \dots b_i}_{\textit{indices running in decreasing order}} a_{i-1} (= b_{i-1})$ is a simple circuit in $(V, E) \Rightarrow (V, E)$ has at least one simple circuit.

qed

30 Eulerian trails and circuits

Task: Look at trails and circuits that traverse every edge of a graph. Derive criteria when such trails and circuits exist.

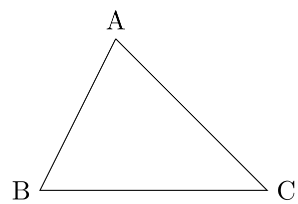
Definition: An <u>Eulerian trail</u> in a graph is a trail that traverses every edge of that graph. In other words, an Eulerian trail is a walk that traverses every edge of the graph exactly once.
Trail $\Rightarrow$ an edge is traversed at most once.
Eulerian $\Rightarrow$ every edge is traversed.

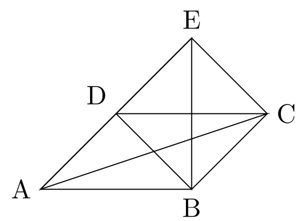
Definition: An <u>Eulerian circuit</u> is a graph is a circuit that traverses every edge of the graph.

Origin of the terminology: Eurlerian comes from the swiss mathematician Leonhard Euler (1707-1783) who solved the problem of the seven bridges of königsbey/kaliningrad (Then Prussia, now Russia) over the river Prugal in 1736. His negative sollution is considered the beginning of graph theory as a subfield of mathematics. We will rederive Euler's results shortly. Google to see the configuration of the bridges on the river Prugel.

Examples:

1. *abc* is an Eulerian trail and an Eulerian circuit. The triangle is k_3.
2. Consider k_5, the complete graph with 5 vertices.
   eabecdbcade is an Eulerian circuit.





In both cases, the degree of the vertices is even for all vertices. We'll see this property is important and derive other necessary and sufficient conditions for the existence of Eulerian trails and circuits.

Theorem: Let (V, E) be a graph, and let $v_0 v_1 \ldots v_m$ be a trail in (V, E). Let $v \in V$ be a vertex, then the number of edges of the trail incident to v is even except when the trail is not closed and the trail starts and finishes at v, in which case the number of edges of the trail incident to the vertex v is odd.

Proof: Note that 0 is an even integer as $0 = 2 \times 0$.

Case 1: $v \neq v_0 \wedge v \neq v_m$. If the trail does not pass through v, the number of dges incident to v belonging to the trail is 0, which is even.
If the trail passes through v, then edges of the trail incident to v are of the form $vi - 1v_i$ and $v_i v_{i+1}$ with $v = v_i$ and $0 < i < m$. Therefore, the number of edges of the trail incident to v equals twice the number of integers i among $1, 2, \ldots, m-1 (0 < i < m)$ s.t. $v = v_i \Rightarrow$ the number of even.

Case 2: $v = v_0$ and the trail is not closed, **i.e.** $v_m \neq v_0$. The edges incident to v are $v_0 v_1$ along with $v_{i-1} v_i$ and $v_i v_{i+1}$ whenever $v = v_i$, hence 1+2. #(instances when $v = v_i$), which is odd.

Case 3: $v = v_m$ and the trail is not closed, **i.e.** $v_m \neq v_0$. Repeat the argument in case 2 with $v_{m-1} v_m$ replacing $v_0 v_1$ to get that the number of edges incident to v is odd.

Case 4: The trail is closed and $v = v_0 = v_m$. The edges incident to v are $v_0 v_1, v_{m-1} v_m$ as well as $v_{i-1} v_i$ and $v_i v_{i+1}$ for each i s.t. $v = v_i \Rightarrow$ once again, the number of edges incident to v is even.

qed

Corollary 1: Let v be a vertex of the graph. Given any circuit in the graph, the number of edges incident to v traversed by that circuit is even.

Proof: Apply the theorem to $v_0 v_1 \ldots v_m$ s.t. $v_0 = v_m$. We deduce that the number of edges incident to v is even.

Corollary 2: If a graph admits an Eulerian circuit, then the degree of every vertex of that graph must be even.

Proof: Let (V, E) be the graph. $\forall v \in V$, the number of edges of any Eulerian circuit incident to v is even by the previous corollary. Since an Eulerian circuit by definition traverses every edge of the graph, every edge incident to v is an edge of the Eulerian circuit $\Rightarrow$ deg v is even $\forall v \in V$ (**NB:** def v could be zero if v is an isolated vertex).

Example: By the previous corollary, k_4, the complete graph on four vertices, cannot have an Eulerian circuit since $\forall v$ in k_4, def $v = 3$ (k_4 is 3-regular as we bserved in a previous lecture).

Corollary 3: If a graph admits an Eulerian trail that is not a circuit, then the degrees of exactly two vertices of the graph must be odd, and the degrees of the remaining vertices must be even. The vertices with odd degrees are exactly the initial and final vertices of the Eulerian trail.

Proof: By the theorem, the initial and final vertices of the Eulerian trail have odd degree, whereas all vertices in between have even degrees.

qed

Next, prove the <u>converse</u> of corollary 2: A non=trivial connected graph has an Eulerian circuit if the deree of each of its vertices is even. The proof is carried out in a series of lemmas:

Lemma A: If the degree of each vertex is even, the $\exists$ a circuit.

Lemma B: If the degree of each vertex is even, if $\exists$ circuit, and if $\exists$ edges not in the circuit incident to a vertex in the circuit, we can construct another circuit.

Lemma C: If we have two circuits with at least one vertex in common, we can combine them.

Lemma D: A criterion for when a trail in Eulerian in a connected graph.

Lemma A: Let vw be an edge of a graph in which the degree of every vertex is even, the $\exists$ circuit of the graph that traverses the edge vw.

Proof: We construct the circuit starting with the edge vw. Let $v_0 = v$ and $v_1 = w$. Let $v_0v_1 \ldots v_k$ be any trail of length $k \geq 1$ traversing the edge vw. Suppose $v_k \neq v = v_0$. As we proved in the previous theorem, since v_k is an endpoint of a non-closed trail, then the number of edges of the trail incident to v_k is odd, but deg v_k is even $\Rightarrow \exists$ edge of the graph incident to v_k that is not traversed by the trail $v_0v_1 \ldots v_k$. Let v_kv_{k+1} be this edge, then $v_0v_1 \ldots v_kv_{k+1}$ is a trail of length $k+1$ that starts at v and traverses vw. Since every edge of the graph is traversed at most once by a trail, the length of any trail in the graph cannot be greater than the number of edges of the graph $\#(E)$. We have shown above that if our trail is not closed, then it can be extended. By successive extensions, we will eventually have constructed a trail that cannot be extended (in at most $\#(E) - 1$ steps). Therefore, that trail must be closed. As the edge vw is traversed, this trail in nontrivial $\Rightarrow$ it is a circuit.

qed

Lemma B: Let (V, E) be a connected graph s.t. $\forall v \in V$, deg v is even, and let some circuit $v_0v_1 \ldots v_{m-1}v_0$, then $\exists$ another circuit in (V, E) passing through v_i that does not traverse any edge traversed by $v_0v_1 \ldots v_{m-1}v_0$.

Proof: Let E' be the set of edges not traversed by $v_0v_1 \ldots v_{m-1}v_0$. $(V, E'$ is a subgraph of (V, E).$\forall v \in V, \#$ of edges of $v_0v_1 \ldots v_{m-1}v_0$ incident to $v = d(v) - d'(v)$, where $d(v) = \text{def}(v) = \#$ of edges in (V, E) incident to v and $d'(v) = \#$ of edges in (V, E') incident to v. By Corollary 1, $d(v) - d'(v)$ is even, but by assumption $d(v) = \text{deg } v$ is even $\Rightarrow d'(v)$ is even $\Rightarrow$ the degree of every vertex in the subgraph (V, E') is even. Now consider the vertex v_i in the statement of Lemme B. Some but not all edges incident to v_i are traversed by $v_0v_1 \ldots v_{m-1}v_0 \Rightarrow d'(v_i) > 0$, **i.e.** at least one edge incident to v_i is in the subgraph (V, E'). We are now exactly in the scenario described by Lemma A $\Rightarrow$ by Lemma A, $\exists$ circuit in (V, E') passing through v_i. This circuit is also a circuit in (V, E) as (V, E') is a subgraph of (V, E), and since all of its edges are in E', this other circuit does not traverse any edge traversed by $v_0v_1 \ldots v_{m-1}v_0$

qed

Lemma C: Suppose that a graph contains a circuit of length m and a circuit of length n. Suppose also that no edges of the graph is traversed by both circuits, and that at least one vertex of the graph is common to both circuits, then the graph contains a circuit of length $m + n$.

Proof: Let v be a vertex of the graph that is common to both circuits. We assume both circuits start and finish at the vertex v. Let the first circuit be $vv_1 \ldots v_{m-1}v$, and let the second circuit be $vw_1w_2 \ldots w_{n-1}v$. We concatenate the two circuits obtaining a circuit $vv_1 \ldots v_{m-1}vw_1w_2 \ldots w_{n-1}w$ of length $m + n$

qed

Lemma D: Let (V, E) be a connected graph, and let some trail in this graph be given. Suppose that no vertex of the graph has the property that not all the edges of the graph incident to that vertex are traversed by the trail. Then the given trail is an Eulerian trail.

Proof: Let v_1 be the set of vertices through which the trail passes, and let v_2 be the set of vertices through which the trail does not pass. $v = v_1 \cup v_2 \wedge v_1 \cap v_2 = \emptyset$. The conclusion of Lemma D amounts to showing $v_2 = \emptyset . \forall u \in v_1, u$ is incident to at least one edge traversed by the trail. $\Rightarrow$ all edges incident to the vertices in v_1 are traversed by the trail, but then every vertex in v adjacent to a vertex in v_1 must belong to $v_1 \Rightarrow$ no edge can join a vertex in v_1 to a vertex in v_2. If $v_2 \neq \emptyset$, then $\exists w \in v_2$, but then w cannot be joined by a path to any vertex in $v_1 \Rightarrow v_1 \wedge v_2$ are in different connected components of the graph $\Rightarrow\Leftarrow$ since the graph is connected $\Rightarrow$ it has only one connected component. Therefore, $v_2 = \emptyset$.

qed

Finally, we can prove Euler's theorem:

Theorem A non-trivial connected graph contains an Eulerian circuit if the degree of every vertex of the graph is even.

Proof: Let (V, E) be a non-trivial connected graph s.t. $\forall v \in V$, deg v is even. By Lemma A, (V, E) contains at least one circuit. It therefore contains a circuit of maximal length (**i.e.** at least as long as any other circuit in the graph). We seek to prove that this circuit of maximal length is indeed Eulerian.
If the graph contains some vertex v s.t. some but not all of the edges of the graph incident to v are traversed by the circuit of maximal length, and v is a vertex on the circuit of maximal length, then by Lemma B $\exists$ a second circuit in (V, E) passing through v, which would not traverse any edge traversed by the circuit of maximal length. By Lemma C, however, we can concatenate the two circuits, obtaining a circuit of length strictly greater than the length of the circuit of maximal length $\Rightarrow\Leftarrow$ we conclude no vertex that belongs to the circuit of maximal length has the property that not all edges incident to it are traversed by the circuit of maximal length. Since (V, E) is connected, by Lemma D, the circuit of maximal length must be Eulerian.

qed

Corollary 2 along with this theorem together gives us:

Theorem: A non-trivial connected graph has an Eulerian circuit $\Leftrightarrow$ the degree of each of its vertices is even.

Corollary: Suppose a connected graph has exactly two vertices whose degree is odd. $\exists$ an Eulerian trail in the graph joining the two vertices with odd degrees.

Proof: We reduce this case to the previous one by embedding the graph (V, E) with vertices v, w that have odd degree into a graph (V', E') s.t. $v' = v \cup \{u\}$ for $u \notin V$ and $E' = E \cup \{uv, uw\}$. (V, E) is a subgraph of $(V'E')$ and (V', E') is connected and each one of its vertices has even degree by construction. By the theorem we just proved, (V', E') has an Eulerian circuit. We reorder the vertices so that the final two edges are the two added edges wu and uv. We now delete the edges wu and uv to obtain an Eulerian trail in the original graph (V, E) from v to w.

qed

31 Hamiltonian Paths and Circuits

Task: Look at paths and circuits that pass through every vertex of a graph.

Definition: A Hamiltonian path in a graph is a path that passed exactly once through every vertex of a graph.
Path $\Rightarrow$ we pass through a vertex at most once (no repeated vertices)
Hamiltonian $\Rightarrow$ we pass through every vertex.

Definition: A Hamiltonian circuit in a graph is a simple circuit that passes through every vertex of the graph.

Origin of the Terminology: Named after William Roman Hamilton (1805-1865) who showed in 1856 that such a circuit exists in the graph consisting of the verices and edges of a dodecahedron (see page 88 in David Wilkins' notes for the picture of a Hamiltonian circuit on a dodecahedron). Hamilton developed a game called Hamilton's puzzle of the icosian game in 1857 whom object was to find Hamiltonian circuits in the dodecahedrom (many solutions exist). This game was marketed in Europe as a pigboard with holes for each vertex of the dodecaherdrom.
NB: The dodecahedron is a Platonic solid, and it turns out every Platonic solid has a Hamiltonian circuit. Recall that the Platonic solids are the tetrahedron (4 faces), the cube (6 faces), the octahedron (8 faces), the dodecahedron (12 faces), and the icosahedron (20 faces). Each of these is a regular graph.

Theorem: Every complete graph k_n for $n \geq 3$ has a Hamiltonian circuit.

Proof: Let $V = \{v_1, v_2, v_3, \ldots v_n\}$ be the set of vertices of k_n, then $v_1 v_2 v_3 \ldots v_n v_1$ is a Hamiltonian circuit. All edges in this circuit are part of k_n because k_n is complete.

qed

32 Forests and Trees

Task: Use the notion of a circuit to define trees.

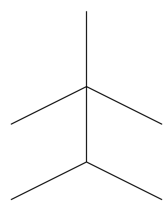
Definition: A graph is called acyclic if it contains no circuits (also known as cycles).

Definition: A forest is an acyclic graph.

Definition: A tree is a connected forest.

Examples:

1. Is a tree and a forest.



2. Is a forest with 2 connected components (**i.e.** it consists of 2 trees.)

Theorem: Every forest contains at least one isolated or pendant vertex.

Proof: Recall that when we studied circuits we proved a theorem that if (V, E) is a graph s.t. $\forall v \in V$ deg $v \geq 2$ (**i.e.** (V, E) has no isolated or pendant vertices), then (V, E) contains at least one simple circuit. The graph (V, E) is a forect, **i.e.** it contains no circuits $\Rightarrow \exists v \in V$ s.t. deg $v = 0$ or deg $v = 1$

qed

Theorem: A non-trivial tree contains at least one pendant vertex.

Proof: A non-trivial tree (V, E) must contain at least 2 vertices. Assum $\exists v \in V$ s.t. deg $v = 0$,**i.e.** v is isolated, then v forms a connected component by itself, but then (V, E) has at least 2 connected components as $\#(V) \geq 2 \Rightarrow\Leftarrow$ to the fact that a tree is by definition connected. Therefore, $\forall v \in V$, deg $v \geq 1$, but by the previous theorem $\exists v \in V$ s.t. $0 \leq$ deg $v \leq 1 \Rightarrow \exists v \in V$ s.t. deg $v = 1$ (since a tree is a forest).

qed

Theorem: Let (V, E) be a tree, then $\#(E) = \#(V) - 1$, where $\#(E)$ is the number of edges of the tree and $\#(V)$ is the number of vertices.

Proof: Use strong induction on $\#(V)$.

Base Case: $\#(V) = 1$. The graph is trivial $\Rightarrow \#(E) = 0$, so $0 = 1 - 1$ as needed.

Inductive Step: Suppose that every tree with m vertices ($\#(V) = m$) has $m - 1 = \#(v) - 1 = \#(E)$ edges we seek to prove that if (V, E) is a tree with $m + 1$ vertices, then it has m edges.
By the previous theorem, (V, E) has one pendent vertex. Let that vertex be v. Since deg $v = 1$, then there is only one edge incident to v. Let vw be that edge. w is then the only vertex of (V, E) adjacent to v. We wish to reduce to the inductive hypothesis, the most natural way is to delete w from v and vw from E. Let $V' = V\backslash\{v\}$ and $E' = E\backslash\{vw\} = \#(E) - 1$. To use the inductive hyptohetsis, we must show (V', E') is a tree, **i.e.** (V', E') is connected and $(V'E')$ contains no circuits. $\forall v_1, v_2 \in V'$, since (V, E) is a tree hence connected, $\exists$ path from v_1 to v_2 in (V, E). This path cannot pass throgh v because deg $v = 1 \Rightarrow$ is would have to pass through v twice contradicting the fact that is it a path (all vertices are distinct) $\Rightarrow$ this path is in $(V', E') \Rightarrow (V'E')$ is connected.
(V', E') is a subgraph of (V, E), which is a tree, hence does not containa ny circuits, so (V', E') contains no circuits.
(V', E') is thus a tree, $\Rightarrow$ by the inductive hypothetsis, $\#(V') = \#(V) - 1 = \#(E') - 1 = \#(E) - 1 - 1 = \#(E) - 2 \Rightarrow \#(V) - 1 = \#(E) - 2 \Leftrightarrow \#(V) = \#(E) - 1$ as needed.

qed

33 Countability of Sets

Task: Understand what it means for a set to be countable, countably infinite and uncountably infinite. Show that the set of all languages over a finite alphabet is uncountably infinite, wheras the set ofa ll regular languages over a finite alphabet is countably infinite.

We want to understand sizes of sets. In the unit on functions last term, when we looked at functions defined on finite sets, we wrote down a set A with m elements as $A = \{a_1, \dots, a_n\}$. This notation mimics the process of counting: a_1 is the first element of A, a_2 is the second element of A, and so on up to a_n is the n^th element of A. In other words, another way of saying A in a set of n elements is that there exists a bijective function $f : A \to \{1, 2, \dots, n\}$.

Definition: A set A has n elements $\leftrightarrow \exists f : A \to J_n$ a bijection.

NB: This definition works $\forall n \geq 1, n \in \mathbb{N}^*$

Notation: $\exists f : A \to J_n$ a bijection is denoted as $A \sim J_n$. More generally, $A \sim B$ means $\exists f : A \to B$ a bijection, and it is a relation on sets. In fact, it is an equivalence relation (check!). $[J_n]$ is the equivalence class of all sets A of size n, i.e. $\#(A) = n$.

Definition: A set A is finite if $A \sim J_n$ for some $n \in \mathbb{N}^*$ or $A = \emptyset$.

Definition: A set A is infinite if A is not finite.

Examples: $\mathbb{N}, \mathbb{Q}, \mathbb{R}$, etc.

To understand sizes of infinite sets, generalize the construction above. Let $J = \mathbb{N}^* = \{1, 2, \dots\}$

Definition: A set A is uncountably infinite if A is neither finite nor countably infinite. In fact, we often treat together the cases A is finite or A is countably infinite since in both of these cases the counting mechanism that is so familiar to us works. Therefore, we have the following definition:

Definition: A set A is countable if A is finite ($A \sim J_n$ or $A = \emptyset$) or A is countably infinite ($A \sim J$).

There is a difference in the terminology regarding countability between CS sources (textbooks, articles, etc.) and maths sources. This is the dictionary:

CS	Maths
countable	at most countable
countably infinite	countable
uncountably infinite	uncountable

It's best to double check which terminology a source is using.

Goal: Characterize $[\mathbb{N}]$, the equivalence class of countably infinite set, and $[\mathbb{R}]$, the equivalence class of uncountably infinite sets the same size as $\mathbb{R}$.

Bad news: Both $[\mathbb{N}]$ and $[\mathbb{R}]$ consist of ininite sets.

Good news: We only care about these two equivalence classes.

NB: There are uncountably infinite sets of size bigger than $[\mathbb{R}]$ that can be obtained from the power set construction, but it is unlikely you will see them in your CS coursework.

To characterize $[\mathbb{N}]$, we need to recall the notion of a sequence:

Definition: A sequence is a set of elements $\{x_1, x_2, \dots\}$ indexed by J, i.e. $\exists f :\to \{x_1, x_2, \dots\}$ s.t. $f(n) = x_n \forall n \in J$.

Recall that sequences and their limits are used to define various notions in calculus (differentiation, integration, etc.). Also, calculators use sequences in order to compute with various rational and irrational numbers.