1. Introduction to logic

① variables, sets and mathematical statements

1) variables : placeholders of values to give names and maintain generality

2) important sets

$\mathbb{N}$, natural numbers (non-negative integers) — $\mathbb{Z}$, integers — $\mathbb{Q}$, rational numbers — $\mathbb{R}$, real numbers

↳ set notation

- superscripts ($^+$, $^-$) — note that 0 is neither positive nor negative
- subscripts ($_{\geq n}$, $_{\leq n}$)
- $\in$ "is a member of"

3) basic mathematical statements

↳ Definition : a sentence that is true or false, but not both

- universal statement: ↳ a certain property is true for all elements in a set
- conditional: ↳ if one thing is true then some other thing also has to be true
- existential: ↳ there is at least one thing for which the property is true

combined ⇒ compound statements

4) fundamentals of proofs

↳ a proof is an inferential argument for a mathematical statement. in principle, a proof can be traced back to self evident or assumed statements (axioms) through accepted rules of inference.

↳ key definitions

1. Definition : precise and unambiguous description of a word by giving only the properties that must be true
2. Axiom / postulate : statement that is assumed to be true without proof. Building block.
3. Theorem : statement that is proved using mathematical reasoning. An important result.
4. lemma : a small theorem; a minor result whose purpose is to help in proving a theorem.
5. Corollary : a result that is a simple deduction from a theorem.
6. Conjecture : statement believed to be true, for which there is no proof.

② Basic properties of integers

1) key properties :

1. closure under addition and multiplication — $(x+y) \in \mathbb{Z}$, $xy \in \mathbb{Z}$
2. commutativity of addition and multiplication — $x+y = y+x$, $xy = yx$
3. associativity of addition and multiplication — $x+y+z = (x+y)+z$, $xyz = (xy)z$
4. distributivity of multiplication — $x(y+z) = xy + xz$
5. trichotomy — $x = y$ or $x > y$ or $x < y$

2) even and odd integers

n is odd $\longleftrightarrow \exists k \in \mathbb{Z}$ s.t. $n = 2k+1$

n is even $\longleftrightarrow \exists k \in \mathbb{Z}$ s.t. $n = 2k$

3) proven results

1. product of two consecutive odds are odd (1.3.4 example #1)
2. For all integers n, if n^2 is even then n is even (4.7.4)
3. There is no greatest integer (4.6.1)
4. n^2 is odd iff n is odd
5. product of any two odd integers is odd

③ Divisibility of integers

1) $d \mid n \longleftrightarrow \exists k \in \mathbb{Z}$ s.t. $n = dk$

d divides n, n is divisible by d

2) proven results

1. divisibility is transitive $a \mid b \wedge b \mid c \rightarrow a \mid c$ (4.3.2)
2. $a \mid b \rightarrow a \leq b$ (4.3.2)
3. The only divisors of 1 are 1 and -1 (4.3.2)

④ rational and irrational numbers

1) Definition: r is rational $\longleftrightarrow \exists a, b \in \mathbb{Z}$ s.t. $r = \frac{a}{b}$ and $b \neq 0$; a real number that is not rational is irrational.

2) proven results

1. $\sqrt{2}$ is irrational (1.3.5 Example #6)
2. There exists irrational number p and q s.t. p^q is rational. (1.3.5 Example #7)
3. The sum of 2 rational numbers is rational (4.2.3)
4. rational numbers are closed under addition

Notes

1. number theory; Bezout's identity $d = \gcd(a,b) \rightarrow \underbrace{ax + by}_{\text{linear combination}} = d$

2. Logic of compound statements

① statements and truth values

1) statement forms, variables and truth values

- **statement form**: expression made of statement variables and logical connectives. Becomes a statement when substituted in.
- statement is either true or false so statement form can be either
  - ⇓ use of truth table to systematically list it out

2) tautologies and contradictions

- **tautology**: a statement form that is always true regardless of truth values of individual statements substituted for its statement variables.
- **contradiction**: a statement form that is always false regardless of truth values of individual statements substituted for its statement variables.

to prove:

- truth table
  - eg. p q r ... ... full impd statement
    T
    T
    T
    ⋮
- logical equivalence to true/false
  - eg. full impd statement ≡ ...
    ⋮
    ≡ false

② logical connectives and equivalence

1) logical equivalence : two statement forms are logically equivalent iff. they have identical truth values for each possible substitution for their statement variables. $P \equiv Q$

↳ to prove logical equivalence / inequivalence

- identical truth values for each row of statement variable boolean values
- inequivalence
  - counter example
  - truth table w/ ≥ row different truth values

2) basic logical connectives

- not (negation) ~

p	~p
T	F
F	T

- and (conjunction) ∧

p	q	p∧q
T	T	T
T	F	F
F	T	F
F	F	F

- or (disjunction) ∨

p	q	p∨q
T	T	T
T	F	T
F	T	T
F	F	F

- conditionals
  - if/then ⟶
  - if and only if (biconditional) ⟷

p	q	p→q	q→p	p↔q	(p→q)∧(q→p)
T	T	T	T	T	T
T	F	F	T	F	F
F	T	T	F	F	F
F	F	T	T	T	T

first

equal order of operations. Requires ( ) to decide order unambiguously.

second

equal order of operations

last

③ conditionals

1) conditional statements

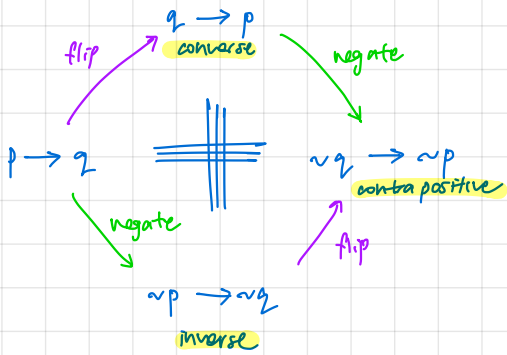
if p, then q — p: hypothesis, q: conclusion

$p \to q$ "p implies q"

p	q	$p \to q$
T	T	T
T	F	F
F	T	T
F	F	T

vacuously true: because statement is not technically false ⇒ must be true

1. contrapositives, converse, inverse



intuition of logical equivalence: if q did not happen, then p must not have happened, since it would otherwise have led to q.

2. "only if": p only if q ⇒ if not q then not p ≡ if p then q (contrapositive)

2) biconditional: p "if and only if" q — p only if q; p if q

$p \leftrightarrow q \equiv (p \to q) \wedge (q \to p)$

p	q	$p \to q$	$q \to p$	$p \leftrightarrow q$	$(p \to q) \wedge (q \to p)$
T	T	T	T	T	T
T	F	F	T	F	F
F	T	T	F	F	F
F	F	T	T	T	T

3) necessary and sufficient conditions

q is a necessary condition for p: $p \to q$

p is a sufficient condition for q: $p \to q$

4) summary of logical equivalences

1	Commutative laws	$p \wedge q \equiv q \wedge p$	$p \vee q \equiv q \vee p$
2	Associative laws	$(p \wedge q) \wedge r \equiv p \wedge (q \wedge r)$	$(p \vee q) \vee r \equiv p \vee (q \vee r)$
3	Distributive laws	$p \wedge (q \vee r) \equiv (p \wedge q) \vee (p \wedge r)$	$p \vee (q \wedge r) \equiv (p \vee q) \wedge (p \vee r)$
4	Identity laws	$p \wedge \mathbf{true} \equiv p$	$p \vee \mathbf{false} \equiv p$
5	Negation laws	$p \vee \sim p \equiv \mathbf{true}$	$p \wedge \sim p \equiv \mathbf{false}$
6	Double negative law	$\sim(\sim p) \equiv p$	
7	Idempotent laws	$p \wedge p \equiv p$	$p \vee p \equiv p$
8	Universal bound laws	$p \vee \mathbf{true} \equiv \mathbf{true}$	$p \wedge \mathbf{false} \equiv \mathbf{false}$
9	De Morgan's laws	$\sim(p \wedge q) \equiv \sim p \vee \sim q$	$\sim(p \vee q) \equiv \sim p \wedge \sim q$
10	Absorption laws	$p \vee (p \wedge q) \equiv p$	$p \wedge (p \vee q) \equiv p$
11	Negation of **true** and **false**	$\sim\mathbf{true} \equiv \mathbf{false}$	$\sim\mathbf{false} \equiv \mathbf{true}$

$p \to q \equiv \sim p \vee q$

$\sim(p \to q) \equiv p \wedge \sim q$

④ Arguments

1) Definition

An **argument (argument form)** is a sequence of statements (statement forms). All statements in an argument (argument form), except for the final one, are called **premises** (or **assumptions** or **hypothesis**). The final statement (statement form) is called the **conclusion**. The symbol •, which is read "therefore", is normally placed just before the conclusion.

To say that an argument form is **valid** means that no matter what particular statements are substituted for the statement variables in its premises, if the resulting premises are all true, then the conclusion is also true.

Argument: $p_1 \wedge p_2 \dots p_n \wedge$ conclusion

($p_1 \dots p_n$: premises)

↳ an argument is a series of assertions.

2) testing an argument for validity

1. construct truth table
2. a row where all premises are true is called a critical row

does any critical row have a false truth value?

- yes → argument is invalid (cite this as counter example)
- no → argument is valid

3) valid argument forms

Rule of inference		
Modus Ponens	$p \to q$ p • q	
Modus Tollens	$p \to q$ $\sim q$ • $\sim p$	
Generalization	p • $p \vee q$	q • $p \vee q$
Specialization	$p \wedge q$ • p	$p \wedge q$ • q
Conjunction	p q • $p \wedge q$	

Rule of inference		
Elimination	$p \vee q$ $\sim q$ • p	$p \vee q$ $\sim p$ • q
Transitivity	$p \to q$ $q \to r$ • $p \to r$	
Proof by Division Into Cases	$p \vee q$ $p \to r$ $q \to r$ • r	
Contradiction Rule	$\sim p \to$ **false** • p	

contradiction rule: if you can show that the supposition that statement is false (negation) leads logically to a contradiction, then you can conclude p is true

4) sound arguments: argument is sound ↔ all premises are true ∧ argument is valid

5) fallacies

1. ambiguous premise
2. circular reasoning
3. jumping to conclusion
4. converse error
   $p \to q$
   q
   • p ⊗
5. inverse error
   $p \to q$
   $\sim p$
   • $\sim q$ ⊗

Notes

1. $(p \to q \wedge q \to r) \to (p \to r)$, but $\not\equiv p \to r$

2. $p \oplus p \equiv$ false $\quad p \oplus q \equiv (p \vee q) \wedge (\neg p \vee \neg q)$

3. Logic of quantified statements

① predicates and quantified statements

1) predicates → evaluate to boolean values

↳ A predicate is a sentence that contains a finite number of variables and become a statement when specific values are substituted for the variables.

↳ the **domain** of a **predicate variable** is the set of all values that may be substituted in place of the variable.

↳ the **truth set** of is the set of all elements (in order) of the respective domains of predicate variables that make $P(x, y \dots)$ true when they are substituted.

eg. x is a student at y Truth set $= \{(x, y \dots) \; x \in D_x, y \in D_y \dots : P(x, y \dots)\}$

(has a Domain) has a Domain

2) quantifiers : another way to convert predicates into statements

for all, $\forall$ → **universal statements**

$\forall x \in D, \; Q(x) \equiv Q(x_1) \wedge Q(x_2) \dots$

↳ true iff $Q(x)$ true for every $x \in D$

↳ false iff $Q(x)$ false for at least one $x \in D$

↳ can be vacuously true

↳ logically equivalent to a generalized 'and' statement

there exists, $\exists$ → **existential statements**

$\exists x \in D \; Q(x) \equiv Q(x_1) \vee Q(x_2) \dots$

↳ true iff $Q(x)$ true for at least one $x \in D$

↳ false iff $Q(x)$ false for all $x \in D$

↳ logically equivalent to a generalized 'or' statement

there exists only one/a unique, $\exists!$

3) equivalent forms of quantified statements

universal

↳ universal conditional statements can be rewritten by narrowing the scope.

eg. $\forall x \in \mathbb{R} \; (x \in \mathbb{Z} \to x \in \mathbb{Q})$

$\equiv \forall x \in \mathbb{Z}, \; x \in \mathbb{Q}$

existential

↳ existential statements' domains can be narrowed down to form an equivalent statement.

$\exists x \text{ s.t. } P(x) \wedge Q(x) \equiv \exists x \in D \text{ s.t. } Q(x)$

$D = \{x : Q(x)\}$

4) implicit quantification : take note of context in informal descriptions to see if statement is quantified.

eg. if $x > 2$ then $x^2 > 4$

⇒ implied to mean $\forall x \in \mathbb{R} \; (x > 2 \to x^2 > 4)$

② negations of quantified statements

1) negation of a universal statement

$\sim(\forall x \in D,\ P(x)) \equiv \exists x \in D,\ \sim P(x)$

2) negation of an existential statement

$\sim(\exists x \in D,\ P(x)) \equiv \forall x \in D,\ \sim P(x)$

③ multi quantified statements

↳ use words like "each", "there is one"

1) order of quantifiers
   - ↳ if contain both $\forall$ and $\exists$ ⇒ order matters. think about it.
   - ↳ if one quantifier immediately follows another of the same type, then order no difference.

2) negations of multi quantified statements
   - ↳ do it part by part

eg. $\sim(\forall x \in D\ \ \exists y \in E \text{ s.t. } P(x,y))$

$\Rightarrow \exists x \in D \text{ s.t. } \sim(\exists y \in E \text{ s.t. } P(x,y))$

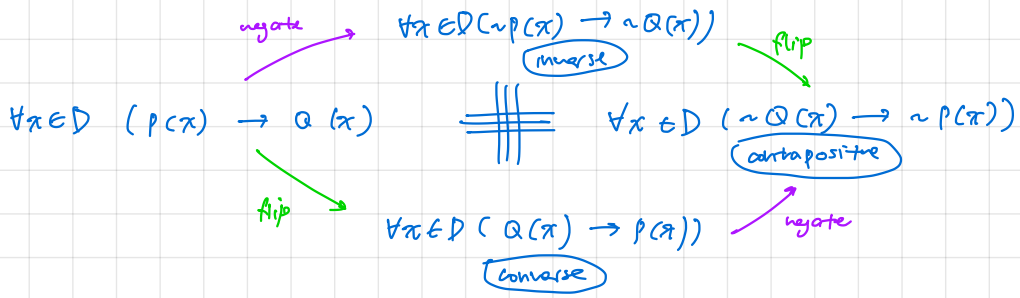
$\Rightarrow \exists x \in D\ \forall y \in E \text{ s.t. } \sim P(x,y)$

④ quantified conditional statements

1) existential conditional statement $\exists x\,(P(x) \to Q(x)) \overset{\sim}{\Rightarrow} \forall x\,(P(x) \wedge \sim Q(x))$

2) universal conditional statement $\forall x\,(P(x) \to Q(x)) \overset{\sim}{\Rightarrow} \exists x\,(P(x) \wedge \sim Q(x))$

3) variants of conditional statements



4) necessary, sufficient, only if

$\forall x$, $r(x)$ is a sufficient condition for $s(x)$ ⇒ $r(x) \to s(x)$

$\forall x$, $r(x)$ is a necessary condition for $s(x)$ ⇒ $\sim r(x) \to \sim s(x) \equiv s(x) \to r(x)$

$\forall x$, $r(x)$ only if $s(x)$ ⇒ $\sim s(x) \to \sim r(x) \equiv r(x) \to s(x)$

⑤ arguments with quantified statements

1) universal instantiation, modus ponens and modus tollens

↳ if some property is true of everything in a set, then it is true of any particular thing in the set

universal modus ponens

$\forall x\ (P(x) \rightarrow Q(x))$

$P(a)$ for a particular a.

$\therefore Q(a)$

universal modus tollens

$\forall x\ (P(x) \rightarrow Q(x))$

$\sim Q(a)$ for a particular a.

$\therefore \sim P(a)$

2) universal transitivity

$\forall x\ (P(x) \rightarrow Q(x))$

$\forall x\ (Q(x) \rightarrow R(x))$

$\forall x\ (P(x) \rightarrow R(x))$

Notes

1. be careful of order of inputs in predicates eg. $P(x,y)$
2. when evaluating truth value of bidirectional $\leftrightarrow$, $\Leftrightarrow$, always consider both directions!

4. Methods of proof

proving exactly one / unique
1. show that ≥ 1
2. show that ≤ 1

① Direct proofs

1) proving existential statements by constructive proof (example)

$\exists x \in D$ s.t. $Q(x)$

↳ find an x (concrete example)
↳ give directions to find x ie. show that x must exist

2) Disproving universal statements by counterexample

↳ showing statement is false is same as showing negation is true
↳ Find a value of $x \in D$ such that negation is true

3) proving universal statements by exhaustion

1. split into cases (essentially considering OR cases)
2. show that all cases lead to statement being true

4) proving universal statements by generalization (if/then, for every)

1. choose an arbitrary but particular element from D
2. show by manipulation, logical equivalence, etc. that statement is true for x.
3. So statement is true for all x (universal generalization)

5) Proof by induction

1. prove that $n=1$ holds
2. state $n=k$ and $n=k+1$ case
3. link them and show that they are equal
4. so it holds for every next number

6) uniqueness proofs

1. prove that at least 1 exists
2. prove that if >1 exist, then they are equal

② indirect proof

1) proof by contradiction

1. suppose negation of statement $\sim S$ is true.
2. show that $\sim S$ logically leads to a contradiction
3. conclude that S is true.

2) proof by contraposition

1. express statement in logically equivalent form (eg. contrapositive of conditional)
2. Do a direct proof to prove S'
3. $S' \equiv S$, so S is true.

(A) universal statement $\overset{\sim}{\longleftrightarrow}$ existential conditional

exhaustion, direct, constructive, contraposition, induction, counterexample

contradiction

(B) uniqueness

Notes

1. For mathematical manipulation, can use 'Note that'
2. make sure to state domain and follow definitions strictly eg. $Q \Rightarrow \frac{a}{b}$, $a, b \in \mathbb{Z}$, $b \neq 0$
3. Not divisible by: $b \mid a \iff \frac{a}{b} = nk + q$, $0 < q < k$
4. definition of modulus: $|f(x)| = \begin{cases} f(x) & \text{if } f(x) \geq 0 \\ -f(x) & \text{if } f(x) < 0 \end{cases}$
5. Fractions $\longrightarrow$ multiply by base to simplify $\frac{x}{y} + \frac{y}{x} \iff x^2 + y^2$
6. For contradiction, start from thing you foresee contradicting.

eg. $(a > b \wedge c > 0) \wedge ac \leq bc$

1.1 assume $ac \leq bc$
1.2 so $c(a-b) \leq 0$
1.3 since $c > 0$, $(a-b) \leq 0$
1.4 but $a > b$

$\longrightarrow$ contradiction

7. "by basic algebra" don't forget
8. "suppose ..." when proving implications or biconditionals

5. Sets

① Definitions and set notation

1) sets : unordered collection of unique elements
   ↳ so $\{1,1\}$ is the same as $\{1\}$

2) notation

roster notation
↳ listing out all elements
$\{1, 2, 3 \dots\}$

set builder notation
↳ filtered subset
$\{\pi \in \mathbb{U} \mid \pi > 2\}$
filter condition

replacement notation
↳ set of processed objects
$\{t(x) \mid x \in \mathbb{U}\}$

3) common sets : $\mathbb{N}$ (natural no.), $\mathbb{Z}$ (integers), $\mathbb{Q}$ (rational no.), $\mathbb{R}$ (real no.), $\mathbb{C}$ (complex no.)

4) cardinality, finite / infinite sets
   ↳ no. of unique elements in set, denoted $|A|$ for finite sets. sets of $|A| = 1$ are singletons.

② subsets and set equality

1) subsets : sets where every element is an element of another set

$$A \subseteq B \iff \forall x\,(x \in A \to x \in B)$$
$$\sim(A \subseteq B) \iff \exists x\,(x \in A \to x \notin B)$$

} by definition. $\emptyset \subseteq A$, $A \subseteq A$

(proper subset) $A \subsetneq B \iff \forall x(x \in A \to x \in B) \wedge \exists y \in B, y \notin A$
$\iff A \subseteq B \wedge B \neq A$

2) set equality

$$A = B \iff A \subseteq B \wedge B \subseteq A \iff \forall z\,(z \in A \leftrightarrow z \in B)$$

↳ proving set equality :
1. suppose x is a particular but arbitrarily chosen element of A
   ⋮
   ∴ $x \in B$, so $A \subseteq B$ (universal generalisation)
2. suppose x is a particular but arbitrarily chosen element of B.
   ⋮
   ∴ $x \in A$, so $B \subseteq A$ (universal generalisation)
3. so $A = B$ (by definition of set equality)

3) set membership vs inclusion
   an element of a subset of

③ operations on sets

union → if sets are mutually disjoint, then $\left|\bigcup_{i=1}^{\infty} A_i\right| = \sum_{i=1}^{\infty} |A_i|$

$$A \cup B = \{ x \in U : x \in A \vee x \in B \}$$

difference

$$A \setminus B = \{ x \in U \mid x \in A \wedge x \notin B \}$$

intersection

$$A \cap B = \{ x \in U : x \in A \wedge x \in B \}$$

complement

$$A^C = U \setminus A = \{ x \in U \mid x \notin A \}$$

④ unique sets

1) empty set : there exists a **unique** set with no elements in it, $\emptyset$

↳ proof :
1. $\{\}$ has no elements.
2. Let A, B be sets with no elements.
   - 2.1 then vacuously, $A \subseteq B$ and $B \subseteq A$.
   - 2.2 so $A = B$

an empty set is a subset of every set

1. suppose the negation is true : $\exists x \in \{\}$ s.t. $x \notin A$.
2. No elements in $\{\}$. contradiction.

OR

1. no elements in $\{\}$.
2. $\forall x \in \{\} \rightarrow x \in A$ is vacuously true.

2) powerset : the set of all possible subsets of A. $\mathcal{P}(A)$. $|\mathcal{P}(A)| = 2^{|A|}$ $|A| = n$ (itself + $\emptyset$)

↳ n reflexive tuples, $n^2 - n$ non.

↳ $\frac{n^2-n}{2}$ symmetric pairs (non-reflexive) + n reflexive & symmetric tuples

3) disjoint sets

sets $A_1, A_2 \dots A_n$ are **pairwise disjoint** $\iff A_i \cap A_j = \emptyset$, $i \neq j$

sets $A_1, A_2 \dots A_n$ are **mutually disjoint** $\iff A_1 \cap A_2 \dots A_n = \emptyset$

4) partitions : a finite or infinite **collection of nonempty sets** $\mathcal{B}$ is a partition of A iff

1. $A = \bigcup_{i=1}^{\infty} A_i$
2. $A_1, A_2 \dots$ are mutually disjoint.

$\iff$

1. all elements are non-empty subsets of A
2. every element of A is in exactly one **element** of $\mathcal{B}$
   ↳ called a **component**

1. $\forall S \in \mathcal{B}, S \neq \emptyset \wedge S \subseteq A$
2. $\forall x \in A, \exists S \in \mathcal{B}$ s.t. $(x \in S) \wedge \forall x \in A, \forall S_1, S_2 \in \mathcal{B} \; (x \in S_1 \wedge x \in S_2 \rightarrow S_1 = S_2)$

3) ordered tuples and cartesian products

↳ represented by $(x, y, \dots)$

cartesian product : set of all ordered tuples of element from each set. $A \times B \times C \dots$

$$\{(x, y, z \dots) \mid x \in A, y \in B, z \in C \dots\}$$

(5) properties of sets

1) subset relations

1. inclusion of intersection $(A \cap B) \subseteq A$ — $x \in A$ specialisation
2. inclusion in union $A \subseteq A \cup B$ — $(x \in A \vee x \in B)$ by generalisation
3. transitive property of subsets $A \subseteq B \wedge B \subseteq C \rightarrow A \subseteq C$ — $x \in A \rightarrow x \in B \wedge x \in B \rightarrow x \in C$ by definition. so $x \in A \rightarrow x \in C$.

2) set identities

For all set A, B, C in a context where U is the universal set, the following hold.

Identity Laws	$A \cup \varnothing = A$	$A \cap U = A$
Universal Bound Laws	$A \cup U = U$	$A \cap \varnothing = \varnothing$
Idempotent Laws	$A \cup A = A$	$A \cap A = A$
Double Complement Law	$\overline{(\overline{A})} = A$	
Commutative Laws	$A \cup B = B \cup A$	$A \cap B = B \cap A$
Associative Laws	$(A \cup B) \cup C = A \cup (B \cup C)$	$(A \cap B) \cap C = A \cap (B \cap C)$
Distributive Laws	$A \cup (B \cap C) = (A \cup B) \cap (A \cup C)$	$A \cap (B \cup C) = (A \cap B) \cup (A \cap C)$
De Morgan's Laws	$\overline{A \cup B} = \overline{A} \cap \overline{B}$	$\overline{A \cap B} = \overline{A} \cup \overline{B}$
Absorption Laws	$A \cup (A \cap B) = A$	$A \cap (A \cup B) = A$
Complement Laws	$A \cup \overline{A} = U$	$A \cap \overline{A} = \varnothing$
Set Difference Law	$A \setminus B = A \cap \overline{B}$	
	$\overline{\varnothing} = U$	$\overline{U} = \varnothing$

Tutorial 3 Q9 $A \subseteq B \leftrightarrow A \cup B = B$

3) proving an intersection is empty: suppose set has an element and derive a contradiction

4) generalised distributive law

$$A \cup \left(\bigcap_{i=1}^{n} B_i\right) = \bigcap_{i=1}^{n} (A \cup B_i)$$

case 1 : $x \in A$

case 2 : $x \in \bigcap_{i=1}^{n} B_i$ or $x \in A \leftrightarrow$

$x \in A \cup \left(\bigcap_{i=1}^{n} B_i\right)$

Notes

1. when checking set equivalence in function domains, be careful in domain/codomain of function eg. $\{3n+1 : n \in \mathbb{Z}\} \neq f: \mathbb{Z} \to \mathbb{Z}, f(n) = 3n+1$, because $0 \in \mathbb{Z}$

2. Complex no. $\mathbb{C}$ can represent $\mathbb{R}$, but not other way around

3. when proving subsets, start in element

eg. $(\ldots) \longrightarrow B \subseteq D$

1. $x \in B$.
   ⋮
   $\therefore x \in D$

4. proving intersection is empty (disjoint): force contradiction by assuming non-empty

eg. $A \cap B = \emptyset \implies A \cap B \neq \emptyset$. Let $x \in A$ ... $x \notin B$

5. "Let A and B be non-empty sets"

6. proving any subset or equality, start there. eg. $A \cup B = B$
   $x \in A \cup B$ → $x \in A$ ...
   $x \in A \cup B$ → $x \in B$ ...

7. proving: • union by generalisation
   • intersection by specialisation
   $x \in A \to x \in A \wedge x \in B \to x \in A \cap B$
   $x \in A \cap x \in B \to x \in A \cap B \to x \in A$
   NOT by definition of $\cup$ or $\cap$, but the underlying principle is specialisation / generalisation.
   don't skip the step of converting definition of $\cup$ or $\cap$
   eg. $x \in \bigcup A_i \iff x \in A_1 \vee x \in A_2 \ldots$
   $x \in \bigcap A_i \iff x \in A_1 \wedge x \in A_2 \ldots$
   $\iff x \in A_i$ by specialisation

8. proving equality, never prove both direction subsets together. Always do separate, and start from left to right. eg. $R = R^{-1}$
   1. $x \in R$ ...
   2. $x \in R^{-1}$ ...

9. For all sets A and B, $\mathcal{P}(A \cup B) = \mathcal{P}(A) \cup \mathcal{P}(B)$

10. union: take away the outside! eg. $\phi \cup \{\phi\} \implies \{\} \cup \{\{\}\}$
    $\implies \{\{\}\} \implies \{\phi\}$

6. Relations

① relations

1) relations between and on sets

↳ relation: a set of ordered pairs containing one object from each set that is a subset of the cartesian product. the elements in the pairs may be related to each other.

$x R y \iff$ some relationship. $\quad R = \{(x,y) \in A \times B : x R y\}$

(A: domain, B: range)

2) inverse of a relation

if R is a relation from A to B, then the inverse relation R^{-1} from B to A is defined as:

(image / pre-image)

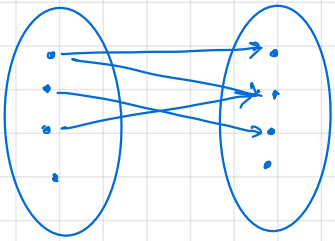
$R = \{(x,y) \in A \times B : x R y\} \quad R^{-1} = \{(y,x) \in B \times A : y R^{-1} x\}$

3) n-ary relations

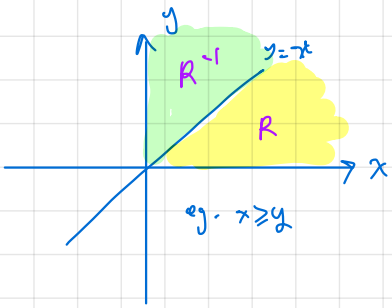
subset of cartesian product of n sets

4) visualising relations

1. finite relations

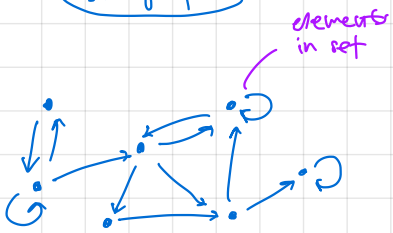


2. infinite relations



relations on a set

3. graphs



② binary relations and their properties

relation on a set ⇒ binary relation; domain = $A \times A$.

1) properties

reflexivity

$\forall x \in A, x R x$

for all points

symmetry

$\forall x, y \in A \ (x R y \longrightarrow y R x)$

for all pairs

transitivity

$\forall x, y, z \in A \ (x R y \wedge y R z \longrightarrow x R z)$

vacuously true if no relations / insufficient relation

2) proofs:

↳ reflexivity: universal generalisation
↳ symmetry: universal generalisation
} direct proofs possible

↳ transitive: exhaustion (direct) or contradiction (indirect)

③ Equivalence relations to describe partitions

1) equivalence relations: a relation on a set that is reflexive, symmetric and transitive, denoted $\sim$

2) equivalence classes: suppose A is a set and R is a relation on A. $\forall a \in A$, the equivalence class of a, denoted $[a]_\sim = \{x \in A : x \, R \, a\}$. An element is a representative

↳ equivalence classes are either equal or disjoint (Lemma 6.34)

$x \sim y \iff [x]_\sim = [y]_\sim \iff [x]_\sim \cap [y]_\sim \neq \emptyset$

$y \sim x$ by symmetry. So for any element $z \in [x]$, $z \sim y$ by transitivity, so $z \in [y]$. vice versa.

$[x] \cap [y] = [x]$ by idempotent law. By reflexivity, $x \sim x$, so $x \in [x]$. So the intersection must contain at least x.

Suppose $z \in [x] \cap [y]$. So $z \sim x$ and $z \sim y$. By transitivity, $x \sim y$.

3) Dividing a set by equivalence relation

$A/\sim \; = \{[x]_\sim : x \in A\}$ — "the quotient of A by $\sim$"

4) Equivalence classes and relations can be used to describe partitions

1. we seek to prove that $A/\sim$ is a partition of set A.
2. $A/\sim$ by definition is a set.
3. we seek to first prove that every element of $A/\sim$ is a nonempty subset of A.
   - 3.1 Let S be a specific but arbitrary element of $A/\sim$. $S \in A/\sim$.
   - 3.2 By definition of $A/\sim$, there exists $x \in A$ s.t. $S = [x]_\sim$
   - 3.3 By definition of equivalence classes and $\subseteq$, $[x]_S \subseteq A$ - so $S \subseteq A$.
   - 3.4 by reflexivity, $x \sim x$, so $x \in [x]$. since $S = [x]$, $x \in S$.
   - ∴ 3.5 so S is a non-empty subset of A.
4. we seek to prove that every element of A is in at least one element of A.
   - 4.1 let $x \in A$
   - 4.2 then by reflexivity, xRx.
   - ∴ 4.3 so $x \in [x] \in A/\sim$.
5. we seek to prove that every element of A is in at most one element of A.
   - 5.1 suppose there exists $x \in A$ that is in two elements of $A/\sim$ S_1 and S_2.
   - 5.2 By definition of $A/\sim$, $S_1 = [y_1]$, $S_2 = [y_2]$, $y_1, y_2 \in A$.
   - 5.3 $x \in S_1 \wedge x \in S_2 \Rightarrow x \in [y_1] \wedge x \in [y_2]$.
   - 5.4 so $[y_1] \wedge [y_2] \neq \emptyset$.
   - 5.5 by lemma 6.3.4, $S_1 = [y_1] = [y_2] = S_2$.

partition ⟷ quotient of an equivalence relation

∴ 6. Dividing a set by equivalence relation produces a partition -

⇒ so equivalence relations (and dividing set by them) can be used to describe partitions, with components as equivalence classes

5) types of equivalence classes

1. equality
2. congruence relations: an equivalence relation s.t. algebraic operations done with equivalent elements will yield equivalent elements.

(eg.) congruence modulo n relation

remainders are same when you divide by n

$a \equiv b \pmod n$ ie. $a - b = nk$, $n, k \in \mathbb{Z}$ ⟷ $n \mid (a-b)$

The difference between a and b is some multiple of n.

is a type of congruence relation because only similar (equivalent) numbers will have this sort of relationship.

↳ $[x] = \{nk + q\}$, $0 \leq q < n$ and $n, k, q \in \mathbb{Z}$

good way to set 'repeating' numbers in terms of divisibility

④ partial orders to describe sets where elements have a must be done before (or at the same time as) relation (eg. sets of tasks)

1) antisymmetry (modelling inability to multi-task)

$$\forall x, y \; (x R y \wedge y R x \rightarrow y = x)$$

2) partial and total orders

Let A be a set and R be a (binary) relation on A.

partial order $\preceq$ — if $x \leq y \wedge x \neq y \Rightarrow x \prec y$

(1) R is antisymmetric
(2) R is transitive
(3) R is reflexive

(A, R) partially ordered set
↳ tuple of sets

total order $\preceq^*$

(1) a total order is a partial order
(2) set is total: $\forall x, y \; (xRy \vee yRx)$

negation: partiality
$\exists x, y$ s.t. $x \not R y \wedge y \not R x$

3) equivalence relations and partial orders

↳ equivalence relations can be partial order, but not necessarily so. symmetry and antisymmetry are not related; they are independent

eg. equality $xRy \iff x = y$ — symmetry, transitivity, reflexivity, antisymmetry } equivalence relation } partial order ↓ total order if $|A| \leq 1$

4) hasse diagrams

Let $\preceq$ be a partial order on set A. The hasse diagram satisfies:

$\forall x, y \in A$, $x \prec y$ and no $z \in A$ is s.t. $x \prec z \prec y$, then x is placed below y with a joining line, else no line joins x to y.

↳ So to convert from a binary relation graph: (1) remove reflexive
(2) remove transitive
(3) "rank"

5) min/max elements

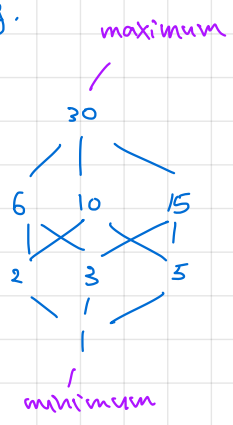
1. minimal: nothing below c (must always exist) — along its chain (smaller)
$\forall x \in A \; (x \preceq c \rightarrow c = x)$

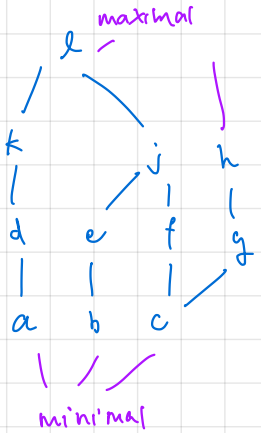
2. minimum: everything above c (need not exist) — connected to anything & smaller
$\forall x \in A \; c \preceq x$

3. maximal: nothing above c
$\forall x \in A \; (x \succeq c \rightarrow x = c)$

4. maximum: everything below c
$\forall x \in A \; c \succeq x$

eg.





(A) smallest element must be minimal

1. Let c be smallest element.
2. take any $x \in A$ s.t. $x \preccurlyeq c$
3. by smallestness, we know $c \preccurlyeq x$
4. so by antisymmetry $x = c$

(B) there must be a minimal element in a finite set

1. take any $c_0 \in A$
2. if c_0 is not minimal, find c_1 s.t. $c_1 \prec c_0$
3. continue this process.
4. $c_{n+1} \neq c_i$ because if so:
   - 4.1 $c_n \prec c_{n-1} \prec \dots \prec c_i = c_{n+1}$
   - 4.2 by transitivity, $c_n \preccurlyeq c_{n+1}$
   - 4.3 so by antisymmetry $c_n = c_{n+1}$ since $c_{n+1} \prec c_n$
   - 4.4 but this contradicts $c_{n+1} \prec c_n$
5. since A is finite, this process must end, at c_n. c_n must be minimal for this to end.

6) linearisation determining a valid sequence

Let A be a set and $\preccurlyeq$ be a partial order on A.
A linearisation of $\preccurlyeq$ is a total order $\preccurlyeq^*$ s.t. $\forall x, y \in A \; (x \preccurlyeq y \Rightarrow x \preccurlyeq^* y)$

Intuition: linearisation is a total order where the arrangement is s.t. if $x \preccurlyeq y$ in the partial order, then $x \preccurlyeq y$ in the total order.

Every partial order $\preccurlyeq$ has a linearisation $\preccurlyeq^*$.

Kahn's algorithm

Proof.

2. Suppose the run produces $A_0, A_1, \dots, A_n, c_0, c_1, \dots, c_{n-1}$ and $\preccurlyeq^*$.
3. Note $A = \{c_0, c_1, \dots, c_{n-1}\}$ because the removal of $c_0, c_1, \dots, c_{n-1}$ from A makes the set empty.
4. Note also that $\preccurlyeq^*$ is a total order on A because it is by definition only a renaming of the total order $\leqslant$ on $\{0, 1, \dots, n-1\}$.
5. 5.1. Take $x \in A$ and $c_j \in A$ such that $x \prec c_j$.
   5.2. Then $x \notin A_j$ as c_j is minimal in A_j.
   5.3. So $x \in A \setminus A_j = \{c_0, c_1, \dots, c_{j-1}\}$ by the choices of $A_0, A_1, \dots, A_j, c_0, c_1, \dots, c_{j-1}$.
   5.4. Let $i \in \{0, 1, \dots, j-1\}$ such that $x = c_i$.
   5.5. Then $x = c_i \preccurlyeq^* c_j$ by the definition of $\preccurlyeq^*$, as $i \leqslant j - 1 < j$. □

so x belongs to the set of all minimal elements of $A_0, A_1, \dots A_{j-1}$

so if we look at x from this set, by definition each element $c_i \preccurlyeq c_j$

so if we do this for any j, we see that the algorithm produces a total order

Kahn's Algorithm (1962)

Input: a finite set A, a partial order $\preccurlyeq$ on A.
(1) Set $A_0 := A$ and $i := 0$.
(2) Repeat until $A_i = \varnothing$:
(2.1) find a minimal element c_i of A_i wrt $\preccurlyeq$;
(2.2) set $A_{i+1} := A_i \setminus \{c_i\}$;
(2.3) set $i := i + 1$.
Output: a linearization $\preccurlyeq^*$ of $\preccurlyeq$ defined by setting, for all indices i, j,

$$c_i \preccurlyeq^* c_j \quad \Leftrightarrow \quad i \leqslant j.$$

(5) modular arithmetic

1) modular equivalences

equivalence classes

Theorem 8.4.1 Modular Equivalences

Let a, b, and n be any integers and suppose $n > 1$. The following statements are all equivalent:

$[x] = \{nk + q\}$

1. $n \mid (a - b)$
2. $a \equiv b \pmod{n}$
3. $a = b + kn$ for some integer k
4. a and b have the same (nonnegative) remainder when divided by n
5. $a \bmod n = b \bmod n$

Proof: We will show that (1) $\Rightarrow$ (2) $\Rightarrow$ (3) $\Rightarrow$ (4) $\Rightarrow$ (5) $\Rightarrow$ (1). It will follow by the transitivity of if-then that all five statements are equivalent.

So let a, b, and n be any integers with $n > 1$.

***Proof that* (1) $\Rightarrow$ (2):** Suppose that $n \mid (a - b)$. By definition of congruence modulo n, we can immediately conclude that $a \equiv b \pmod{n}$.

***Proof that* (2) $\Rightarrow$ (3):** Suppose that $a \equiv b \pmod{n}$. By definition of congruence modulo n, $n \mid (a - b)$. Thus, by definition of divisibility, $a - b = kn$, for some integer k. Adding b to both sides gives that $a = b + kn$.

***Proof that* (3) $\Rightarrow$ (4):** Suppose that $a = b + kn$, for some integer k. Use the quotient-remainder theorem to divide a by n to obtain

$$a = qn + r \quad \text{where } q \text{ and } r \text{ are integers and } 0 \leq r < n.$$

So r is the remainder obtained when a is divided by n. Substituting $b + kn$ for a in the equation $a = qn + r$ gives that

$$b + kn = qn + r,$$

and subtracting kn from both sides and factoring out n yields

$$b = (q - k)n + r.$$

Now since $0 \leq r < n$, the uniqueness property of the quotient-remainder theorem guarantees that r is also the remainder obtained when b is divided by n. Thus a and b have the same remainder when divided by n.

***Proof that* (4) $\Rightarrow$ (5):** Suppose that a and b have the same remainder when divided by n. It follows immediately from the definition of the *mod* function that $a \bmod n = b \bmod n$.

***Proof that* (5) $\Rightarrow$ (1):** Suppose that $a \bmod n = b \bmod n$. By definition of the *mod* function, a and b have the same remainder when divided by n. Thus, by the quotient-remainder theorem, we can write

$$a = q_1 n + r \quad \text{and} \quad b = q_2 n + r \quad \text{where } q_1, q_2, \text{ and } r \text{ are integers and } 0 \leq r < n.$$

It follows that

$$a - b = (q_1 n + r) - (q_2 n + r) = (q_1 - q_2)n.$$

Therefore, since $q_1 - q_2$ is an integer, $n \mid (a - b)$.

2) modular arithmetic

Theorem 8.4.3 Modular Arithmetic

Let a, b, c, d, and n be integers with $n > 1$, and suppose

$$a \equiv c \pmod{n} \quad \text{and} \quad b \equiv d \pmod{n}.$$

Then

1. $(a + b) \equiv (c + d) \pmod{n}$
2. $(a - b) \equiv (c - d) \pmod{n}$
3. $ab \equiv cd \pmod{n}$
4. $a^m \equiv c^m \pmod{n}$ for every positive integer m.

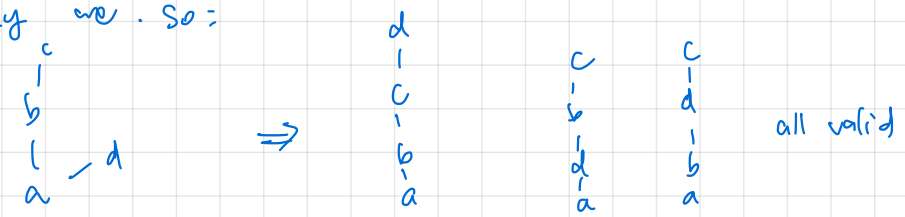
Notes

1- if R is reflexive, then minimum $|R| = |A| = n$, since all elements of A would be related to itself

2. because $[x]$ is itself a partition, with equivalence classes for congruence mod m, you just need to show that $\exists\, x, y \in \mathbb{Z}$ s.t. $[x]_{am} \subseteq [y]_{an}$
ie- one of the equivalence classes fulfils

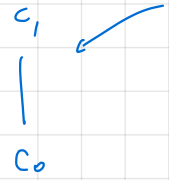
3. be careful of converse errors when proving min/max

4. in a partial order, linearising it only requires that all that need to be done consecutively are. So:



all valid

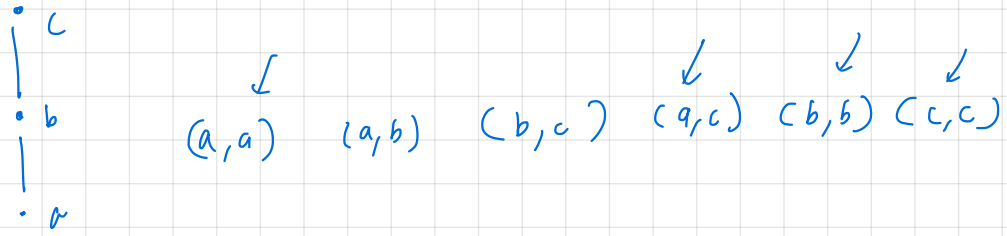
5. $\succ$ → order $\not\succ$ → not the same as $\succ$, since not necessarily opposite.
eg. reflexive relations



$c_0 \prec c_1$

$c_0 \not\prec c_1 = c_0$ → if $|\text{set}| \leq 0$, then even if $\not\prec$ it is vacuously true that $c_1 = c_0$

6. when reading Hasse diagrams, remember to include all along main and itself.



Functions

① function

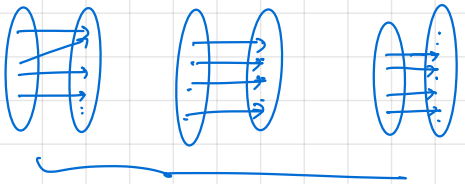
↳ a function from A to B is an assignment to each element of A to exactly one element in B.

$f: A \to B$ $\qquad f: x \mapsto f(x)$ (image)

A: domain, B: codomain

codomain ≠ NOT output set, but just domain

eg $f(x) = \begin{cases} \text{output set for range of } x \\ 0 \quad \text{otherwise} \end{cases}$ } domain



all well defined
↳ not everything in codomain must be mapped to
↳ everything in domain must be mapped from, and only to 1 thing (1 arrow)

1) identity function

↳ denoted id_A, is the function $f: A \to A$ s.t. $\forall x \in A \quad f(x) = x = id_A(x)$

2) equality of functions

1. domains & codomains are equal
2. $\forall x \in A \quad f(x) = g(x)$

} with all input elements from domain, corresponding outputs are identical

② objects as functions

sequences bijection

$a_1, a_2 \ldots$ can be represented by $a: \mathbb{Z}_{\geq 0} \to Y \quad a: n \mapsto a_n$

↳ so any function with domain $\mathbb{Z}_{\geq m}$ for some $m \in \mathbb{Z}_{\geq 0}$ is equivalent to a sequence

strings

1) definition

a string or a word is an expression of the form $a_0 a_1 \ldots a_{l-1}$, l is length. order & repetition matter.

↳ $l \in \mathbb{Z}_{\geq 0}$ and $a_0, a_1 \ldots a_{l-1} \in A$.

↳ A = { set of characters } $\quad A^*$ = { set of all possible strings over A }

↳ ε = empty string = string of length 0

2) function definition

↳ $a_0 a_1 \ldots a_{l-1}$ can be represented by $a: \{0, 1, \ldots l-1\} \to A$, set of all characters,

$a(n) = a_n$

↳ so all functions $a: \{m, m+1 \ldots m+l-1\} \to A$ where $m \in \mathbb{Z}$ and $l \in \mathbb{Z}_{\geq 0}$ represent a string of length l over a. $\quad a(m)\, a(m+1)\, a(m+2) \ldots a(m+l-1)$

③ Function composition

definition $f: A \to B$ & $g: B \to C$ $\quad g \circ f : A \to C$ $\quad g \circ f = g(f(x))$

↳ for $g \circ f$ to be well defined, the codomain of f must equal the domain of g.

1) idempotency : $f \circ f = f$
2) no commutativity : $g \circ f \neq f \circ g$
3) associativity : $(h \circ g) \circ f = h \circ (g \circ f)$.

↳ same domain & codomain

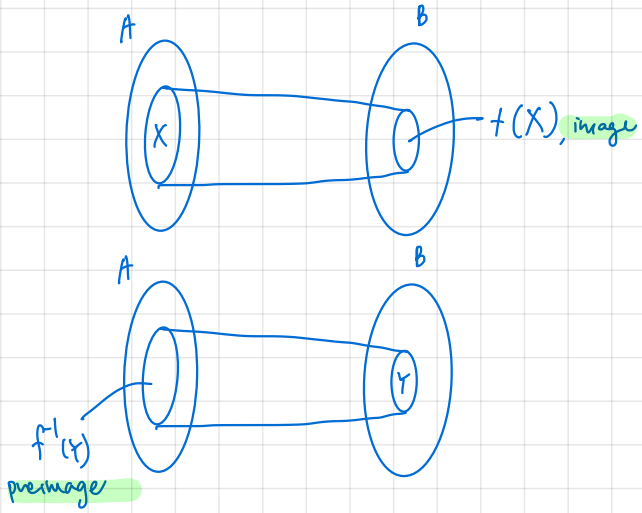
↳ $((h \circ g) \circ f)(x) = h \circ g(f(x)) = h(g(f(x))) = h((g \circ f)(x)) = h \circ (g \circ f)(x)$

④ set wise image & preimage

1) definitions

$X \subseteq A \longrightarrow f(X) = \{ f(x) : x \in X \}$

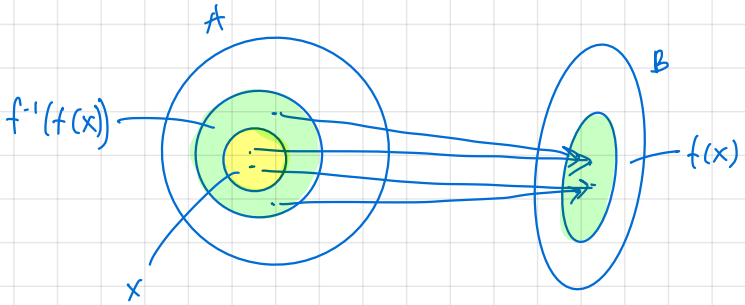
$Y \subseteq B \longrightarrow f^{-1}(Y) = \{ x \in A : f(x) \in Y \}$



2) properties

1. preimage of $f(X)$ will always contain X, but the converse is not true.

↳ intuition : well defined function → each $y \in B$ can have multiple inputs mapped to it, including $x \in X$. But not all $x \to y$ may be in X.

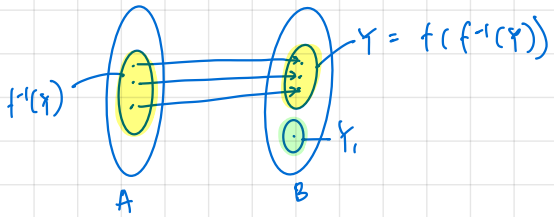


$X \subseteq f^{-1} f(X)$ ✓

$f^{-1} f(X) \subseteq X$ ✗

2. image of $f^{-1}(Y)$ will always be equal to Y, but the converse is not true.

↳ intuition : well defined function can have elements of domain not mapped from domain. So their preimages is an empty set, and that image is also an empty set.



$Y = f f^{-1}(Y)$ ✓

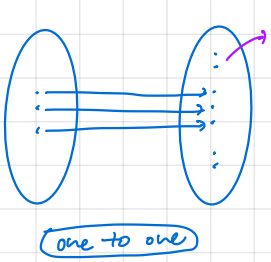
$f f^{-1}(Y) \subseteq Y$ ✗

⑤ injections

definition

$\forall x_1, x_2 \in A \quad f(x_1) = f(x_2) \longrightarrow x_1 = x_2$

to prove: find x_1, x_2 s.t. $f(x_1) = f(x_2) \Rightarrow$ show $x_1 = x_2$



one to one

injectivity of composed functions

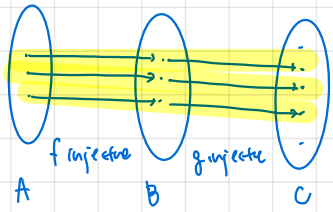
$f: A \to B \qquad g: B \to C$

(both) f, g injective $\longrightarrow$ g∘f injective

f is injective $\longleftrightarrow$ one can left compose it w some function to give an injection. ie. $gf(x)$ is injective.

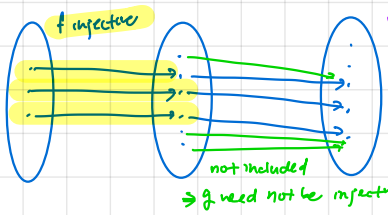
$\Uparrow$

g∘f injective $\longrightarrow$ f injective (inner)



g∘f injective

intuition: both injective → straight mapping all the way from A to C



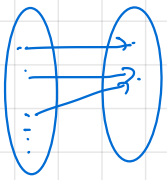
⇒ g need not be injective

intuition - for g∘f to be injective, elements of A map straight to g. so f must be injective. g need not be, since it can have other elements not mapped to from A that could have mapped to C.

⑥ surjections

definition $\forall y \in B, \exists x \in A \text{ s.t. } y = f(x)$

to prove: take $y \in B$. find $x \in A$ for each case. show that $f(x) = y$.



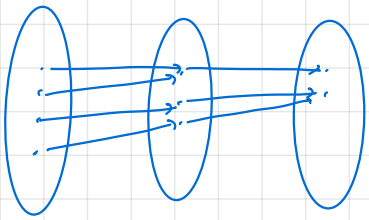
fully connected

ie. preimage exists

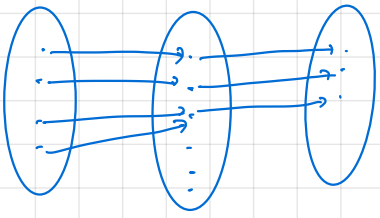
surjectivity of composed functions

(both) f, g surjective $\longrightarrow$ g∘f surjective

g∘f surjective $\longrightarrow$ g is surjective (outer)



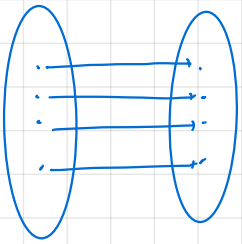
intuition: f fully connects, g fully connects f, g must fully connect



intuition: for g∘f to be fully connected from A to C, we know that all in C map to all in A. so all in C must have something mapped to in B at the intermediate step. But it is possible to have unmapped intermediate elements. so f need not be surjective.

⑦ bijections & inverses

definition $\forall y \in B \; \exists! x \in A \quad y = f(x)$



one to one & fully connected

f bijective iff f surjective ∧ f injective.

inverses

g is an inverse of f iff $\forall x \in A \; \forall y \in B \quad y = f(x) \leftrightarrow x = g(y)$. Denoted f^{-1}.

i) inverses are unique (proposition 9.3.17)

g_1, g_2 are inverses to $f: A \to B \longrightarrow g_1 = g_2$

2) f is bijective $\longleftrightarrow$ f^{-1} exists (theorem 9.3.19)

↳ intuition: well defined f^{-1} exists. By well defined nature of f & f^{-1}, f is surjective & injective.

bijective. Define $g: B \to A$, $g(y)$ unique $x \in A$ s.t. $y = f(x)$ $\forall y \in B$. g is well defined and fits definition of inverse.

3) composition of inverses

$(g \circ f)^{-1} = f^{-1} \circ g^{-1}$

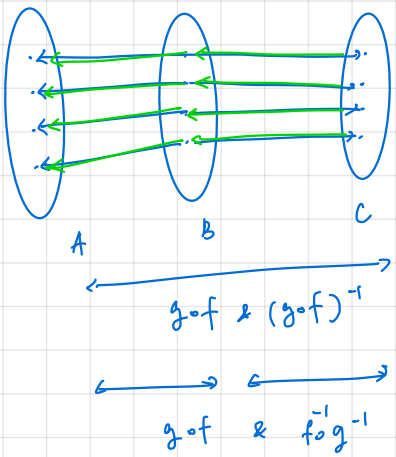
$z = g \circ f(x) = g(f(x))$

$g^{-1}(z) = f(x)$

$f^{-1}(g^{-1}(z)) = x$

compose

$f^{-1} g^{-1}(z) = x = g \circ f(x)$



4) order of bijections

↳ the least $n \in \mathbb{Z}^+$ s.t. $\underbrace{f \circ f \circ f \ldots}_{n\text{-many } f\text{'s}} = f$

Notes

1. well defined: { in defined domain; 1-1 } ϕ cannot see graph of function w/ non-empty domain

2. proofs w/ functions: consider $f: A \to B$ $f: x \mapsto y$ ($y = f(x)$)

   so $x \in A$, $y = f(x) \in B$, $(x, f(x)) \in$ output set $S \in A \times B$.

3. proving $\exists!$ → exists
   → unique

4. well defined functions w/ equivalence classes
   - intuition: put in $[x], [y] \mapsto f([x], [y])$
   - to prove, take a_1, a_2, b_1, b_2 s.t. $[a_1] \sim [b_1]$ ...
     ↳ by lemma, $a_1 \sim a_2$, $b_1 \sim b_2$
     ↳ expand algebraically
     ↳ show that final form $a_1 \sim b_1$, $a_2 \sim b_2 \Rightarrow [a_1 R b_1] = [a_2 R b_2]$

Induction & recursion

① sequences & sets

sequences and functions

sequence a_n $f(0)=a_0$ $f(1)=a_1$ ...

$\mathbb{Z}_{\geq 0}$ 0 1 ...

subsequence a_m $f(m)=a_m$ $f(m+1)=a_{m+1}$ ...

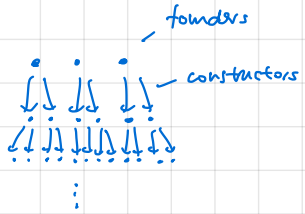
$\mathbb{Z}_{\geq 0}$ m $m+1$ ...

recursively defined sequences: definition of a_n involves a_0, a_1, a_{n-1} ... for all but finitely many $n \in \mathbb{Z}_{\geq 0}$

eg. F_0, F_1, F_2 ... $F_0 = 0$, $F_1 = 1$, $F_{n+2} = F_{n+1} + F_n$

↳

a_n	0	1	$F(0)+F(1)$	$F(1)+F(2)$
$\mathbb{Z}_{\geq 0}$	0	1	2	3



recursively defined sets

(base clause) certain elements, founders are in S. $c \in S$.

(recursion clause) specify certain functions, called constructors, under which the set is closed.

if $x \in S \longrightarrow f(x) \in S$.

(minimality clause) membership for S can always be demonstrated by finitely many applications of the base & recursion clauses.

④ useful manipulations in sequences

summation

1. $\sum_{k=m}^{n} a_k = \sum_{k=m}^{n-1} a_k + a_n$

2. method of cancellation by compensation to create subtraction of successive terms

eg. $\sum_{k=1}^{n} \frac{1}{k(k+1)} = \sum_{1} \frac{k+1-k}{k(k+1)} = \sum_{1} \frac{1}{k} - \frac{1}{k+1}$

$= \frac{1}{1} - \cancel{\frac{1}{2}} + \cancel{\frac{1}{2}} - \cancel{\frac{1}{3}} + \cancel{\frac{1}{3}} - \frac{1}{4}$ ... $= 1 - \frac{1}{n+1}$

3. $c \sum_{k=m}^{n} a_k + d \sum_{k=m}^{n} b_k = \sum_{k=m}^{n} (c \cdot a_k + d \cdot b_k)$

↳ same in $\prod_{k=m}^{n}$

4. change of variable (just sub)

② mathematical induction on sequences

1) intuition: trying to prove that $\forall n \in \mathbb{Z}_{\geq m}$, $P(n)$ is true.

1. show that $P(m)$ is true.
2. show that if $P(n)$ is true, $P(n+1)$ is true.
3. Then true, since cascading implication.

eg. $n \in \mathbb{Z}_{\geq 0}$.

1. $P(0)$
2. $P(n) \rightarrow P(n+1)$

so $P(0) \rightarrow P(1)$, $P(1) \rightarrow P(2)$, ⋮ } $P(n)$ $\forall n \in \mathbb{Z}_{\geq 0}$

2) techniques

1. (base step) prove that base case is true.
2. (induction step) suppose $P(n)$ is true for all $n \in \mathbb{Z}_{\geq m}$.
   - ↳ express the $n+1^{th}$ case in terms of the n^{th} case, such that it can apply to base.
   - ↳ use the truth of the n^{th} case $P(n)$ to prove truth of $n+1^{th}$ case.

③ strong MI on recursively defined sequences

1) intuition: to prove that $\forall n \in \mathbb{Z}_{\geq m}$ $P(n)$. eg. 2 base cases

1. show that $P(m)$, $P(m+1)$ several base cases are true.
2. show that $P(n) \wedge P(n+1) \wedge P(n+2) \rightarrow P(n+3)$
3. Then true, since cascading implication

eg. $n \in \mathbb{Z}_{\geq 0}$.

1. $P(0)$, $P(1)$, $P(2)$.
2. $P(n) \wedge P(n+1) \wedge P(n+2) \rightarrow P(n+3)$

so $P(0) \wedge P(1) \wedge P(2) \rightarrow P(3)$
$P(1) \wedge P(2) \wedge P(3) \rightarrow P(4)$
⋮

2) technique

1. (base step) prove that base cases are true.
2. (induction step) suppose $P(n)$, $P(n+1)$ ... are true.
   - ↳ express the $n+\ell^{th}$ case in terms of the prior cases.
   - ↳ use the truth of the prior cases to prove the truth of the $n+\ell^{th}$ case.

3) Fandation of MI and strong MI

↳ MI is a particular case of strong MI

↳ in general MI is to prove a property about a sequence (function) where domain is natural numbers, each linked consecutively

↳ its truth is based on the idea that you can reach any case by incrementing from the least element finitely many times

⇓

utilises the well ordering principle (Theorem 8.2.10)

Every non-empty subset of $\mathbb{Z}_{\geq 0}$ has a smallest element.

④ structural induction on recursively defined sets

↳ MI & strong MI are particular cases of strong MI, where set is $\mathbb{Z}_{\geq 0}$. So founder element is 0 (or m) and constructor f is $f : x \to x+1$

↳ we can generalise inductive proof technique to any recursively defined set

(technique)

1. define some property $P(x)$ about an element x in the set S.
2. prove $P(c)$ for all founders c.
3. prove that $P(x) \longrightarrow P(f(x))$ for all cases.

- quantifiers — how to think about it? → testing and being precise

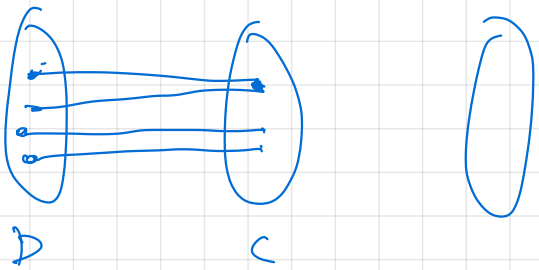
- proofs — how to be precise?
  - proofread
  - trying to attack it

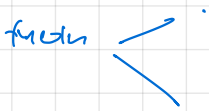
- eqv. classes — concept!
- well defined functions for eqv. class

- how to visualise concepts? — quantifiers — eqv. classes & relations } as sets
  - functions and relations!
    - injective / surjective / bijective

- definitions in proofs?

- constructing proofs?





~~5~~ — 7 3 — modulo 3

7 — reflexive | does exist

11 — itself

Cardinality

① Cardinality

1) finite sets : sets with a finite number of elements.

2) infinite sets : a set B is infinite $\iff \exists A$ s.t. $A \subset B$ and $|A| = |B|$ (T8 Q3)

intuition : $A = B \setminus \{b\}$. $|B \cup c|$

3) cardinality : the abstract notion of size → finite sets — number of elements
→ infinite sets — ?

② functions & cardinality

1) bijections & same cardinality

finite sets, pwof by MI ↑ Cantor, infinite sets ↑

$|A| = |B| \iff$ there is a bijection $f : A \to B$ (Theorem 10.1.1, 10.2.1)

2) properties of equal cardinality → it is an equivalence relation

reflexivity

$\forall A, |A| = |A|$.

bijection : id_A

symmetry

$|A| = |B| \to |B| = |A|$

bijection exists. Find inverse. inverse also a bijection, shows that $|B| = |A|$.

transitivity

$|A| = |B|$ and $|B| = |C|$, $|A| = |C|$

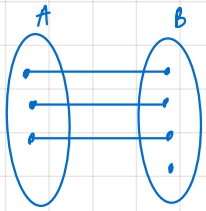
bijection $f: A \to B$, $g: B \to C$ exists. $(g \circ f)^{-1} = f^{-1} \circ g^{-1}$. $(g \circ f)^{-1}$ exists, so bijection $g \circ f$ exists. $|A| = |C|$.

3) pigeonhole & dual pigeonhole principle

pigeonhole principle
Theorem 10.1.2

A and B are finite.

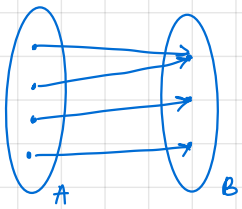
$|A| \leq |B| \iff$ injection $A \to B$.



dual pigeonhole principle
Theorem 10.1.3

A and B are finite.

Then $|A| \geq |B| \iff$ there is a surjection $A \to B$.



② Countability of sets

1) Cantor's definition of countability : a set is countable ↔ finite or same cardinality as $\mathbb{Z}_{\geq 0}$

2) sequences and countability

Note 10.3.4 : An infinite set is countable ↔ there is a sequence $b_0, b_1, \ldots$ in which every element of B appears exactly once.

↳ using representation of sequences through functions (bijections)

Lemma 10.3.7 : An infinite set B is countable ↔ there is a sequence $b_0, b_1, \ldots$ in which every element of B appears (w/ addtl stuff or repetition).

↳ take away non-B, take away repetitions → 10.3.4 → countable

3) key properties of countable sets → consider cases of both finite & infinite

1. any subset A of a countable set B is countable (proposition 10.3.5)
   ↳ A ↗ finite ⇒ countable
   ↘ infinite → B has sequence → take away elements in B not A → left w/ sequence of A ⇒ countable
2. Every infinite set has a countable (infinite) subset (proposition 10.3.6)
   ↳ for every $n \in \mathbb{Z}_{\geq 0}$ choose $b_n \in B$. $b_n \neq b_j$ since set is infinite. So this subset is countable.

③ Cantor's diagonal argument

1) intuition — For sets that have elements that themselves have infinite elements, and their equality is judged by whether they 'contain' all the same elements.

1. proof by contradiction. suppose A is countable. Then A has a sequence that contains all elements of A.
2. enumerate them

$$\begin{matrix} a_{00} & a_{01} & a_{02} & \ldots \\ a_{10} & a_{11} & a_{12} & \ldots \\ a_{20} & a_{21} & a_{22} & \ldots \end{matrix}$$

3. Find some b s.t. $b_0 \neq a_{00}$, $b_1 \neq a_{11}$, $b_2 \neq a_{22}$ .. → then b is not in the sequence since it can never equal any a_{ij}. This contradicts 1.

2) countable & uncountable sets

$\mathbb{Z}$ $\mathbb{Q}$

$\mathbb{R}$

$P(A)$, where A is countable infinite (Theorem 10.4.3)

$\mathbb{C}$

④ countability of operated sets

1) unions :

1. countable infinite ∪ finite → finite sequence first, append after (T8 Q1)
2. countable infinite ∪ countable infinite → alternating sequence (proposition 10.4.1)

Let A, B be countable infinite sets. Then $A \cup B$ is countable.

3. finite ∪ finite → finite
4. B is infinite, C is finite. bijection $B \cup C \to B$ exists. so $|B| = |B \cup C|$ (T8Q2) (finite, infinite)

2) cartesian product : $\mathbb{Z}^+_{10} \times \mathbb{Z}^+_{10}$ is countable (Theorem 10.4.2)

3) infinite unions

1. $\bigcup_{i \in \mathbb{Z}^+_0} S_i = S_0 \cup S_1 \cup S_2 \ldots S_i$ is countable if S_i is countable infinite. (T8Q4)

↳ can be written as

S_0 a_{01} a_{02} ...
S_1 a_{11} a_{12} ...
⋮ a_{21} a_{22} ...
S_i ⋮

, and each a_{ij} can be mapped to $x \in \mathbb{Z}^+_{10} \times \mathbb{Z}^+_{10}$, as a bijection.

$|\mathbb{Z}^+_{10} \times \mathbb{Z}^+_{10}| = |\mathbb{Z}^+_{10}|$. countable.

2. $\bigcup_{i \in \mathbb{Z}^+_0} S_i$ is countable if S_i is countable. (T8Q5)

↳ $\bigcup_{i \in \mathbb{Z}} S_i \subseteq \bigcup_{i \in \mathbb{Z}} A_i$, where A_i is countable infinite. Any subset of countable is countable.

use this

Notes

1. if not stated, then must consider both finite & infinite cases.
2.

10. Counting and probability

① Basics of counting

i) no. of elements in a list & application

Theorem 9.1.1 The Number of Elements in a List

If m and n are integers and $m \le n$, then there are
$$n - m + 1$$
integers from m to n inclusive.

Q: from 100 to 999, how many divisible by 5?

100 101 102 103 104 105 106 107 108 109 110 ... 994 995 996 997 998 999

5×20 5×21 5×22 5×199

↳ 20, 21, 22 ... 199

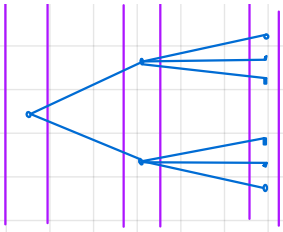
creating a subset ⇒ so there are $199 - 20 + 1 = 180$

2) rules of counting

(multiplication rule)

Theorem 9.2.1 The Multiplication/Product Rule

If an operation consists of k steps and
the first step can be performed in n_1 ways,
the second step can be performed in n_2 ways
(regardless of how the first step was performed),
:
the k^{th} step can be performed in n_k ways
(regardless of how the preceding steps were performed),
Then the entire operation can be performed in
$n_1 \times n_2 \times n_3 \times \dots \times n_k$ ways.



only holds if each step on tree has same no. of ways for each branch

↳ be careful if there are restriction

(addition rule)

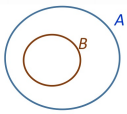
Theorem 9.3.1 The Addition/Sum Rule

Suppose a finite set A equals the union of k distinct mutually disjoint subsets $A_1, A_2, \dots, A_k$. Then
$$|A| = |A_1| + |A_2| + \dots + |A_k|.$$

(difference rule)

Theorem 9.3.2 The Difference Rule

If A is a finite set and $B \subset A$, then
$$|A \setminus B| = |A| - |B|.$$



The difference rule holds for the following reason:
If $B \subseteq A$, then the two sets B and $A \setminus B$ have no elements in common and $B \cup (A \setminus B) = A$.

Hence, by the addition rule,
$$|B| + |A \setminus B| = |A|.$$

Subtracting $|B|$ from both sides gives the equation
$$|A \setminus B| = |A| - |B|.$$

(inclusion / exclusion rule)

Theorem 9.3.3 The Inclusion/Exclusion Rule for 2 or 3 Sets

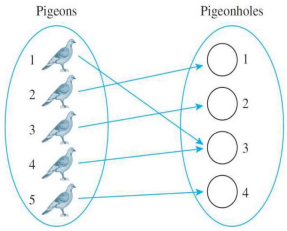
If A, B, and C are any finite sets, then
$$|A \cup B| = |A| + |B| - |A \cap B|$$
and
$$|A \cup B \cup C| = |A| + |B| + |C| - |A \cap B| - |A \cap C| - |B \cap C| + |A \cap B \cap C|$$

② pigeonhole principle

(PHP)

Pigeonhole Principle (PHP)

A function from one finite set to a smaller finite set cannot be one-to-one: There must be at least 2 elements in the domain that have the same image in the co-domain.

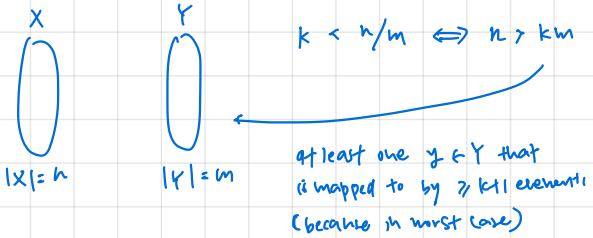


ie. not injective

(generalised PHP)

Generalized Pigeonhole Principle

For any function f from a finite set X with n elements to a finite set Y with m elements and for any positive integer k, if $k < n/m$, then there is some $y \in Y$ such that y is the image of at least $k + 1$ distinct elements of X.



③ counting subsets of a set : PnC

1) permutations of a set (no repetition, order matters)

Theorem 9.2.2 Permutations

The number of permutations of a set with n ($n \geq 1$) elements is **$n!$**

2) r combinations & permutations

r permutations → remove permutations r! → r combinations

Theorem 9.2.3 r-permutations from a set of n elements

If n and r are integers and $1 \leq r \leq n$, then the number of r-permutations of a set of n elements is given by the formula

$$P(n, r) = n(n-1)(n-2)\ldots(n-r+1)$$ first version

or, equivalently,

$$P(n, r) = \frac{n!}{(n-r)!}$$ second version

but permuted

Theorem 9.5.1 Formula for $\binom{n}{r}$

The number of subsets of size r (or r-combinations) that can be chosen from a set of n elements, $\binom{n}{r}$, is given by the formula

$$\binom{n}{r} = \frac{P(n,r)}{r!}$$

Recall that $P(n,r) = \frac{n!}{(n-r)!}$

or, equivalently,

$$\binom{n}{r} = \frac{n!}{r!(n-r)!}$$

where n and r are non-negative integers with $r \leq n$.

3) permutation of a set w/ indistinguishable objects

Theorem 9.5.2 Permutations with Sets of Indistinguishable Objects

Suppose a collection consists of n objects of which

n_1 are of type 1 and are indistinguishable from each other
n_2 are of type 2 and are indistinguishable from each other
:
n_k are of type k and are indistinguishable from each other

and suppose that $n_1 + n_2 + \ldots + n_k = n$. Then the number of distinguishable permutations of the n objects is

$$\binom{n}{n_1}\binom{n-n_1}{n_2}\binom{n-n_1-n_2}{n_3}\cdots\binom{n-n_1-n_2-\cdots-n_{k-1}}{n_k} = \frac{n!}{n_1!n_2!n_3!\cdots n_k!}$$

intuition: choosing $n_1, n_2 \ldots n_k$ positions to put these indistinguishable objects in n positions

another intuition: deprermuting double counts

4) multisets of size r as partitions

Definition: Multiset

An ***r*-combination with repetition allowed**, or **multiset of size *r***, chosen from a set X of n elements is an unordered selection of elements taken from X with repetition allowed.

If $X = \{x_1, x_2, \cdots, x_n\}$, we write an r-combination with repetition allowed as $[x_{i_1}, x_{i_2}, \cdots, x_{i_r}]$ where each x_{i_j} is in X and some of the x_{i_j} may equal each other.

Theorem 9.6.1 Number of r-combinations with Repetition Allowed

The number of r-combination with repetition allowed (multisets of size r) that can be selected from a set of n elements is:

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

This equals the number of ways r objects can be selected from n categories of objects with repetitions allowed.

Intuition: r to be chosen w/ repetition allowed ⇒ categorising ⇒ need n-1 partitions

↳ total: r + (n-1) positions

choose r positions for objects

choose n-1 positions for partitions

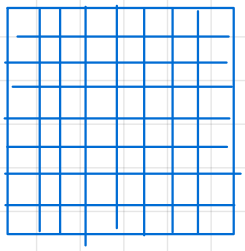
↳ = ↵

	Category 1	Category 2	Category 3	Category 4
[1, 1, 1]:	xxx	|	|	|
[1, 3, 4]:	x	|	| x	| x
[2, 4, 4]:		| x	|	| xx

④ The rook problem

in how many ways can one arrange k rooks on an $m \times n$ board so that they do not attack each other?

$k \leq m, n \Rightarrow$ k rooks must be different column AND different row



because there are $m \times n$ choices initially → place one, then $m \times n - 1$ ... permutation formula. But better to think of it as using multiplication rule.

total no. of ways $= m \times n \, P_k$

Application : modelling scenario where there are m category 1, n category 2, and any 2 objects cannot be same m or n.

⑤ Theorems & combinatorial proofs

1) combinatorial proofs : "intuitive" proofs
   ↳ uses counting as the basis of proofs
   ↳ bijective proofs (eg. in cardinality)
   ↳ double counting : use two equivalent ways to count to derive an identity

2) identities

Pascal's formula

Theorem 9.7.1 Pascal's Formula

Let n and r be positive integers, $r \le n$. Then

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

(x chosen / x not chosen)

intuition : mutually exclusive ways of choosing - chosen set either includes some element x. So you sum these outcomes.

Binomial theorem

$\binom{n}{r}$ is called binomial coefficient

Theorem 9.7.2 Binomial Theorem

Given any real numbers a and b and any non-negative integer n,

$$(a+b)^n = \sum_{k=0}^{n} \binom{n}{k} a^{n-k} b^k$$
$$= a^n + \binom{n}{1}a^{n-1}b^1 + \binom{n}{2}a^{n-2}b^2 + \cdots + \binom{n}{n-1}a^1 b^{n-1} + b^n$$

34

intuition: in expanding $(a+b)^n$, there are essentially n 'groups' of $(a+b)$, and from these you choose k b's and n-k a's, to get the powers. Since there are diff ways to choose, you combinatorially sum them.

using to simplify a sum:

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

Sum of choices

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

(left: equivalent to choosing subsets in powerset; right: size of powerset)

Lecture 12 Example 8 symmetry

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

⑥ Events and probability

1) Relative frequency interpretation

Equally Likely Probability Formula

If S is a finite sample space in which all outcomes are equally likely and E is an event in S, then the **probability** of E, denoted $P(E)$, is

$$P(E) = \frac{\text{The number of outcomes in } E}{\text{The total number of outcomes in } S} = \frac{|E|}{|S|}$$

9

Formula for the Probability of the Complement of an Event

If S is a finite sample space and A is an event in S, then

$$P(\bar{A}) = 1 - P(A)$$

2) probability axioms

Probability Axioms

Let S be a sample space. A probability function P from the set of all events in S to the set of real numbers satisfies the following axioms: For all events A and B in S,

1. $0 \le P(A) \le 1$
2. $P(\varnothing) = 0$ and $P(S) = 1$
3. If A and B are disjoint events ($A \cap B = \varnothing$), then $P(A \cup B) = P(A) + P(B)$

3) expected value

Definition: Expected Value

Suppose the possible outcomes of an experiment, or random process, are real numbers $a_1, a_2, a_3, \cdots, a_n$ which occur with probabilities $p_1, p_2, p_3, \cdots, p_n$ respectively. The **expected value** of the process is

$$\sum_{k=1}^{n} a_k p_k = a_1 p_1 + a_2 p_2 + a_3 p_3 + \cdots + a_n p_n$$

probability-weighted average

The expected value of the sum of random variables is equal to the sum of their individual expected values, regardless of whether they are independent.

For random variables X and Y (which may be dependent),

$$E[X + Y] = E[X] + E[Y]$$

More generally, for random variables $X_1, X_2, \cdots, X_n$ and constants $c_1, c_2, \cdots, c_n$,

$$E\left[\sum_{i=1}^{n} c_i \cdot X_i\right] = \sum_{i=1}^{n} (c_i \cdot E[X_i])$$

48

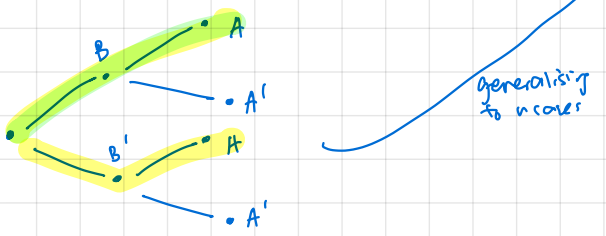
4) conditional probability

Definition: Conditional Probability

Let A and B be events in a sample space S. If $P(A) \neq 0$, then the **conditional probability of B given A**, denoted $P(B|A)$, is

$$P(B|A) = \frac{P(A \cap B)}{P(A)} \qquad 9.9.1$$

intuition: shrinking sample space since A must have occured



generalising to n cases

5) Bayes' theorem

Theorem 9.9.1 Bayes' Theorem

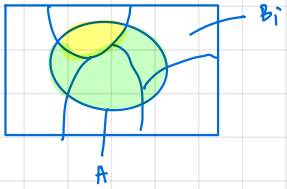
Suppose that a sample space S is a union of mutually disjoint events $B_1, B_2, B_3, \ldots, B_n$.

Suppose A is an event in S, and suppose A and all the B_i have non-zero probabilities.

If k is an integer with $1 \leq k \leq n$, then

$$P(B_k|A) = \frac{P(A|B_k) \cdot P(B_k)}{P(A|B_1) \cdot P(B_1) + P(A|B_2) \cdot P(B_2) + \cdots + P(A|B_n) \cdot P(B_n)}$$

$= P(A \cap B_k)$, where B occurs first then A, given B has occured



intuition: given that A has occured and it can occur in k ways, what is the probability that it happened through that one way?

7) independent events

1) independence

Definition: Independent Events

If A and B are events in a sample space S, then A and B are **independent**, if and only if,

$$P(A \cap B) = P(A) \cdot P(B)$$

2) independence of sets of events

Definition: Pairwise Independent and Mutually Independent

Let A, B and C be events in a sample space S. A, B and C are **pairwise independent**, if and only if, they satisfy conditions 1 – 3 below. They are **mutually independent** if, and only if, they satisfy all four conditions below.

1. $P(A \cap B) = P(A) \cdot P(B)$
2. $P(A \cap C) = P(A) \cdot P(C)$
3. $P(B \cap C) = P(B) \cdot P(C)$
4. $P(A \cap B \cap C) = P(A) \cdot P(B) \cdot P(C)$

1–3: pairwise; 1–4: mutual

79

11 - Graphs

① Graphs

1) Basic terminology

↳ a graph is defined by its vertices and edges. $G = (V, E)$

$\{v_1, v_2, \dots\}$ set of vertices $\quad \{e_1, e_2 \dots\}$ set of edges

representation depends on direction

(undirected graph)

Definition: Undirected Graph

An undirected **graph** G consists of 2 finite sets: a nonempty set V of **vertices** and a set E of **edges**, where each (undirected) edge is associated with a set consisting of either one or two vertices called its **endpoints**.

An edge is said to **connect** its endpoints; two vertices that are connected by an edge are called **adjacent vertices**; and a vertex that is an endpoint of a loop is said to be **adjacent to itself**.

An edge is said to be **incident on** each of its endpoints, and two edges incident on the same endpoint are called **adjacent edges**.

We write $e = \{v, w\}$ for an undirected edge e incident on vertices v and w.

(directed graph)

Definition: Directed Graph

A **directed graph**, or **digraph**, G, consists of 2 finite sets: a nonempty set V of **vertices** and a set E of **directed edges**, where each (directed) edge is associated with an ordered pair of vertices called its **endpoints**.

We write $e = (v, w)$ for a directed edge e from vertex v to vertex w.

2) Degree

(Handshake theorem)

↳ because edge is shared by two nodes → degree = 2 × no. of edges

Definition: Degree of a Vertex and Total Degree of an Undirected Graph

Let G be a undirected graph and v a vertex of G. The **degree** of v, denoted **deg(v)**, equals the number of edges that are incident on v, with an edge that is a loop counted twice.

The **total degree of G** is the sum of the degrees of all the vertices of G.

Theorem 10.1.1 The Handshake Theorem

If G is any graph, then the sum of the degrees of all the vertices of G equals twice the number of edges of G. Specifically, if the vertices of G are $v_1, v_2, \dots, v_n$, where $n \geq 0$, then

The total degree of $G = \deg(v_1) + \deg(v_2) + \dots + \deg(v_n)$

$= 2 \times$ (the number of edges of G).

(Total degree is even)

Corollary 10.1.2

The total degree of a graph is even.

Proposition 10.1.3

In any graph there are an even number of vertices of odd degree.

3) Types of graphs

(simple graph) no parallel edges or loops

Definition: Simple Graph

A **simple graph** is an undirected graph that does not have any loops or parallel edges. (That is, there is at most one edge between each pair of distinct vertices.)

Simple graph

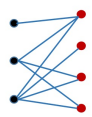
Non simple graph

Non simple graph

(bipartite graphs)

Definition: Bipartite Graph

A **bipartite graph** (or bigraph) is a simple graph whose vertices can be divided into two disjoint sets U and V such that every edge connects a vertex in U to one in V.

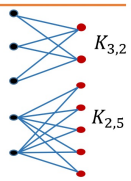


Bipartite graph

Definition: Complete Bipartite Graph

A **complete bipartite graph** is a bipartite graph on two disjoint sets U and V such that every vertex in U connects to every vertex in V.

If $|U| = m$ and $|V| = n$, the complete bipartite graph is denoted as $K_{m,n}$.

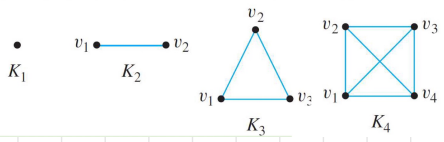


no. of edges = $m \times n$

(complete graph) fully connected

Definition: Complete Graph

A **complete graph** on ***n* vertices**, $n > 0$, denoted K_n, is a simple graph with n vertices and exactly one edge connecting each pair of distinct vertices .



Draw K_5.

edges in complete graph = ${}^nC_2$

(subgraphs)

Definition: Subgraph of a Graph

A graph H is said to be a **subgraph** of graph G if and only if every vertex in H is also a vertex in G, every edge in H is also an edge in G, and every edge in H has the same endpoints as it has in G.

H subgraph of $G \Leftrightarrow \forall v \in H, e \in H, v \in G$ & $e \in G$

↳ same idea as subset

② trails, paths and circuits

walk

A **walk from *v* to *w*** is a finite alternating sequence of adjacent vertices and edges of *G*. Thus a walk has the form

$$v_0 e_1 v_1 e_2 \dots v_{n-1} e_n v_n,$$

The **trivial walk** from *v* to *v* consists of the single vertex *v*.
A **closed walk** is a walk that starts and ends at the same vertex.

trail: no repeated edge

A **trail from *v* to *w*** is a walk from *v* to *w* that does not contain a repeated edge.

path: no repeated edge or vertices

A **path from *v* to *w*** is a trail that does not contain a repeated vertex.

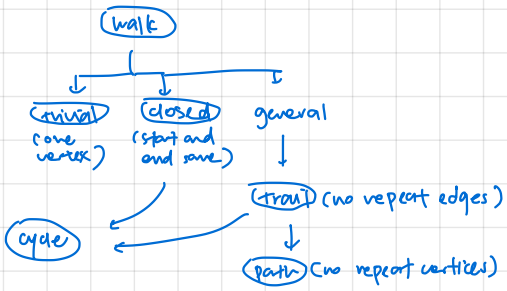
circuit/cycle: closed walk, no repeated edges (not necessarily all edges)

Circuit (or **cycle**): Let $n \in \mathbb{Z}_{\geq 3}$. An undirected graph $G(V, E)$ where $V = \{x_1, x_2, \cdots, x_n\}$ and $E = \{\{x_1, x_2\}, \{x_2, x_3\}, \cdots, \{x_{n-1}, x_n\}, \{x_n, x_1\}\}$ is called **a circuit/cycle**.

(A cycle is a closed walk that does not contain a repeated edge.)

A **simple circuit** (or **simple cycle**) is a circuit that does not have any other repeated vertex except the first and last.

An undirected graph is **cyclic** if it contains a loop or a cycle; otherwise, it is **acyclic**.



③ connectedness

A graph is connected if it is possible to travel from any vertex to any other vertex along a sequence of adjacent edges of the graph.

Definition: Connectedness

Two vertices *v* and *w* of a graph *G*=(*V*,*E*) are **connected** if and only if there is a walk from *v* to *w*.

The graph *G* is connected if and only if given *any* two vertices *v* and *w* in *G*, there is a walk from *v* to *w*. Symbolically,

G is connected iff ∀ vertices *v*, *w* ∈*V*, ∃ a walk from *v* to *w*.

proof: direct

show that any two vertices are connected or some share a common consin vertex

by contradiction

↳ suppose any two not connected

↳ (neighborhood) summed > n vertices directly connected to a vertex

by PHP

↳ total max no. of edges of vertices by definition

↳ total max no. possible (each can connect to n-1)

⇓

PHP

connected component: largest connected subgraph

Definition: Connected Component

A graph *H* is a **connected component** of a graph *G* if and only if

1. The graph *H* is a subgraph of *G*;
2. The graph *H* is connected; and
3. No connected subgraph of *G* has *H* as a subgraph and contains vertices or edges that are not in *H*.

circuits and connectedness

Lemma 10.5.3

If *G* is any connected graph, *C* is any circuit in *G*, and one of the edges of *C* is removed from *G*, then the graph that remains is still connected.

Essentially, the reason why Lemma 10.5.3 is true is that any two vertices in a circuit are connected by 2 distinct paths. It is possible to draw the graph so that one of these goes "clockwise" and the other goes "counter-clockwise" around the circuit.

④ Euler and hamiltonian circuits

1) Eulerian: traversing every edge exactly once, every vertex at least once

Eulerian circuits: every vertex at least once, every edge once, must return to same spot

Definition: Euler Circuit

Let *G* be a graph. An **Euler circuit** for *G* is a circuit that contains every vertex and traverses every edge of *G* exactly once.

→ **Theorem 10.2.4**

A graph *G* has an Euler circuit if and only if *G* is connected and every vertex of *G* has positive even degree.

Eulerian graph

Definition: Eulerian Graph

An **Eulerian graph** is a graph that contains an Euler circuit.

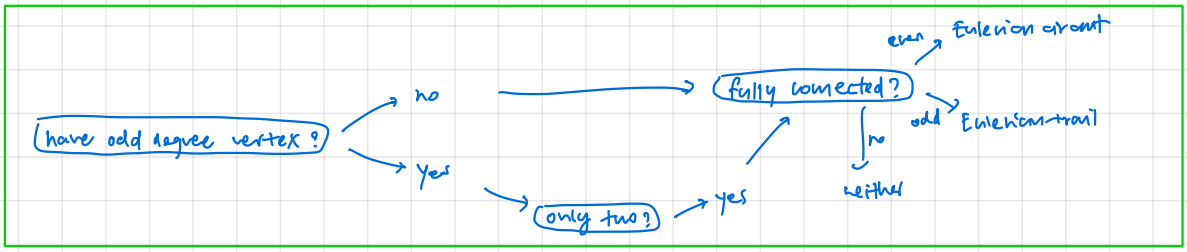
Eulerian trail: traverse every edge once, every vertex at least once

Definition: Euler Trail

Let *G* be a graph, and let *v* and *w* be two distinct vertices of *G*. An **Euler trail/path from *v* to *w*** is a sequence of adjacent edges and vertices that starts at *v*, ends at *w*, passes through every vertex of *G* at least once, and traverses every edge of *G* exactly once.

→ **Corollary 10.2.5**

Let *G* be a graph, and let *v* and *w* be two distinct vertices of *G*. There is an Euler trail from *v* to *w* if and only if *G* is connected, *v* and *w* have odd degree, and all other vertices of *G* have positive even degree.



2) Hamiltonian: travelling every vertex exactly once

no ways to prove that a graph contains a Hamiltonian circuit other than through trial and error

⇓

but can prove that it does not contain

Definition: Hamiltonian Circuit

Given a graph G, a **Hamiltonian circuit** for G is a simple circuit that includes every vertex of G. (That is, every vertex appears exactly once, except for the first and the last, which are the same.)

Definition: Hamiltonian Graph

A **Hamiltonian graph** (also called **Hamilton graph**) is a graph that contains a Hamiltonian circuit.

Proposition 10.2.6

If a graph G has a Hamiltonian circuit, then G has a subgraph H with the following properties:

1. H contains every vertex of G.
2. H is connected.
3. H has the same number of edges as vertices.
4. Every vertex of H has degree 2.

↳ find all possible subgraphs & test

5) matrix representation of graphs

1) matrix definitions

Definition: Matrix Multiplication

Let $\mathbf{A} = (a_{ij})$ be an $m \times k$ matrix and $\mathbf{B} = (b_{ij})$ an $k \times n$ matrix with real entries. The (matrix) product of **A** times **B**, denoted **AB**, is that matrix (c_{ij}) defined as follows:

$$\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1k} \\ a_{21} & a_{22} & \cdots & a_{2k} \\ \vdots & \vdots & & \vdots \\ a_{i1} & a_{i2} & \cdots & a_{ik} \\ \vdots & \vdots & & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mk} \end{bmatrix} \begin{bmatrix} b_{11} & b_{12} & \cdots & b_{1j} & \cdots & b_{1n} \\ b_{21} & b_{22} & \cdots & b_{2j} & \cdots & b_{2n} \\ & & & \cdot & & \\ & & & \cdot & & \\ & & & \cdot & & \\ b_{k1} & b_{k2} & \cdots & b_{kj} & \cdots & b_{kn} \end{bmatrix} = \begin{bmatrix} c_{11} & c_{12} & \cdots & c_{1j} & \cdots & c_{1n} \\ c_{21} & c_{22} & \cdots & c_{2j} & \cdots & c_{2n} \\ \vdots & \vdots & & \vdots & & \vdots \\ c_{i1} & c_{i2} & \cdots & c_{ij} & \cdots & c_{in} \\ \vdots & \vdots & & \vdots & & \vdots \\ c_{m1} & c_{m2} & \cdots & c_{mj} & \cdots & c_{mn} \end{bmatrix}$$

where

$$c_{ij} = a_{i1}b_{1j} + a_{i2}b_{2j} + \cdots + a_{ik}b_{kj} = \sum_{r=1}^{k} a_{ir}b_{rj}.$$

for all $i = 1, 2, \ldots, m$ and $j = 1, 2, \ldots, n$.

Definition: Identity Matrix

For each positive integer n, the $\boldsymbol{n \times n}$ **identity matrix**, denoted $I_n = (\delta_{ij})$ or just **I** (if the size of the matrix is obvious from context), is the $n \times n$ matrix in which all the entries in the main diagonal are 1's and all other entries are 0's. In other words,

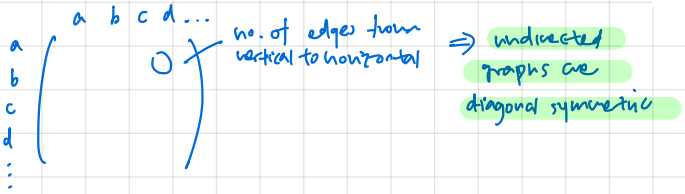
$$\delta_{ij} = \begin{cases} 1, & \text{if } i = j \\ 0, & \text{if } i \neq j \end{cases} \qquad \text{for all } i, j = 1, 2, \ldots, n.$$

2) representing graphs in matrices

Definition: Adjacency Matrix of a Directed Graph

Let G be a directed graph with ordered vertices $v_1, v_2, \ldots v_n$. The **adjacency matrix of *G*** is the $n \times n$ matrix $\mathbf{A} = (a_{ij})$ over the set of non-negative integers such that

a_{ij} = the number of arrows from v_i to v_j for all $i, j = 1, 2, \ldots, n$.



3) counting walks of length N

Theorem 10.3.2

If G is a graph with vertices $v_1, v_2, \ldots, v_m$ and **A** is the adjacency matrix of G, then for each positive integer n and for all integers $i, j = 1, 2, \ldots, m$,
the ij-th entry of $\mathbf{A}^n$ = the number of walks of length n from v_i to v_j.

6) isomorphic graphs and complements

1) isomorphic: same graph but orientated differently

checking for isomorphism

1. $|V_1| = |V_2|$ $|E_1| = |E_2|$ (bijection can exist)
2. can 'rearrange' orientation of nodes to get graph, without disconnecting / connecting anything

Two graphs G and G' that are the same except for the labeling of their vertices and edges are called *isomorphic*. In other words, there exists matching between the vertices such that two vertices are connected by an edge in G if and only if corresponding vertices are connected by an edge in G'.

Definition: Isomorphic Graph

Let $G = (V_G, E_G)$ and $G' = (V_{G'}, E_{G'})$ be two graphs.

G* is isomorphic to *G', denoted $G \cong G'$, if and only if there exist bijections $g: V_G \to V_{G'}$ and $h: E_G \to E_{G'}$ that preserve the edge-endpoint functions of G and G' in the sense that for all $v \in V_G$ and $e \in E_G$,

v is an endpoint of $e \Leftrightarrow g(v)$ is an endpoint of $h(e)$.

Alternative definition

Let $G = (V_G, E_G)$ and $G' = (V_{G'}, E_{G'})$ be two graphs.

G* is isomorphic to *G' if and only if there exists a permutation $\pi: V_G \to V_{G'}$ such that $\{u, v\} \in E_G \Leftrightarrow \{\pi(u), \pi(v)\} \in E_{G'}$.

Theorem 10.4.1 Graph Isomorphism is an Equivalence Relation

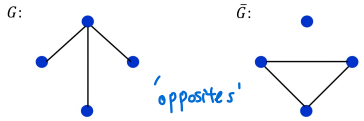
Let S be a set of graphs and let $\cong$ be the relation of graph isomorphism on S. Then $\cong$ is an equivalence relation on S.

1. reflexive: identity function (bijection)
2. symmetric: bijection → inverse also bijection
3. transitive: composed bijection still bijection

2) Complements — union of graph & complement ⇒ complete graph

Definition 1. If G is a simple graph, the *complement* of G, denoted $\bar{G}$, is obtained as follows: the vertex set of $\bar{G}$ is identical to the vertex set of G. However, two distinct vertices v and w of $\bar{G}$ are connected by an edge if and only if v and w are not connected by an edge in G.

The figure below shows a graph G and its complement $\bar{G}$.



A graph G and its complement $\bar{G}$.

Definition 2. A *self-complementary* graph is isomorphic with its complement.

7) planar graphs

Definition: Planar Graph

A **planar graph** is a graph that can be drawn on a (two-dimensional) plane without edges crossing.

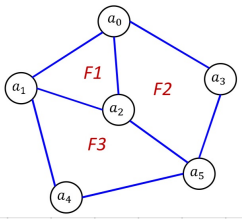
Check if planar graph — complete graphs > 4 non-planar

1. try to rearrange graph (find isomorphic) so that no edges cross

Euler's formula for counting regions

Euler's Formula

For a connected planar simple graph $G = (V, E)$ with $e = |E|$ and $v = |V|$, if we let f be the number of faces, then
$f = e - v + 2$



outer face counts

$e = 8$
$v = 6$
$f = 8 - 6 + 2 = 4$

8) Modelling graph problems

1. nodes → meaning
2. edges → meaning
3. some algorithm/operation → meaning ⇒ impact on properties
   - (total) degree
   - (total) weight
   - (total) no. of edges

Colouring problems

1. nodes → entity
2. edge → some r/s some cannot have in each other
3. algorithm → vertex colouring ⇒ four groups needed

Notes:

1. Each vertex can be connected to 1 ... n-1 vertices - OR 0 ... n-2 vertices, because n-1 case connected to all others.
2. A vertex can have n-1 edges iff there is no vertex w/ 0 edges
3. There are $2^{\binom{n}{2}}$ simple graphs. ← intuition: n vertices, $\binom{n}{2}$ edges to include or not.
4. Total no. of spanning trees in a complete graph = n^{n-2}

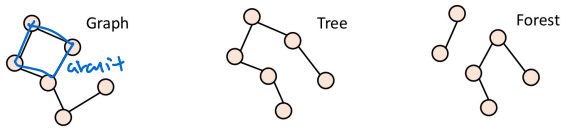
12. Trees

① Trees (special forms of undirected graphs)

1) trees

Definition: Tree

(The graph is assumed to be undirected here.)
A **graph** is said to be **circuit-free** if and only if it has no circuits.
A graph is called a **tree** if and only if it is circuit-free and connected.
A **trivial tree** is a graph that consists of a single vertex.
A graph is called a **forest** if and only if it is circuit-free and not connected.



4

2) vertices in a tree

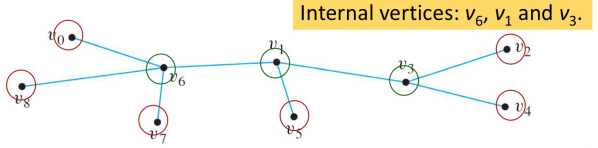
degree: $=1 \rightarrow$ leaf; $>1 \rightarrow$ internal

Definitions: Terminal vertex (leaf) and internal vertex

Let T be a tree. If T has only one or two vertices, then each is called a **terminal vertex** (or **leaf**). If T has at least three vertices, then a vertex of degree 1 in T is called a **terminal vertex** (or **leaf**), and a vertex of degree greater than 1 in T is called an **internal vertex**.

Example: Find all terminal vertices (leaves) and all internal vertices in the following tree:

Leaves: v_0, v_2, v_4, v_5, v_7 and v_8.
Internal vertices: v_6, v_1 and v_3.



9

3) properties of trees

non-trivial trees have ≥ 1 leaf

1. **Lemma 10.5.1**

Any non-trivial tree has at least one vertex of degree 1.

Proof: Let T be a particular but arbitrarily chosen non-trivial tree.
Step 1: Pick a vertex v of T and let e be an edge incident on v.
Step 2: While deg(v) > 1, repeat steps 2a, 2b and 2c:
2a: Choose e' to be an edge incident on v such that $e' \neq e$.
2b: Let v' be the vertex at the other end of e' from v.
2c: Let $e = e'$ *and* $v = v'$.
The algorithm must eventually terminate because the set of vertices of the tree T is finite and T is circuit-free. When it does, a vertex v of degree 1 will have been found.

7

3. Any non-trivial tree has at least two vertices of degree 1.

Proof:

1. Let T be a non-trivial tree.
2. Let S be the set of all paths in T.
3. Let P be the largest path in S (though not necessarily unique)
   3-1 Since T is non-trivial and trees are n-1 edges in T, a path can be of length 1 to n-1
   3-2 So there is a path of maximum length
4. The initial and final vertices in P must have degree 1
   4-1 suppose terminal vertex has degree 2.
   4-2 case 1: the other vertex joined by the other edge is not in P. Then adding it in would increase |P|. This contradicts that P is greatest.
   4-3 case 2: the other vertex is in P. Then it is cyclic. Then T is not a tree. This contradicts.

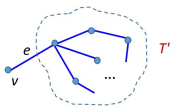
2. **Theorem 10.5.2**

Any tree with n vertices ($n > 0$) has $n - 1$ edges.

(intuition) Because trees are no circuits, can be rearranged to form a straight line

Proof: By mathematical induction.
Let the property $P(n)$ be "any tree with n vertices has $n - 1$ edges".
$P(1)$: Let T be a tree with one vertex. Then T has no edges.
So $P(1)$ is true.
Show that for all integers $k \geq 1$, if $P(k)$ is true then $P(k+1)$ is true.
Suppose $P(k)$ is true.

1. Let T be a particular but arbitrarily chosen tree with $k + 1$ vertices.
2. Since k is positive, $(k + 1) \geq 2$, and so T has more than one vertex.
3. Hence, by Lemma 10.5.1, T has a vertex v of degree 1, and has at least another vertex in T besides v.
4. Thus, there is an edge e connecting v to the rest of T.
5. Define a subgraph T' of T so that $V_{T'} = V_T - \{v\}$ and $E_{T'} = E_T - \{e\}$.
   5.1 The number of vertices of T' is $(k + 1) - 1 = k$.
   5.2 T' is circuit-free.
   5.3 T' is connected.
6. Hence by definition, T' is a tree.
7. Since T' has k vertices, by inductive hypothesis, number of edges of T' = (number of vertices of T') $- 1 = k - 1$.
8. But number of edges of T = (number of edges of T') $+ 1 = k$.
9. Hence $P(k+1)$ is true.



11

4) proving a graph is a tree ie. proving it is circuit free given connected

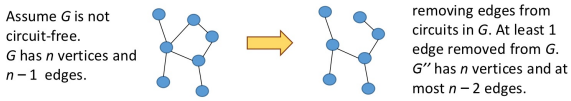
Theorem 10.5.4

If G is a connected graph with n vertices and $n-1$ edges, then G is a tree.

no circuits

Proof:

1. Suppose G is a particular but arbitrarily chosen graph that is connected and has n vertices and $n-1$ edges.
2. Since G is connected, it suffices to show that G is circuit-free.
3. Suppose G is not circuit free
   - 3.1 Let C be the circuit in G.
   - 3.2 By Lemma 10.5.3, an edge of C can be removed from G to obtain a graph G' that is connected.
   - 3.3 If G' has a circuit, then repeat this process: Remove an edge of the circuit from G' to form a new connected graph.
   - 3.4 Continue the process of removing edges from the circuits until eventually a graph G'' is obtained that is connected and is circuit-free.
   - 3.5 By definition, G'' is a tree.
   - 3.6 Since no vertices were removed from G to form G'', G'' has n vertices.
   - 3.7 Thus, by Theorem 10.5.2, G'' has $n-1$ edges.
   - 3.8 But the supposition that G has a circuit implies that at least one edge of G is removed to form G''.
   - 3.9 Hence G'' has no more than $(n-1)-1 = n-2$ edges, which contradicts its having $n-1$ edges.
   - 3.10 So the supposition is false.
4. Hence G is circuit-free, and therefore G is a tree.



16

5) Finding no. of non-isomorphic trees

1. $n \rightarrow n-1$ edges
2. total degree $= 2(n-1)$ by Handshake theorem
3. Find possible combinations of degrees

By Theorem 10.5.2, any tree with 4 vertices has 3 edges. And so by the Handshake Theorem, the tree has a total degree of 6.

Theorem 10.1.1 The Handshake Theorem

Given a graph $G=(V, E)$, the total degree of $G = 2|E|$.

Also, every non-trivial tree has at least two vertices of degree 1.

The only possible combinations of degrees for the 4 vertices are:

1, 1, 1, 3 and **1, 1, 2, 2**

Therefore, there are two non-isomorphic trees with 4 vertices.

13

(2) Rooted trees

Definitions: Rooted Tree, Level, Height

A **rooted tree** is a tree in which there is one vertex that is distinguished from the others and is called the **root**.

The **level** of a vertex is the number of edges along the unique path between it and the root.

The **height** of a rooted tree is the maximum level of any vertex of the tree.

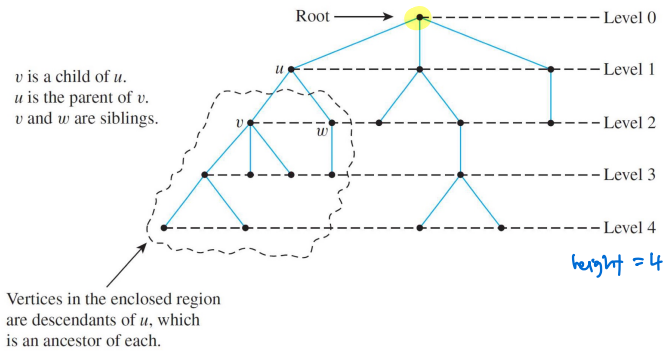
Definitions: Child, Parent, Sibling, Ancestor, Descendant

Given the root or any internal vertex v of a rooted tree, the **children** of v are all those vertices that are adjacent to v and are one level farther away from the root than v.

If w is a child of v, then v is called the **parent** of w, and two distinct vertices that are both children of the same parent are called **siblings**.

Given two distinct vertices v and w, if v lies on the unique path between w and the root, then v is an **ancestor** of w, and w is a **descendant** of v.

19



(3) Binary trees

1) binary trees

this matters!

Definitions: Binary Tree, Full Binary Tree

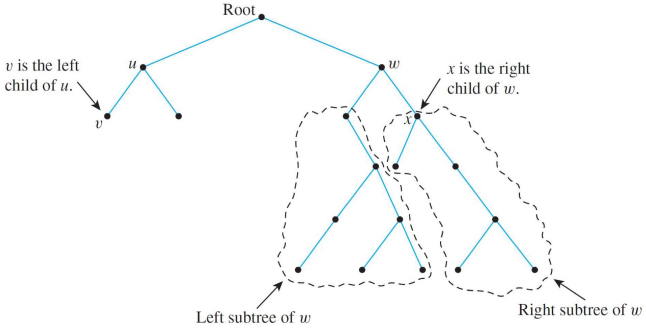
A **binary tree** is a rooted tree in which every parent has at most two children. Each child is designated either a **left child** or a **right child** (but not both), and every parent has at most one left child and one right child.

A **full binary tree** is a binary tree in which each parent has exactly two children.

Definitions: Left Subtree, Right Subtree

Given any parent v in a binary tree T, if v has a left child, then the **left subtree** of v is the binary tree whose root is the left child of v, whose vertices consist of the left child of v and all its descendants, and whose edges consist of all those edges of T that connect the vertices of the left subtree.

The **right subtree** of v is defined analogously.



2) properties of binary trees

1.

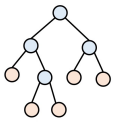
Theorem 10.6.1: Full Binary Tree Theorem

If T is a full binary tree with k internal vertices, then T has a total of $2k + 1$ vertices and has $k + 1$ terminal vertices (leaves).

↳ k internal, k + 1 terminal

Proof:

1. Every vertex, except the root, has a parent.
2. Since every internal vertex of a full binary tree has exactly two children, the number of vertices that have a parent is twice the number of parents, or $2k$.
   #vertices of T = #vertices that have a parent + #vertices that do not have a parent
   $= 2k + 1$
3. #terminal vertices = #vertices – #internal vertices
   $= 2k + 1 - k = k + 1$
4. Therefore T has a total of **$2k + 1$ vertices** and has **$k + 1$ terminal vertices**.



27

2.

Theorem 10.6.2

For non-negative integers h, if T is any binary tree with height h and t terminal vertices (leaves), then

$$t \le 2^h$$

Equivalently,

$$\log_2 t \le h$$

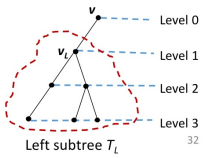
This theorem says that the maximum number of terminal vertices (leaves) of a binary tree of height h is 2^h. Alternatively, a binary tree with t terminal vertices (leaves) has height of at least $\log_2 t$.

29

Proof: By strong mathematical induction

1. Let $P(h)$ be "If T is any binary tree of height h, then the number of leaves of T is at most 2^h.
2. $P(0)$: T consists of one vertex, which is a terminal vertex. Hence $t = 1 = 2^0$.
3. Show that for all integers $k \ge 0$, if $P(i)$ is true for all integers i from 0 through k, then $P(k+1)$ is true.
4. Let T be a binary tree of height $k + 1$, root v, and t leaves.
5. Since $k \ge 0$, hence $k + 1 \ge 1$ and so v has at least one child.
6. We consider two cases: If v has only one child, or if v has two children.

7. Case 1 (v has only one child):
   - 7.1 Without loss of generality, assume that v's child is a left child and denote it by v_L. Let T_L be the left subtree of v.
   - 7.2 Because v has only one child, v has degree 1 (leaf), so the total number of leaves in T equals the number of leaves in $T_L + 1$. Thus, if t_L is the number of leaves in T_L, then $t = t_L + 1$.
   - 7.3 By inductive hypothesis, $t_L \le 2^k$ because the height of T_L is k, one less than the height of T.
   - 7.4 Also, because v has a child, $k+1 \ge 1$ and so $2^k \ge 2^0 = 1$.
   - 7.5 Therefore,
     $$t = t_L + 1 \le 2^k + 1 \le 2^k + 2^k = 2^{k+1}$$



32

8. Case 2 (v has two children):
   - 8.1 Now v has a left child v_L and a right child v_R, and they are the roots of a left subtree T_L and a right subtree T_R respectively.
   - 8.2 Let h_L and h_R be the heights of T_L and T_R respectively.
   - 8.3 Then $h_L \le k$ and $h_R \le k$ since T is obtained by joining T_L and T_R and adding a level.
   - 8.4 Let t_L and t_R be the number of leaves of T_L and T_R respectively.
   - 8.5 Then, since both T_L and T_R have heights less than $k + 1$, by inductive hypothesis, $t_L \le 2^{h_L}$ and $t_R \le 2^{h_R}$.
   - 8.6 Therefore,
     $$t = t_L + t_R \le 2^{h_L} + 2^{h_R} \le 2^k + 2^k \le 2^{k+1}$$
9. In both cases, $P(k+1)$ is true.
10. Hence if T is any binary tree with height h and t terminal vertices (leaves), then $t \le 2^h$.

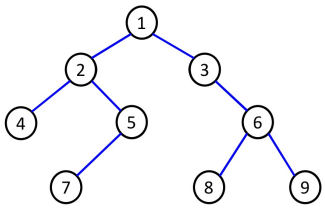
34

A) Binary tree traversal

1) Breadth first search

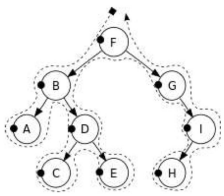
In breadth-first search (by E.F. Moore), it starts at the root and visits its adjacent vertices, and then moves to the next level.

The figure shows the order of the vertices visited.

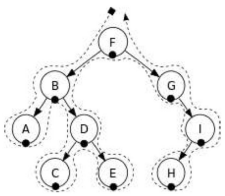


left → right per level, level by level

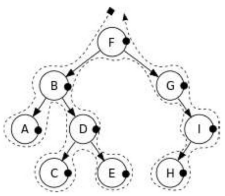
2) Depth first search



Pre-order:
F, B, A, D, C, E, G, I, H



In-order:
A, B, C, D, E, F, G, H, I



Post-order:
A, C, E, D, B, H, I, G, F

1. print current
2. left subtree recursively
3. right subtree recursively

1. go left, not, print
2. print root
3. go right, recursive

1. go left
2. go right
3. root

recursive

Inferring tree structure from in order (+)

pre order: node – left – right →

← left – right – node : post order

1. use pre/post order to get root
2. using adjacency in pre/post, get left/right subtree from in order → until terminal

⑤ spanning trees

1) spanning trees

Definition: Spanning Tree

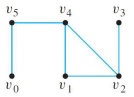
A **spanning tree** for a graph G is a subgraph of G that contains every vertex of G and is a tree.

Proposition 10.7.1

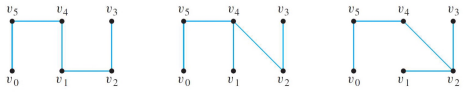
1. Every connected graph has a spanning tree.
2. Any two spanning trees for a graph have the same number of edges.

(how to get spanning trees from graphs)

1. if already tree, done
2. remove any combination of edges from circuits in the graph to give a tree



The graph G has one circuit $v_2v_1v_4v_2$ and removal of any edge of the circuit gives a tree. Hence there are three spanning trees for G.



46

2) weighted graphs and minimum spanning trees

Definitions: Weighted Graph, Minimum Spanning Tree

A **weighted graph** is a graph for which each edge has an associated positive real number **weight** . The sum of the weights of all the edges is the **total weight** of the graph.

→ each edge has weight

→ total weight = $\sum$ weight of edge i

A **minimum spanning tree** for a connected weighted graph is a spanning tree that has the least possible total weight compared to all other spanning trees for the graph.

If G is a weighted graph and e is an edge of G, then $\boldsymbol{w(e)}$ denotes the weight of e and $\boldsymbol{w(G)}$ denotes the total weight of G.

3) algorithms to find minimum spanning trees

Kruskal

Joseph B. Kruskal (1928 – 2010)

In Kruskal's algorithm, the edges of a connected weighted graph are examined one by one in order of increasing weight.

At each stage the edge being examined is added to what will become the minimum spanning tree, provided that this addition does not create a circuit.

After $n - 1$ edges have been added (where n is the number of vertices of the graph), these edges, together with the vertices of the graph, form a minimum spanning tree for the graph.

49

Algorithm 10.7.1 Kruskal

Input: G [a connected weighted graph with n vertices]

Algorithm:

1. Initialize T to have all the vertices of G and no edges.
2. Let E be the set of all edges of G, and let $m = 0$.
3. While $(m < n - 1)$
   - 3a. Find an edge e in E of least weight.
   - 3b. Delete e from E.
   - 3c. If addition of e to the edge set of T does not produce a circuit, then add e to the edge set of T and set $m = m + 1$

   End while

Output: T [T is a minimum spanning tree for G]

Prim

Prim's algorithm works differently from Kruskal's. It builds a minimum spanning tree T by expanding outward in connected links from some vertex.

One edge and one vertex are added at each stage. The edge added is the one of least weight that connects the vertices already in T with those not in T, and the vertex is the endpoint of this edge that is not already in T.

Algorithm 10.7.2 Prim

Input: G [a connected weighted graph with n vertices]

Algorithm:

1. Pick a vertex v of G and let T be the graph with this vertex only.
2. Let V be the set of all vertices of G except v.
3. For $i = 1$ to $n - 1$
   - 3a. Find an edge e of G such that (1) e connects T to one of the vertices in V, and (2) e has the least weight of all edges connecting T to a vertex in V. Let w be the endpoint of e that is in V.
   - 3b. Add e and w to the edge and vertex sets of T, and delete w from V.

Output: T [T is a minimum spanning tree for G]

if tree is a unique minimum, algorithms will produce the same spanning tree

⑥ Modeling problems using trees

1. probability
2. binary trees & tree traversal : algebraic operations