What is discrete math?

- The real numbers are continuous in the senses that:
  - ∗ between any two real numbers there is a real number
- The integers do not share this property.

  In this sense the integers are lumpy, or "discrete"

So discrete math is the study of mathematical objects that are discrete.

"It's all the math that counts"

Some discrete mathematical concepts:

- Integers: Between two integers there is not another integer.
- Propositions: Either true or false, there are no 1/2 truths (in math)
- Sets: An item is either in a set or not in a set, never partly in and partly out.
- Relations: A pair of items are related or not.
- Networks (graphs): Between two terminals of a network connection there are no terminals.

Propositional Logic

Propositions are statements that are either true or false.

Operators $\wedge$, $\vee$, $\neg$, $\rightarrow$, $\leftrightarrow$

Algebraic laws: Commutativity, Associativity, Distributivity ($\wedge$ over $\vee$, **and** $\vee$ over $\wedge$) Involution, De Morgan, definition of $\rightarrow$, definition of $\leftrightarrow$ etc.

Tautology, equivalence, and inference

Defn: *propositional statement*, *tautology*, *contradiction*, *conditional statement*, *equivalence* $\Leftrightarrow$, *inference* $\Rightarrow$

Principles: *Substituting an equivalent statement. Replacing a logic variable in a tautology.*

Defn *algebraic proof*

Applications

- Simplification of digital circuits
- Simplification of computer programs
- Provides foundation for proofs in more sophisticated realms of math.

Sets

Defns *Set builder notation*:

$$\{x \in S \mid P(x)\}$$

empty set ($\emptyset$)

Is $\{\emptyset\} = \emptyset$?

Operations: *Union* $S \cup T$,*Intersection* $S \cap T$, *Difference* $S - T$, *Symmetric Difference* $S \Delta T$, *Complement* $\overline{S}$, *Cardinality* $|S|$, *Power Set* $\mathcal{P}(S)$, *Cartesian product* $S \times T$

Relations: Equality (=), subset ($\subseteq$), proper subset ($\subset$), disjoint

Applications

- Designing algorithms
- Error correcting codes

Proofs about sets

Proofs of equality ($=$)

Proofs of subset ($\subseteq$)

Predicate Logic

Predicates (boolean expressions)

A **predicate** is an expression that is either true or false, but that might depend on the values of one or more variables.

Concept Free and bound variables.

- Bound occurrences are entirely local to some construct:

$$\{\underline{\underline{x}} \in S \mid P(\underline{x})\} \qquad \sum_{\underline{\underline{x}} \in S} f(\underline{x}) \qquad \int_a^b f(\underline{x})\,\mathrm{d}\underline{\underline{x}}$$

$$\forall \underline{\underline{x}} \in S, P(\underline{x}) \qquad \exists \underline{\underline{x}} \in S, P(\underline{x})$$

- Free variables define the interface of the expression to the outside world.

Quantifiers

$$(\forall j \in S, f(j)) = f(s_0) \wedge f(s_1) \wedge \ldots\ldots$$
$$(\exists j \in S, f(j)) = f(s_0) \vee f(s_1) \vee \ldots\ldots$$
$$DeMorgan\prime s \text{ law } \neg(\forall j \in S, f(j)) \Leftrightarrow (\exists j \in S, \neg f(j))$$

$DeMorgan's$ law $\neg(\exists j \in S, f(j)) \Leftrightarrow (\forall j \in S, \neg f(j))$

$$(\exists x \in S, f(x)) \Leftrightarrow (\{x \mid x \in S \wedge f(x)\} \neq \emptyset)$$
$$(\forall x \in S, f(x)) \Leftrightarrow (\{x \mid x \in S \wedge f(x)\} = S)$$

Equivalence revisited. We say that an expressions f and g ($f \Leftrightarrow g$) are equivalent if they are equal for all values of their free variables.

Applications

- Fundamental to all branches of math
- Describing states of programs.
- Describing the relation between the initial and final states of subprograms
  - Preconditions: free variables are variables representing program state.
  - Postcondition: free variables are
    - plain variables representing the initial program state
    - primed variables representing the final program state

Integers & Mathematical Reasoning

Theorem: *"The Euclidean Division Algorithm"* For $a, b \in \mathbb{Z}$ with $b \neq 0$, there exists a unique pair of integers (q, r) such that $a = qb + r$ and $0 \leq r < |b|$

Operations: $a \operatorname{div} b$, $a \bmod b$, $\gcd(a, b)$

Relations: *divisibility* $b \mid a$, *congruence*. we write $a_0 \equiv a_1 (\bmod b)$ to mean that $b \mid (a_0 - a_1)$.

Primes

Properties: *prime, composite, prime decomposition*

Theorem: *"The fundamental theorem of arithmetic"*

Proof methods

- Direct proof.
  - $P \to Q$
    - Assume P; show Q
  - $P \wedge Q$
    - Show P; then show Q
    - Show P; then assume P and show Q.
  - $P \vee Q$
    - Assume $\neg P$; show Q.

- $\forall x \in S, P(x)$
  - Let x be an arbitrary member of S; show $P(x)$
- $\exists y \in S, P(x)$
  - Let y be some specified member of S; show $P(x)$

• Proof by contradiction

- P
  - Assume $\neg P$; show $0 = 1$ or some other contradiction.

• Proof by contrapositive

- $P \rightarrow Q$
  - Assume $\neg Q$; show $\neg P$

• Proof by cases

- P
  - Assume Q and show P; then assume $\neg Q$ and show P.

• Use of "let"

- If you are assuming $\exists x \in S, P(x)$, then give a name to the item that is assumed to exist.

Applications

- Integers: Public Key Cryptography (e.g. the RSA method)
- Mathematical Reasoning: Providing a tight argument about anything. E.g. that a design is correct with respect to some property.

Induction

Properties. A property of integer numbers is a partial function from the integer numbers to $\{T, F\}$.

Simple induction

Principle: The principle of (simple) mathematical induction

For any property P of integer numbers and $n_0 \in \mathbb{Z}$

- if
  - [base step] $P(n_0)$ and
  - [induction step] for all $k \geq n_0$,
    - if $\underline{P(k)}$
    - then $P(k+1)$
- then $\forall n \geq n_0, P(n)$

Using simple induction for proofs

- Prove the base step
- Prove inductive step
  - Assume k is an arbitrary integer $\geq n_0$
  - Assume that the inductive hypothesis ($P(k)$) is true

∗ Show that $P(k+1)$ is true under these assumptions

Complete Induction

Principle The principle of complete induction (extended version)

For any n_0 and n_1 in $\mathbb{Z}$ with $0 \leq n_0 \leq n_1$ and property P of the integers

- If
  - ∗ [base step(s)] $P(n_0)$ and $P(n_0+1)$ and ... and $P(n_1-1)$ and
  - ∗ [induction step] for all $k \geq n_1$
    - · if for all j, with $n_0 \leq j < k$, $P(j)$
    - · then $P(k)$
- then, for all $n \in \{n_0, n_0+1, ...\}$, $P(n)$.

Here there are $n_1 - n_0$ base steps

Using complete induction for proofs

- Prove the base step(s)
- Prove inductive step
  - ∗ Assume k is any large enough integer ($k \geq n_1$)

    * Assume that the inductive hypothesis is true
    * Show that $P(k)$ is true under these assumptions

Applications

- Proof of many useful theorems including proving solutions to recurrences
- Proving looping programs and recursive programs

Recurrence Relations

- Substitute and simplify method
- Linear Homogeneous Recurrence Relations with Constant Coefficients of Degree k
- Degree 2 case:

$$a(n) = c_1 a(n-1) + c_2 a(n-2) \qquad (*)$$

- If r is root of the polynomial

$$x^n - c_1 x^{n-1} - c_2 x^{n-2} = 0 \qquad (**)$$

  then

$$r^n = c_1 r^{n-1} + c_2 r^{n-2}$$

  so

$$a(n) = r^n$$

  is a solution to (*).
- If $b(n)$ is a solution, then so is $\theta b(n)$ for any θ .
- If $b(n)$ is a solution and $c(n)$ is a solution then $b(n)+c(n)$ is a solution.
- So $a(n) = \theta_1 r_1^n + \theta_2 r_2^n$ is a solution.

Repeated roots

Theorem: A RR of the form $a(n) = c_1 a(n-1) + c_2 a(n-1)$ has a characteristic polynomial with one root r then nr^n is a solution.

Thus, for any $\alpha_0, \alpha_1 \in \mathbb{R}$, it has a solution of the form

$$a(n) = (\alpha_0 + \alpha_1 n) r^n$$

Functions and Relations

Defn: *binary relation*, *domain*, *range*, *graph*.

Properties. *partial function, total relation, function*, *one-one*, *onto*

Operations. Inverse, composition.

Inverse of R is a partial function iff R is one-one.

Inverse of R is a total relation iff R is onto.

Inverse of R is a function iff R is one-one and onto.

Relational Databases

Defn: *table* (or *n-ary relation*)

Operations: *projection, attribute renaming, selection, join*

Applications

- Relations in system design (modeling meaning of components)
- Relations in databases
- Relations in many branches of mathematics

Graphs

Defns: *(undirected) graph, directed graph, vertex, edge, loop, incidence function.*

Properties: *simple*, *connected*

Defns: *walk (undirected)*

$(v_0, e_1, v_1, e_2, ..., e_k, v_k) \quad \phi(e_i) = \{v_{i-1}, v_i\}$ for all $i \in \{1, 2, ..., k\}$

walk (directed)

$(v_0, e_1, v_1, e_2, ..., e_k, v_k) \quad \phi(e_i) = (v_{i-1}, v_i)$ for all $i \in \{1, 2, ..., k\}$

Defn: *adjacency matrix* (use of adjacency matrix to count walks and calculate connectivity)

Algorithms: Dijkstra's.

Colouring graphs

Defn: k*-colouring*

if $\phi(e) = \{u, v\}$ then $f(u) \neq f(v)$, for all $u, v \in V, e \in E$

Applications

- Modeling conflict over resources (colouring as solution)
- Modeling structure of systems
- Optimization: Shortest paths, minimum spanning tree, maximal flow.
- Many many others

Automata

Defns: *finite state automaton, behaviour, complete behaviour.*

For describing

- behaviour of synchronous systems
- behaviour of asynchronous systems
- languages (sets of finite sequences)

Many-many other applications