Discrete Mathematics I

Computer Science Tripos, Part 1A
Paper 1

Natural Sciences Tripos, Part 1A,
Computer Science option

Politics, Psychology and Sociology, Part 1,
Introduction to Computer Science option

2009–10

Peter Sewell

Computer Laboratory

University of Cambridge

`Time-stamp: <2009-11-01 13:25:09 pes20>`

©Peter Sewell 2008, 2009

Contents

Syllabus

Lecturer: Dr P.M. Sewell

No. of lectures: 8

This course is a prerequisite for all theory courses as well as Discrete Mathematics II, Algorithms I, Security (Part IB and Part II), Artificial Intelligence (Part IB and Part II), Information Theory and Coding (Part II).

Aims

This course will develop the intuition for discrete mathematics reasoning involving numbers and sets.

Lectures

- **Logic.** The languages of propositional and predicate logic and their relationship to informal statements, truth tables, validity [3 lectures]
- **Proof.** Proof of predicate logic formulas. [2 lectures]
- **Sets.** Basic set constructions and relation properties, including equivalence relations, DAGs, pre-, partial and total orders, and functions. [2 lectures]
- **Induction.** Proof by induction, including proofs about total functional programs over natural numbers and lists. [1 lecture]

Objectives

On completing the course, students should be able to

- write a clear statement of a problem as a theorem in mathematical notation;
- prove and disprove assertions using a variety of techniques.

Recommended reading

Biggs, N.L. (1989). *Discrete mathematics.* Oxford University Press.
Bornat, R. (2005). *Proof and Disproof in Formal Logic.* Oxford University Press.
Devlin, K. (2003). *Sets, functions, and logic: an introduction to abstract mathematics.* Chapman and Hall/CRC Mathematics (3rd ed.).
Mattson, H.F. Jr (1993). *Discrete mathematics.* Wiley.
Nissanke, N. (1999). *Introductory logic and sets for computer scientists.* Addison-Wesley.
Pólya, G. (1980). *How to solve it.* Penguin.
(*) Rosen, K.H. (1999). *Discrete mathematics and its applications* (6th ed.). McGraw-Hill.
(*) Velleman, D. J. (1994). *How to prove it (a structured approach).* CUP.

For Supervisors (and Students too)

The main aim of the course is to enable students to confidently use the language of propositional and predicate logic, and set theory.

We first introduce the language of propositional logic, discussing the relationship to natural-language argument. We define the meaning of formulae with the truth semantics w.r.t. assumptions on the propositional variables, and, equivalently, with truth tables. We also introduce equational reasoning using tautologies, to make instantiation and reasoning-in-context explicit.

We then introduce quantifiers, again emphasising the intuitive reading of formulae and defining the truth semantics. We introduce the notions of free and bound variable (but not alpha equivalence).

We do not develop any metatheory, and we treat propositional assumptions, valuations of variables, and models of atomic predicate symbols all rather informally. There are no turnstiles, but we talk about valid formulae and (briefly) about satisfiable formulae.

We then introduce 'structured' proof. This is essentially natural deduction proof, but laid out on the page in roughly the box-and-line style used by Bornat (2005). We don't give the usual horizontal-line presentation of the rules. The rationale here is to introduce a style of proof for which one can easily define what is (or is not) a legal proof, but where the proof text on the page is reasonably close to the normal mathematical 'informal but rigorous' practice that will be used in most of the rest of the Tripos. We emphasise how to prove and how to use each connective, and talk about the pragmatics of finding and writing proofs.

The set theory material introduces the basic notions of set, element, union, intersection, powerset, and product, relating to predicates (e.g. relating predicates and set comprehensions, and the properties of union to those of disjunction), with some more small example proofs. We then define some of the standard properties of relations (reflexive, symmetric, transitive, antisymmetric, acyclic, total) to characterise directed graphs, undirected graphs, equivalence relations, pre-orders, partial orders, and functions). These are illustrated with simple examples to introduce the concepts, but their properties and uses are not explored in any depth (for example, we do not define what it means to be an injection or surjection).

Finally, we recall inductive proof over the naturals, making the induction principle explicit in predicate logic, and over lists, talking about inductive proof of simple pure functional programs (taking examples from the previous SWEng II notes).

I'd suggest 3 supervisons. A possible schedule might be:

1. After the first 2–3 lectures (on or after Thurs 19 Nov)
   Example Sheets 1 and 2, covering Propositional and Predicate Logic
2. After the next 3–4 lectures (on or after Thurs 26 Nov)
   Example Sheets 3 and the first part of 4, covering Structured Proof and Sets
3. After all 8 lectures (on or after Thurs 3 Dec)
   Example Sheet 4 (the remainder) and 5, covering Inductive Proof

Learning Guide

Notes: These notes include all the slides, but by no means everything that'll be said in lectures.

Exercises: There are some exercises at the end of the notes. I suggest you do all of them. Most should be rather straightforward; they're aimed at strengthening your intuition about the concepts and helping you develop quick (but precise) manipulation skills, not to provide deep intellectual challenges. A few may need a bit more thought. Some are taken (or adapted) from Devlin, Rosen, or Velleman. More exercises and examples can be found in any of those.

Tripos questions: This version of the course was new in 2008.

Feedback: Please do complete the on-line feedback form at the end of the course, and let me know during it if you discover errors in the notes or if the pace is too fast or slow.

Errata: A list of any corrections to the notes will be on the course web page.

1 Introduction

Discrete Mathematics 1

Computer Science Tripos, Part 1A

Natural Sciences Tripos, Part 1A, Computer Science

Politics, Psychology and Sociology Part 1, Introduction to Computer Science

Peter Sewell

1A, 8 lectures

2009 – 2010

Introduction

At the start of the Industrial Revolution, we built bridges and steam engines without enough applied maths, physics, materials science, etc.

FIG. 5. AN ILLUSTRATION OF WHAT EXPLOSION DID TO STAYS AND BRACES

Fix: understanding based on continuous-mathematics models — calculus, matrices, complex analysis,...

Introduction

Now, we build computer systems, and sometimes...

[Ariane 501]

Introduction

Now, we build computer systems, and sometimes...

[Ariane 501]

But, computer systems are large and complex, and are *discrete*:
we can't use approximate continuous models for correctness reasoning.
So, need *applied discrete maths* — logic, set theory, graph theory, combinatorics, abstract algebra, ...

Logic and Set Theory — Pure Mathematics

Origins with the Greeks, 500–350 BC, philosophy and geometry:

Aristotle, Euclid

Formal logic in the 1800s:

De Morgan, Boole, Venn, Peirce, Frege

Set theory, model theory, proof theory; late 1800s and early 1900s:

Cantor, Russell, Hilbert, Zermelo, Frankel, Goedel, Turing, Bourbaki, Gentzen, Tarski

Focus then on the foundations of mathematics — but what was developed then turns out to be unreasonably effective in Computer Science. This is the core of the applied maths that we need.

Logic and Set Theory — Applications in Computer Science

- modelling digital circuits (1A Digital Electronics, 1B ECAD)
- proofs about particular algorithms and code (1A Algorithms 1, 1B Algorithms 2)
- proofs about what is (or is not!) computable and with what complexity (1B Computation Theory, Complexity Theory)
- proofs about programming languages and type systems (1A Regular Languages and Finite Automata, 1B Semantics of Programming Languages, 2 Types)
- foundation of databases (1B Databases)
- automated reasoning and model-checking tools (1B Logic & Proof, 2 Specification & Verification)

Outline

- Propositional Logic
- Predicate Logic
- Sets
- Inductive Proof

Focus on *using* this material, rather than on metatheoretic study.

More (and more metatheory) in Discrete Maths 2 and in Logic & Proof.

New course last year — feedback welcome.

Supervisons

Not rocket science (?), but *needs practice* to become fluent.

Five example sheets. Many more suitable exercises in the books.

Up to your DoS and supervisor, but I'd suggest 3 supervisons. A possible schedule might be:

1. After the first 2–3 lectures (on or after Thurs 19 Nov)
   Example Sheets 1 and 2, covering Propositional and Predicate Logic
2. After the next 3–4 lectures (on or after Thurs 26 Nov)
   Example Sheets 3 and the first part of 4, covering Structured Proof and Sets
3. After all 8 lectures (on or after Thurs 3 Dec)
   Example Sheet 4 (the remainder) and 5, covering Inductive Proof

2 Propositional Logic

Propositional Logic

Propositional Logic

Starting point is informal natural-language argument:

Socrates is a man. All men are mortal. So Socrates is mortal.

Propositional Logic

Starting point is informal natural-language argument:

Socrates is a man. All men are mortal. So Socrates is mortal.

If a person runs barefoot, then his feet hurt. Socrates' feet hurt. Therefore, Socrates ran barefoot

It will either rain or snow tomorrow. It's too warm for snow. Therefore, it will rain.

It will either rain or snow tomorrow. It's too warm for snow. Therefore, it will rain.

Either the butler is guilty or the maid is guilty. Either the maid is guilty or the cook is guilty. Therefore, either the butler is guilty or the cook is guilty.

It will either rain or snow tomorrow. It's too warm for snow. Therefore, it will rain.

Either the framger widget is misfiring or the wrompal mechanism is out of alignment. I've checked the alignment of the wrompal mechanism, and it's fine. Therefore, the framger widget is misfiring.

Either the framger widget is misfiring or the wrompal mechanism is out of alignment. I've checked the alignment of the wrompal mechanism, and it's fine. Therefore, the framger widget is misfiring.

Either p *or* q. *Not* q. *Therefore,* p

2.1 The Language of Propositional Logic

Atomic Propositions

$1 + 1 = 2$

$10 + 10 = 30$

Tom is a student

Is Tom a student? $\times$

Give Tom food! $\times$

$x + 7 = 10$ $\times$

$1 + 2 + ... + n = n(n + 1)/2$ $\times$

Atomic Propositions

When we're studying logic, instead of fixing some particular language of atomic propositions, we'll use *propositional variables* p, q, etc. In a particular context, each of these might be true or false (but not 21.5).

Compound Propositions

We'll build more complex *compound propositions* out of those of atomic propositions. Any *propositional variable* p, q, etc., is trivially a compound proposition.

We'll write p, q, etc. for arbitrary propositional variables.

We'll write P, Q, etc. for arbitrary compound propositions.

Building Compound Propositions: Truth and Falsity

We'll write T for the constant true proposition, and F for the constant false proposition.

Building Compound Propositions: Conjunction

If P and Q are two propositions, $P \wedge Q$ is a proposition.

Pronounce $P \wedge Q$ as 'P and Q'. Sometimes written with $\&$ or .

Definition: $P \wedge Q$ is true if (and only if) P is true and Q is true

Examples:

Tom is a student $\wedge$ Tom has red hair

$(1 + 1 = 2) \wedge (7 \leq 10)$

$(1 + 1 = 2) \wedge (2 = 3)$

$((1 + 1 = 2) \wedge (7 \leq 10)) \wedge (5 \leq 5)$

$(\mathrm{p} \wedge \mathrm{q}) \wedge \mathrm{p}$

Building Compound Propositions: Conjunction

We defined the meaning of $P \wedge Q$ by saying '$P \wedge Q$ is true if and only if P is true and Q is true'.

We could instead, equivalently, have defined it by enumerating all the cases, in a *truth table*:

P	Q	$P \wedge Q$
T	T	T
T	F	F
F	T	F
F	F	F

According to this definition, is $((1 + 1 = 2) \wedge (7 \leq 10)) \wedge (5 \leq 5)$ true or false?

Building Compound Propositions: Conjunction

We pronounce $P \wedge Q$ as 'P and Q', but not all uses of the English 'and' can be faithfully translated into $\wedge$.

Tom and Alice had a dance.

Grouping

Tom went to a lecture and had lunch.

Temporal ordering?

The Federal Reserve relaxed banking regulations, and the markets boomed.

Causality?

When we want to talk about time or causality in CS, we'll do so explicitly; they are not built into this logic.

Building Compound Propositions: Conjunction

Basic properties:

The order doesn't matter: whatever P and Q are, $P \wedge Q$ means the same thing as $Q \wedge P$.

Check, according to the truth table definition, considering each of the 4 possible cases:

P	Q	$P \wedge Q$	$Q \wedge P$
T	T	T	T
T	F	F	F
F	T	F	F
F	F	F	F

In other words, $\wedge$ is *commutative*

Building Compound Propositions: Conjunction

...and:

The grouping doesn't matter: whatever P, Q, and R are, $P \wedge (Q \wedge R)$ means the same thing as $(P \wedge Q) \wedge R$.

(Check, according to the truth table definition, considering each of the 8 possible cases).

In other words, $\wedge$ is *associative*

So we'll happily omit *some* parentheses, e.g. writing $P_1 \wedge P_2 \wedge P_3 \wedge P_4$ for $P_1 \wedge (P_2 \wedge (P_3 \wedge P_4))$.

Building Compound Propositions: Disjunction

If P and Q are two propositions, $P \vee Q$ is a proposition.

Pronounce $P \vee Q$ as 'P or Q'. Sometimes written with $|$ or $+$

Definition: $P \vee Q$ is true if and only if P is true or Q is true

Equivalent truth-table definition:

P	Q	$P \vee Q$
T	T	T
T	F	T
F	T	T
F	F	F

Building Compound Propositions: Disjunction

You can see from that truth table that $\vee$ is an *inclusive* or: $P \vee Q$ if *at least one* of P and Q.

$(2+2=4) \vee (3+3=6)$ is true

$(2+2=4) \vee (3+3=7)$ is true

The English 'or' is sometimes an *exclusive* or: P xor Q if *exactly* one of P and Q. 'Fluffy is either a rabbit or a cat.'

P	Q	$P \vee Q$	P xor Q
T	T	T	F
T	F	T	T
F	T	T	T
F	F	F	F

(we won't use xor so much).

Building Compound Propositions: Disjunction

Basic Properties

$\vee$ is also commutative and associative:

$P \vee Q$ and $Q \vee P$ have the same meaning

$P \vee (Q \vee R)$ and $(P \vee Q) \vee R$ have the same meaning

$\wedge$ distributes over $\vee$:

$P \wedge (Q \vee R)$ and $(P \wedge Q) \vee (P \wedge R)$ have the same meaning

'P and either Q or R' 'either (P and Q) or (P and R)'

and the other way round: $\vee$ distributes over $\wedge$

$P \vee (Q \wedge R)$ and $(P \vee Q) \wedge (P \vee R)$ have the same meaning

When we mix $\wedge$ and $\vee$, we take care with the parentheses!

Building Compound Propositions: Negation

If P is some proposition, $\neg P$ is a proposition.

Pronounce $\neg P$ as 'not P'. Sometimes written as $\sim P$ or $\overline{P}$

Definition: $\neg P$ is true if and only if P is false

Equivalent truth-table definition:

P	$\neg P$
T	F
F	T

Building Compound Propositions: Implication

If P and Q are two propositions, $P \Rightarrow Q$ is a proposition.

Pronounce $P \Rightarrow Q$ as 'P implies Q'. Sometimes written with $\longrightarrow$

Definition: $P \Rightarrow Q$ is true if (and only if), whenever P is true, Q is true

Equivalent truth-table definition:

P	Q	$P \Rightarrow Q$
T	T	T
T	F	F
F	T	T
F	F	T

Building Compound Propositions: Implication

That can be confusing. First, the logic is not talking about causation, but just about truth values.

$(1+1=2) \Rightarrow (3<4)$ is true

Second, $P \Rightarrow Q$ is vacuously true if P is false.

'If I'm a giant squid, then I live in the ocean'

For that to be true, either:

(a) I really am a giant squid, in which case I must live in the ocean, or

(b) I'm not a giant squid, in which case we don't care where I live.

$P \Rightarrow Q$ and $(P \wedge Q) \vee \neg P$ and $Q \vee \neg P$ all have the same meaning

Building Compound Propositions: Implication

Basic properties:

$P \Rightarrow Q$ and $\neg Q \Rightarrow \neg P$ have the same meaning

$\Rightarrow$ is not commutative: $P \Rightarrow Q$ and $Q \Rightarrow P$ do not have the same meaning

$P \Rightarrow (Q \wedge R)$ and $(P \Rightarrow Q) \wedge (P \Rightarrow R)$ have the same meaning

$(P \wedge Q) \Rightarrow R$ and $(P \Rightarrow R) \wedge (Q \Rightarrow R)$ do not

$(P \wedge Q) \Rightarrow R$ and $P \Rightarrow Q \Rightarrow R$ do

Building Compound Propositions: Bi-Implication

If P and Q are two propositions, $P \Leftrightarrow Q$ is a proposition.

Pronounce $P \Leftrightarrow Q$ as 'P if and only if Q'. Sometimes written with $\leftrightarrow$ or $=$.

Definition: $P \Leftrightarrow Q$ is true if (and only if) P is true whenever Q is true, and vice versa

Equivalent truth-table definition:

P	Q	$P \Leftrightarrow Q$
T	T	T
T	F	F
F	T	F
F	F	T

The Language of Propositional Logic

Summarising, the formulae of propositional logic are the terms of the grammar

$$P, Q ::= p \mid \mathsf{T} \mid \mathsf{F} \mid \neg P \mid P \wedge Q \mid P \vee Q \mid P \Rightarrow Q \mid P \Leftrightarrow Q$$

where p ranges over atomic propositions p, q, etc., and we use parentheses (P) as necessary to avoid ambiguity.

For any such formula P, assuming the truth value of each atomic proposition p it mentions is fixed (true or false), we've defined whether P is true or false.

Example Compound Truth Table

Given an arbitrary formula P, we can calculate the meaning of P for all possible assumptions on its atomic propositions by enumerating the cases in a truth table.

For example, consider $P \stackrel{\text{def}}{=} ((\mathrm{p} \vee \neg\mathrm{q}) \Rightarrow (\mathrm{p} \wedge \mathrm{q}))$. It mentions two atomic propositions, p and q, so we have to consider 2^2 possibilities:

p	q	$\neg\mathrm{q}$	$\mathrm{p} \vee \neg\mathrm{q}$	$\mathrm{p} \wedge \mathrm{q}$	$(\mathrm{p} \vee \neg\mathrm{q}) \Rightarrow (\mathrm{p} \wedge \mathrm{q})$
T	T	F	T	T	T
T	F	T	T	F	F
F	T	F	F	F	T
F	F	T	T	F	F

Notice that this calculation is *compositional* in the structure of P.

The Binary Boolean Functions of one and two variables

$2^{(2^1)}$ functions of one variable

P	T	P	$\neg P$	F
T	T	T	F	F
F	T	F	T	F

$2^{(2^2)}$ functions of two variables

P	Q	T	$\vee$		P	$\Rightarrow$	Q	$\Leftrightarrow$	$\wedge$	nand	xor						F
T	T	T	T	T	T	T	T	T	T	F	F	F	F	F	F	F	F
T	F	T	T	T	T	F	F	F	F	T	T	T	T	F	F	F	F
F	T	T	T	F	F	T	T	F	F	T	T	F	F	T	T	F	F
F	F	T	F	T	F	T	F	T	F	T	F	T	F	T	F	T	F

(what are the complete subsets of those functions?) (why stop at 2?)

A Few More Equivalences

Identity:

$P \wedge$ T and P have the same meaning

$P \vee$ F and P have the same meaning

Complement:

$P \wedge \neg P$ and F have the same meaning

$P \vee \neg P$ and T have the same meaning

De Morgan:

$\neg(P \wedge Q)$ and $\neg P \vee \neg Q$ have the same meaning

$\neg(P \vee Q)$ and $\neg P \wedge \neg Q$ have the same meaning

Translating away $\Leftrightarrow$:

$P \Leftrightarrow Q$ and $(P \Rightarrow Q) \wedge (Q \Rightarrow P)$ have the same meaning

2.2 Equational Reasoning

Tautologies

Say P is a *tautology*, or is *valid*, if it is always true — i.e., if, whatever assumption we make about the truth values of its atomic propositions, then P is true.

When we say 'P and Q have the same meaning', we really mean 'whatever assumption we make about the truth values of their atomic propositions, P and Q have the same truth value as each other'.

We write that as P iff Q

(Strictly, this P iff Q is a meta-statement about two propositions, not itself a proposition. But P iff Q if and only if $P \Leftrightarrow Q$ is a tautology.)

Equational Reasoning

Tautologies are really useful — because they can be used anywhere.

In more detail, this P iff Q is a proper notion of equality. You can see from its definition that

- it's *reflexive*, i.e., for any P, we have P iff P
- it's *symmetric*, i.e., if P iff Q then Q iff P
- it's *transitive*, i.e., if P iff Q and Q iff R then P iff R

Moreover, if P iff Q then we can replace a subformula P by Q in any context, without affecting the meaning of the whole thing. For example, if P iff Q then $P \wedge R$ iff $Q \wedge R$, $R \wedge P$ iff $R \wedge Q$, $\neg P$ iff $\neg Q$, etc.

Equational Reasoning

Now we're in business: we can do equational reasoning, replacing equal subformulae by equal subformulae, just as you do in normal algebraic manipulation (where you'd use $2 + 2 = 4$ without thinking).

This complements direct verification using truth tables — sometimes that's more convenient, and sometimes this is. Later, we'll see a third option — structured proof.

Some Collected Tautologies, for Reference

For any propositions P, Q, and R

Commutativity:	Unit:
$P \wedge Q$ iff $Q \wedge P$ (and-comm)	$P \wedge$ F iff F (and-unit)
$P \vee Q$ iff $Q \vee P$ (or-comm)	$P \vee$ T iff T (or-unit)
Associativity:	Complement:
$P \wedge (Q \wedge R)$ iff $(P \wedge Q) \wedge R$ (and-assoc)	$P \wedge \neg P$ iff F (and-comp)
$P \vee (Q \vee R)$ iff $(P \vee Q) \vee R$ (or-assoc)	$P \vee \neg P$ iff T (or-comp)
Distributivity:	De Morgan:
$P \wedge (Q \vee R)$ iff $(P \wedge Q) \vee (P \wedge R)$ (and-or-dist)	$\neg(P \wedge Q)$ iff $\neg P \vee \neg Q$ (and-DM)
$P \vee (Q \wedge R)$ iff $(P \vee Q) \wedge (P \vee R)$ (or-and-dist)	$\neg(P \vee Q)$ iff $\neg P \wedge \neg Q$ (or-DM)
Identity:	Defn:
$P \wedge$ T iff P (and-id)	$P \Rightarrow Q$ iff $Q \vee \neg P$ (imp)
$P \vee$ F iff P (or-id)	$P \Leftrightarrow Q = (P \Rightarrow Q) \wedge (Q \Rightarrow P)$ (bi)

Equational Reasoning — Example

Suppose we wanted to prove a 3-way De Morgan law

$\neg(P_1 \wedge P_2 \wedge P_3)$ iff $\neg P_1 \vee \neg P_2 \vee \neg P_3$

We could do so either by truth tables, checking 2^3 cases, or by equational reasoning:

$\neg(P_1 \wedge P_2 \wedge P_3)$	iff	$\neg(P_1 \wedge (P_2 \wedge P_3))$	choosing an $\wedge$ association
	iff	$\neg P_1 \vee \neg(P_2 \wedge P_3)$	by (and-DM)

(and-DM) is $\neg(P \wedge Q)$ iff $\neg P \vee \neg Q$. Instantiating the metavariables P and Q as

$$\begin{array}{ccc} P & \mapsto & P_1 \\ Q & \mapsto & P_2 \wedge P_3 \end{array}$$

we get exactly the $\neg(P_1 \wedge (P_2 \wedge P_3))$ iff $\neg P_1 \vee \neg(P_2 \wedge P_3)$ needed.

$\neg(P_1 \wedge P_2 \wedge P_3)$	iff	$\neg(P_1 \wedge (P_2 \wedge P_3))$	choosing an $\wedge$ association
	iff	$\neg P_1 \vee \neg(P_2 \wedge P_3)$	by (and-DM)
	iff	$\neg P_1 \vee (\neg P_2 \vee \neg P_3)$	by (and-DM)

(and-DM) is $\neg(P \wedge Q)$ iff $\neg P \vee \neg Q$. Instantiating the metavariables P and Q as

$$\begin{array}{ccc} P & \mapsto & P_2 \\ Q & \mapsto & P_3 \end{array}$$

we get $\neg(P_2 \wedge P_3)$ iff $\neg P_2 \vee \neg P_3$. Using that in the context $\neg P_1 \vee \ldots$ gives us exactly the equality $\neg P_1 \vee \neg(P_2 \wedge P_3))$ iff $\neg P_1 \vee (\neg P_2 \vee \neg P_3)$.

	iff	$\neg P_1 \vee \neg P_2 \vee \neg P_3$	forgetting the $\vee$ association

So by transitivity of iff, we have $\neg(P_1 \wedge P_2 \wedge P_3)$ iff $\neg P_1 \vee \neg P_2 \vee \neg P_3$

There I unpacked the steps in some detail, so you can see what's really going on. Later, we'd normally just give the brief justification on each line; we wouldn't write down the boxed reasoning (instantiation, context, transitivity) — but it should be clearly in your head when you're doing a proof.

If it's not clear, write it down — *use* the written proof as a tool for thinking.

Still later, you'll use equalities like this one as single steps in bigger proofs.

Equational reasoning from those tautologies is *sound*: however we instantiate them, and chain them together, if we deduce that P iff Q then P iff Q.

Pragmatically important: if you've faithfully modelled some real-world situation in propositional logic, then you can do any amount of equational reasoning, and the result will be meaningful.

Is equational reasoning from those tautologies also *complete*? I.e., if P iff Q, is there an equational proof of that?
Yes (though proving completeness is beyond the scope of DM1).

Pragmatically: if P iff Q, and you systematically explore all possible candidate equational proofs, eventually you'll find one. But there are infinitely many candidates: at any point, there might be several tautologies you could try to apply, and sometimes there are infinitely many instantiations (consider T iff $P \vee \neg P$).

...so naive proof search is not a decision procedure (but sometimes you can find short proofs).

In contrast, we had a terminating algorithm for checking tautologies by truth tables (but that's exponential in the number of propositional variables).

Satisfiability

Recall P is a *tautology*, or is *valid*, if it is always true — i.e., if, whatever assumption we make about the truth values of its atomic propositions, then P is true.

Say P is a *satisfiable* if, under *some* assumption about the truth values of its atomic propositions, P is true.

$\mathrm{p} \land \neg\mathrm{q}$ satisfiable (true under the assumption $\mathrm{p} \mapsto$ T, $q \mapsto$ F)

$\mathrm{p} \land \neg\mathrm{p}$ unsatisfiable (not true under $\mathrm{p} \mapsto$ T or $\mathrm{p} \mapsto$ F)

P unsatisfiable iff $\neg P$ valid

Object, Meta, Meta-Meta,...

We're taking care to distinguish the connectives of the object language (propositional logic) that we're studying, and the informal mathematics and English that we're using to talk about it (our meta-language).

For now, we adopt a simple discipline: the former in symbols, the latter in words.

Later, you'll use logic to talk about logic.

Application: Combinational Circuits

Use T and F to represent high and low voltage values on a wire.

Logic gates (AND, OR, NAND, etc.) compute propositional functions of their inputs.

Notation: T, F, $\land$, $\lor$, $\neg$ vs 0, 1, $.$, $+$, $\overline{\ }$

SAT solvers: compute satisfiability of formulae with 10 000's of propositional variables.

3 Predicate Logic

Predicate Logic

Predicate Logic

(or Predicate Calculus, or First-Order Logic)

Socrates is a man. All men are mortal. So Socrates is mortal.

Predicate Logic

(or Predicate Calculus, or First-Order Logic)

Socrates is a man. All men are mortal. So Socrates is mortal.

Can we formalise in propositional logic?

Write p for *Socrates is a man*
Write q for *Socrates is mortal*
$\mathrm{p} \qquad \mathrm{p} \Rightarrow \mathrm{q} \qquad \mathrm{q}$

?

Predicate Logic

Often, we want to talk about properties of things, not just atomic propositions.

All lions are fierce.
Some lions do not drink coffee.
Therefore, some fierce creatures do not drink coffee.
[Lewis Carroll, 1886]

Let x range over creatures. Write $\mathrm{L}(x)$ for 'x is a lion'. Write $\mathrm{C}(x)$ for 'x drinks coffee'. Write $\mathrm{F}(x)$ for 'x is fierce'.

$\forall\, x.\mathrm{L}(x) \Rightarrow \mathrm{F}(x)$
$\exists\, x.\mathrm{L}(x) \wedge \neg\mathrm{C}(x)$
$\exists\, x.\mathrm{F}(x) \wedge \neg\mathrm{C}(x)$

3.1 The Language of Predicate Logic

Predicate Logic

So, we extend the language.

Variables x, y, etc., ranging over some specified domain.

Atomic predicates $\mathrm{A}(x)$, $\mathrm{B}(x)$, etc., like the earlier atomic propositions, but with truth values that depend on the values of the variables. Write $A(x)$ for an arbitrary atomic predicate. E.g.:

Let $\mathrm{A}(x)$ denote $x + 7 = 10$, where x ranges over the natural numbers. $\mathrm{A}(x)$ is true if $x = 3$, otherwise false, so $\mathrm{A}(3) \wedge \neg A(4)$

Let $\mathrm{B}(n)$ denote $1 + 2 + ... + n = n(n + 1)/2$, where n ranges over the naturals. $\mathrm{B}(n)$ is true for any value of n, so $\mathrm{B}(27)$.

Add these to the language of formulae:

$$P, Q ::= A(x) \mid \mathsf{T} \mid \mathsf{F} \mid \neg P \mid P \wedge Q \mid P \vee Q \mid P \Rightarrow Q \mid P \Leftrightarrow Q$$

Predicate Logic — Universal Quantifiers

If P is a formula, then $\forall\ x.P$ is a formula

Pronounce $\forall\ x.P$ as 'for all x, P'.

Definition: $\forall\ x.P$ is true if (and only if) P is true for all values of x (taken from its specified domain).

Sometimes we write $P(x)$ for a formula that might mention x, so that we can write (e.g.) $P(27)$ for the formula with x instantiated to 27.

Then, if x is ranging over the naturals,
$\forall\ x.P(x)$ if and only if $P(0)$ and $P(1)$ and $P(2)$ and ...

Or, if x is ranging over $\{\mathrm{red}, \mathrm{green}, \mathrm{blue}\}$,then
$(\forall\ x.P(x))$ iff $P(\mathrm{red}) \wedge P(\mathrm{green}) \wedge P(\mathrm{blue})$.

Predicate Logic — Existential Quantifiers

If P is a formula, then $\exists\ x.P$ is a formula

Pronounce $\exists\ x.P$ as 'exists x such that P'.

Definition: $\exists\ x.P$ is true if (and only if) there is at least one value of x (taken from its specified domain) such that P is true.

So, if x is ranging over $\{\mathrm{red}, \mathrm{green}, \mathrm{blue}\}$, then
$(\exists\ x.P(x))$ iff $P(\mathrm{red}) \vee P(\mathrm{green}) \vee P(\mathrm{blue})$.

Because the domain might be infinite, we don't give truth-table definitions for $\forall$ and $\exists$.

Note also that we don't allow infinitary formulae — I carefully *didn't* write
$(\forall\ x.P(x))$ iff $P(0) \wedge P(1) \wedge P(2) \wedge \ldots$ $\times$

The Language of Predicate Logic

Summarising, the formulae of predicate logic are the terms of the grammar

$$\begin{aligned} P, Q \quad ::= \quad & A(x) \mid \mathsf{T} \mid \mathsf{F} \mid \neg P \mid P \wedge Q \mid P \vee Q \mid P \Rightarrow Q \mid \\ & P \Leftrightarrow Q \mid \forall\ x.P \mid \exists\ x.P \end{aligned}$$

Convention: the scope of a quantifier extends as far to the right as possible, so (e.g.) $\forall\ x.\mathrm{A}(x) \wedge \mathrm{B}(x)$ is $\forall x.(\mathrm{A}(x) \wedge \mathrm{B}(x))$, not $(\forall\ x.\mathrm{A}(x)) \wedge \mathrm{B}(x)$.

(other convention — no dot, always parenthesise: $\forall\ x(P)$)

Predicate Logic — Extensions

n-ary atomic predicates $\mathrm{A}(x, y)$, $\mathrm{B}(x, y, z)$,...

(regard our old p, q, etc. as 0-ary atomic predicates)

Equality as a special binary predicate $(e = e')$ where e and e' are some mathematical expressions (that might mention variables such as x), and similarly for $<, >, \leq, \geq$ over numbers.

$(e \neq e')$ iff $\neg(e = e')$

$(e \leq e')$ iff $(e < e') \vee (e = e')$

Predicate Logic — Examples

What do these mean? Are they true or false?

$\exists\, x.(x^2 + 2x + 1 = 0)$ where x ranges over the integers

$\forall\, x.(x < 0) \vee (x = 0) \vee (x \geq 0)$ where x ranges over the reals

$\forall\, x.(x \geq 0) \Rightarrow (2x > x)$ where x ranges over the reals

Predicate Logic — Examples

Formalise:

If someone learns discrete mathematics, then they will find a good job. (*)

Let x range over all people.

Write $\mathrm{L}(x)$ to mean 'x learns discrete mathematics'
Write $\mathrm{J}(x)$ to mean 'x will find a good job'

Then $\forall\, x.\mathrm{L}(x) \Rightarrow \mathrm{J}(x)$ is a reasonable formalisation of (*).

Is it true? We'd need to know more...

Predicate Logic — Nested Quantifers

What do these mean? Are they true?

$\forall\, x.\forall\, y.(x + y = y + x)$ where x, y range over the integers

$\forall\, x.\exists\, y.(x = y - 10)$ where x, y range over the integers

$\exists\, x.\forall\, y.(x \geq y)$ where x, y range over the integers

$\forall\, y.\exists\, x.(x \geq y)$ where x, y range over the integers

$\exists\, x.\exists\, y.(4x = 2y) \wedge (x + 1 = y)$ where x, y range over the integers

Predicate Logic — Examples

Formalise:

Every real number except 0 has a multiplicative inverse

$\forall\, x.(\neg(x = 0)) \Rightarrow \exists\, y.(x\ y = 1)$ where x ranges over the reals

Predicate Logic — Examples

Formalise:

Everyone has exactly one best friend.

Let x, y, z range over all people.

Write $\mathrm{B}(x, y)$ to mean y is a best friend of x

Then $\forall\, x.\exists\, y.\mathrm{B}(x, y) \wedge \forall\, z.\mathrm{B}(x, z) \Rightarrow z = y$ is one reasonable formalisation.

Equivalently $\forall\, x.\exists\, y.\mathrm{B}(x, y) \wedge \forall\, z.(\neg(z = y)) \Rightarrow \neg\mathrm{B}(x, z)$.

Um. what about $y = x$?

Predicate Logic — Basic Properties

De Morgan laws for quantifiers:

$(\neg\forall\ x.P)$ iff $\exists\ x.\neg P$

$(\neg\exists\ x.P)$ iff $\forall\ x.\neg P$

Distributing quantifiers over $\wedge$ and $\vee$:

$(\forall\ x.P \wedge Q)$ iff $(\forall\ x.P) \wedge (\forall\ x.Q)$

$(\exists\ x.P \wedge Q)$ ~~iff~~ $(\exists\ x.P) \wedge (\exists\ x.Q)$ × (left-to-right holds)

$(\forall\ x.P \vee Q)$ ~~iff~~ $(\forall\ x.P) \vee (\forall\ x.Q)$ × (right-to-left holds)

$(\exists\ x.P \vee Q)$ iff $(\exists\ x.P) \vee (\exists\ x.Q)$

Predicate Logic — Free and Bound Variables

A slightly odd (but well-formed) formula:

$\mathrm{A}(x) \wedge (\forall\ x.\mathrm{B}(x) \Rightarrow \exists\ x.\mathrm{C}(x,x))$

Really there are 3 different x's here, and it'd be clearer to write

$\mathrm{A}(x) \wedge (\forall\ x'.\mathrm{B}(x') \Rightarrow \exists\ x''.\mathrm{C}(x'',x''))$ or

$\mathrm{A}(x) \wedge (\forall\ y.\mathrm{B}(y) \Rightarrow \exists\ z.\mathrm{C}(z,z))$

Say an occurrence of x in a formula P is *free* if it is not inside any $(\forall\ x....)$ or $(\exists\ x....)$

All the other occurrences of x are *bound* by the closest enclosing $(\forall\ x....)$ or $(\exists\ x....)$

The *scope* of a quantifier in a formula $...(\forall\ x.P)...$ is all of P (except any subformulae of P of the form $\forall\ x....$ or $\exists\ x....$).

Truth Semantics

Whether a formula P is true or false might depend on

1. an interpretation of the atomic predicate symbols used in P (generalising the 'assumptions on its atomic propositions' we had before)

2. the values of the free variables of P

Often 1 is fixed (as it is for $e = e'$)

Application: Databases

4 Proof

Proof

Proof

We've now got a rich enough language to express some non-trivial conjectures, e.g.

$\forall\, n.(n \geq 2) \Rightarrow \neg\exists\, x, y, z.x \neq 0 \wedge y \neq 0 \wedge z \neq 0 \wedge x^n + y^n = z^n$

(where n ranges over the naturals)

Is that true or false?

Proof

$\forall\, n.(n \geq 2) \Rightarrow \neg\exists\, x, y.x \neq 0 \wedge y \neq 0 \wedge z \neq 0 \wedge x^n + y^n = z^n$

We have to be able to reason about this kind of thing, to *prove* that it's true (or to disprove it — to prove its negation...).

This course: 'informal' rigorous proof (normal mathematical practice). A proof is a rigorous argument to convince a very skeptical reader. It should be completely clear, and the individual steps small enough that there's no question about them.

(Later, study 'formal' proofs, as mathematical objects themselves...)

Non-Proofs

There are *lots*.

'I have discovered a truly remarkable proof which this margin is too small to contain.'

'I'm your lecturer, and I say it's true'

'The world would be a sad place if this wasn't true'

'I can't imagine that it could be false'

Statements

Theorem 1 [associativity of $+$] $\forall\, x, y, z.x + (y + z) = (x + y) + z$

Often leave top-level universal quantifiers implicit (but only in these top-level statements):

Theorem 2 $x + (y + z) = (x + y) + z$

Proposition — a little theorem

Lemma — a little theorem written down as part of a bigger proof

Corollary — an easy consequence of some theorem

any of those should come with a proof attached

Conjecture $x \text{ mod } 2 = 0 \vee x \text{ mod } 3 = 0 \vee x \text{ mod } 5 = 0$

Structured Proof

The truth-table and equational reasoning from before is still sound, but we need more, to reason about the quantifiers. And truth tables aren't going to help there.

Going to focus instead on the structure of the formulae we're trying to prove (and of those we can use).

Practice on statements about numbers — not that we care about these results particularly, but just to get started.

Example

Theorem? The sum of two rationals is rational.

Example

Theorem? The sum of two rationals is rational.

Clarify the logical form:

Theorem? $(\mathrm{Rational}(x) \wedge \mathrm{Rational}(y)) \Rightarrow \mathrm{Rational}(x+y)$

Theorem? The sum of two rationals is rational.

Clarify the logical form:

Theorem?
$\forall\, x.\forall\, y.(\mathrm{Rational}(x) \wedge \mathrm{Rational}(y)) \Rightarrow \mathrm{Rational}(x+y)$

and the definitions:

Say $\mathrm{Rational}(x)$ iff $\exists\, n, m.(x = n/m)$

where x and y range over real numbers and n and m range over integers.

Sometimes this clarification is a major intellectual activity (and the subsequent proof might be easy); sometimes it's easy to state the problem (but the proof is very hard).

How *far* we have to clarify the definitions depends on the problem — here I didn't define the reals, integers, addition, or division.

$\forall\, x.\forall\, y.(\mathrm{Rational}(x) \wedge \mathrm{Rational}(y)) \Rightarrow \mathrm{Rational}(x+y)$

1. Consider an arbitrary real x

now we aim to prove $\forall\, y.(\mathrm{Rational}(x) \wedge \mathrm{Rational}(y)) \Rightarrow \mathrm{Rational}(x+y)$

$\forall\, x.\forall\, y.(\mathrm{Rational}(x) \wedge \mathrm{Rational}(y)) \Rightarrow \mathrm{Rational}(x+y)$

1. Consider an arbitrary real x
2. Consider an arbitrary real y

now we aim to prove $(\mathrm{Rational}(x) \wedge \mathrm{Rational}(y)) \Rightarrow \mathrm{Rational}(x+y)$

$\forall\, x.\forall\, y.(\mathrm{Rational}(x) \wedge \mathrm{Rational}(y)) \Rightarrow \mathrm{Rational}(x+y)$

1. Consider an arbitrary real x
2. Consider an arbitrary real y

3. Assume $\mathrm{Rational}(x) \wedge \mathrm{Rational}(y)$

now we aim to prove $\mathrm{Rational}(x+y)$

$\forall\, x.\forall\, y.(\mathrm{Rational}(x) \wedge \mathrm{Rational}(y)) \Rightarrow \mathrm{Rational}(x+y)$

1. Consider an arbitrary real x
2. Consider an arbitrary real y

3. Assume $\mathrm{Rational}(x) \wedge \mathrm{Rational}(y)$
4. $\mathrm{Rational}(x)$ from 3 by $\wedge$-elimination

now we aim to prove $\mathrm{Rational}(x+y)$

$\forall\, x.\forall\, y.(\mathrm{Rational}(x) \wedge \mathrm{Rational}(y)) \Rightarrow \mathrm{Rational}(x+y)$

1. Consider an arbitrary real x
2. Consider an arbitrary real y

3. Assume $\mathrm{Rational}(x) \wedge \mathrm{Rational}(y)$
4. $\mathrm{Rational}(x)$ from 3 by $\wedge$-elimination
5. $\mathrm{Rational}(y)$ from 3 by $\wedge$-elimination

now we aim to prove $\mathrm{Rational}(x+y)$

$\forall\, x.\forall\, y.(\mathrm{Rational}(x) \wedge \mathrm{Rational}(y)) \Rightarrow \mathrm{Rational}(x+y)$

1. Consider an arbitrary real x
2. Consider an arbitrary real y

3. Assume $\mathrm{Rational}(x) \wedge \mathrm{Rational}(y)$
4. $\mathrm{Rational}(x)$ from 3 by $\wedge$-elimination
5. $\mathrm{Rational}(y)$ from 3 by $\wedge$-elimination
6. $\exists\, n, m.(x = n/m)$ from 4 by unfolding the defn of $\mathrm{Rational}$
7. $\exists\, n, m.(y = n/m)$ from 5 by unfolding the defn of $\mathrm{Rational}$

now we aim to prove $\mathrm{Rational}(x+y)$

$\forall\, x.\forall\, y.(\text{Rational}(x) \wedge \text{Rational}(y)) \Rightarrow \text{Rational}(x+y)$

1. Consider an arbitrary real x
2. Consider an arbitrary real y

3. Assume $\text{Rational}(x) \wedge \text{Rational}(y)$
4. $\text{Rational}(x)$ from 3 by $\wedge$-elimination
5. $\text{Rational}(y)$ from 3 by $\wedge$-elimination
6. $\exists\, n, m.(x = n/m)$ from 4 by unfolding the defn of Rational
7. $\exists\, n, m.(y = n/m)$ from 5 by unfolding the defn of Rational
8. For some actual (integer) n_1 and m_1, $x = n_1/m_1$ from 6 by $\exists$-elimination
9. For some actual (integer) n_2 and m_2, $y = n_2/m_2$ from 7 by $\exists$-elimination

now we aim to prove $\text{Rational}(x+y)$

$\forall\, x.\forall\, y.(\text{Rational}(x) \wedge \text{Rational}(y)) \Rightarrow \text{Rational}(x+y)$

1. Consider an arbitrary real x
2. Consider an arbitrary real y

3. Assume $\text{Rational}(x) \wedge \text{Rational}(y)$

...

8. For some actual (integer) n_1 and m_1, $x = n_1/m_1$ from 6 by $\exists$-elimination
9. For some actual (integer) n_2 and m_2, $y = n_2/m_2$ from 7 by $\exists$-elimination
10. $x + y = (n_1/m_1) + (n_2/m_2)$ from 8 and 9, adding both sides
11. $= \frac{n_1\ m_2}{m_1\ m_2} + \frac{m_1\ n_2}{m_1\ m_2}$ from 10, by arithmetic
12. $= \frac{n_1\ m_2 + m_1\ n_2}{m_1\ m_2}$ from 11, by arithmetic

now we aim to prove $\text{Rational}(x+y)$

$\forall\, x.\forall\, y.(\text{Rational}(x) \wedge \text{Rational}(y)) \Rightarrow \text{Rational}(x+y)$

1. Consider an arbitrary real x
2. Consider an arbitrary real y

3. Assume $\text{Rational}(x) \wedge \text{Rational}(y)$

...

10. $x + y = (n_1/m_1) + (n_2/m_2)$ from 8 and 9, adding both sides
11. $= \frac{n_1\ m_2}{m_1\ m_2} + \frac{m_1\ n_2}{m_1\ m_2}$ from 10, by arithmetic
12. $= \frac{n_1\ m_2 + m_1\ n_2}{m_1\ m_2}$ from 11, by arithmetic
13. $\exists\, n, m. x{+}y = n/m$ from 10–12, witness $n = n_1\ m_2 + m_1\ n_2$
    $m = m_1\ m_2$

now we aim to prove $\text{Rational}(x+y)$

$\forall\, x.\forall\, y.(\mathrm{Rational}(x) \wedge \mathrm{Rational}(y)) \Rightarrow \mathrm{Rational}(x+y)$

1. Consider an arbitrary real x

2. Consider an arbitrary real y

3. Assume $\mathrm{Rational}(x) \wedge \mathrm{Rational}(y)$

...

10. $x + y = (n_1/m_1) + (n_2/m_2)$ from 8 and 9, adding both sides

11. $\quad = \frac{n_1\ m_2}{m_1\ m_2} + \frac{m_1\ n_2}{m_1\ m_2}$ from 10, by arithmetic

12. $\quad = \frac{n_1\ m_2 + m_1\ n_2}{m_1\ m_2}$ from 11, by arithmetic

13. $\exists\, n, m. x+y = n/m$ from 10–12, witness $n = n_1\ m_2 + m_1\ n_2$, $m = m_1\ m_2$

14. $\mathrm{Rational}(x+y)$ from 13, folding the defn of $\mathrm{Rational}$

now we aim to prove $\mathrm{Rational}(x+y)$ — but we have! so:

$\forall\, x.\forall\, y.(\mathrm{Rational}(x) \wedge \mathrm{Rational}(y)) \Rightarrow \mathrm{Rational}(x+y)$

1. Consider an arbitrary real x

2. Consider an arbitrary real y

3. Assume $\mathrm{Rational}(x) \wedge \mathrm{Rational}(y)$

...

10. $x + y = (n_1/m_1) + (n_2/m_2)$ from 8 and 9, adding both sides

11. $\quad = \frac{n_1\ m_2}{m_1\ m_2} + \frac{m_1\ n_2}{m_1\ m_2}$ from 10, by arithmetic

12. $\quad = \frac{n_1\ m_2 + m_1\ n_2}{m_1\ m_2}$ from 11, by arithmetic

13. $\exists\, n, m. x+y = n/m$ from 10–12, witness $n = n_1\ m_2 + m_1\ n_2$, $m = m_1\ m_2$

14. $\mathrm{Rational}(x+y)$ from 13, folding the defn of $\mathrm{Rational}$

15. $(\mathrm{Rational}(x) \wedge \mathrm{Rational}(y)) \Rightarrow \mathrm{Rational}(x+y)$ by $\Rightarrow$-I, 3–14

now we aim to prove $\forall\, x.\forall\, y.(\mathrm{Rational}(x) \wedge \mathrm{Rational}(y)) \Rightarrow \mathrm{Rational}(x+y)$

$\forall\, x.\forall\, y.(\mathrm{Rational}(x) \wedge \mathrm{Rational}(y)) \Rightarrow \mathrm{Rational}(x+y)$

1. Consider an arbitrary real x.
2. Consider an arbitrary real y.
3. Assume $\mathrm{Rational}(x) \wedge \mathrm{Rational}(y)$.
4. $\mathrm{Rational}(x)$ from 3 by $\wedge$-elimination
5. $\mathrm{Rational}(y)$ from 3 by $\wedge$-elimination
6. $\exists\, n, m.(x = n/m)$ from 4 by unfolding the defn of $\mathrm{Rational}$
7. $\exists\, n, m.(y = n/m)$ from 5 by unfolding the defn of $\mathrm{Rational}$
8. For some actual (integer) n_1 and m_1, $x = n_1/m_1$ from 6 by $\exists$-elimination
9. For some actual (integer) n_2 and m_2, $y = n_2/m_2$ from 7 by $\exists$-elimination
10. $x + y = (n_1/m_1) + (n_2/m_2)$ from 8 and 9, adding both sides
11. $\quad = \frac{n_1\ m_2}{m_1\ m_2} + \frac{m_1\ n_2}{m_1\ m_2}$ from 10, by arithmetic
12. $\quad = \frac{n_1\ m_2 + m_1\ n_2}{m_1\ m_2}$ from 11, by arithmetic
13. $\exists\, n, m. x + y = n/m$ from 10–12, witness $n = n_1\ m_2 + m_1\ n_2$, $m = m_1\ m_2$
14. $\mathrm{Rational}(x+y)$ from 13, folding the defn of $\mathrm{Rational}$
15. $(\mathrm{Rational}(x) \wedge \mathrm{Rational}(y)) \Rightarrow \mathrm{Rational}(x+y)$ by $\Rightarrow$-introduction, from 3–14
16. $\forall\, y.(\mathrm{Rational}(x) \wedge \mathrm{Rational}(y)) \Rightarrow \mathrm{Rational}(x+y)$ by $\forall$-introduction, from 2–15
17. $\forall\, x.\forall\, y.(\mathrm{Rational}(x) \wedge \mathrm{Rational}(y)) \Rightarrow \mathrm{Rational}(x+y)$ by $\forall$-introduction, from 1–16

Theorem $(\mathrm{Rational}(x) \wedge \mathrm{Rational}(y)) \Rightarrow \mathrm{Rational}(x+y)$

Proof

1. Consider an arbitrary real x.
2. Consider an arbitrary real y.
3. Assume $\mathrm{Rational}(x) \wedge \mathrm{Rational}(y)$.
4. $\mathrm{Rational}(x)$ from 3 by $\wedge$-elimination
5. $\mathrm{Rational}(y)$ from 3 by $\wedge$-elimination
6. $\exists\ n, m.(x = n/m)$ from 4 by unfolding the defn of $\mathrm{Rational}$
7. $\exists\ n, m.(y = n/m)$ from 5 by unfolding the defn of $\mathrm{Rational}$
8. For some actual (integer) n_1 and m_1, $x = n_1/m_1$ from 6 by $\exists$-elimination
9. For some actual (integer) n_2 and m_2, $y = n_2/m_2$ from 7 by $\exists$-elimination
10. $x + y = (n_1/m_1) + (n_2/m_2)$ from 8 and 9, adding both sides
11. $= \frac{n_1\ m_2}{m_1\ m_2} + \frac{m_1\ n_2}{m_1\ m_2}$ from 10, by arithmetic
12. $= \frac{n_1\ m_2 + m_1\ n_2}{m_1\ m_2}$ from 11, by arithmetic
13. $\exists\ n, m. x + y = n/m$ from 10–12, witness $n = n_1\ m_2 + m_1\ n_2$, $m = m_1\ m_2$
14. $\mathrm{Rational}(x+y)$ from 13, folding the defn of $\mathrm{Rational}$
15. $(\mathrm{Rational}(x) \wedge \mathrm{Rational}(y)) \Rightarrow \mathrm{Rational}(x+y)$ by $\Rightarrow$-introduction, from 3–14
16. $\forall\ y.(\mathrm{Rational}(x) \wedge \mathrm{Rational}(y)) \Rightarrow \mathrm{Rational}(x+y)$ by $\forall$-introduction, from 2–15
17. $\forall\ x.\forall\ y.(\mathrm{Rational}(x) \wedge \mathrm{Rational}(y)) \Rightarrow \mathrm{Rational}(x+y)$ by $\forall$-introduction, from 1–16 □

What is a Proof (in this stylised form)?

A list of lines, each of which is either:

- a formula of predicate logic, with a justification (‘P, from ... by ...’)
- an assumption of some formula (‘Assume P’)
- an introduction of a arbitrary variable (‘Consider an arbitrary x (from the appropriate domain)’)
- an introduction of some actual witness variables and a formula (‘For some actual n, P’)

When we make an assumption, we open a box. We have to close it before we can discharge the assumption (by $\Rightarrow$-introduction at step 15).

(Actually also for introductions of arbitrary and witness variables. But if these are just at the top level, and we do $\forall$-introduction on them at the end, we might not draw them.)

What is a Proof (in this stylised form)?

Lines are numbered

Introduced variables must be *fresh* (not free in any preceeding formula).

The justifications must not refer to later lines (no circular proofs, please!)

1. P by ... from 15 $\times$

...

15. Q by ... from 1

What is a Proof (in this stylised form)?

The justifications must not refer to lines inside any earlier box

1. Assume P

...

15. U from ... by ...

...

27. Q from ... by ...

28. $P \Rightarrow Q$ by $\Rightarrow$-introduction, from 1–27

29. Assume R

...

1007. ... from 15 by ... $\times$

(earlier in an enclosing box is ok)

What is a Justification (in this stylised form)?
Back to the Connectives — And

To use a conjunction: if we know $P \wedge Q$, then we can deduce P, or we can deduce Q (or both, as often as we like)

...

$m.$ $P \wedge Q$ from ...

...

$n.$ P from m by $\wedge$-elimination

or

...

$m.$ $P \wedge Q$ from ...

...

$n.$ Q from m by $\wedge$-elimination

What is a Justification (in this stylised form)?
Back to the Connectives — And

To prove a conjunction: we can prove $P \wedge Q$ by proving P and proving Q.

...

$l.$ P from ...

...

$m.$ Q from ...

...

$n.$ $P \wedge Q$ from l and m by $\wedge$-introduction

(it doesn't matter in what order l and m are in)

What is a Justification (in this stylised form)?
Back to the Connectives — Or

To prove a disjunction: to prove $P \vee Q$, we could prove P, or we could prove Q. (could even use $\neg Q$ or $\neg P$ resp.)

...

m. P from ...

...

n. $P \vee Q$ from m by $\vee$-introduction

or

...

m. Q from ...

...

n. $P \vee Q$ from m by $\vee$-introduction

What is a Justification (in this stylised form)?
Back to the Connectives — Or

To use a disjunction: if we know $P \vee Q$, and by assuming P we can prove R, and by assuming Q we can prove R, then we can deduce R (a form of case analysis).

l. $P \vee Q$ from ... by ...

...

m_1. Assume P
...
m_2. R

...

n_1. Assume Q
...
n_2. R

...

o. R from l, m_1–m_2, n_1–n_2 by $\vee$-elimination

(it doesn't matter what order l, m_1–m_2, and n_1–n_2 are in)

What is a Justification (in this stylised form)?
Back to the Connectives — Implication

To prove an implication: to prove $P \Rightarrow Q$, assume P, prove Q, and discharge the assumption.

...

m. Assume P
...
n. Q from ... by ...

$n+1$. $P \Rightarrow Q$ from m–n, by $\Rightarrow$-introduction

What is a Justification (in this stylised form)?

Back to the Connectives — Implication

To use an implication: if we know $P \Rightarrow Q$, and we know P, we can deduce Q

...
l. $P \Rightarrow Q$ by ...
...
m. P by ...
...
n. Q from l and m by $\Rightarrow$-elimination

(also known as *modus ponens*)

What is a Justification (in this stylised form)?

Back to the Connectives — Negation

To prove a negation: to prove $\neg P$, assume P, prove F, and discharge the assumption.

...
m. Assume P
...
n. F from ... by ...
$n+1$. $\neg P$ from m–n, by $\neg$-introduction

That's a lot like $\Rightarrow$-introduction (not a surprise, as $\neg P$ iff $(P \Rightarrow \mathsf{F})$).

What is a Justification (in this stylised form)?

Back to the Connectives — Negation

To use a negation: if we know $\neg P$, and we know P, we can deduce F

...
l. P by ...
...
m. $\neg P$ by ...
...
n. F from l and m by $\neg$-elimination

What is a Justification (in this stylised form)?

Back to the Connectives — Truth

To prove T: nothing to do

...
n. T

That's not very useful, though... because:

To use T: you can't do anything with it.

What is a Justification (in this stylised form)?
Contradiction

To prove P by contradiction: if, from assuming $\neg P$, we can prove F, then we can deduce P

...
m. Assume $\neg P$
...
n. F from ... by ...
$n+1$. P from m–n, by contradiction

Note that in the other rules either a premise (for elimination rules) or the conclusion (for introduction rules) had some particular form, but here the conclusion is an arbitrary P.

What is a Justification (in this stylised form)?
Contradiction′

To prove P by contradiction: if we can deduce F, then we can deduce any P

...
m. F from ... by ...
...
n. P from m, by contradiction

(hopefully this would be under some assumption(s)...)

Example

Theorem $(P \wedge Q) \Rightarrow (P \vee Q)$

Proof:

1. Assume $P \wedge Q$
2. P from 1 by $\wedge$-elim
3. $P \vee Q$ from 2 by $\vee$-intro
4. $(P \wedge Q) \Rightarrow (P \vee Q)$ from 1–3 by $\Rightarrow$-intro

$\square$

Example

Theorem ? $(P \vee Q) \Rightarrow (P \wedge Q)$

Proof ?:

1. Assume $P \vee Q$
2. ...use $\vee$-elim somehow? prove by contradiction?

????

$n-2$. P from ? by ?
$n-1$. Q from ? by ?
n. $(P \wedge Q)$ from $n-1$, $n-2$ by $\wedge$-intro
$n+1$. $(P \vee Q) \Rightarrow (P \wedge Q)$ from 1–n by $\Rightarrow$-intro

Counterexample? Prove negation?

What is a Justification (in this stylised form)?

Back to the Connectives — For all

To use a universally quantified formula: if we know $\forall x.P(x)$, then we can deduce $P(v)$ for any v (of the appropriate domain)

...

$m.$ $\forall\ x.P(x)$ from ...

...

$n.$ $P(v)$ from m by $\forall$-elimination

What is a Justification (in this stylised form)?

Back to the Connectives — For all

To prove a universally quantified formula $\forall\ x.P(x)$, consider an arbitrary fresh variable x (ranging over the appropriate domain) and prove $P(x)$, then discharge the assumption.

...

> $m.$ Consider an arbitrary x (from domain ...)
>
> ...
>
> $n.$ $P(x)$ by ...

$n+1.$ $\forall\ x.P(x)$ from m–n by $\forall$-introduction

What is a Justification (in this stylised form)?

Back to the Connectives — Exists

To prove an existentially quantified formula $\exists\ x.P(x)$, prove $P(v)$ for some witness v (from the appropriate domain).

...

$m.$ $P(v)$

...

$n.$ $\exists\ x.P(x)$ from m by $\exists$-introduction with witness $x = v$

What is a Justification (in this stylised form)?

Back to the Connectives — Exists

To use an existentially quantified formula $\exists\ x.P(x)$, introduce a fresh variable (ranging over the appropriate domain) x_1, about which we know only $P(x_1)$

...

$m.$ $\exists\ x.P(x)$

...

$n.$ For some actual x_1, $P(x_1)$ from m by $\exists$-elimination

That's a special case of this more general rule:

$l.$ $\exists\ x.P(x)$

...

> $m.$ For some actual x_1, $P(x_1)$
>
> ...
>
> $n.$ Q (where x_1 not free in Q)

...

$o.$ Q from l, m–n, by $\exists$-elimination

Example

Many theorems have a similar top-level structure, e.g.

$\forall\, x, y, z.(P \wedge Q \wedge R) \Rightarrow S$

1. Consider an arbitrary x, y, z.

2. Assume $P \wedge Q \wedge R$.

3. P from 2 by $\wedge$-elimination

4. Q from 2 by $\wedge$-elimination

5. R from 2 by $\wedge$-elimination

...

215. S by ...

216. $(P \wedge Q \wedge R) \Rightarrow S$ from 2–215 by $\Rightarrow$-introduction

217. $\forall\, x, y, z.(P \wedge Q \wedge R) \Rightarrow S$ by $\forall$-introduction, from 1–216

What is a Proof (in this stylised form)?

NB This particular stylised form is only one way to write down rigorous paper proofs. It's a good place to start, but its not always appropriate. Later, you'll sometimes take bigger steps, and won't draw the boxes.

But however they are written, they *have* to be written down clearly — a proof is a communication tool, to persuade. Each step needs a justification.

In questions, we'll say specifically "by structured proof", "by equational reasoning", "by truth tables", or, more generally "prove".

(This is basically the 'box and line' proofs from Bornat 2005, which are a linear notation for *natural deduction* proofs. More on that in 1B Logic & Proof. If you want to try mechanised proofs, see `jape.org.uk` or `prover.cs.ru.nl` (experiments! — let me know how it goes))

Soundness and Completeness?

Are these proof rules *sound*? (i.e., are all the provable formulae valid?)

Are these proof rules *complete*? (i.e., are all valid formulae provable?)

Think about *proof search*

Aside: Writing Discrete Maths

By hand

In ASCII

```
P ::= T | F | p | A(x) | P /\ Q | P \/ Q
    | P=>Q | P<=>Q | !x.P | ?x.P
```

In LaTeX (but don't forget that typesetting is *not* real work)

Pragmatics

Given some conjecture:

1. Ensure the statement is well-defined, and that you know the definitions of whatever it uses.
2. Understand intuitive what it's saying. Verbalize it.
3. Intuitively, why is it true? (or false?)
4. What are the hard (or easy) cases likely to be?
5. Choose a strategy — truth tables, equational reasoning, structured proof, induction, ...
6. Try it! (but be prepared to backtrack)
7. Expand definitions and make abbreviations as you need them.
8. Writing — to communicate, and to help you think.

9. Choose variable names carefully; take care with parentheses
10. Use enough words and use enough symbols, but keep them properly nested. Don't use random squiggles ("$\Rightarrow$" or "$\because$") for meta-reasoning.
11. If it hasn't worked yet... either
    (a) you've make some local mistake (mis-instantiated, re-used a variable name, not expanded definitions enough, forgotten a useful assumption). Fix it and continue.
    (b) you've found that the conjecture is false. Construct a simple counterexample and check it.
    (c) you need to try a different strategy (different induction principle, strengthened induction hypothesis, proof by contradictions,...)
    (d) you didn't really understand intuitively what the conjecture is saying, or what the definitions it uses mean. Go back to them again.

12. If it has worked: read through it, skeptically. Maybe *re-write* it.
13. Finally, give it to someone else, as skeptical and careful as you can find, to see if they believe it — to see if they believe that *what you've written down is a proof*, not that they believe that *the conjecture is true*.

...more fallacies

5 Set Theory

Set Theory

Set Theory

Now we've got some reasoning techniques, but not much to reason *about*.

Let's add *sets* to our language.

What is a set? An unordered collection of *elements*:

$\{0,3,7\} = \{3,0,7\}$

might be empty:

$\{\} = \emptyset = \varnothing$

might be infinite:

$$\begin{aligned} \mathbb{N} &= \{0,1,2,3...\} \\ \mathbb{Z} &= \{...,-1,0,1,...\} \\ \mathbb{R} &= \text{...all the real numbers} \end{aligned}$$

Some more interesting sets

the set of nodes in a network (encode with $\mathbb{N}$?)

the set of paths between such nodes (encode ??)

the set of polynomial-time computable functions from naturals to naturals

the set of well-typed programs in some programming language (encode???)

the set of executions of such programs

the set of formulae of predicate logic

the set of valid proofs of such formulae

the set of all students in this room (?)

the set of all sets $\times$

Basic relationships

membership $x \in A$

$3 \in \{1,3,5\}$

$2 \notin \{1,3,5\}$

(of course $(2 \notin \{1,3,5\})$ iff $\neg(2 \in \{3,5,1\})$)

equality between sets $A = \mathrm{B}$ iff $\forall\, x.x \in A \Leftrightarrow x \in \mathrm{B}$

$\{1,2\} = \{2,1\} = \{2,1,2,2\} \qquad \{\} \neq \{\{\}\}$

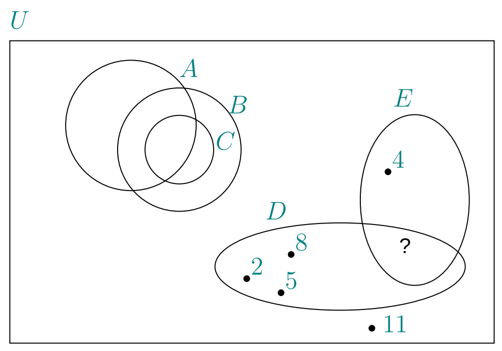
inclusion or *subset* $A \subseteq \mathrm{B}$ iff $\forall\, x.x \in A \Rightarrow x \in \mathrm{B}$

Properties: $\subseteq$ is reflexive, transitive,

and antisymmetric ($(A \subseteq \mathrm{B} \wedge \mathrm{B} \subseteq A) \Rightarrow A = \mathrm{B}$)

but not total: $\{1,2\} \not\subseteq \{1,3\} \not\subseteq \{1,2\}$

Venn Diagrams



Bounded Quantifiers

Write

$\forall\, x \in A . P$ for $\forall\, x . x \in A \Rightarrow P$

$\exists\, x \in A . P$ for $\exists\, x . x \in A \wedge P$

where A is a subset of the domain that x ranges over.

Define Even to be the set of all even naturals

Then can write $\forall\, n \in \mathrm{Even} . \exists\, m \in \mathbb{N} . n = 3m$

Building interesting subsets with set comprehension

$\mathrm{Even} \overset{\mathrm{def}}{=} \{n \mid \exists\, m \in \mathbb{N} . n = 2m\}$

$\{x \mid x \in \mathbb{N} \wedge \neg\exists\, y, z \in \mathbb{N} . y > 1 \wedge z > 1 \wedge y\ z = x\}$

$\{x \mid x \in \mathbb{N} \wedge \forall\, y \in \mathbb{N} . y > x\}$

$\{2\ x \mid x \in \mathbb{N}\}$

From sets to predicates, and back again

From sets to predicates: given a set A, can define a predicate
$P(x) \overset{\mathrm{def}}{=} x \in A$

From predicates to sets: given $P(x)$ and some set U, can build a set
$A \overset{\mathrm{def}}{=} \{x \mid x \in U \wedge P(x)\}$

(in some logics we'd really identify the two concepts – but not here)

Property of comprehensions: $x \in \{y \mid P(y)\}$ iff $P(x)$

Building new sets from old ones: union, intersection, and difference

$A \cup B \overset{\text{def}}{=} \{x \mid x \in A \vee x \in B\}$

$A \cap B \overset{\text{def}}{=} \{x \mid x \in A \wedge x \in B\}$

$A - B \overset{\text{def}}{=} \{x \mid x \in A \wedge x \notin B\}$

A and B are *disjoint* iff $A \cap B = \{\}$ (symm, not refl or tran)

Building new sets from old ones: union, intersection, and difference

$\{1,2\} \cup \{2,3\} = \{1,2,3\}$

$\{1,2\} \cap \{2,3\} = \{2\}$

$\{1,2\} - \{2,3\} = \{1\}$

Properties of union, intersection, and difference

Recall $\vee$ is associative: $P \vee (Q \vee R)$ iff $(P \vee Q) \vee R$

Theorem $A \cup (B \cup C) = (A \cup B) \cup C$

Proof

$A \cup (B \cup C)$

1. $= \{x \mid x \in A \vee x \in (B \cup C)\}$ unfold defn of union
2. $= \{x \mid x \in A \vee x \in \{y \mid y \in B \vee y \in C\}\}$ unfold defn of union
3. $= \{x \mid x \in A \vee (y \in B \vee y \in C)\}$ comprehension property
4. $= \{x \mid (x \in A \vee y \in B) \vee y \in C\}$ by $\vee$ assoc
5. $= (A \cup B) \cup C$ by the comprehension property and folding defn of union twice $\square$

Some Collected Set Equalities, for Reference

For any sets A, B, and C, all subsets of U

Commutativity:

$A \cap B = B \cap A$ ($\cap$-comm)

$A \cup B = B \cup A$ ($\cup$-comm)

Associativity:

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

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

Distributivity:

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

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

Identity:

$A \cap U = A$ ($\cap$-id)

$A \cup \{\} = A$ ($\cup$-id)

Unit:

$A \cap \{\} = \{\}$ ($\cap$-unit)

$A \cup U = U$ ($\cup$-unit)

Complement:

$A \cap (U - A) = \{\}$ ($\cap$-comp)

$A \cup (U - A) = U$ ($\cup$-comp)

De Morgan:

$U - (A \cap B) = (U - A) \cup (U - B)$ ($\cap$-DM)

$U - (A \cup B) = (U - A) \cap (U - B)$ ($\cup$-DM)

Example Proof

Theorem $\{\} \subseteq A$

Proof

$\{\} \subseteq A$

1. iff $\forall x . x \in \{\} \Rightarrow x \in A$ unfolding defn of $\subseteq$
2. iff $\forall x .$F $\Rightarrow x \in A$ use defn of $\in$
3. iff $\forall x .$T equational reasoning with (F $\Rightarrow P$) iff T
4. iff T using defn of $\forall$ $\square$

Another Proof of the Same Theorem

Theorem $\{\} \subseteq A$

Another Proof (using the structured rules more explicitly)

1. $\{\} \subseteq A$ iff $\forall x . x \in \{\} \Rightarrow x \in A$ unfolding defn of $\subseteq$

We prove the r.h.s.:

2. Consider an arbitrary x
    3. Assume $x \in \{\}$
    4. F by defn of $\in$
    5. $x \in A$ from 4, by contradiction
6. $x \in \{\} \Rightarrow x \in A$ from 3–5, by $\Rightarrow$-introduction

7. $\forall x . x \in \{\} \Rightarrow x \in A$ from 2–6, by $\forall$-introduction $\square$

Building new sets from old ones: powerset

Write $\mathcal{P}(A)$ for the set of all subsets of a set A.

$\mathcal{P}\{\} = \{\{\}\}$

$\mathcal{P}\{7\} = \{\{\}, \{7\}\}$

$\mathcal{P}\{1, 2\} = \{\{\}, \{1\}, \{2\}, \{1, 2\}\}$

$A \in \mathcal{P}(A)$

(why 'power' set?)

Building new sets from old ones: product

Write (a, b) (or sometimes $\langle a, b\rangle$) for an ordered pair of a and b

$A \times \mathrm{B} \overset{\text{def}}{=} \{(a, b) \mid a \in A \wedge b \in \mathrm{B}\}$

Similarly for triples $(a, b, c) \in A \times \mathrm{B} \times \mathrm{C}$ etc.

Pairing is *non-commutative*: $(a, b) \neq (b, a)$ unless $a = b$

Pairing is *non-associative* and distinct from 3-tupling etc:
$(a, (b, c)) \neq (a, b, c) \neq ((a, b), c)$ and
$A \times (\mathrm{B} \times \mathrm{C}) \neq A \times \mathrm{B} \times \mathrm{C} \neq (A \times \mathrm{B}) \times \mathrm{C}$

Why 'product'?
$\{1, 2\} \times \{\mathrm{red}, \mathrm{green}\} = \{(1, \mathrm{red}), (2, \mathrm{red}), (1, \mathrm{green}), (2, \mathrm{green})\}$

We know $(a, b) = (b, a) \Rightarrow a = b$ for pairs

so why not lift the result to set product?

Theorem ? $(A \times B = B \times A) \Rightarrow A = B$

Proof?

The first components of the pairs in $A \times B$ are from A.

The first components of the pairs in $B \times A$ are from B.

If $A \times B = B \times A$ then these must be the same, so $A = B$.

Theorem ? $(A \times B = B \times A) \Rightarrow A = B$

Proof?

1. Assume $A \times B = B \times A$

We prove $A = B$, i.e. $\forall x.x \in A \Leftrightarrow x \in B$

2. Consider an arbitrary x.

We first prove the $\Rightarrow$ implication.

3. Assume $x \in A$.

4. Consider an arbitrary $y \in B$.

5. $(x, y) \in A \times B$ by defn $\times$

6. $(x, y) \in B \times A$ by 1

7. $x \in B$ by defn $\times$

8. $x \in A \Rightarrow x \in B$ from 3–7 by $\Rightarrow$-introduction

9. The proof of the $\Leftarrow$ implication is symmetric

10. $\forall x.x \in A \Leftrightarrow x \in B$ from 2–9 by $\forall$-introduction

---

Theorem $(A \times B = B \times A) \wedge A \neq \emptyset \wedge B \neq \emptyset \Rightarrow A = B$

Proof

0. Assume $A \neq \emptyset$ and $B \neq \emptyset$

1. Assume $A \times B = B \times A$

We prove $A = B$, i.e. $\forall x.x \in A \Leftrightarrow x \in B$

2. Consider an arbitrary x.

We first prove the $\Rightarrow$ implication.

3. Assume $x \in A$.

4. Consider an arbitrary $y \in B$ using 0

5. $(x, y) \in A \times B$ by defn $\times$

6. $(x, y) \in B \times A$ by 1

7. $x \in B$ by defn $\times$

8. $x \in A \Rightarrow x \in B$ from 3–7 by $\Rightarrow$-introduction

9. The proof of the $\Leftarrow$ implication is symmetric

10. $\forall x.x \in A \Leftrightarrow x \in B$ from 2–9 by $\forall$-introduction $\square$

Theorem $(A \times B = B \times A) \wedge A \neq \emptyset \wedge B \neq \emptyset \Rightarrow A = B$

or equivalently

Theorem $(A \times B = B \times A) \Rightarrow A = B \vee A = \emptyset \vee B = \emptyset$

using $((P \wedge R) \Rightarrow Q)$ iff $(P \Rightarrow Q \vee \neg R)$ and De Morgan

Aside

Let $A \overset{\text{def}}{=} \{n \mid n = n+1\}$

Is $\forall\, x \in A. x = 7$ true?

Or $\forall\, x \in A. x = x+1$? Or $\forall\, x \in A. 1 = 2$?

Is $\exists\, x \in A. 1+1 = 2$ true?

5.1 Relations, Graphs, and Orders

Relations, Graphs, and Orders

Using Products: Relations

Say a (binary) *relation* R between two sets A and B is a subset of all the (a, b) pairs (where $a \in A$ and $b \in \mathrm{B}$)

$R \subseteq A \times \mathrm{B}$ (or, or course, $R \in \mathcal{P}(A \times \mathrm{B})$)

Extremes: $\varnothing$ and $A \times \mathrm{B}$ are both relations between A and B

$1_A \overset{\text{def}}{=} \{(a, a) \mid a \in A\}$ is the *identity relation* on A

$\varnothing \subseteq 1_A \subseteq A \times A$

Sometimes write infix: $a\ R\ b \overset{\text{def}}{=} (a, b) \in R$

Relational Composition

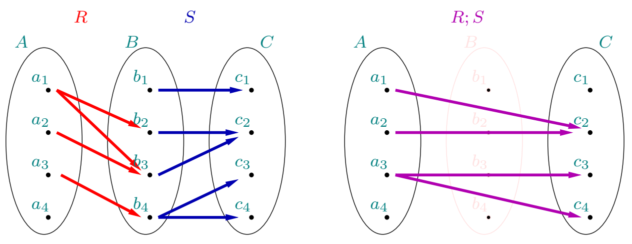
Given $R \subseteq A \times \mathrm{B}$ and $S \subseteq \mathrm{B} \times \mathrm{C}$, their *relational composition* is

$R; S \overset{\text{def}}{=} \{(a, c) \mid \exists\, b.(a, b) \in R \wedge (b, c) \in S\}$

$R; S \subseteq A \times \mathrm{C}$

Sometimes write that the other way round: $S \circ R \overset{\text{def}}{=} R; S$
(to match function composition)

Relational Composition



$A \overset{\text{def}}{=} \{a_1, a_2, a_3, a_4\}$ B $\overset{\text{def}}{=} \{b_1, b_2, b_3, b_4\}$ C $\overset{\text{def}}{=} \{c_1, c_2, c_3, c_4\}$

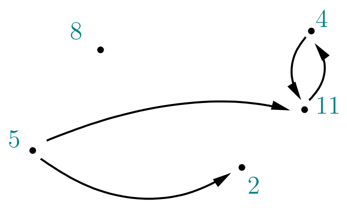
$R \overset{\text{def}}{=} \{(a_1, b_2), (a_1, b_3), (a_2, b_3), (a_3, b_4)\}$

$S \overset{\text{def}}{=} \{(b_1, c_1), (b_2, c_2), (b_3, c_2), (b_4, c_3), (b_4, c_4)\}$

$R; S = \{(a_1, c_2), (a_2, c_2), (a_3, c_3), (a_3, c_4)\}$

Relations as Directed Graphs

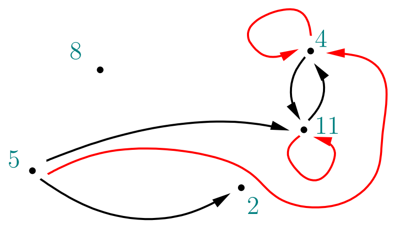
Relations from a set to itself



$G \subseteq \mathbb{N} \times \mathbb{N}$

$G = \{(5, 2), (5, 11), (4, 11), (11, 4)\}$

Transitivity



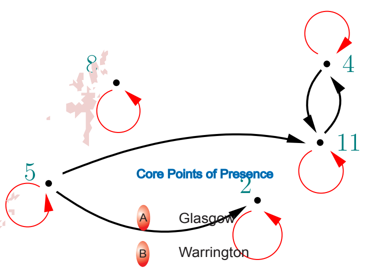
$R \subseteq A \times A$

$R^+ \overset{\text{def}}{=} R \cup (R; R) \cup (R; R; R) \cup \ldots$

$G^+ = \{(5, 2), (5, 11), (4, 11), (11, 4)\} \cup \{(5, 4), (11, 11), (4, 4)\}$

R *is transitive* if $R = R^+$

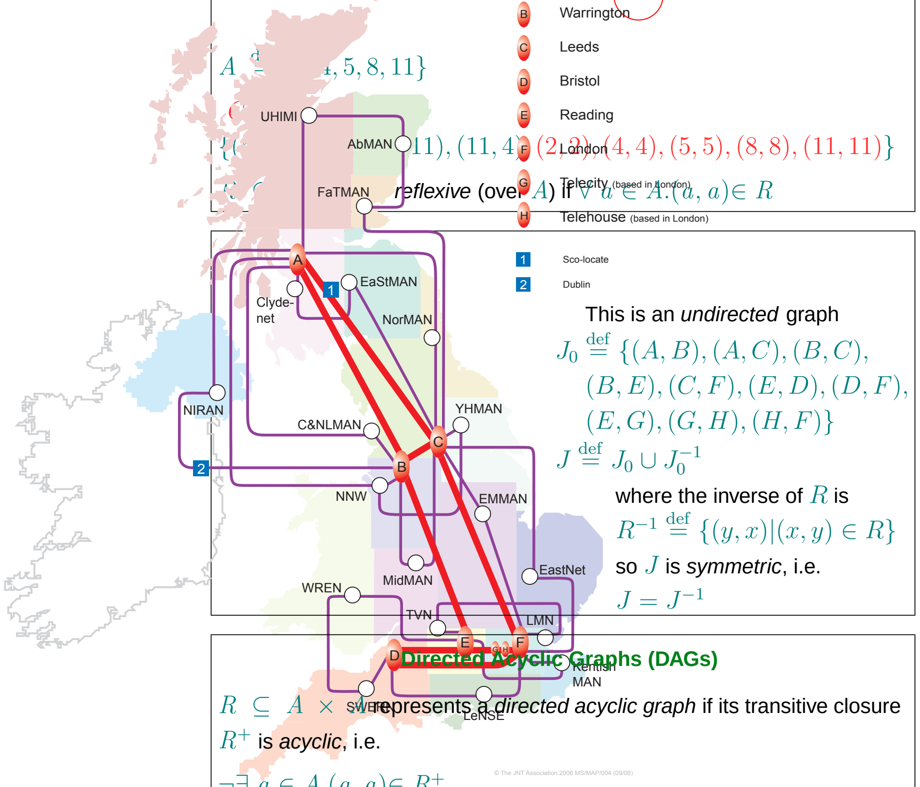
Reflexivity



$A \stackrel{\text{def}}{=}$ [illegible]$4,5,8,11\}$

$\{($[illegible]$11),(11,4),(2,2),(4,4),(5,5),(8,8),(11,11)\}$

[illegible] *reflexive* (over A) if $\forall\ a \in A.(a,a)\in R$



This is an *undirected* graph

$J_0 \stackrel{\text{def}}{=} \{(A,B),(A,C),(B,C),$
$(B,E),(C,F),(E,D),(D,F),$
$(E,G),(G,H),(H,F)\}$

$J \stackrel{\text{def}}{=} J_0 \cup J_0^{-1}$

where the inverse of R is

$R^{-1} \stackrel{\text{def}}{=} \{(y,x)|(x,y)\in R\}$

so J is *symmetric*, i.e.

$J = J^{-1}$

Directed Acyclic Graphs (DAGs)

$R \subseteq A \times A$ represents a *directed acyclic graph* if its transitive closure R^+ is *acyclic*, i.e.

$\neg\exists\ a \in A.(a,a)\in R^+$

Equivalence Relations

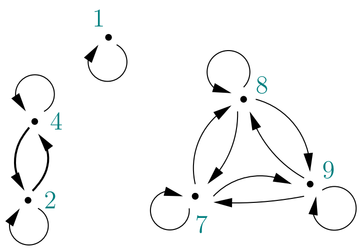
$R \subseteq A \times A$ is an *equivalence relation* (over A) if:

- R is reflexive, i.e. $\forall\ a \in A.(a,a)\in R$
- R is transitive,
  i.e. $\forall\ a_1,a_2,a_3 \in A.((a_1,a_2)\in R \wedge (a_2,a_3)\in R) \Rightarrow (a_1,a_3)\in R$
- R is symmetric, i.e. $\forall\ a_1,a_2 \in A.(a_1,a_2)\in R \Rightarrow (a_2,a_1)\in R$

e.g. $\{(m,n) \mid m \bmod 3 = n \bmod 3\}$ (over $\mathbb{N}$)

The *equivalence class* of $a \in A$ is all the things related to it, i.e.
$\{a' \mid (a,a')\in R\}$

Equivalence Relations



An equivalence relation over $\{1,2,4,7,8,9\}$

$\{(1,1),(2,2),(4,4),(2,4),(4,2),(7,7),(8,8),(9,9),(7,8),(8,7),(8,9),(9,8),(9,7),(7,9)\}$

with three equivalence classes: $\{1\}$, $\{2,4\}$, and $\{7,8,9\}$

Pre-Orders

Reflexive transitive relations are known as *pre-orders* .

Suppose $\leq \subseteq A \times A$ is a pre-order over A.
By the definition, $a \leq a$, and if $a_1 \leq a_2 \leq a_3$ then $a_1 \leq a_3$.
But we can have $a_1 \leq a_2 \leq a_1$ for $a_1 \neq a_2$.

(Note that we drew pairs (a_1, a_2) as $a_1 \longrightarrow a_2$, but write $(a_1, a_2) \in \leq$ or $a_1 \leq a_2$)

Partial Orders

A *partial order* $\leq$ over A is a reflexive transitive relation (so a pre-order) that is also *antisymmetric*,

$\forall\ a_1, a_2 \in A.(a_1 \leq a_2 \wedge a_2 \leq a_1) \Rightarrow (a_1 = a_2)$

For example, here's part of the $\subseteq$ relation over sets:

$$\begin{tikzcd} & \{1,2,3\} & \\ \{1,2\} \arrow[ur] & & \{1,3\} \arrow[ul] \\ & \{1\} \arrow[ul] \arrow[ur] & \end{tikzcd}$$

(when we draw a partial order, we usually omit the refl and tran edges — these are *Hasse diagrams*)

Total Orders

A *total order* (or *linear order*) $\leq$ over A is a reflexive, transitive, antisymmetric relation (so a partial order) that is also *total*,

$\forall\ a_1, a_2 \in A.(a_1 \leq a_2 \vee a_2 \leq a_1)$

(in fact the reflexivity condition is redundant)

For example, here's a Hasse diagram of part of the usual $\leq$ relation over $\mathbb{N}$:

$$\begin{tikzcd} 2 \\ 1 \arrow[u] \\ 0 \arrow[u] \end{tikzcd}$$

Special Relations — Summary

A relation $R \subseteq A \times A$ is a directed graph. Properties:

- transitive $\forall\ a_1, a_2, a_3 \in A.(a_1\ R\ a_2 \wedge a_2\ R\ a_3) \Rightarrow a_1\ R\ a_3$
- reflexive $\forall\ a \in A.(a\ R\ a)$
- symmetric $\forall\ a_1, a_2 \in A.(a_1\ R\ a_2 \Rightarrow a_2\ R\ a_1)$
- acyclic $\forall\ a \in A.\neg(a\ R^+ a)$
- antisymmetric $\forall\ a_1, a_2 \in A.(a_1\ R\ a_2 \wedge a_2\ R\ a_1) \Rightarrow a_1 = a_2$
- total $\forall\ a_1, a_2 \in A.(a_1\ R\ a_2 \vee a_2\ R\ a_1)$

Combinations of properties: R is a ...

- directed acyclic graph if the transitive closure is acyclic
- undirected graph if symmetric
- equivalence relation if reflexive, transitive, and symmetric
- pre-order if reflexive and transitive,
- partial order if reflexive, transitive, and antisymmetric
- total order if reflexive, transitive, antisymmetric, and total

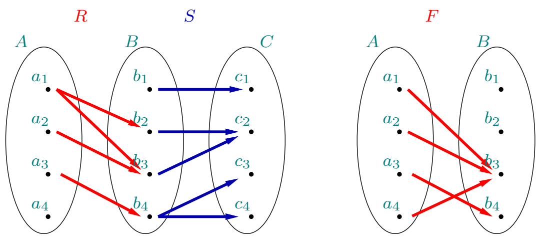
Functions

A *function* from A to B is just a relation which identifies exactly one element of B for each element of A.

$R \subseteq A \times \mathrm{B}$ is *functional* iff

$\forall\ a \in A.\exists\ b \in \mathrm{B}.(a, b) \in R$ and

$\forall\ a \in A.\forall\ b, b' \in \mathrm{B}.((a, b) \in R \wedge (a, b') \in R) \Rightarrow b = b'$



Application — Relaxed Memory: One Intel/AMD Example

Initial shared memory values: $x = 0 \quad y = 0$

Per-processor registers: $r_A \quad r_B$

Processor A	Processor B
`store` $x := 1$	`store` $y := 1$
`load` $r_A := y$	`load` $r_B := x$

Processor A	Processor B
MOV [x]⟵\$1	MOV [y]⟵\$1
MOV EAX⟵[y]	MOV EBX⟵[x]

Final register values: $r_A =?$ $\quad r_B =?$

Application — Relaxed Memory: One Intel/AMD Example

Initial shared memory values: $x = 0 \qquad y = 0$

Per-processor registers: $r_A \qquad r_B$

Processor A	Processor B
`store` $x := 1$	`store` $y := 1$
`load` $r_A := y$	`load` $r_B := x$

Processor A	Processor B
MOV [x]←$1	MOV [y]←$1
MOV EAX←[y]	MOV EBX←[x]

Final register values: $r_A = ? \qquad r_B = ?$

Each processor can do its own store action before the store of the other processor.

Makes it hard to understand what your programs are doing!

Already a real problem for OS, compiler, and library authors.

Application — Relaxed Memory: part of the formalisation

$$
\begin{array}{l}
\text{preserved\_program\_order } \mathrm{E} = \\
\quad \{(e_1, e_2) \mid (e_1, e_2) \in (\text{po\_strict } \mathrm{E}) \wedge \\
\qquad ((\exists p\ r.(\text{loc } e_1 = \text{loc } e_2) \wedge \\
\qquad\qquad (\text{loc } e_1 = \mathsf{Some}\ (\mathsf{Location\_reg}\ p\ r))) \vee \\
\qquad (\text{mem\_load } e_1 \wedge \text{mem\_load } e_2) \vee \\
\qquad (\text{mem\_store } e_1 \wedge \text{mem\_store } e_2) \vee \\
\qquad (\text{mem\_load } e_1 \wedge \text{mem\_store } e_2) \vee \\
\qquad (\text{mem\_store } e_1 \wedge \text{mem\_load } e_2 \wedge (\text{loc } e_1 = \text{loc } e_2)) \vee \\
\qquad ((\text{mem\_load } e_1 \vee \text{mem\_store } e_1) \wedge \text{locked } \mathrm{E}\ e_2) \vee \\
\qquad (\text{locked } \mathrm{E}\ e_1 \wedge (\text{mem\_load } e_2 \vee \text{mem\_store } e_2)))\}
\end{array}
$$

6 Induction

Induction

Example

Theorem $\sum_{i=1}^{n} i = n * (n+1)/2$

Proof By induction on n.

Base case (0): $\sum_{i=1}^{0} i = 0 = 0 * 1/2$

Inductive case ($n+1$): Assume $\sum_{i=1}^{n} i = n * (n+1)/2$ as the inductive hypothesis, then we have to prove
$\sum_{i=1}^{n+1} i = (n+1) * ((n+1)+1)/2$.
But $\sum_{i=1}^{n+1} i = \sum_{i=1}^{n} i + (n+1) = n * (n+1)/2 + (n+1) = (n+1) * (n+1+1)/2$ $\square$

What's really going on?

Using a fact about $\mathbb{N}$, the *induction principle*

$(P(0) \wedge (\forall\, n.P(n) \Rightarrow P(n+1))) \Rightarrow \forall\, n.P(n)$

(really a schema — that's true for any predicate P)

We think of an induction hypothesis, here taking

$P(n) \stackrel{\text{def}}{=} \sum_{i=1}^{n} i = n * (n+1)/2$

and instantiate the schema with it:

$$\begin{array}{l}(\quad (\sum_{i=1}^{0} i = 0*(0+1)/2) \wedge \\ \quad (\forall\, n.(\sum_{i=1}^{n} i = n*(n+1)/2) \\ \qquad \Rightarrow \\ \qquad (\sum_{i=1}^{n+1} i = (n+1)*((n+1)+1)/2))) \\ \Rightarrow \\ \forall\, n. \sum_{i=1}^{n} i = n*(n+1)/2 \end{array}$$

$$\begin{array}{l}(\quad (\sum_{i=1}^{0} i = 0*(0+1)/2) \wedge \\ \quad (\forall\, n.(\sum_{i=1}^{n} i = n*(n+1)/2) \\ \qquad \Rightarrow \\ \qquad (\sum_{i=1}^{n+1} i = (n+1)*((n+1)+1)/2))) \\ \Rightarrow \\ \forall\, n. \sum_{i=1}^{n} i = n*(n+1)/2 \end{array}$$

Then we prove the antecedents of the top-level implication (with our normal proof techniques), and use modus ponens to conclude the consequent.

Induction on lists

An ML function to append two lists:

```
fun app ([], ys)    = ys
  | app (x::xs, ys) = x :: app(xs,ys)
```

This is *terminating* and *pure* (no mutable state, no IO, no exceptions). So we can regard it as a mathematical function app.

It operates on *lists*. Suppose they are lists of elements of a set A.

Is app associative?

Induction on lists

Theorem

$\forall\, xs, ys, zs.\mathsf{app}(\mathsf{app}(xs, ys), zs) = \mathsf{app}(xs, \mathsf{app}(ys, zs))$

Proof We use the *induction schema for lists*

$(P([]) \wedge (\forall\, xs.P(xs) \Rightarrow \forall\, x.P(x :: xs))) \Rightarrow \forall\, xs.P(xs)$

with the induction hypothesis

$P(xs) \stackrel{\text{def}}{=} \ \forall\, ys, zs.\mathsf{app}(\mathsf{app}(xs, ys), zs) = \mathsf{app}(xs, \mathsf{app}(ys, zs))$

Base case: we have to prove $P([])$,
i.e. $\forall\, ys, zs.\mathsf{app}(\mathsf{app}([], ys), zs) = \mathsf{app}([], \mathsf{app}(ys, zs))$

a. Consider arbitrary ys and zs.
b. $\mathsf{app}(\mathsf{app}([], ys), zs) = \mathsf{app}(ys, zs)$ by the first clause of the defn of app
c. ... $= \mathsf{app}([], \mathsf{app}(ys, zs))$ by the first clause of the defn of app (backwards)

Inductive step: we have to prove $(\forall\ xs.P(xs) \Rightarrow \forall\ x.P(x :: xs)))$

1. Consider an arbitrary xs.

2. Assume $P(xs)$
3. $\forall\ ys, zs.\mathtt{app}(\mathtt{app}(xs, ys), zs) = \mathtt{app}(xs, \mathtt{app}(ys, zs))$ from 2, unfolding defn of P
4. Consider an arbitrary x
   (now we have to prove $P(x :: xs)$, i.e.
   $\forall\ ys, zs.\mathtt{app}(\mathtt{app}(x :: xs, ys), zs) = \mathtt{app}(x :: xs, \mathtt{app}(ys, zs))$)
5. Consider arbitrary ys and zs
6. $\mathtt{app}(\mathtt{app}(x :: xs, ys), zs) = \mathtt{app}(x :: \mathtt{app}(xs, ys), zs)$ by the second clause of $\mathtt{app}$
7. ... $= x :: \mathtt{app}(\mathtt{app}(xs, ys), zs)$ by the second clause of $\mathtt{app}$
8. ... $= x :: \mathtt{app}(xs, \mathtt{app}(ys, zs))$ instantiating 3 with $ys = ys$,$zs = zs$ under $x :: \ldots$
9. ... $= \mathtt{app}(x :: xs, \mathtt{app}(ys, zs))$ by the second clause of $\mathtt{app}$ (backwards)
10. $P(x :: xs)$ from 5–9, by $\forall$-introduction and folding the defn of P
11. $\forall\ x.P(x :: xs)$ from 4–10 by $\forall$-introduction

12. $P(xs) \Rightarrow \forall\ x.P(x :: xs)$ from 2–11 by $\Rightarrow$-introduction
13. $\forall\ xs.P(xs) \Rightarrow \forall\ x.P(x :: xs)$ from 1–12 by $\forall$-introduction

Now from the induction scheme, (c), and (13), we have $\forall xs.P(xs)$, which (unfolding the defn of P) is exactly the theorem statement.

Simpler proof structure: first rearrange the quantifiers

$$\forall\ xs, ys, zs.\mathtt{app}(\mathtt{app}(xs, ys), zs) = \mathtt{app}(xs, \mathtt{app}(ys, zs))$$
iff
$$\forall\ ys, zs.\forall\ xs.\mathtt{app}(\mathtt{app}(xs, ys), zs) = \mathtt{app}(xs, \mathtt{app}(ys, zs))$$

Then consider arbitrary ys and zs, and inside that do induction on lists, with induction hypothesis

$$P(xs) \stackrel{\text{def}}{=} \mathtt{app}(\mathtt{app}(xs, ys), zs) = \mathtt{app}(xs, \mathtt{app}(ys, zs))$$

(instead of $P(xs) \stackrel{\text{def}}{=} \forall\ ys, zs.\mathtt{app}(\mathtt{app}(xs, ys), zs) = \mathtt{app}(xs, \mathtt{app}(ys, zs))$)

OK, as we don't need to instantiate P at different ys and zs

Generalizing an Induction Hypothesis

ML functions for the length of a list:

```
fun nlength []      = 0
  | nlength (x::xs) = 1 + nlength xs
fun addlen (k,[])    = k
  | addlen (k,x::xs) = addlen(k+1,xs)
```

(compiler optimization?) Both are terminating and pure.

Theorem ? $\mathtt{addlen}(0, \ell) = \mathtt{nlength}(\ell)$

Induction on ℓ — but which induction hypothesis?

$P''(\ell) \stackrel{\text{def}}{=} \mathtt{addlen}(0, \ell) = \mathtt{nlength}(\ell)$ too weak

$P'(\ell) \stackrel{\text{def}}{=} \mathtt{addlen}(k, \ell) = k + \mathtt{nlength}(\ell)$ too rigid: need to vary k

$P(\ell) \stackrel{\text{def}}{=} \forall\ k.\mathtt{addlen}(k, \ell) = k + \mathtt{nlength}(\ell)$ just right

Base case: we need to show $P([])$, i.e. $\forall\ k.\texttt{addlen}(k, []) = k + \texttt{nlength}([])$

1. Consider an arbitrary k.
2. $\texttt{addlen}(k, []) = k = k + 0 = k + \texttt{nlength}(0)$ by the defn $\texttt{addlen}$ and $\texttt{nlength}$

Inductive step: we need to show $(\forall\ \ell.P(\ell) \Rightarrow \forall\ x.P(x :: \ell)))$

3. Consider an arbitrary ℓ
4. Assume the induction hypothesis $P(\ell)$, i.e. $\forall\ k.\texttt{addlen}(k, \ell) = k + \texttt{nlength}(\ell)$
5. Consider an arbitrary x
   (now we have to show $P(x :: \ell)$, i.e. $\forall\ k.\texttt{addlen}(k, x :: \ell) = k + \texttt{nlength}(x :: \ell)$)
6. Consider an arbitrary k
7. $\texttt{addlen}(k, x :: \ell) = \texttt{addlen}(k + 1, \ell)$ by defn $\texttt{addlen}$
8. ... $= (k + 1) + \texttt{nlength}(\ell)$ by 4, instantiating k with $k + 1$
9. ... $= k + \texttt{nlength}(x :: \ell)$ by defn $\texttt{nlength}$
11. $\forall\ k.\texttt{addlen}(k, x :: \ell) = k + \texttt{nlength}(x :: \ell)$ from 6–9 by $\forall$-introduction
12. $P(x :: \ell)$ from 11 by folding defn P
13. $\forall\ x.P(x :: \ell)$ from 5–12 by $\forall$-introduction
14. $P(\ell) \Rightarrow \forall\ x.P(x :: \ell)$ from 4–13 by $\Rightarrow$-introduction
15. $\forall\ \ell.P(\ell) \Rightarrow \forall\ x.P(x :: \ell)$ from 3–14 by $\forall$-introduction

The theorem follows by instantiating P with $k = 0$ $\square$

...rewriting that semi-structured proof more idiomatically:

Theorem $\texttt{addlen}(0, \ell) = \texttt{nlength}(\ell)$

Proof Induction on ℓ, with I.H. $P(\ell) \overset{\text{def}}{=} \forall\ k.\texttt{addlen}(k, \ell) = k + \texttt{nlength}(\ell)$
in induction schema $(P([]) \wedge (\forall\ xs.P(xs) \Rightarrow \forall\ x.P(x :: xs))) \Rightarrow \forall\ xs.P(xs)$

Base case: we need to show $P([])$
Consider an arbitrary k, then $\texttt{addlen}(k, []) = k = k + 0 = k + \texttt{nlength}(0)$ by defn $\texttt{addlen}$ and $\texttt{nlength}$

Inductive step: consider an arbitrary ℓ, assume $P(\ell)$, and consider an arbitrary x. We have to show $P(x :: \ell)$.

Consider an arbitrary k.
$\texttt{addlen}(k, x :: \ell) = \texttt{addlen}(k + 1, \ell)$ by defn $\texttt{addlen}$
... $= (k + 1) + \texttt{nlength}(\ell)$ by $P(\ell)$, instantiating k with $k + 1$
... $= k + \texttt{nlength}(x :: \ell)$ by defn $\texttt{nlength}$

7 Conclusion

Conclusion

We've introduced a good part of the language of discrete mathematics

(vocabulary, grammar, pragmatics...)

Fluency comes with use; you'll see that this is a remarkably flexible tool for formulating and analysing computational problems.

The End

A Recap: How To Do Proofs

The purpose of this appendix is summarise some of the guidelines about how to prove theorems. This should give you some help in answering questions that begin with "Show that the following is true ...". It draws on earlier notes by Myra VanInwegen; with many thanks for making the originals available.

The focus here is on doing *informal but rigorous proofs*. These are rather different from the *formal proofs*, in Natural Deduction or Sequent Calculus, that will be introduced in the 1B Logic and Proof course. Formal proofs are derivations in one of those proof systems – they are in a completely well-defined form, but are often far too verbose to deal with by hand (although they can be machine-checked). Informal proofs, on the other hand, are the usual mathematical notion of proof: written arguments to persuade the reader that you could, if pushed, write a fully formal proof.

This is important for two reasons. Most obviously, you should learn how to do these proofs. More subtly, but more importantly, only by working with the mathematical definitions in some way can you develop a good intuition for what they mean — trying to do some proofs is the best way of understanding the definitions.

A.1 How to go about it

Proofs differ, but for many of those you meet the following steps should be helpful.

1. Make sure the statement of the conjecture is precisely defined. In particular, make sure you understand any strange notation, and find the definitions of all the auxiliary gadgets involved (e.g. definitions of any typing or reduction relations mentioned in the statement, or any other predicates or functions).
2. Try to understand at an intuitive level what the conjecture is saying – verbalize out loud the basic point. For example, for a Type Preservation conjecture, the basic point might be something like "if a well-typed configuration reduces, the result is still well-typed (with the same type)".
3. Try to understand intuitively why it is true (or false...). Identify what the most interesting cases might be — the cases that you think are most likely to be suspicious, or hard to prove. Sometimes it's good to start with the easy cases (if the setting is unfamiliar to you); sometimes it's good to start with the hard cases (to find any interesting problems as soon as possible).
4. Think of a good basic strategy. This might be:
   (a) simple logic manipulations;
   (b) collecting together earlier results, again by simple logic; or
   (c) some kind of induction.
5. Try it! (remembering you might have to backtrack if you discover you picked a strategy that doesn't work well for this conjecture). This might involve any of the following:
   (a) Expanding definitions, inlining them. Sometimes you can just blindly expand all definitions, but more often it's important to expand only the definitions which you want to work with the internal structure of — otherwise things just get too verbose.
   (b) Making abbreviations — defining a new variable to stand for some complex gadget you're working with, saying e.g.

       ```
       where e = (let x:int = 7+2 in x+x)
       ```

       Take care with choosing variable names.

(c) Doing *equational reasoning*, e.g.

```
e = e1   by ...
  = e2   by ...
  = e3   as ...
```

Here the `e` might be any mathematical object — arithmetic expressions, or expressions of some grammar, or formulae. Some handy equations over formulae are given in §A.2.2.

(d) Proving a formula based on its structure. For example, to prove a formula $\forall x \in S.P(x)$ you would often assume you have an arbitrary x and then try to prove $P(x)$.

```
Take an arbitrary $x \in S$.
We now have to show $P(x)$:
```

This is covered in detail in §A.2.3. Much proof is of this form, automatically driven by the structure of the formula.

(e) Using an assumption you've made above.

(f) Induction. As covered in the 1B Semantics notes, there are various kinds of induction you might want to use: mathematical induction over the natural numbers, structural induction over the elements of some grammar, or rule induction over the rules defining some relation (especially a reduction or typing relation). For each, you should:

i. Decide (and state!) what kind of induction you're using. This may need some thought and experience, and you might have to backtrack.

ii. Remind yourself what the induction principle is exactly.

iii. Decide on the induction hypothesis you're going to use, writing down a predicate Φ which is such that the conclusion of the induction principle implies the thing you're trying to prove. Again, this might need some thought. Take care with the quantifiers here — it's suspicious if your definition of Φ has any globally-free variables...

iv. Go through each of the premises of the induction principle and prove each one (using any of these techniques as appropriate). Many of those premises will be implications, e.g. $\forall x \in \mathbb{N}.\Phi(x) \Rightarrow \Phi(x+1)$, for which you can do a proof based on the structure of the formula — taking an arbitrary x, assuming $\Phi(x)$, and trying to prove $\Phi(x+1)$. Usually at some point in the latter you'd make use of the assumption $\Phi(x)$.

6. In all of the above, remember: the point of doing a proof on paper is to *use* the formalism to *help you think* — to help you cover all cases, precisely — and also to *communicate with the reader*. For both, you need to write clearly:

(a) Use enough words! "Assume", "We have to show", "By such-and-such we know", "Hence",...

(b) Don't use random squiggles. It's good to have formulae properly nested within text, with and no "$\Rightarrow$" or "$\because$" between lines of text.

7. If it hasn't worked yet... either

(a) you've make some local mistake, e.g. mis-instantiated something, or used the same variable for two different things, or not noticed that you have a definition you should have expanded or an assumption you should have used. Fix it and continue.

    (b) you've discovered that the conjecture is really false. Usually at this point it's a good idea to construct a counterexample that is as simple as possible, and to check carefully that it really is a counterexample.
    (c) you need to try a different strategy — often, to use a different induction principle or to strengthen your induction hypothesis.
    (d) you didn't really understand intuitively what the conjecture is saying, or what the definitions it uses mean. Go back to them again.

8. If it has worked: read through it, skeptically, and check. Maybe you'll need to *re-write* it to make it comprehensible: proof *discovery* is not the same as proof *exposition.* See the example proofs in the Semantics notes.
9. Finally, give it to someone else, as skeptical and careful as you can find, to see if they believe it — to see if they believe that *what you've written down is a proof*, not that they believe that *the conjecture is true.*

A.2 And in More Detail...

In the following, I'll call any logic statement a formula. In general, what we'll be trying to do is *prove* a formula, using a collection of formulas that we know to be true or are assuming to be true. There's a big difference between *using* a formula and *proving* a formula. In fact, what you do is in many ways opposite. So, I'll start by explaining how to *prove* a formula.

A.2.1 Meet the Connectives

Here are the logical connectives and a very brief decription of what each means.

$P \wedge Q$	P and Q are both true
$P \vee Q$	P is true, or Q is true, or both are true
$\neg P$	P is not true (P is false)
$P \Rightarrow Q$	if P is true then Q is true
$P \Leftrightarrow Q$	P is true exactly when Q is true
$\forall x \in S.P(x)$	for all x in S, P is true of x
$\exists x \in S.P(x)$	there exists an x in S such that P holds of x

A.2.2 Equivalences

These are formulas that mean the same thing, and this is indicated by a iff between them. The fact that they are equivalent to each other is justified by the truth tables of the connectives.

definition of $\Rightarrow$	$P \Rightarrow Q$	iff	$\neg P \vee Q$
definition of $\Leftrightarrow$	$P \Leftrightarrow Q$	iff	$(P \Rightarrow Q) \wedge (Q \Rightarrow P)$
definition of $\neg$	$\neg P$	iff	$P \Rightarrow \text{false}$
de Morgan's Laws	$\neg(P \wedge Q)$	iff	$\neg P \vee \neg Q$
	$\neg(P \vee Q)$	iff	$\neg P \wedge \neg Q$
extension to quantifiers	$\neg(\forall x.P(x))$	iff	$\exists x.\neg P(x)$
	$\neg(\exists x.P(x))$	iff	$\forall x.\neg P(x)$
distributive laws	$P \vee (Q \wedge R)$	iff	$(P \vee Q) \wedge (P \vee R)$
	$P \wedge (Q \vee R)$	iff	$(P \wedge Q) \vee (P \wedge R)$
coalescing quantifiers	$(\forall x.P(x)) \wedge (\forall x.Q(x))$	iff	$\forall x.(P(x) \wedge Q(x))$
	$(\exists x.P(x)) \vee (\exists x.Q(x))$	iff	$\exists x.(P(x) \vee Q(x))$
these ones apply if	$(\forall x.P(x)) \wedge Q$	iff	$(\forall x.P(x) \wedge Q)$
x is not free in Q	$(\forall x.P(x)) \vee Q$	iff	$(\forall x.P(x) \vee Q)$
	$(\exists x.P(x)) \wedge Q$	iff	$(\exists x.P(x) \wedge Q)$
	$(\exists x.P(x)) \vee Q$	iff	$(\exists x.P(x) \vee Q)$

A.2.3 How to Prove a Formula

For each of the logical connectives, I'll explain how to handle them.

$\boxed{\forall x \in S.P(x)}$ This means "For all x in S, P is true of x." Such a formula is called a universally quantified formula. The goal is to prove that the property P, which has some xs somewhere in it, is true no matter what value in S x takes on. Often the "$\in S$" is left out. For example, in a discussion of lists, you might be asked to prove $\forall l.\texttt{length}\, l > 0 \Rightarrow \exists x.\,\texttt{member}(x, l)$. Obviously, l is a list, even if it isn't explicitly stated as such.

There are several choices as to how to prove a formula beginning with $\forall x$. The standard thing to do is to just prove $P(x)$, not assuming anything about x. Thus, in doing the proof you sort of just mentally strip off the $\forall x$. What you would write when doing this is "Let x be any S". However, there are some subtleties—if you're already using an x for something else, you can't use the same x, because then you *would* be assuming something about x, namely that it equals the x you're already using. In this case, you need to use alpha-conversion[1] to change the formula you want to prove to $\forall y \in S.P(y)$, where y is some variable you're not already using, and then prove $P(y)$. What you could write in this case is "Since x is already in use, we'll prove the property of y".

An alternative is induction, if S is a set that is defined with a structural definition. Many objects you're likely to be proving properties of are defined with a structural definition. This includes natural numbers, lists, trees, and terms of a computer language. Sometimes you can use induction over the natural numbers to prove things about other objects, such as graphs, by inducting over the number of nodes (or edges) in a graph.

You use induction when you see that during the course of the proof you would need to use the property P for the subparts of x in order to prove it for x. This usually ends up being the case if P involves functions defined recursively (i.e., the return value for the function depends on the function value on the subparts of the argument).

A special case of induction is case analysis. It's basically induction where you don't use the inductive hypothesis: you just prove the property for each possible form that x could have. Case analysis can be used to prove the theorem about lists above.

A final possibility (which you can use for all formulas, not just for universally quantified ones) is to assume the contrary, and then derive a contradiction.

$\boxed{\exists x \in S.P(x)}$ This says "There exists an x in S such that P holds of x." Such a formula is called an existentially quantified formula. The main way to prove this is to figure out what x has to be (that is, to find a concrete representation of it), and then prove that P holds of that value. Sometimes you can't give a completely specified value, since the value you pick for x has to depend on the values of other things you have floating around. For example, say you want to prove

$$\forall x, y \in \Re. x < y \wedge \sin x < 0 \wedge \sin y > 0 \Rightarrow \exists z. x < z \wedge z < y \wedge \sin z = 0$$

where $\Re$ is the set of real numbers. By the time you get to dealing with the $\exists z. x < z \wedge z < y \wedge \sin z = 0$, you will have already assumed that x and y were any real numbers. Thus the value you choose for z has to depend on whatever x and y are.

An alternative way to prove $\exists x \in S.P(x)$ is, of course, to assume that no such x exists, and derive a contradiction.

To summarize what I've gone over so far: to *prove* a universally quantified formula, you must prove it for a generic variable, one that you haven't used before. To prove an existentially quantified formula, you get to choose a value that you want to prove the property of.

$\boxed{P \Rightarrow Q}$ This says "If P is true, then Q is true". Such a formula is called an implication, and it is often pronounced "P implies Q". The part before the $\Rightarrow$ sign (here P) is called the antecedent, and the part after the $\Rightarrow$ sign (here Q) is called the consequent. $P \Rightarrow Q$ is equivalent to $\neg P \vee Q$, and so if P is false, or if Q is true, then $P \Rightarrow Q$ is true.

The standard way to prove this is to assume P, then use it to help you prove Q. Note that I said that you will be *using* P. Thus you will need to follow the rules in Section A.2.4 to deal with the logical connectives in P.

Other ways to prove $P \Rightarrow Q$ involve the fact that it is equivalent to $\neg P \vee Q$. Thus, you can prove $\neg P$ without bothering with Q, or you can just prove Q without bothering with P. To reason by contradiction you assume that P is true and that Q is not true, and derive a contradiction.

[1]Alpha-equivalence says that the name of a bound variable doesn't matter, so you can change it at will (this is called alpha-conversion). You'll get to know the exact meaning of this soon enough so I won't explain this here.

Another alternative is to prove the contrapositive: $\neg Q \Rightarrow \neg P$, which is equivalent to it.

$\boxed{P \Leftrightarrow Q}$ This says "P is true if and only if Q is true". The phrase "if and only if" is usually abbreviated "iff". Basically, this means that P and Q are either both true, or both false.

Iff is usually used in two main ways: one is where the equivalence is due to one formula being a definition of another. For example, $A \subseteq B \Leftrightarrow (\forall x.\, x \in A \Rightarrow x \in B)$ is the standard definition of subset. For these iff statements, you don't have to prove them. The other use of iff is to state the equivalence of two different things. For example, you could define an SML function `fact`:

```
fun fact 0 = 1
  | fact n = n * fact (n - 1)
```

Since in SML whole numbers are integers (both positive and negative) you may be asked to prove: `fact` x terminates $\Leftrightarrow x \geq 0$. The standard way to do this is us the equivalence $P \Leftrightarrow Q$ is equivalent to $P \Rightarrow Q \wedge Q \Rightarrow P$. And so you'd prove that (`fact` x terminates $\Rightarrow x \geq 0) \wedge (x \geq 0 \Rightarrow$ `fact` x terminates).

$\boxed{\neg P}$ This says "P is not true". It is equivalent to $P \Rightarrow$ false, thus this is one of the ways you prove it: you assume that P is true, and derive a contradiction (that is, you prove false). Here's an example of this, which you'll run into later this year: the undecidability of the halting problem can be rephrased as $\neg\exists x \in RM.\, x$ *solves the halting problem*, where RM is the set of register machines. The proof of this in your Computation Theory notes follows exactly the pattern I described—it assumes there is such a machine and derives a contradiction.

The other major way to prove $\neg P$ is to figure out what the negation of P is, using equivalences like De Morgan's Law, and then prove that. For example, to prove $\neg\forall x \in \mathcal{N}.\, \exists y \in \mathcal{N}.\, x = y^2$, where $\mathcal{N}$ is the set of natural numbers, you could push in the negation to get: $\exists x \in \mathcal{N}.\, \forall y \in \mathcal{N}.\, x \neq y^2$, and then you could prove that.

$\boxed{P \wedge Q}$ This says "P is true and Q is true". Such a formula is called a conjunction. To prove this, you have to prove P, and you have to prove Q.

$\boxed{P \vee Q}$ This says "P is true or Q is true". This is *inclusive* or: if P and Q are both true, then $P \vee Q$ is still true. Such a formula is called a disjunction. To prove this, you can prove P or you can prove Q. You have to choose which one to prove. For example, if you need to prove $(5 \bmod 2 = 0) \vee (5 \bmod 2 = 1)$, then you'll choose the second one and prove that.

However, as with existentials, the choice of which one to prove will often depend on the values of other things, like universally quantified variables. For example, when you are studying the theory of programming languages (you will get a bit of this in Semantics), you might be asked to prove

$$\begin{aligned} \forall P \in ML.\quad & \textit{P is properly typed} \Rightarrow \\ & (\textit{the evaluation of P runs forever}) \vee (\textit{P evaluates to a value}) \end{aligned}$$

where ML is the set of all ML programs. You don't know in advance which of these will be the case, since some programs do run forever, and some do evaluate to a value. Generally, the best way to prove the disjunction in this case (when you don't know in advance which will hold) is to use the equivalence with implication. For example, you can use the fact that $P \vee Q$ is equivalent to $\neg P \Rightarrow Q$, then assume $\neg P$, then use this to prove Q. For example, your best bet to proving this programming languages theorem is to assume that the evaluation of P doesn't run forever, and use this to prove that P evaluates to a value.

A.2.4 How to Use a Formula

You often end up using a formula to prove other formulas. You can use a formula if someone has already proved that it's true, or you are assuming it because it was in an implication, namely, the A in $A \Rightarrow B$. For each logical connective, I'll tell you how to use it.

$\boxed{\forall x \in S.P(x)}$ This formula says that something is true of *all* elements of S. Thus, when you use it, you can pick any value at all to use instead of x (call it v), and then you can use $P(v)$.

$\boxed{\exists x \in S.P(x)}$ This formula says that there is some x that satisfies P. However, you do not know what it is, so you can not assume anything about it. The usual approach it to just say that the thing that is being said to exist is just x, and use the fact that P holds of x to prove something else. However, if you're already using an x for something else, you have to pick another variable to represent the thing that exists.

To summarize this: to *use* a universally quantified formula, you can choose any value, and use that the formula holds for that variable. To use an existentially quantified formula, you must not assume anything about the value that is said to exists, so you just use a variable (one that you haven't used before) to represent it. Note that this is more or less opposite of what you do when you prove a universally or existentially quantified formula.

$\boxed{\neg P}$ Usually, the main use of this formula is to prove the negation of something else. An example is the use of reduction to prove the unsolvability of various problems in the Computation Theory (you'll learn all about this in 1B). You want to prove $\neg Q$, where Q states that a certain problem (Problem 1) is decidable (in other words, you want to prove that Problem 1 is not decidable). You know $\neg P$, where P states that another problem (Problem 2) is decidable (i.e. $\neg P$ says that Problem 2 is not decidable). What you do basically is this. You first prove $Q \Rightarrow P$, which says that if Problem 1 is decidable, then so is Problem 2. Since $Q \Rightarrow P$ iff $\neg P \Rightarrow \neg Q$, you have now proved $\neg P \Rightarrow \neg Q$. You already know $\neg P$, so you use modus ponens[2] to get that $\neg Q$.

$\boxed{P \Rightarrow Q}$ The main way to use this is that you prove P, and then you use modus ponens to get Q, which you can then use.

$\boxed{P \Leftrightarrow Q}$ The main use of this is to replace an occurrence of P in a formula with Q, and vise versa.

$\boxed{P \wedge Q}$ Here you can use both P and Q. Note, you're not *required* to use both of them, but they are both true and are waiting to be used by you if you need them.

$\boxed{P \vee Q}$ Here, you know that one of P or Q is true, but you do not know which one. To use this to prove something else, you have to do a split: first you prove the thing using P, then you prove it using Q.

Note that in each of the above, there is again a difference in the way you use a formula, verses the way you prove it. They are in a way almost opposites. For example, in proving $P \wedge Q$, you have to prove both P and Q, but when you are using the formula, you don't have to use both of them.

A.3 An Example

Here, we'll look forward to Regular Languages and Finite Automata (this might not make much sense until you get to that). The Pumping Lemma for regular sets (PL for short) is

[2]Modus ponens says that if $A \Rightarrow B$ and A are both true, then B is true.

an astonishingly good example of the use of quantifiers. We'll go over the proof and use of the PL, paying special attention to the logic of what's happening.

A.3.1 Proving the PL

My favorite book on regular languages, finite automata, and their friends is the Hopcroft and Ullman book *Introduction to Automata Theory, Languages, and Computation.* You should locate this book in your college library, and if it isn't there, insist that your DoS order it for you.

In the *Automata Theory* book, the Pumping Lemma is stated as: "Let L be a regular set. Then there is a constant n such that if z is any word in L, and $|z| \geq n$, we may write $z = uvw$ in such a way that $|uv| \leq n$, $|v| \geq 1$, and for all $i \geq 0$, uv^iw is in L." The Pumping Lemma is, in my experience, one of the most difficult things about learning automata theory. It is difficult because people don't know what to do with all those logical connectives. Let's write it as a logical formula.

$$\begin{array}{l}\forall L \in \mathit{RegularLanguages}.\\ \ \exists n.\ \forall z \in L.\ |z| \geq n \Rightarrow\\ \ \ \exists u\,v\,w.\ z = uvw \land |uv| \leq n \land |v| \geq 1 \land\\ \ \ \ \forall i \geq 0.\ uv^iw \in L\end{array}$$

Complicated, eh? Well, let's prove it, using the facts that Hopcroft and Ullman have established in the chapters previous to the one wih the PL. I'll give the proof and put in square brackets comments about what I'm doing.

Let L be any regular language. [Here I'm dealing with the $\forall L \in \mathit{RegularLanguages}$ by stating that I'm not assuming anything about L.] Let M be a minimal-state deterministic finite state machine accepting L. [Here I'm *using* a fact that Hopcroft and Ullman have already proved about the equivalence of regular languages and finite automata.] Let n be the number of states in this finite state machine. [I'm dealing with the $\exists n$ by giving a very specific value of what it will be, based on the arbitrary L.] Let z be any word in L. [Thus I deal with $\forall z \in L$.] Assume that $|z| \geq n$. [Thus I'm taking care of the $\Rightarrow$ by assuming the antecedent.]

Say z is written $a_1a_2 \ldots a_m$, where $m \geq n$. Consider the states that M is in during the processing of the first n symbols of z, $a_1a_2 \ldots a_n$. There are $n + 1$ of these states. Since there are only n states in M, there must be a duplicate. Say that after symbols a_j and a_k we are in the same state, state s (i.e. there's a loop from this state that the machine goes through as it accepts z), and say that $j < k$. Now, let $u = a_1a_2 \ldots a_j$. This represents the part of the string that gets you to state s the first time. Let $v = a_{j+1} \ldots a_k$. This represents the loop that takes you from s and back to it again. Let $w = a_{k+1} \ldots a_m$, the rest of word z. [We have chosen definite values for u, v, and w.] Then clearly $z = uvw$, since u, v, and w are just different sections of z. $|uv| \leq n$ since u and v occur within the first n symbols of z. $|v| \geq 1$ since $j < k$. [Note that we're dealing with the formulas connected with $\land$ by proving each of them.]

Now, let i be a natural number (i.e. ≥ 0). [This deals with $\forall i \geq 0$.] Then $uv^iw \in L$. [Finally our conclusion, but we have to explain why this is true.] This is because we can repeat the loop from s to s (represented by v) as many times as we like, and the resulting word will still be accepted by M.

A.3.2 Using the PL

Now we use the PL to prove that a language is not regular. This is a rewording of Example 3.1 from Hopcroft and Ullman. I'll show that $L = \{0^{i^2} | i \text{ is an integer}, i \geq 1\}$ is not regular. Note that L consists of all strings of 0's whose length is a perfect square. I will *use* the PL. I want to prove that L is not regular. I'll assume the negation (i.e., that L is regular) and derive a contradiction. So here we go. Remember that what I'm emphasizing here is not the finite automata stuff itself, but how to use a complicated theorem to prove something else.

Assume L is regular. We will use the PL to get a contradiction. Since L is regular, the PL applies to it. [We note that we're using the $\forall$ part of the PL for this particular L.] Let n be as described in the PL. [This takes care of using the $\exists n$. Note that we *are not* assuming anything about its actual value, just that it's a natural number.] Let $z = 0^{n^2}$. [Since the PL says that something is true of *all* zs, we can choose the one we want to use it for.] So by the PL there exist u, v, and w such that $z = uvw$, $|uv| \leq n$, $|v| \geq 1$. [Note that we don't assume anything about what the u, v, and w actually are; the only thing we know about them is what the PL tells us about them. This is where people trying to use the PL usually screw up.] The PL then says that for any i, then $uv^iw \in L$. Well, then $uv^2w \in L$. [This is using the $\forall i \geq 0$ bit.] However, $n^2 < |uv^2w| \leq n^2 + n$, since $1 \leq |v| \leq n$. But $n^2 + n < (n+1)^2$. Thus $|uv^2w|$ lies properly between n^2 and $(n+1)^2$ and is thus not a perfect square. Thus uv^2w is not in L. This is a contradiction. Thus our assumption (that L was regular) was incorrect. Thus L is not a regular language.

B Exercises

Exercise Sheet 1: Propositional Logic

1. Let p stand for the proposition "I bought a lottery ticket" and q for "I won the jackpot". Express the following as natural English sentences:
   (a) $\neg p$
   (b) $p \vee q$
   (c) $p \wedge q$
   (d) $p \Rightarrow q$
   (e) $\neg p \Rightarrow \neg q$
   (f) $\neg p \vee (p \wedge q)$
2. Formalise the following in terms of atomic propositions R, B, and W, first making clear how they correspond to the English text.
   (a) Berries are ripe along the path, but rabbits have not been seen in the area.
   (b) Rabbits have not been seen in the area, and walking on the path is safe, but berries are ripe along the path.
   (c) If berries are ripe along the path, then walking is safe if and only if rabbits have not been seen in the area.
   (d) It is not safe to walk along the path, but rabbits have not been seen in the area and the berries along the path are ripe.
   (e) For walking on the path to be safe, it is necessary but not sufficient that berries not be ripe along the path and for rabbits not to have been seen in the area.
   (f) Walking is not safe on the path whenever rabbits have been seen in the area and berries are ripe along the path.
3. Formalise these statements and determine (with truth tables or otherwise) whether they are consistent (i.e. if there are some assumptions on the atomic propositions that make it true): "The system is in a multiuser state if and only if it is operating normally. If the system is operating normally, the kernel is functioning. Either the kernel is not functioning or the system is in interrupt mode. If the system is not in multiuser state, then it is in interrupt mode. The system is not in interrupt mode."
4. When is a propositional formula P *valid*? When is P *satisfiable*?
5. For each of the following propositions, construct a truth table and state whether the proposition is valid or satisfiable. (For brevity, you can just write one truth table with many columns.)
   (a) $p \wedge \neg p$
   (b) $p \vee \neg p$
   (c) $(p \vee \neg q) \Rightarrow q$
   (d) $(p \vee q) \Rightarrow (p \wedge q)$
   (e) $(p \Rightarrow q) \Leftrightarrow (\neg q \Rightarrow \neg p)$
   (f) $(p \Rightarrow q) \Rightarrow (q \Rightarrow p)$
6. For each of the following propositions, construct a truth table and state whether the proposition is valid or satisfiable.

(a) $p \Rightarrow (\neg q \vee r)$

(b) $\neg p \Rightarrow (q \Rightarrow r)$

(c) $(p \Rightarrow q) \vee (\neg p \Rightarrow r)$

(d) $(p \Rightarrow q) \wedge (\neg p \Rightarrow r)$

(e) $(p \Leftrightarrow q) \vee (\neg q \Leftrightarrow r)$

(f) $(\neg p \Leftrightarrow \neg q) \Leftrightarrow (q \Leftrightarrow r)$

7. Formalise the following and, by writing truth tables for the premises and conclusion, determine whether the arguments are valid.

   (a) Either John isn't stupid and he is lazy, or he's stupid.
   John is stupid.
   Therefore, John isn't lazy.

   (b) The butler and the cook are not both innocent
   Either the butler is lying or the cook is innocent
   Therefore, the butler is either lying or guilty

8. Use truth tables to determine which of the following are equivalent to each other:

   (a) P

   (b) $\neg P$

   (c) $P \Rightarrow \mathrm{F}$

   (d) $P \Rightarrow \mathrm{T}$

   (e) $\mathrm{F} \Rightarrow P$

   (f) $\mathrm{T} \Rightarrow P$

   (g) $\neg\neg P$

9. Use truth tables to determine which of the following are equivalent to each other:

   (a) $(P \wedge Q) \vee (\neg P \wedge \neg Q)$

   (b) $\neg P \vee Q$

   (c) $(P \vee \neg Q) \wedge (Q \vee \neg P)$

   (d) $\neg(P \vee Q)$

   (e) $(Q \wedge P) \vee \neg P$

10. Imagine that a logician puts four cards on the table in front of you. Each card has a number on one side and a letter on the other. On the uppermost faces, you can see E, K, 4, and 7. He claims that if a card has a vowel on one side, then it has an even number on the other. How many cards do you have to turn over to check this?

11. Give a truth-table definition of the ternary boolean operation **if** P **then** Q **else** R.

12. Given the truth table for an arbitrary n-ary function $f(p_1, .., p_n)$ (from n propositional variables $p_1, .., p_n$ to $\{\mathrm{T}, \mathrm{F}\}$), describe how one can build a proposition, using only $p_1, .., p_n$ and the connectives $\wedge$, $\vee$, and $\neg$, that has the same truth table as f. (Hint: first consider each *line* of the truth table separately, and then how to combine them.)

13. Show, by equational reasoning from the axioms in the notes, that $\neg(P \wedge (Q \vee R \vee S))$ iff $\neg P \vee (\neg Q \wedge \neg R \wedge \neg S)$

Exercise Sheet 2: Predicate Logic

1. Formalise the following statements in predicate logic, making clear what your atomic predicate symbols stand for and what the domains of any variables are.
   (a) Anyone who has forgiven at least one person is a saint.
   (b) Nobody in the calculus class is smarter than everybody in the discrete maths class.
   (c) Anyone who has bought a Rolls Royce with cash must have a rich uncle.
   (d) If anyone in the college has the measles, then everyone who has a friend in the college will have to be quarantined.
   (e) Everyone likes Mary, except Mary herself.
   (f) Jane saw a bear, and Roger saw one too.
   (g) Jane saw a bear, and Roger saw it too.
   (h) If anyone can do it, Jones can.
   (i) If Jones can do it, anyone can.
2. Translate the following into idiomatic English.
   (a) $\forall\, x.(H(x) \wedge \forall\, y.\neg M(x,y)) \Rightarrow U(x)$ where $H(x)$ means x is a man, $M(x,y)$ means x is married to y, $U(y)$ means x is unhappy, and x and y range over people.
   (b) $\exists\; z.P(z,x) \wedge S(z,y) \wedge W(y)$ where $P(z,x)$ means z is a parent of x, $S(z,y)$ means z and y are siblings, $W(y)$ means y is a woman, and x, y, and z range over people.
3. State whether the following are true or false, where x, y and z range over the integers.
   (a) $\forall\, x.\exists\, y.(2x - y = 0)$
   (b) $\exists\, y.\forall\, x.(2x - y = 0)$
   (c) $\forall\, x.\exists\, y.(x - 2y = 0)$
   (d) $\forall\, x.x < 10 \Rightarrow \forall\, y.(y < x \Rightarrow y < 9)$
   (e) $\exists\, y.\exists\, z.y + z = 100$
   (f) $\forall\, x.\exists\, y.(y > x \wedge \exists\, z.y + z = 100)$
4. What changes above if x, y and z range over the reals?
5. Formalise the following (over the real numbers):
   (a) Negative numbers don't have square roots
   (b) Every positive number has exactly two square roots

Exercise Sheet 3: Structured Proof

1. Give structured proofs of
   (a) $(P \Rightarrow Q) \Rightarrow ((Q \Rightarrow R) \Rightarrow (P \Rightarrow R))$
   (b) $(P \Rightarrow Q) \Rightarrow ((R \Rightarrow \neg Q) \Rightarrow (P \Rightarrow \neg R))$
   (c) $(P \Rightarrow (Q \Rightarrow R)) \Rightarrow (\neg R \Rightarrow (P \Rightarrow \neg Q))$
2. Consider the following non-Theorem. What's wrong with the claimed proof?

   Non-Theorem Suppose x and y are reals, and $x + y = 10$. Then $x \neq 3$ and $y \neq 8$.

   Proof Suppose the conclusion of the Theorem is false. Then $x = 3$ and $y = 8$. But then $x + y = 11$, which contradicts the assumption that $x + y = 10$. Hence the conclusion must be true.
3. Give a structured proof of $((\forall\, x.\mathrm{L}(x) \Rightarrow \mathrm{F}(x)) \wedge (\exists\, x.\mathrm{L}(x) \wedge \neg \mathrm{C}(x))) \Rightarrow \exists\, x.\mathrm{F}(x) \wedge \neg \mathrm{C}(x)$
4. Give a structured proof of $(\exists\, x.(P(x) \Rightarrow Q(x))) \Rightarrow ((\forall\, x.P(x)) \Rightarrow \exists\, x.Q(x))$
5. Prove that, for any $n \in \mathbb{N}$, n is even iff n^3 is even (hint: first define what 'even' means).
6. Prove that the following are equivalent:
   (a) $\exists\, x.P(x) \wedge \forall\, y.(P(y) \Rightarrow y = x)$
   (b) $\exists\, x.\forall\, y.P(y) \Leftrightarrow y = x$

Exercise Sheet 4: Sets

1. Consider the set $A \overset{\text{def}}{=} \{\{\}, \{\{\}\}, \{\{\{\}\}\}\}$. If $x \in A$, how many elements might x have?
2. Prove that if $A \subseteq \mathrm{B}$ then $A \cup \mathrm{B} = \mathrm{B}$
3. Prove that if $A \subseteq A'$ and $\mathrm{B} \subseteq B'$ then $A \times \mathrm{B} \subseteq A' \times B'$
4. What can you say about sets A and B if you know that
   (a) $A \cup \mathrm{B} = A$
   (b) $A \cap \mathrm{B} = A$
   (c) $A - \mathrm{B} = A$
   (d) $A \cap \mathrm{B} = \mathrm{B} \cap A$
   (e) $A - \mathrm{B} = \mathrm{B} - A$
5. Draw the Hasse diagram for the subset relation $\subseteq$ among the sets $A \overset{\text{def}}{=} \{2, 4, 6\}$, $\mathrm{B} \overset{\text{def}}{=} \{2, 6\}$, $\mathrm{C} \overset{\text{def}}{=} \{4, 6\}$, and $D \overset{\text{def}}{=} \{4, 6, 8\}$.
6. Is $\mathcal{P}(A \cap \mathrm{B}) = \mathcal{P}(A) \cap \mathcal{P}(\mathrm{B})$ true for all sets A and B? Either prove it or give a counterexample.
7. Is $\mathcal{P}(A \cup \mathrm{B}) = \mathcal{P}(A) \cup \mathcal{P}(\mathrm{B})$ true for all sets A and B? Either prove it or give a counterexample.
8. Draw pictures illustrating the following subsets of $\mathbb{R}^2$.
   (a) $\{(x, y) \mid y = x^2 - x - 2\}$
   (b) $\{(x, y) \mid y < x\}$
   (c) $\{(x, y) \mid (y > 0 \wedge y = x)\} \cup \{(2, y) \mid y > 1\} \cup \{(0, 0)\}$

9. Let S be a set of students, R a set of college rooms, P a set of professors, and C a set of courses. Let $\mathrm{L} \subseteq S \times R$ be the relation containing (s, r) if student s lives in room r. Let $E \subseteq S \times \mathrm{C}$ be the relation containing (s, c) if student s is enrolled for course c. Let $\mathrm{T} \subseteq \mathrm{C} \times P$ be the relation containing (c, p) if course c is lectured by professor p. Describe the following relations.

   (a) E^{-1}

   (b) $\mathrm{L}^{-1}; E$

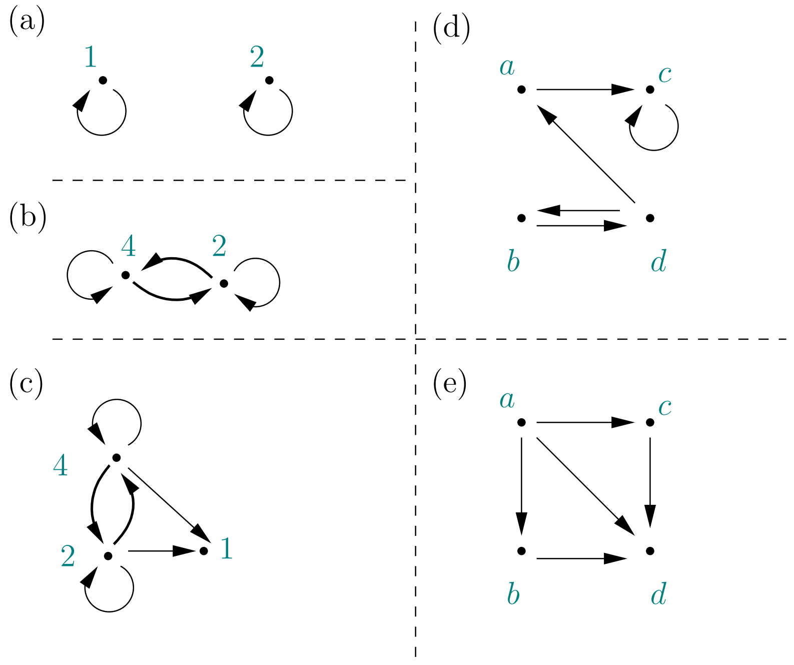
   (c) $E; E^{-1}$

   (d) $(\mathrm{L}^{-1}; E); \mathrm{T}$

   (e) $\mathrm{L}^{-1}; (E; \mathrm{T})$

   (f) $(\mathrm{L}^{-1}; \mathrm{L})^{+}$

10. For each of the following 5 relations, list its ordered pairs. Give a table showing for each whether it is reflexive, symmetric, transitive, acyclic, antisymmetric, and/or total.



11. Give a table showing, for each of the following relations over $\mathbb{N}$, whether it is reflexive, symmetric, transitive, or functional.

   (a) $n \; R \; m \overset{\text{def}}{=} n = 2m$

   (b) $n \; R \; m \overset{\text{def}}{=} 2n = m$

   (c) $n \; R \; m \overset{\text{def}}{=} \exists\, k . k \geq 2 \wedge k \text{ divides } n \wedge k \text{ divides } m$

12. (a) If R and S are directed acyclic graphs over a set A, is $R; S$? Either prove it or give a counterexample.

   (b) If R and S are directed acyclic graphs over a set A, is $R \cup S$? Either prove it or give a counterexample.

   (c) If R and S are directed acyclic graphs over a set A, is $R \cap S$? Either prove it or give a counterexample.

   (d) If R is a relation over a set A, can it be both symmetric and antisymmetric? Either give an example or prove it cannot.

Exercise Sheet 5: Inductive Proof

In all of the following, please state your induction hypothesis explicitly as a predicate.

1. Prove that, for all natural numbers n, $\sum_{i=1}^{n} i^2 = n(n+1)(2n+1)/6$.
2. Prove that, for all natural numbers x, m, and n, $x^{m+n} = x^m\, x^n$.
3. Prove that for all $n \geq 3$, if n distinct points on a circle are joined by consecutive order by straight lines, then the interior angles of the resulting polygon add up to $180(n-2)$ degrees.
4. Prove that, for any positive integer n, a $2^n \times 2^n$ square grid with any one square removed can be tiled with L-shaped pieces consisting of 3 squares.
5. Consider the following pair of ML function declarations:

   ```
   fun takew p [] = []
     | takew p (x::xs) = if p x then x ::  takew p xs else []
   fun dropw p [] = []
     | dropw p (x::xs) = if p x then dropw p xs else x::xs
   ```

   Prove `(takew p xs) @ (dropw p xs) = xs` using induction. (Assume that function `p` always terminates.) [Software Engineering II, 2001, p.2, q.9b]
6. Consider the following two ML functions:

   ```
    fun sumfiv [] = 0
      | sumfiv (x::xs) = 5*x + sumfiv xs
    fun summing z [] = z
      | summing z (x::xs) = summing (z + x) xs
   ```

   Prove that `sumfiv xs` is equal to `5 * summing 0 xs`. [Software Engineering II, 1999, p.2, q.9c]