Part I

Proofs

Mathematical Proofs

This text is all about methods for constructing and understanding proofs. In fact, we could have titled the book *Proofs, Proofs, and More Proofs*. We will begin in Part I with a description of basic proof techniques. We then apply these techniques in chapter 4 to establish some very important facts about numbers, facts that form the underpinning of the world's most widely used cryptosystem.

Simply put, a proof is a method of establishing truth. Like beauty, "truth" sometimes depends on the eye of the beholder, however, and it should not be surprising that what constitutes a proof differs among fields. For example, in the judicial system, *legal* truth is decided by a jury based on the allowable evidence presented at trial. In the business world, *authoritative* truth is specified by a trusted person or organization, or maybe just your boss. In fields such as physics and biology, *scientific* truth[1] is confirmed by experiment. In statistics, *probable* truth is

[1] Actually, only scientific *falsehood* can really be demonstrated by an experiment—when the experiment fails to behave as predicted. But no amount of experiment can confirm that the *next* experiment won't fail. For this reason, scientists rarely speak of truth, but rather of *theories* that accurately predict

established by statistical analysis of sample data.

Philosophical proof involves careful exposition and persuasion typically based on a series of small, plausible arguments. The best example begins with "Cogito ergo sum," a Latin sentence that translates as "I think, therefore I am." It comes from the beginning of a 17th century essay by the mathematician/philosopher, René Descartes, and it is one of the most famous quotes in the world: do a web search on the phrase and you will be flooded with hits.

Deducing your existence from the fact that you're thinking about your existence is a pretty cool and persuasive-sounding idea. However, with just a few more lines of argument in this vein, Descartes goes on to conclude that there is an infinitely beneficent God. Whether or not you believe in a beneficent God, you'll probably agree that any very short proof of God's existence is bound to be far-fetched. So even in masterful hands, this approach is not reliable.

Mathematics has its own specific notion of "proof."

mathematical

Definition. A ~~formal~~ *proof* of a *proposition* is a chain of *logical deductions* leading to

---

past, and anticipated future, experiments.

the proposition from a base set of *axioms*.

The three key ideas in this definition are highlighted: proposition, logical deduction, and axiom. These three ideas are explained in the following chapters, beginning with propositions in chapter 1. We will then provide *lots* of examples of proofs and even some examples of "false proofs" (*i.e.*, arguments that look like a proof but that contain mis-steps, or deductions that aren't so logical when examined closely). False proofs are often even more important as examples than correct proofs, because ~~is only through a step-by-step verification of a "proof" of an absurd proposition that you can hone your skills at spotting fallacies making~~ they are uniquely helpful with honing your skills at ~~math~~ making sure each step of an argument a proof follows logically from prior steps.

~~We conclude Part I by listing~~
~~Chapter 2 contains a dozen~~
~~Chapters 3 and 3 contain descriptions of and templates~~

Creating a good proof is a lot like creating a beautiful work of art. In fact, mathematicians often refer to really good proofs as being "elegant" or "beautiful." ~~It will take practice~~ ~~It may take some~~

→ As with any endeavor, it will probably take a little practice before your fellow students use such praise when referring to your proofs, but to get you started in the right direction, we will ~~desert the~~ provide templates for the most useful proof techniques in chapters 2 and 3. We then apply these techniques in chapter 4 to establish some important facts about numbers; facts that form the underpinning of one of the world's most widely used cryptosystems.

Chapter 1

Propositions

Definition. A *proposition* is a mathematical statement that is either true or false.

For example, both of the following statements are propositions. The first is true and the second is false.

Proposition 1.0.1. $2 + 3 = 5$.

Proposition 1.0.2. $1 + 1 = 3$.

Being true or false doesn't sound like much of a limitation, but it does exclude

statements such as, "Wherefore art thou Romeo?" and "Give me an *A*!".

Unfortunately it is not always easy to decide if a proposition is true or false, or even what the proposition means. In part, this is because the English language is riddled with ambiguities. For example, here are some statements that illustrate the issue:

1. "You may have cake, or you may have ice cream."

2. "If pigs can fly, then you can understand the Chebyshev bound."

3. "If you can solve any problem we come up with, then you get an *A* for the course."

4. "Every American has a dream."

What *precisely* do these sentences mean? Can you have both cake and ice cream or must you choose just one dessert? If the second sentence is true, then is the Chebyshev bound incomprehensible? If you can solve some problems we come up with but not all, then do you get an *A* for the course? And can you still get an *A* even if you can't solve any of the problems? Does the last sentence imply that all

Americans have the same dream or might some of them have different dreams?

Some uncertainty is tolerable in normal conversation. But when we need to formulate ideas precisely—as in mathematics and programming—the ambiguities inherent in everyday language can be a real problem. We can't hope to make an exact argument if we're not sure exactly what the statements mean. So before we start into mathematics, we need to investigate the problem of how to talk about mathematics.

To get around the ambiguity of English, mathematicians have devised a special mini-language for talking about logical relationships. This language mostly uses ordinary English words and phrases such as "or", "implies", and "for all". But mathematicians endow these words with definitions more precise than those found in an ordinary dictionary. Without knowing these definitions, you might sometimes get the gist of statements in this language, but you would regularly get misled about what they really meant.

Surprisingly, in the midst of learning the language of mathematics, we'll come

across the most important open problem in computer science—a problem whose solution could change the world.

1.1 Compound Propositions

In English, we can modify, combine, and relate propositions with words such as "not", "and", "or", "implies", and "if-then". For example, we can combine three propositions into one like this:

If all humans are mortal **and** all Greeks are human, **then** all Greeks are mortal.

For the next while, we won't be much concerned with the internals of propositions—whether they involve mathematics or Greek mortality—but rather with how propositions are combined and related. So we'll frequently use variables such as P and Q in place of specific propositions such as "All humans are mortal" and "$2 + 3 = 5$". The understanding is that these variables, like propositions, can take on only the values T (true) and F (false). Such true/false variables are sometimes called *Boolean variables* after their inventor, George—you guessed it—Boole.

1.1.1 NOT, AND, OR

We can precisely define these special words using *truth tables*. For example, if P denotes an arbitrary proposition, then the truth of the proposition "NOT(P)" is defined by the following truth table:

P	NOT(P)
T	F
F	T

The first row of the table indicates that when proposition P is true, the proposition "NOT(P)" is false. The second line indicates that when P is false, "NOT(P)" is true. This is probably what you would expect.

In general, a truth table indicates the true/false value of a proposition for each possible setting of the variables. For example, the truth table for the proposition "P AND Q" has four lines, since the two variables can be set in four different ways:

P	Q	P AND Q
T	T	T
T	F	F
F	T	F
F	F	F

According to this table, the proposition "P AND Q" is true only when P and Q are both true. This is probably the way you think about the word "and."

There is a subtlety in the truth table for "P OR Q":

P	Q	P OR Q
T	T	T
T	F	T
F	T	T
F	F	F

The third row of this table says that "P OR Q" is true even if *both* P and Q are true. This isn't always the intended meaning of "or" in everyday speech, but this is the standard definition in mathematical writing. So if a mathematician says, "You may have cake, or you may have ice cream," he means that you *could* have both.

If you want to exclude the possibility of both having and eating, you should use "exclusive-or" (XOR):

P	Q	P XOR Q
T	T	F
T	F	T
F	T	T
F	F	F

1.1.2 IMPLIES

The least intuitive connecting word is "implies." Here is its truth table, with the lines labeled so we can refer to them later.

P	Q	P IMPLIES Q	
T	T	T	(tt)
T	F	F	(tf)
F	T	T	(ft)
F	F	T	(ff)

Let's experiment with this definition. For example, is the following proposition true or false?

"If the Riemann Hypothesis is true, then $x^2 \geq 0$ for every real number x."

The Riemann Hypothesis is a famous unresolved conjecture in mathematics (*i.e.*, no one knows if it is true or false). But that doesn't prevent you from answering the question! This proposition has the form $P \Rightarrow Q$ where the *hypothesis*, P, is "the Riemann Hypothesis is true" and the *conclusion*, Q, is "$x^2 \geq 0$ for every real number x". Since the conclusion is definitely true, we're on either line (tt) or line (ft) of the truth table. Either way, the proposition as a while is *true*!

One of our original examples demonstrates an even stranger side of implications.

"If pigs can fly, then you can understand the Chebyshev bound."

Don't take this as an insult; we just need to figure out whether this proposition is true or false. Curiously, the answer has *nothing* to do with whether or not you can understand the Chebyshev bound. Pigs cannot fly, so we're on either line (ft) or

line (ff) of the truth table. In both cases, the proposition is *true*!

In contrast, here's an example of a false implication:

"If the moon shines white, then the moon is made of white cheddar."

Yes, the moon shines white. But, no, the moon is not made of white cheddar cheese. So we're on line (tf) of the truth table, and the proposition is false.

The truth table for implications can be summarized in words as follows:

An implication is true exactly when the if-part is false or the then-part is true.

This sentence is worth remembering; a large fraction of all mathematical statements are of the if-then form!

1.1.3 IFF

Mathematicians commonly join propositions in one additional way that doesn't arise in ordinary speech. The proposition "P if and only if Q" asserts that P and Q

are logically equivalent; that is, either both are true or both are false.

P	Q	P IFF Q
T	T	T
T	F	F
F	T	F
F	F	T

For example, the following if-and-only-if statement is true for every real number x:

$$x^2 - 4 \geq 0 \quad \text{iff} \quad |x| \geq 2$$

For some values of x, *both* inequalities are true. For other values of x, *neither* inequality is true . In every case, however, the proposition as a whole is true.

1.1.4 Notation

Mathematicians have devised symbols to represent words like "AND" and "NOT". The most commonly-used symbols are summarized in the table below.

English	Cryptic Notation
NOT(P)	$\neg P$ (alternatively, $\overline{P}$)
P AND Q	$P \wedge Q$
P OR Q	$P \vee Q$
P IMPLIES Q	$P \longrightarrow Q$
if P then Q	$P \longrightarrow Q$
P IFF Q	$P \longleftrightarrow Q$ (alternatively, P iff Q)

For example, using this notation, "If P AND NOT(Q), then R" would be written:

$$(P \wedge \overline{Q}) \longrightarrow R$$

This symbolic language is helpful for writing complicated logical expressions compactly. But words such as "OR" and "IMPLIES" generally serve just as well as the cryptic symbols $\vee$ and $\longrightarrow$, and their meaning is easy to remember. We will use them interchangeably and you can feel free to use whichever convention is easiest for you.

1.1.5 Logically Equivalent Implications

Do these two sentences say the same thing?

If I am hungry, then I am grumpy.

If I am not grumpy, then I am not hungry.

We can settle the issue by recasting both sentences in terms of propositional logic.[1] Let P be the proposition "I am hungry", and let Q be "I am grumpy". The first sentence says "P IMPLIES Q" and the second says "NOT(Q) IMPLIES NOT(P)". We

[1] This sounds scary, but don't worry, propositional logic is easy. Illegible compound propositions.

can compare these two statements in a truth table:

P	Q	P IMPLIES Q	NOT(Q) IMPLIES NOT(P)
T	T	T	T
T	F	F	F
F	T	T	T
F	F	T	T

Sure enough, the columns of truth values under these two statements are the same, which precisely means they are equivalent. In general, "NOT(Q) IMPLIES NOT(P)" is called the *contrapositive* of the implication "P IMPLIES Q." And, as we've just shown, the two are just different ways of saying the same thing.

In contrast, the *converse* of "P IMPLIES Q" is the statement "Q IMPLIES P". In terms of our example, the converse is:

If I am grumpy, then I am hungry.

This sounds like a rather different contention, and a truth table confirms this suspicion:

P	Q	P IMPLIES Q	Q IMPLIES P
T	T	T	T
T	F	F	T
F	T	T	F
F	F	T	T

Thus, an implication *is* logically equivalent to its contrapositive but is *not* equivalent to its converse.

One final relationship: an implication and its converse together are equivalent to an iff statement, specifically, to these two statements together. For example,

If I am grumpy, THEN I am hungry, AND

if I am hungry, THEN I am grumpy.

are equivalent to the single statement:

I am grumpy IFF I am hungry.

Once again, we can verify this with a truth table:

P	Q	(P IMPLIES Q)	(Q IMPLIES P)	(P IMPLIES Q) AND (Q IMPLIES P)	P IFF Q
T	T	T	T	T	T
T	F	F	T	F	F
F	T	T	F	F	F
F	F	T	T	T	T

1.1.6 Problems

Class Problems

Homework Problems

all problems will be at the end of each chapter, so pls remove them from sections.

1.2 Propositional Logic in Computer Programs

Propositions and logical connectives arise all the time in computer programs. For example, consider the following snippet, which could be either C, C++, or Java:

```
if ( x > 0 || (x <= 0 && y > 100) )
```

⋮

(further instructions)

The symbol `||` denotes "OR", and the symbol `&&` denotes "AND". The *further instructions* are carried out only if the proposition following the word `if` is true. On closer inspection, this big expression is built from two simpler propositions. Let A be the proposition that `x > 0`, and let B be the proposition that `y > 100`. Then we can rewrite the condition "A OR (NOT(A) AND B)". A truth table reveals that

this complicated expression is logically equivalent to "A OR B".

A	B	A OR (NOT(A) AND B)	A OR B
T	T	T	T
T	F	T	T
F	T	T	T
F	F	F	F

This means that we can simplify the code snippet without changing the program's behavior:

```
if ( x > 0 || y > 100 )
    ⋮
(further instructions)
```

Rewriting a logical expression involving many variables in the simplest form is both difficult and important. Simplifying expressions in software can increase the speed of your program. Chip designers face a similar challenge—instead of minimizing `&&` and `||` symbols in a program, their job is to minimize the number of analogous physical devices on a chip. The payoff is potentially enormous: a chip with fewer devices is smaller, consumes less power, has a lower defect rate, and is cheaper to manufacture.

1.2.1 Problems

Class Problems

Homework Problems

1.2.2 Problems

Class Problems

1.3 Predicates and Quantifiers

David: we'll need to ensure uniformity on capitalization. Any convention is OK by me.

1.3.1 Propositions with infinitely many cases

Most of the examples of propositions that we have considered thus far have been nice in the sense that it has been relatively easy to determine if they are true or false. At worse, there were only a few cases to check in a truth table. Unfortunately, not all propositions are so easy to check. That is because some propositions may involve a large or infinite number of possible cases. For example, consider the following proposition involving prime numbers. (A *prime* is an integer greater

than 1 that is divisible only by itself and 1. For example, 2, 3, 5, 7, and 11 are primes, but 4, 6, and 9 are not. A number greater than 1 that is not prime is said to be *composite*.)

Proposition 1.3.1. *For every nonnegative integer, n, the value of $n^2 + n + 41$ is prime.*

It is not immediately clear whether this proposition is true or false. In such circumstances, it is tempting to try to determine its veracity by computing the value of[2]

$$p(n) ::= n^2 + n + 41. \tag{1.1}$$

for several values of n and then checking to see if they are prime. If any of the computed values is not prime, then we will know that the proposition is false. If all the computed values are indeed prime, then we might be tempted to conclude that the proposition is true.

We begin with $p(0) = 41$ which is prime. $p(1) = 43$ is also prime. So is $p(2) = 47$, $p(3) = 53$,..., and $p(20) = 461$, all of which are prime. Hmmm... It is starting to look like $p(n)$ is

[2]The symbol ::= means "equal by definition." It's always ok to simply write "=" instead of ::=, but reminding the reader that an equality holds by definition can be helpful.

a prime for every nonnegative integer n. In fact, we can keep checking through $n = 39$ and confirm that $p(39) = 1601$ is prime. The proposition certainly does seem to be true.

But $p(40) = 40^2 + 40 + 41 = 41 \cdot 41$, which is not prime. So it's *not* true that the expression is prime *for all* nonnegative integers, and thus the proposition is false!

EDITING NOTE: In fact, it's not hard to show that *no* polynomial with integer coefficients can map all natural numbers into prime numbers, unless it's a constant.

Although surprising, this example is not as contrived or rare as you might suspect. As we will soon see, there are many examples of propositions that seem to be true when you check a few cases, but which turn out to be false. The key to remember is that you can't check a claim about an infinite set by checking a finite set of its elements, no matter how large the finite set.

Propositions that involve *all* numbers are so common that there is a special

notation for them. For example, Proposition 1.3.1 can also be written as

$$\forall n \in \mathbb{N}.\ p(n) \text{ is prime.} \tag{1.2}$$

Here the symbol $\forall$ is read "for all". The symbol $\mathbb{N}$ stands for the set of *nonnegative integers*, namely, 0, 1, 2, 3, ... (ask your instructor for the complete list). The symbol "$\in$" is read as "is a member of," or "belongs to," or simply as "is in". The period after the $\mathbb{N}$ is just a separator between phrases.

Here is another example of a proposition that, at first, seems to be true but which turns out to be false.

It was checked by humans and then by computers for many values of a, b, c and d over the next two centuries.

Proposition 1.3.2. *$a^4 + b^4 + c^4 = d^4$ has no solution when a, b, c, d are positive integers.*

this

Euler (pronounced "oiler") conjectured proposition to be true this in 1769. Ultimately the proposition was proven false in 1987 by Noam Elkies. The solution he found was $a = 95800, b = 217519, c = 414560, d = 422481$. No wonder it took 218 years to show the proposition is false!

In logical notation, Proposition 1.3.2 could be written,

$$\forall a \in \mathbb{Z}^+ \forall b \in \mathbb{Z}^+ \forall c \in \mathbb{Z}^+ \forall d \in \mathbb{Z}^+.\ a^4 + b^4 + c^4 \neq d^4.$$

Here, $\mathbb{Z}^+$ is a symbol for the positive integers. Strings of $\forall$'s are usually abbreviated for easier reading, as follows:

$$\forall a, b, c, d \in \mathbb{Z}^+ .\ a^4 + b^4 + c^4 \neq d^4.$$

The following proposition is even nastier.

Proposition 1.3.3. *$313(x^3 + y^3) = z^3$ has no solution when $x, y, z \in \mathbb{Z}^+$.*

This proposition is also false, but the smallest counterexample values for x, y, and z have more than 1000 digits! Even the world's largest computers would not be able to get that far with brute force. Of course, you may be wondering why anyone would care whether or not there is a solution to $313(x^3 + y^3) = z^3$ where x, y, and z are positive integers. It turns out that finding solutions to such equations is important in the field of elliptic curves, which turns out to be important to the study of factoring large integers, which turns out (as we will see in Chapter 4) to be important in cracking commonly-used cryptosystems, which is why mathematicians went to the effort to find the solution with thousands of digits.

# of course, not all propositions

[Illegible] that have infinitely many cases to check turn out to be false. The

following proposition (known as the "Four-Color Theorem") turns out to be true.

Proposition 1.3.4. *Every map can be colored with 4 colors so that adjacent*[3] *regions have different colors.*

The proof of this proposition is difficult and took over a century to perfect. Along the way, many incorrect proofs were proposed, including one that stood for 10 years in the late 19th century before the mistake was found. An extremely laborious proof was finally found in 1976 by mathematicians Appel and Haken, who used a complex computer program to categorize the four-colorable maps; the program left a few thousand maps uncategorized, and these were checked by hand by Haken and his assistants—including his 15-year-old daughter. There was a lot of debate about whether this was a legitimate proof: the proof was too big to be checked without a computer, and no one could guarantee that the computer calculated correctly, nor did anyone have the energy to recheck the four-colorings of the thousands of maps that were done by hand. Within the past decade, a mostly

[3]Two regions are adjacent only when they share a boundary segment of positive length. They are not considered to be adjacent if their boundaries meet only at a few points.

intelligible proof of the Four-Color Theorem was found, though a computer is still needed to check the colorability of several hundred special maps.[4]

In some cases, we do not know whether or not a proposition is true. For example, the following simple proposition (known as Goldbach's Conjecture) has been heavily studied since 1742 but we still do not know if it is true. Of course, it has been checked by computer for many values of n, but as we have seen, that is not sufficient to conclude that it is true.

Proposition 1.3.5 (Goldbach). *Every even integer greater than 2 is the sum of two primes.*

While the preceding propositions are important in mathematics, computer scientists are often interested in propositions concerning the "correctness" of programs and systems, to determine whether a program or system does what it's

[4]See http://www.math.gatech.edu/~thomas/FC/fourcolor.html

The story of the Four-Color Proof is told in a well-reviewed popular (non-technical) book: "Four Colors Suffice. How the Map Problem was Solved." *Robin Wilson*. Princeton Univ. Press, 2003, 276pp. ISBN 0-691-11533-8.

supposed to. Programs are notoriously buggy, and there's a growing community of researchers and practitioners trying to find ways to prove program correctness. These efforts have been successful enough in the case of CPU chips that they are now routinely used by leading chip manufacturers to prove chip correctness and avoid mistakes like the notorious Intel division bug in the 1990's.

EDITING NOTE: ref needed

Developing mathematical methods to verify programs and systems remains an active research area. We'll consider some of these methods later in the text.

1.3.2 Predicates

A *predicate* is a proposition whose truth depends on the value of one or more variables. Most of the propositions above were defined in terms of predicates. For example,

"n is a perfect square"

is a predicate whose truth depends on the value of n. The predicate is true for $n = 4$ since four is a perfect square, but false for $n = 5$ since five is not a perfect square.

Like other propositions, predicates are often named with a letter. Furthermore, a function-like notation is used to denote a predicate supplied with specific variable values. For example, we might name our earlier predicate P:

$$P(n) ::= \text{“}n \text{ is a perfect square”}$$

Now $P(4)$ is true, and $P(5)$ is false.

This notation for predicates is confusingly similar to ordinary function notation. If P is a predicate, then $P(n)$ is either *true* or *false*, depending on the value of n. On the other hand, if p is an ordinary function, like $n^2 + n$, then $p(n)$ is a *numerical quantity*. **Don't confuse these two!**

1.3.3 Quantifiers

There are a couple of assertions commonly made about a predicate: that it is *sometimes* true and that it is *always* true. For example, the predicate

$$\text{“}x^2 \geq 0\text{”}$$

is always true when x is a real number. On the other hand, the predicate

$$\text{“}5x^2 - 7 = 0\text{”}$$

is only sometimes true; specifically, when $x = \pm\sqrt{7/5}$.

There are several ways to express the notions of “always true” and “sometimes true” in English. The table below gives some general formats on the left and specific examples using those formats on the right. You can expect to see such phrases hundreds of times in mathematical writing!

Always True	
For all n, $P(n)$ is true.	For all $x \in \mathbb{R}$, $x^2 \geq 0$.
$P(n)$ is true for every n.	$x^2 \geq 0$ for every $x \in \mathbb{R}$.
Sometimes True	
There exists an n such that $P(n)$ is true.	There exists an $x \in \mathbb{R}$ such that $5x^2 - 7 = 0$.
$P(n)$ is true for some n.	$5x^2 - 7 = 0$ for some $x \in \mathbb{R}$.
$P(n)$ is true for at least one n.	$5x^2 - 7 = 0$ for at least one $x \in \mathbb{R}$.

All these sentences quantify how often the predicate is true. Specifically, an assertion that a predicate is always true, is called a *universally quantified* statement. An assertion that a predicate is sometimes true, is called an *existentially quantified* statement.

Sometimes English sentences are unclear about quantification:

"If you can solve any problem we come up with, then you get an *A* for the course."

The phrase "you can solve any problem we can come up with" could reasonably be interpreted as either a universal or existential statement. It might mean:

"You can solve *every* problem we come up with,"

or maybe

"You can solve *at least one* problem we come up with."

In the preceding example, the quantified phrase appears inside a larger if-then statement. This is quite normal; quantified statements are themselves propositions and can be combined with AND, OR, IMPLIES, etc., just like any other proposition.

1.3.4 Notation

There are symbols to represent universal and existential quantification, just as there are symbols for "AND" ($\wedge$), "IMPLIES" ($\longrightarrow$), and so forth. In particular, to say that a predicate, $P(x)$, is true for all values of x in some set, D, we write:

$$\forall x \in D.\ P(x) \tag{1.3}$$

The *universal quantifier* symbol $\forall$ is read "for all," so this whole expression (1.3) is read "For all x in D, $P(x)$ is true." Remember that upside-down "A" stands for "**A**ll."

To say that a predicate $P(x)$ is true for at least one value of x in D, we write:

$$\exists x \in D.\ P(x) \tag{1.4}$$

The *existential quantifier* symbol $\exists$, is read "there exists." So expression (1.4) is read, "There exists an x in D such that $P(x)$ is true." Remember that backward "E" stands for "**E**xists."

The symbols $\forall$ and $\exists$ are always followed by a variable—typically with an indication of the set the variable ranges over—and then a predicate, as in the two

examples above.

As an example, let Probs be the set of problems we come up with, Solves(x) be the predicate "You can solve problem x", and G be the proposition, "You get an A for the course." Then the two different interpretations of

> "If you can solve any problem we come up with, then you get an A for the course."

can be written as follows:

$$(\forall x \in \text{Probs. Solves}(x)) \text{ IMPLIES } G,$$

or maybe

$$(\exists x \in \text{Probs. Solves}(x)) \text{ IMPLIES } G.$$

1.3.5 Mixing Quantifiers

Many mathematical statements involve several quantifiers. For example, *Goldbach's Conjecture* states:

> "Every even integer greater than 2 is the sum of two primes."

Let's write this more verbosely to make the use of quantification clearer:

> For every even integer n greater than 2, there exist primes p and q such that $n = p + q$.

Let Evens be the set of even integers greater than 2, and let Primes be the set of primes. Then we can write Goldbach's Conjecture in logic notation as follows:

$$\underbrace{\forall n \in \text{Evens}}_{\text{for every even integer } n > 2} \underbrace{\exists p \in \text{Primes}\ \exists q \in \text{Primes}.}_{\text{there exist primes } p \text{ and } q \text{ such that}}\ n = p + q.$$

The proposition can also be written more simply as

$$\forall n \in \text{Evens}\ \exists p, q \in \text{Primes}.\ n = p + q.$$

1.3.6 Order of Quantifiers

Swapping the order of different kinds of quantifiers (existential or universal) usually changes the meaning of a proposition. For example, let's return to one of our initial, confusing statements:

> "Every American has a dream."

This sentence is ambiguous because the order of quantifiers is unclear. Let A be the set of Americans, let D be the set of dreams, and define the predicate $H(a, d)$ to be "American a has dream d." Now the sentence could mean that there is a single dream that every American shares:

$$\exists\, d \in D.\, \forall a \in A.\, H(a, d)$$

For example, it might be that every American shares the dream of owning their own home.

Or it could mean that every American has a personal dream:

$$\forall a \in A.\, \exists\, d \in D.\, H(a, d)$$

For example, some Americans may dream of a peaceful retirement, while others dream of continuing practicing their profession as long as they live, and still others may dream of being so rich they needn't think at all about work.

Swapping quantifiers in Goldbach's Conjecture creates a patently false statement; namely that every even number ≥ 2 is the sum of *the same* two primes:

Insert

$$\underbrace{\exists\, p, q \in \text{Primes}}_{\substack{\text{there exist primes} \\ p \text{ and } q \text{ such that}}} \underbrace{\forall n \in \text{Evens}.}_{\substack{\text{for every even} \\ \text{integer } n > 2}} \; n = p + q.$$

1.3.7 Variables Over One Domain

When all the variables in a formula are understood to take values from the same nonempty set, D, it's conventional to omit mention of D. For example, instead of $\forall x \in D\ \exists y \in D.\ Q(x, y)$ we'd write $\forall x \exists y.\ Q(x, y)$. The unnamed nonempty set that x and y range over is called the *domain of discourse*, or just plain *domain*, of the formula.

It's easy to arrange for all the variables to range over one domain. For example, Goldbach's Conjecture could be expressed with all variables ranging over the domain $\mathbb{N}$ as

$$\forall n.\, (n \in \text{Evens}) \text{ IMPLIES } (\exists p\, \exists q.\ p \in \text{Primes AND } q \in \text{Primes AND } n = p + q).$$

1.3.8 Negating Quantifiers

There is a simple relationship between the two kinds of quantifiers. The following two sentences mean the same thing:

It is not the case that everyone likes to snowboard.

There exists someone who does not like to snowboard.

In terms of logic notation, this follows from a general property of predicate formulas:

$$\text{NOT}\,(\forall x.\ P(x)) \quad \text{is equivalent to} \quad \exists x.\ \text{NOT}(P(x)).$$

Similarly, these sentences mean the same thing:

There does not exist anyone who likes skiing over magma.

Everyone dislikes skiing over magma.

We can express the equivalence in logic notation this way:

$$\text{NOT}\,(\exists x.\ P(x)) \ \text{IFF}\ \forall x.\ \text{NOT}(P(x)). \tag{1.5}$$

The general principle is that *moving a "not" across a quantifier changes the kind of quantifier.*

1.4 Validity

A propositional formula is called *valid* when it evaluates to **T** no matter what truth values are assigned to the individual propositional variables. For example, the propositional version of the Distributive Law is that P AND $(Q$ OR $R)$ is equivalent to $(P$ AND $Q)$ OR $(P$ AND $R)$. This is the same as saying that

$$[P \text{ AND } (Q \text{ OR } R)] \text{ IFF } [(P \text{ AND } Q) \text{ OR } (P \text{ AND } R)]$$

(eqn A)

is valid. This can be verified by checking the truth table for P AND (Q or R) and (P AND Q) OR (P AND R):

— INSERT D goes here —

will be referenced in new 2.6.3

The same idea extends to predicate formulas, but to be valid, a formula now must evaluate to true no matter what values its variables may take over any unspecified domain, and no matter what interpretation a predicate variable may be given. For example, we already observed that the rule for negating a quantifier is captured by the valid assertion (1.5).

INSERT D

P	Q	R	P AND (Q OR R)	(P AND Q) OR (P AND R)
T	T	T	T	T
T	T	F	T	T
T	F	T	T	T
T	F	F	F	F
F	T	T	F	F
F	T	F	F	F
F	F	T	F	F
F	F	F	F	F

Another useful example of a valid assertion is

$$\exists x \forall y.\ P(x,y) \text{ IMPLIES } \forall y \exists x.\ P(x,y). \tag{1.6}$$

Here's an explanation why this is valid:

Let D be the domain for the variables and P_0 be some binary predicate[5] on D. We need to show that if

$$\exists x \in D\ \forall y \in D.\ P_0(x,y) \tag{1.7}$$

holds under this interpretation, then so does

$$\forall y \in D\ \exists x \in D.\ P_0(x,y). \tag{1.8}$$

So suppose (1.7) is true. Then by definition of $\exists$, this means that some element $d_0 \in D$ has the property that

$$\forall y \in D.\ P_0(d_0,y).$$

By definition of $\forall$, this means that

$$P_0(d_0,d)$$

[5]That is, a predicate that depends on two variables.

is true for all $d \in D$. So given any $d \in D$, there is an element in D, namely, d_0, such that $P_0(d_0, d)$ is true. But that's exactly what (1.8) means, so we've proved that (1.8) holds under this interpretation, as required.

We hope this is helpful as an explanation, although purists would not really want to call it a "proof." The problem is that with something as basic as (1.6), it's hard to see what more elementary axioms are ok to use in proving it. What the explanation above did was translate the logical formula (1.6) into English and then appeal to the meaning, in English, of "for all" and "there exists" as justification.

In contrast to (1.6), the formula

$$\forall y \exists x.\ P(x, y) \text{ IMPLIES } \exists x \forall y.\ P(x, y). \tag{1.9}$$

is *not* valid. We can prove this by describing an interpretation where the hypothesis, $\forall y \exists x.\ P(x, y)$, is true but the conclusion, $\exists x \forall y.\ P(x, y)$, is not true. For example, let the domain be the integers and $P(x, y)$ mean $x > y$. Then the hypothesis would be true because, given a value, n, for y we could, for example, choose the

value of x to be $n+1$. But under this interpretation the conclusion asserts that there is an integer that is bigger than all integers, which is certainly false. An interpretation like this which falsifies an assertion is called a *counter model* to the assertion.

1.5 Satisfiability

A proposition is **satisfiable** if some setting of the variables makes the proposition true. For example, P AND $\overline{Q}$ is satisfiable because the expression is true if P is true or Q is false. On the other hand, P AND $\overline{P}$ is not satisfiable because the expression as a whole is false for both settings of P. But determining whether or not a more complicated proposition is satisfiable is not so easy. How about this one?

$$(P \text{ OR } Q \text{ OR } R) \text{ AND } (\overline{P} \text{ OR } \overline{Q}) \text{ AND } (\overline{P} \text{ OR } \overline{R}) \text{ AND } (\overline{R} \text{ OR } \overline{Q})$$

The general problem of deciding whether a proposition is satisfiable is called *SAT*. One approach to SAT is to construct a truth table and check whether or not a T ever appears. But this approach is not very efficient; a proposition with n variables has a truth table with 2^n lines, so the effort required to decide about a

proposition grows exponentially with the number of variables. For a proposition with just 30 variables, that's already over a billion lines to check!

Is there a more *efficient* solution to SAT? In particular, is there some, presumably very ingenious, procedure that determines in a number of steps that grows *polynomially*—like n^2 or n^{14}—instead of exponentially, whether any given proposition is satisfiable or not? No one knows. And an awful lot hangs on the answer. An efficient solution to SAT would immediately imply efficient solutions to many, many other important problems involving packing, scheduling, routing, and circuit verification, among other things. This would be wonderful, but there would also be worldwide chaos. Decrypting coded messages would also become an easy task (for most codes). Online financial transactions would be insecure and secret communications could be read by everyone.

Recently there has been exciting progress on *sat-solvers* for practical applications like digital circuit verification. These programs find satisfying assignments with amazing efficiency even for formulas with millions of variables. Unfortu-

nately, it's hard to predict which kind of formulas are amenable to sat-solver methods, and for formulas that are NOT satisfiable, sat-solvers generally take exponential time to verify that.

So no one has a good idea how to solve SAT in polynomial time, or how to prove that it can't be done—researchers are completely stuck. The problem of determining whether or not SAT has a polynomial time solution is known as the "**P** vs. **NP**" problem. It is the outstanding unanswered question in theoretical computer science. It is also one of the seven Millenium Problems: the Clay Institute will award you $1,000,000 if you solve the **P** vs. **NP** problem.

1.6 Problems

1.6.1 Problems

Class Problems

Homework Problems

Problems for section 1.1

Problems for ~~Section~~ 1.2

etc.

~~Editor's Notes~~

David: the problem section of the end of each chapter will have subsections but they won't be numbered and they should not appear in the Contents. They will just be italics as follows