A Guide for Making Proofs

Mathijs de Weerdt

This document is loosely based on MIT OpenCourseWare [2].

Abstract. In principle, a proof can be *any* sequence of logical deductions from axioms, definitions, and previously-proved statements that concludes with the proposition in question. This freedom in constructing a proof can seem overwhelming at first. How do you even *start* a proof?
Here is the good news: many proofs follow one of a handful of standard templates. Proofs all differ in the details, of course, but these templates at least provide you with an outline to fill in. We will go through several of these standard patterns, pointing out the basic idea and common pitfalls and giving some examples. Many of these templates fit together; one may give you a top-level outline while others help you at the next level of detail. And we will show you other, more sophisticated proof techniques later on.
The recipes below are very specific at times, telling you exactly which words to write down on your piece of paper. You are certainly free to say things your own way instead; we are just giving you something you *could* say so that you are never at a complete loss.

1 The Axiomatic Method

The standard procedure for establishing truth in mathematics was invented by Euclid, a mathematician working in Alexandria, Egypt around 300 BC. His idea was to begin with five assumptions about geometry, which seemed undeniable based on direct experience. (For example, there is a straight line segment between every pair of points.) Propositions like these that are simply accepted as true are called *axioms* or observations.

Starting from these axioms, Euclid established the truth of many additional propositions by providing "proofs". A *proof* is a sequence of logical deductions from axioms and previously-proved statements that concludes with the proposition in question. There are several common terms for a proposition that has been proved. The different terms hint at the role of the proposition within a larger body of work.

- Important propositions are called *theorems.*
- A *lemma* is a preliminary proposition useful for proving later propositions.
- A *corollary* is an afterthought, a proposition that follows in just a few logical steps from a theorem.

These definitions are not precise. In fact, sometimes a good lemma turns out to be far more important than the theorem it was originally used for.

Euclid's axiom-and-proof approach, now called the axiomatic method, is the foundation for mathematics today.

2 Proving an Implication

An enormous number of mathematical claims have the form "If P, then Q" or, equivalently, "P implies Q" (in propositional logic, $P \rightarrow Q$). Here are some examples:

- (Quadratic Formula) If $ax^2+bx+c=0$ and $a \neq 0$, then $x = \frac{(-b \pm \sqrt{b^2-4ac})}{2a}$.
- (Goldbachs Conjecture) If n is an even integer greater than 2, then n is a sum of two primes.
- If $0 \leq x \leq 2$, then $-x^3+4x+1>0$.

There are a couple standard methods for proving an implication.

2.1 Method #1

In order to prove that P implies Q:

1. Write, "Assume P."
2. Show that Q logically follows.

This method is equivalent to the Fitch rule for the introduction of the implication [5]:

1	P	(hypothesis)
2	$\vdots$	
3	Q	
4	$P \rightarrow Q$	($\rightarrow$-intro, 1,3)

Example

Theorem 1. *If* $0 \leq x \leq 2$*, then* $-x^3+4x+1>0$*.*

Before we write a proof of this theorem, we have to do some scratch-work to figure out why it is true.

The inequality certainly holds for $x=0$; then $-x^3+4x+1$ is equal to 1 and $1>0$. As x grows, the $4x$ term (which is positive) initially seems to have greater magnitude than $-x^3$ (which is negative). For example, when $x=1$, we have $4x=4$, but $-x^3=-1$ only. In fact, it looks like $-x^3$doesn't begin to dominate until $x>2$. So it seems the $-x^3+4x$ part should be nonnegative for all x between 0 and 2, which would imply that $-x^3+4x+1$ is positive.

So far, so good. But we still have to replace all those "seems like" phrases with solid, logical arguments. We can get a better handle on the critical $-x^3 + 4x$ part by factoring it, which is not too hard:

$$-x^3 + 4x = x(2 - x)(2 + x)$$

Aha! For x between 0 and 2, all of the terms on the right side are non-negative. And a product of non-negative terms is also non-negative. Let us organize this blizzard of observations into a clean proof.

Proof. Assume $0 \leq x \leq 2$. Then x, $2-x$, and $2+x$ are all non-negative. Therefore, the product of these terms is also non-negative. Adding 1 to this product gives a positive number, so:

$$x(2 - x)(2 + x) + 1 > 0$$

Multiplying out on the left side proves that

$$-x^3 + 4x + 1 > 0$$

as claimed. □

There are a couple points here that apply to all proofs:

- You will often need to do some scratch-work while you are trying to figure out the logical steps of a proof. Your scratch-work can be as disorganized as you like – full of dead-ends, strange diagrams, obscene words, whatever. But keep your scratch-work separate from your final proof, which should be clear and concise.
- Proofs typically begin with the word "Proof" and end with some sort of doohickey like □ or "q.e.d.". The only purpose for these conventions is to clarify where proofs begin and end.

2.2 Method #2: Prove the Contra-positive

Remember that an implication ("P implies Q") is logically equivalent to its contra-positive ("not Q implies not P"); proving one is as good as proving the other. And often proving the contra-positive is easier than proving the original statement. If so, then you can proceed as follows:

1. Write, "We prove the contra-positive:" and then state the contra-positive.
2. Proceed as in Method #1.

In propositional logic, this method relies on the fact that $(P \to Q) \leftrightarrow (\neg Q \to \neg P)$ is a tautology.

Example

Theorem 2. *If r is irrational, then $\sqrt{r}$ is also irrational.*

Recall that rational numbers are equal to a ratio of integers and irrational numbers are not. So we must show that if r is not a ratio of integers, then $\sqrt{r}$ is also not a ratio of integers. That's pretty convoluted! We can eliminate both "not"'s and make the proof straightforward by considering the contra-positive instead.

Proof. We prove the contra-positive: if $\sqrt{r}$ is rational, then r is rational. Assume that $\sqrt{r}$ is rational. Then there exists integers a and b such that:

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

Squaring both sides gives:

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

Since a^2 and b^2 are integers, r is also rational. □

In the book on "Algorithm Design" [3] the argumentation for (3.14) on page 95 follows this method, as well as the proof of the Marriage theorem (p.372–373,7.40).

2.3 Necessary and sufficient

If $P \to Q$ holds, we say that Q is a *necessary* condition for P. This means that P can never be true without Q being the case as well.

If $Q \to P$ holds, we say that Q is a *sufficient* condition for P. This means that whenever Q is valid, P will also be true. If a condition is both necessary and sufficient, the condition is equivalent, denoted by $P \leftrightarrow Q$. Proving equivalence is discussed in the next section.

3 Proving an "If and Only If"

Many mathematical theorems assert that the following statements are logically equivalent (TFAE); that is, one holds if and only if the other does. Sometimes "iff" is used as a short for "if and only if". Here are some examples:

- An integer is a multiple of 3 if and only if the sum of its digits is a multiple of 3.
- Two triangles have the same side lengths iff all angles are the same.
- A positive integer $p \geq 2$ is prime if and only if $1 + (p - 1) \times (p - 2) \times \ldots \times 3 \times 2 \times 1$ is a multiple of p.

3.1 Method #1: Prove Each Statement Implies the Other

The statement "P if and only if Q" ($P \leftrightarrow Q$) is equivalent to the two statements "P implies Q" and "Q implies P". So you can prove an "if and only if" by proving *two* implications:

1. Write, "We prove that P implies Q and vice-versa."
2. Write, "First, we show that P implies Q." Do this by one of the methods in Section 2.
3. Write, "Now, we show that Q implies P." Again, do this by one of the methods in Section 2.

Example Two sets are defined to be equal if they contain the same elements; that is, $X = Y$ means $z \in X$ if and only if $z \in Y$. So set equivalence proofs often have an "if and only if" structure.

Theorem 3. *(DeMorgan's Law for Sets). Let A, B, and C be sets. Then:*

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

Proof. We show that $z \in A \cap (B \cup C)$ implies that $z \in (A \cap B) \cup (A \cap C)$ and vice-versa.

First, we show that $z \in A \cap (B \cup C)$ implies that $z \in (A \cap B) \cup (A \cap C)$. Assume that $z \in A \cap (B \cup C)$. Then z is in A and z is also in B or C. Thus, z is in either $A \cap B$ or $A \cap C$, which implies $z \in (A \cap B) \cup (A \cap C)$

Now, we show that $z \in (A \cap B) \cup (A \cap C)$ implies that $z \in A \cap (B \cup C)$. Assume that $z \in (A \cap B) \cup (A \cap C)$. Then z is in both A and B or else z is in both A and C. Thus, z is in A and z is also in B or C. This implies that $z \in A \cap (B \cup C)$. □

Also proofs for other equalities (=) have an "if and only if" structure. For example as in the proofs of the following propositions [3]:

- the flow value lemma (p.346–347, 7.6), where a chain of equalities is proven (see Method #2 below), and
- *the size of a maximum cardinality matching is equal to the size of the maximum flow* (p.369, 7.37), which can be proved by first showing that the size of *the size of a maximum cardinality matching is less than or equal (≤) to the size of the maximum flow* and then showing that *the size of the maximum flow is less than or equal to the size of a maximum cardinality matching.*

3.2 Method #2: Construct a Chain of Iffs

In order to prove that P is true if and only if Q is true:

1. Write, "We construct a chain of if-and-only-if implications."
2. Prove P is equivalent to a second statement which is equivalent to a third statement and so forth until you reach Q.

This method is generally more difficult than the first, but the result can be a short, elegant proof.

Example The standard deviation of some values $x_1, x_2, ..., x_n$ is defined to be:

$$\sqrt{(x_1 - \mu)^2 + (x_2 - \mu)^2 + \ldots + (x_n - \mu)^2}$$

where μ is the average of the values:

$$\mu = \frac{x_1 + x_2 + \ldots + x_n}{n}$$

Theorem 4. *The standard deviation of some values $x_1, \ldots, x_n$ is zero if and only if all values are equal to the mean.*

For example, the standard deviation of test scores is zero if and only if everyone scored exactly the class average.

Proof. We construct a chain of "if and only if" implications. The standard deviation of $x_1, \ldots, x_n$ is zero if and only if:

$$\sqrt{(x_1 - \mu)^2 + (x_2 - \mu)^2 + ... + (x_n - \mu)^2} = 0$$

where μ is the average of $x_1, \ldots, x_n$. This equation holds if and only if

$$(x_1 - \mu)^2 + (x_2 - \mu)^2 + \ldots + (x_n - \mu)^2 = 0$$

since zero is the only number whose square root is zero. Every term in this equation is non-negative, so this equation holds if and only every term is actually 0. But this is true if and only if every value x_i is equal to the mean μ. □

3.3 Method #3: Prove a Cycle of Implications

Sometimes you need to prove the equivalence of three or more statements. In that case, it is a good idea to prove a cycle of implications. For example, if you need to prove that P, Q, and R are all equivalent, it suffices to show that $P \to Q$, $Q \to R$, and $R \to P$. This is done in the slides of the course on Algorithms to simultaneously prove the augmenting path theorem and the max-flow min-cut theorem (p.348–350, 7.9-7.13).

4 Proof by Contradiction

In a *proof by contradiction* or *indirect proof*, you show that if a proposition were false, then some logical contradiction or absurdity would follow. Thus, the proposition must be true. Proof by contradiction can be used for any type of proposition. However, as the name suggests, indirect proofs can be a little convoluted. So direct proofs are generally preferable as a matter of clarity.

4.1 Method

In order to prove a proposition P by contradiction:

1. Write, "We use proof by contradiction."
2. Write, "Suppose P is false."
3. Deduce a logical contradiction.
4. Write, "This is a contradiction. Therefore, P must be true."

The equivalent structure in a Fitch proof is as follows:

1	$\quad \neg P$	(hypothesis)
2	$\quad \vdots$	
3	$\quad Q$	
4	$\quad \neg Q$	
5	$\neg\neg P$	($\neg$-intro, 1,3,4)
6	P	($\neg$-elim, 5)

Example Remember that a number is *rational* if it is equal to a ratio of integers. For example, $3.5 = 7/2$ and $0.1111... = 1/9$ are rational numbers. On the other hand, we will prove by contradiction that $\sqrt{2}$ is irrational.

Theorem 5. *$\sqrt{2}$ is irrational.*

Proof. We use proof by contradiction. Suppose the claim is false; that is, $\sqrt{2}$ is rational. Then we can write $\sqrt{2}$ as a fraction of integers a/b *in lowest terms.*

Squaring both sides gives $2 = a^2/b^2$ and so $2b^2 = a^2$. This implies that a is even; that is, a is a multiple of 2. Therefore, a^2 must be a multiple of 4. Because of the equality $2b^2 = a^2$, we know $2b^2$ must also be a multiple of 4. This implies that b^2 is even and so b must be even. But since a and b are both even, the fraction a/b is not in lowest terms. This is a contradiction. Therefore, $\sqrt{2}$ must be irrational. □

In the book on "Algorithm Design" [3], many proofs follow this method. See for example:

- When Gale-Shapley's algorithm (G-S) is applied to a matching problem with as many men as women, it returns a perfect and stable matching (p.8, 1.5 and 1.6).
- G-S always returns a man-optimal and woman-pessimal matching (p.10–12, 1.7 and 1.8).
- If a directed graph G has a topological ordering, then G is a DAG (p.101, 3.18).
- In every DAG G there is a node v with no incoming edges (p.102, 3.19).
- Greedy interval scheduling is optimal (p.121, 4.3).
- The cut-property and the cycle-proerty (p.145–148, 4.17 and 4.20).

4.2 Potential Pitfall

Often students use an indirect proof when a direct proof would be simpler. Such proofs are not wrong; they just are not excellent. Let us look at an example.

Definition 1. *A function f is* strictly increasing *if $f(x) > f(y)$ for all real x and y such that $x > y$.*

Theorem 6. *If f and g are strictly increasing functions, then $f + g$ is a strictly increasing function.*

Let us first look at a simple, direct proof.

Proof. Let x and y be arbitrary real numbers such that $x > y$. Then:

$$f(x) > f(y) \quad \text{(since } f \text{ is strictly increasing)}$$

$$g(x) > g(y) \quad \text{(since } g \text{ is strictly increasing)}$$

Adding these inequalities gives:

$$f(x) + g(x) > f(y) + g(y)$$

Thus, $f + g$ is strictly increasing as well. □

Now we could prove the same theorem by contradiction, but this makes the argument needlessly complicated.

Proof. We use proof by contradiction. Suppose that $f + g$ is not strictly increasing. Then there must exist real numbers x and y such that $x > y$, but

$$f(x) + g(x) \leq f(y) + g(y)$$

This inequality can only hold if either $f(x) \leq f(y)$ or $g(x) \leq g(y)$. Either way, we have a contradiction because both f and g were defined to be strictly increasing. Therefore, $f + g$ must actually be strictly increasing. □

5 Case Analysis

The proof of a statement can sometimes be broken down into several cases, which then can be tackled individually.

5.1 The Method

In order to prove a proposition P using case analysis:

1. Write, "We use case analysis."
2. Identify a sequence of conditions, at least one of which must hold. (If this is not obvious, you must prove it.)
3. For each condition:
   (a) State the condition.
   (b) Prove P assuming that the condition holds.

In a Fitch-style proof, this approach is equivalent to using the rule for the elimination of the $\vee$:

1	$A \vee B$	
2	$\vdots$	
3	$\quad$ | A	(hypothesis)
4	$\quad$ | $\vdots$	
5	$\quad$ | P	
6	$\vdots$	
7	$\quad$ | B	(hypothesis)
8	$\quad$ | $\vdots$	
9	$\quad$ | P	
10	P	($\vee$-elim, 1, 3-5, 7-9)

Often case analysis arguments extend to several levels. The most difficult challenge in a case analysis argument is try to decide how to break up the problem. The most common error is failing to construct a complete set of cases.

Example

Theorem 7. *There exist irrational numbers p and q such that p raised to the power q is rational.*

This is an ingenious proof, not the sort of thing one would think up in a few minutes.

Proof. We use case analysis. Let $v = \sqrt{2}^{\sqrt{2}}$. There are two cases:

Case 1: v is rational. Let $p = q = \sqrt{2}$. Then $p^q = v$ is rational, so the claim holds.

Case 2: v is irrational. Let $p = v$ and $q = \sqrt{2}$. Then:

$$p^q = v^{\sqrt{2}} = \left(\sqrt{2}^{\sqrt{2}}\right)^{\sqrt{2}} = \sqrt{2}^2 = 2$$

Since 2 is rational, the claim holds.

Since we have checked that the claim holds in every case, we know that the claim holds. □

6 Predicate Logic

Sometimes you need to prove a theorem or proposition that includes a phrase like "for all..." or "there exists...". For such theorems you need to use one of the following approaches.

6.1 Proving a "For all..."

In order to prove $\forall_x P(x)$, i.e., that a certain proposition P holds for all objects x, you use the following approach:

1. Write, "Let any object a be given."
2. Then show that $P(a)$, without making any assumptions on the properties of a.
3. Write, "Since we have not made any assumptions on a, we may now conclude that $\forall_x P(x)$."

This method is equivalent to the Fitch-rule of the introduction of the $\forall$. However, note that this rule can only be applied if there are no hypotheses about a.

1	$P(a)$	
2	$\vdots$	
3	$\forall x P(x)$	($\forall$-intro,1)

In the book on "Algorithm Design" [3] this method is used for the proof that all $k-$clusterings have a smaller (or equal) spacing than the clustering produced by deleting the $k-1$ most expensive edges from the minimum spanning tree (p.160-161, 4.26).

6.2 Proving not-for-all

Sometimes you need to prove that it does not hold for all objects x that $P(x)$. This is can be done by showing that there is one object for which $P(x)$ does not hold. This follows immediately from the tautology $\neg\forall x P(x) \leftrightarrow \exists x \neg P(x)$.

1. Write, "Consider the object x^* with the following properties."
2. Show that for this specific object x^* proposition P does not hold (i.e., it holds that $\neg P(x^*)$).
3. Write, "Since we have shown that P does not hold for x^*, we may now conclude that it is not the case that P holds for all objects x."

6.3 Proving a "There exists..."

In order to prove $\exists_x P(x)$, i.e., that there exists at least one object x for which proposition P holds, you use the following approach:

1. Write, "Consider the object x^* with the following properties."

2. Show that for this specific object x^* it holds that $P(x^*)$.
3. Write, "Since proposition P holds for object x^*, we may now conclude that there is an object for which P holds.

This method is equivalent to the Fitch-rule of the introduction of the $\exists$.

1	$P(a)$	
2	$\vdots$	
3	$\exists x P(x)$	($\exists$-intro,1)

Sometimes you need to prove that it does not hold that there is an object x for which $P(x)$. This can be done by showing that for every object $P(x)$ does not hold. See the previous subsection for how to deal with this.

6.4 Using a "For all..." or an "There exists..."

When you know that some proposition P holds for all objects, and if you have an object a in your proof, you may immediately conclude that P also holds for your object a.

When you know that there exists an object x for which some proposition P holds, you are not allowed to do this. However, you are allowed to reason about this specific object x^* for which P holds. Sometimes you can draw more general conclusions from that. The Fitch scheme for this proof structure looks as follows.

1	$\exists P(x)$	
2	$\vdots$	
3	│ $P(x^*)$	(hypothesis)
4	│ $\vdots$	
5	│ Q	
6	Q	($\exists$-elim, 1,3)

7 Proof by induction

For many algorithms, a *proof by induction* is usually the best way to go. See the separate document on induction for how to construct proofs by induction [1].

In the book on "Algorithm Design", the following proofs follow this method [3]:

– If G is a DAG, then G has a topological ordering (p.102, 3.20).
– Dijkstra's algorithm returns the shortest path (p.139, 4.14).
– The run-time of Mergesort is bounded by $O(n \cdot \log n)$ (p.213, 5.2, using substitution).

8 Proofs using an invariant

If your proof is about the correctness of an algorithm with a while loop, it is usually a good idea to first come up with a so-called *invariant*. An invariant is a proposition that is true in every iteration of the while loop. It should always be easy to verify that an invariant is correct: it should hold when you start executing the while loop, and it should also hold at the end of an iteration. You can now conlucde that it also holds when all iterations are finished. Moreover, you are also allowed to use the fact that the condition for the while loop does not hold when the while loop is finished.

9 How to Write Good Proofs

The purpose of a proof is to provide the reader with definitive evidence of an assertion's truth. To serve this purpose effectively, more is required of a proof than just logical correctness: a good proof must also be clear. These goals are usually complimentary; a well-written proof is more likely to be a correct proof, since mistakes are harder to hide. Here are some tips on writing good proofs:

State your game plan. A good proof begins by explaining the general line of reasoning, e.g. "We use case analysis" or "We argue by contradiction". This creates a rough mental picture into which the reader can fit the subsequent details.

Keep a linear flow. We sometimes see proofs that are like mathematical mosaics, with juicy tidbits of reasoning sprinkled across the page. This is not good. The steps of your argument should follow one another in a sequential order.

A proof is an essay, not a calculation. Many students initially write proofs the way they compute integrals. The result is a long sequence of expressions without explanation. This is bad. A good proof usually looks like an essay with some equations thrown in.

Use complete sentences. Avoid excessive symbolism. Your reader is probably good at understanding words, but much less skilled at reading arcane mathematical symbols. So use words where you reasonably can.

Simplify. Long, complicated proofs take the reader more time and effort to understand and can more easily conceal errors. So a proof with fewer logical steps is a better proof.

Introduce notation thoughtfully. Sometimes an argument can be greatly simplified by introducing a variable, devising a special notation, or defining a new term. But do this sparingly since you are requiring the reader to remember all that new stuff. And remember to actually define the meanings of new variables, terms, or notations; do not just start using them!

Structure long proofs. Long programs are usually broken into a hierarchy of smaller procedures. Long proofs are much the same. Facts needed in your proof that are easily stated, but not readily proved are best pulled out and proved in preliminary lemmas. Also, if you are repeating essentially the same argument over and over, try to capture that argument in a general lemma, which you can cite repeatedly instead.

Do not bully. Words such as "clearly", "trivially", and "obviously" serve no logical function. Rather, they almost always signal an attempt to bully the reader into accepting something which the author is having trouble justifying rigorously. Do not use these words in your own proofs and go on the alert whenever you read one.

Finish. At some point in a proof, you will have established all the essential facts you need. Resist the temptation to quit and leave the reader to draw the "obvious" conclusion. What is obvious to you as the author is not likely to be obvious to the reader. Instead, tie everything together yourself and explain why the original claim follows.

The analogy between good proofs and good programs extends beyond structure. The same rigorous thinking needed for proofs is essential in the design of critical computer system. When algorithms and protocols only "mostly work" due to reliance on hand-waving arguments, the results can range from problematic to catastrophic. An early example was the Therac 25, a machine that provided radiation therapy to cancer victims, but occasionally killed them with massive overdoses due to a software race condition [4]. More recently, for example in August 2004, but also in December 2007, computer failures caused the entire fleet of big air transport companies and all their passengers to be grounded![1]

It is a certainty that we will all one day be at the mercy of critical computer systems designed by you and your classmates. So we really hope that you will develop the ability to formulate rock-solid logical arguments that a system actually does what you think it does!

References

1. Srini Devadas and Eric Lehman. Induction I: Lecture notes 3 for 6.042/18.062j mathematics for computer science, 2005. http://ocw.mit.edu/OcwWeb/Electrical-Engineering-and-Computer-Science/6-042JSpring-2005/LectureNotes/index.htm.
2. Srini Devadas and Eric Lehman. Lecture notes 2 for 6.042/18.062j mathematics for computer science, 2005. http://ocw.mit.edu/OcwWeb/Electrical-Engineering-and-Computer-Science/6-042JSpring-2005/LectureNotes/index.htm.
3. Jon Kleinberg and Eva Tardos. *Algorithm Design*. Addison-Wesley, 2005.
4. Nancy Leveson and Clark S. Turner. An investigation of the therac-25 accidents. *IEEE Computer*, 26(7):18–41, 1993.
5. Hans Tonino. *Logica*. Delft University of Technology, 2002.

[1] Search for example for `flight computer failure` on the internet.