A Short Guide to Proof by Induction
Donald Chinn  (October 27, 2006)

This document describes proof by induction. We will discuss when a proof by induction is an appropriate proof technique and how to prove something by induction. We will examine specific examples from very different domains to try to understand the structure of a proof by induction.

Why Proof by Induction?

Proof by induction is just another proof technique. Proofs in general are a form of argument. An argument is a sequence of statements designed to support a claim of fact. A proof is really just a mathematical argument. What distinguishes a proof from an argument in everyday speech (for example, “my argument for why we should cut taxes”) is that mathematical proofs have mathematical precision and they obey the rules of logic and inference. Proofs can be used to support a wide variety of mathematical claims. They can also be used to show that computer programs compute what they were intended to compute, which is a form of mathematical claim.

A Simple Proof

Generally speaking, proofs begin with axioms, postulates, or statements that are given. For example, if we wanted to prove the mathematical statement “If \( x \) an integer and is even, then \( x^2 \) is even,” then we would first suppose that we have an even integer \( x \) that is even. What does it mean for an integer to be even? It means that it can be expressed as the product of 2 and an integer. So, we know that \( x = 2k \), where \( k \) is some integer. What we need to show is that \( x^2 \) is even. But \( x = 2k \), where \( k \) is some integer, so \( x^2 = (2k)^2 = 4k^2 = 2(2k^2) \). Since \( x^2 \) can be expressed as the product of 2 and an integer, then \( x^2 \) is even.

We now have the following proof:

**Claim:** If \( x \) is an even integer, then \( x^2 \) is even.

**Proof:** Suppose \( x \) is an even integer. Then \( x = 2k \), where \( k \) is some integer. So \( x^2 = (2k)^2 = 4k^2 = 2(2k^2) \). Since \( x^2 \) can be expressed as the product of 2 and the integer \( 2k^2 \), then \( x^2 \) is even.

Do you see how each step follows from what is previously known? Also, note that the conclusion of the proof (“\( x^2 \) is even”) is what we needed to prove. You need to thoroughly understand what a proof is before you can understand how to do a proof by induction, because the basic elements of a proof are contained in a proof by induction.

Another important proof technique is proof by contradiction. In proof by contradiction, we proceed by supposing that the claim we want to prove is actually false. From this supposition, we then proceed to prove to be true a statement that is known to be false. Since the reason we achieved this contradiction was by supposing that the original claim was false, then it must be true.
Proof by Induction

A proof by induction has three parts:
1. In the first part, we must clearly define a statement $P(n)$ that is parameterized by an integer $n$.
2. In the second part, we must perform a basis step, where we show that $P(c)$ is true, where $c$ is some small integer, typically 0.
3. In the third part, called the induction step, we must show that for all $n \geq c$, the statement $P(n)$ implies the statement $P(n + 1)$.

Some important things to observe about a proof by induction. Note that $P(n)$ represents an infinite family of statements. For example, consider the following claim:

**Claim 1:** $n \leq 2^n$ for all positive integers $n$.

Notice what the claim says. It says that for any positive integer $n$, something is true about that $n$, namely that $n \leq 2^n$. That is an infinite number of statements, because it says that $1 \leq 2^1$, $2 \leq 2^2$, $3 \leq 2^3$, $4 \leq 2^4$, $5 \leq 2^5$, $6 \leq 2^6$, $7 \leq 2^7$, etc.

The basis step does not have to involve proving $P(0)$. In the claim above, the basis step of a proof by induction would be to prove that $P(1)$ is true. If we wanted to prove the following claim:

**Claim 2:** $n^2 \leq 2^n$ for all integers $n \geq 4$.

then we would want to show that $P(4)$ is true for our basis step. It would be futile to show that $n^2 \leq 2^n$ for all integers $n \geq 0$, because it simply isn’t true ($n = 3$ is a counterexample).

Sometimes in a proof by induction, we need to have more than one basis step. We will see an example of that later in one of the exercises.

In the induction step, we are allowed to assume that $P(n)$ is true for some fixed $n \geq c$, and we must show that $P(n + 1)$ is true. In this step, the statement $P(n)$ is called the induction hypothesis. It is often a good practice when writing out a proof by induction to explicitly write out the induction hypothesis before writing out the induction step so that you know what you are allowed to assume.

Let us return to Claim 1 above. How would we prove it by induction? We must first get an intuitive understanding for why the statement is true. In the case of Claim 1, we can intuitively see that the left side of the inequality, $n$, grows linearly in $n$, whereas the right side of the inequality grows by a factor of 2 each time we increase $n$ by 1. While this intuition is exactly right, it is not a proof of anything.

To prove this claim by induction, we start with clearly defining $P(n)$. In this case, $P(n)$ is the statement “$n \leq 2^n$.” It is improper to write this as $P(n) = n \leq 2^n$. Instead, we should say “Let $P(n)$ be the statement $n \leq 2^n$.” Again, note that for every value of $n$, the statement $P(n)$ is either true or false.
We now need to pick a value $c$ for the basis step. In the case of Claim 1, we should choose $c = 1$, because that is the smallest integer for which we need to show that $P(n)$ is true. Since $1 \leq 2^1$, we can easily see that $P(1)$ is true.

For the induction step, we need to take advantage of our intuition that the right side of the inequality grows very rapidly compared to the left side of the inequality. We are allowed to suppose that for some fixed $n \geq 1$, the induction hypothesis $P(n)$, which is that, $n \leq 2^n$, is true. We then need to show that $P(n + 1)$, which is the statement that $n+1 \leq 2^{n+1}$, is true.

We observe that since $n \geq 1$ and $n \leq 2^n$ (the induction hypothesis), then $n + 1 \leq n + n \leq 2^n + 2^n = 2^{n+1}$. So, $n + 1 \leq 2^{n+1}$. This last statement is $P(n+1)$.

Note that the induction hypothesis was used when we made the step that said that $n + n \leq 2^n + 2^n$. It is usually a good idea to indicate in a proof by induction where we used the induction hypothesis, because (as in this case), the induction hypothesis is not necessarily true for all values of $n$.

Here is the full proof.

**Claim:** $n \leq 2^n$ for all positive integers $n$.

**Proof:** We prove the claim by induction on $n$.

Let $P(n)$ be the statement “$n \leq 2^n$.”

**Basis step:** $n = 1$. Then $n = 1 \leq 2^1 = 2^n$.

**Induction hypothesis:** Suppose that for some fixed $n \geq 1$, $n \leq 2^n$.

**Induction step:** We need to show that $n + 1 \leq 2^{n+1}$.

\[
\begin{align*}
    n + 1 & \leq n + n \quad \text{(since $n \geq 1$)} \\
    & \leq 2^n + 2^n \quad \text{(by the induction hypothesis)} \\
    & = 2^{n+1} 
\end{align*}
\]

This concludes the proof by induction.

Let us look at another classic example of proof by induction.

---

A precise mathematical statement.

Indicate to readers that a proof by induction follows, so that they know what to expect.

Precisely state what $P(n)$ is.

Show that $P(c)$ is true.

Explicitly state the induction hypothesis $P(n)$ so that you don’t get confused about what you can assume in the induction step.

Indicate that the proof by induction is done, so that any other work that follows is not confused with the proof by induction.

Show that $P(n+1)$ is true. Justify each non-obvious step and indicate where the induction hypothesis is used.
Claim: For all $n \geq 0$, \( \sum_{i=0}^{n} i = \frac{n(n+1)}{2} \).

Proof: We prove the claim by induction on $n$.

Let $P(n)$ be the statement \( \sum_{i=0}^{n} i = \frac{n(n+1)}{2} \).

Basis step: $n = 0$. Then \( \sum_{i=0}^{n} i = 0 = \frac{0(1)}{2} = \frac{n(n+1)}{2} \).

Induction hypothesis: Suppose that for some fixed $n \geq 0$, \( \sum_{i=0}^{n} i = \frac{n(n+1)}{2} \).

Induction step: We need to show that \( \sum_{i=0}^{n+1} i = \frac{(n+1)((n+1)+1)}{2} \).

\[
\sum_{i=0}^{n+1} i = (n+1) + \sum_{i=0}^{n} i
\]

\[
= (n+1) + \frac{n(n+1)}{2} \quad \text{(by the induction hypothesis)}
\]

\[
= \frac{2n + 2 + n^2 + n}{2}
\]

\[
= \frac{n^2 + 3n + 2}{2}
\]

\[
= \frac{(n+1)(n+2)}{2}
\]

\[
= \frac{(n+1)((n+1)+1)}{2}
\]

This concludes the proof by induction.

Notice how in the induction step, the summation was broken down into a summation that had a slightly smaller upper limit for the index of summation, so that the induction hypothesis could be used.

Proofs Involving Program Correctness

Proof by induction can be used to prove that programs (or algorithms) compute what the program designer intended for them to compute. Here is a classic example.

```c
int factorial(int n) {
    if (n == 1) {
        return 1;
    } else {
        return n * factorial(n-1);
    }
}
```
What we want to show here is that the program, when passed in \( n \) as a parameter, really does compute the factorial of \( n \). Our intuition about how this recursive program was written tells us that if we want to compute \( n! \), we first compute \((n-1)!\) and then multiply it by \( n \), which is exactly what the code does. Again, this is not a proof, but it provides the key idea for the proof. How do we prove that the program is doing what we intended it to do? By induction, of course.

**Claim:** For all \( n \geq 1 \), \( \text{factorial}(n) \) returns \( n! \).

**Proof:** We prove the claim by induction on \( n \).

Let \( P(n) \) be the statement “\( \text{factorial}(n) \) returns \( n! \).”

**Basis step:** \( n = 1 \). Then \( \text{factorial}(n) \) follows the then part of the if statement and returns 1, which is \( 1! \).

**Induction hypothesis:** Suppose that for some fixed \( n \geq 1 \), \( \text{factorial}(n) \) returns \( n! \).

**Induction step:** We need to show that \( \text{factorial}(n+1) \) returns \( (n+1)! \).

Since \( n \geq 1 \), then \( n+1 \geq 2 \), and so \( \text{factorial}(n+1) \) follows the else part of the if statement and returns \((n+1)\) times the value that \( \text{factorial}(n+1-1) \) returns. But by the induction hypothesis, \( \text{factorial}(n) \) returns \( n! \), and so \( \text{factorial}(n+1) \) returns \( (n+1)\cdot n! = (n+1)! \).

This concludes the proof by induction.

**Proofs Involving Recursive Structures**

Proof by induction is an especially good proof technique when the objects involved are recursively defined. For example, binary trees are defined recursively (a root with at most two subtrees).

**Claim:** The number of nodes in any binary tree of height \( h \) is at most \( 2^{h+1} - 1 \).

**Proof:** We prove the claim by induction on \( h \).

Let \( P(h) \) be the statement “The number of nodes in any binary tree of height \( h \) is at most \( 2^{h+1} - 1 \).”

**Basis step:** \( h = -1 \). Then the tree has 0 nodes, and \( 0 \leq 2^{h+1} - 1 \).

**Induction hypothesis:** Suppose that for some fixed \( h \) and for all \(-1 \leq m \leq h\), the number of nodes in any tree of height \( m \) is at most \( 2^{m+1} - 1 \).

**Induction step:** We need to show that the number of nodes in any binary tree of height \( h+1 \) is at most \( 2^{(h+1)+1} - 1 \).

Let \( T \) be a binary tree of height \( h + 1 \). It consists of a root and at most two subtrees. Each of the subtrees has height at most \( h \) (and at least height -1). Let us say that the left subtree has height \(-1 \leq k \leq h \) and the right subtree has height \(-1 \leq l \leq h \). By the induction hypothesis, these subtrees contain at most \( 2^{k+1} - 1 \) and \( 2^{l+1} - 1 \) nodes, respectively. Note that since \( k \leq h \) and \( l \leq h \), then the number of nodes in each subtree is at most \( 2^{h+1} - 1 \).

Therefore, the total number of nodes in \( T \) is at most 1 (for the root) plus \( 2^{h+1} - 1 \) (the left subtree) plus \( 2^{h+1} - 1 \) (the right subtree). That is, the number of nodes in \( T \) is at most \( 1 + (2^{h+1} - 1) + (2^{h+1} - 1) = 2^{h+2} - 1 \).

This concludes the proof by induction.
This proof by induction is called *strong induction*. In strong induction, the induction hypothesis is that $P(m)$ is true for all $m$ from $c$ up to some fixed but arbitrary $n$. (It’s called strong induction because the induction hypothesis contains more statements, or is logically stronger.) The goal of the induction step is still to show that $P(n+1)$ is true. In this example, we need to use strong induction because the left and right subtrees of the tree T could have any height from -1 up to $h$. In this example, $h$ takes the role of $n$ in our general scheme of proof by induction. If we only were allowed to assume that $P(h)$ were true, then we would not be able to say anything about the number of nodes in the left and right subtrees.

In the case of binary trees, if you are proving a statement by induction on the number of nodes of the tree or on the height of the tree, it is often a good strategy in the induction step to break up the tree into three parts, its root and its two subtrees. The reason this tends to be a good strategy is that we know that the subtrees (if any) have fewer nodes than the original tree (because we are not counting the root as part of the subtree) and each of the subtrees has height less than the original tree. This makes it possible to use the induction hypothesis to conclude something about the subtrees, as we did in this last example.

**Exercises**

Prove the following claims.

**Claim:** $n^2 \leq 2^n$ for all integers $n \geq 4$.

Rewrite your proof to attempt to show that $n^2 \leq 2^n$ for all integers $n \geq 0$. Where does your proof fail?

**Claim:** For all $n \geq 1$, \[ \sum_{i=1}^{n} i^2 = \frac{n(n+1)(2n+1)}{6}. \]

Define $F_n$ as follows:

$F_0 = 1$
$F_1 = 1$
$F_n = F_{n-1} + F_{n-2}$, for all $n \geq 2$.

```c
int fib(int n) {
    if ((n == 0) || (n == 1)) {
        return 1;
    } else {
        return fib(n-1) + fib(n-2);
    }
}
```
**Claim:** For all \( n \geq 0 \), the \( \text{fib}(n) \) returns the value \( F_n \).

Note that you must have two basis steps and must use *strong induction.* (Why?)

```c
int factorial(int n) {
    int product = 1;
    for (int i = 1; i <= n; i++) {
        product = product * i;
    }
    return product;
}
```

**Claim:** \( \text{factorial}(n) \) returns \( n! \).

**Subclaim:** Before the \( i \)th iteration of the for loop in \( \text{factorial}() \), the variable \( \text{product} \) contains the value \( i! \).

Show how the subclaim can be used to show the claim.

**Claim:** Show that a binary tree with height \( h \geq 0 \) has no more than \( 2^h \) leaves. (Warning: you can’t use the induction hypothesis if the height of the tree you are considering has a height of \( -1 \).)