CS 374: Additional notes on Induction

Chandra Chekuri
University of Illinois at Urbana-Champaign

September 3, 2015

1 Introduction

These notes are intended to supplement existing notes and references on induction available at our institution and else where. I assume that the reader is already familiar with the basics of induction. Many students have trouble with induction on non-recursively defined structures. The goal of these notes is to give some simple examples and explicitly point out some aspects in the hope that some fraction of the students may benefit.

A significant number of proofs that one encounters in discrete mathematics are of the form "*for all objects of a certain type a property P is true*". Thus this is a universally quantified statement. A simple example of such a statement is the following.

$$\forall n \in \mathbb{N}, \sum_{i=1}^{n} i = n(n+1)/2.$$

Typically the interesting case for universally quantified statements is when the quantification is over an infinite set such as the natural numbers or other discrete structures or even uncountable sets such as real numbers. Here is a statement on graphs that we will see again.

In any undirected graph the number of odd degree nodes is even.

A common reason place why universally quantified statements arise in computer science is because we want to prove that algorithms or machines or code that we develop are correct. What does it mean to prove that an algorithm $\mathcal{A}$ is correct for some computational problem. It can be stated as "*for all inputs I, the algorithm $\mathcal{A}$ on input I correctly computes the desired answer*". For instance you can think of

an algorithm for sorting a given set of numbers. Another example is an algorithm that given as input a graph $G = (V, E)$ and two nodes $s, t \in V$, checks whether s can reach t in G.

Some general techniques for proving correctness of universal statements are (i) a direct argument (ii) induction and (iii) proof by contradiction. A comment here is that coming up with a proof is not the same as writing down a proof. However, both are closely related in the sense that a systematic way to come with a proof is to try sequentially all the ways a proof could be written and try them out.

An inductive proof for a claim $\forall n, P(n)$, would typically show the following:

- Prove that the claim is true when $n = 0$, i.e., show $P(0)$ is true. This is the **base case**.
- For any arbitrary integer $k > 0$, we *assume* that P holds for all smaller numbers $i = 0, \ldots, k-1$, and then prove that P holds for k.
  In other words, for any arbitrary integer $k > 0$, we assume that $P(0), \ldots, P(k-1)$ hold, and prove $P(k)$. This is the **induction step**. The assumption is often separately stated and is called the **induction hypothesis**.

It turns out that if we argue the above, then that constitutes a valid proof that $P(n)$ holds for every $n \in \mathbb{N}$. This is the principle of induction and I will assume you have already seen the justification for this.

Note that the inductive step is itself a universally quantified statement since we have to prove it for all $k > 0$. We usually give a direct proof for it.

Here we are concerned with proving statements of the form $\forall I \in \mathcal{I}, P(I)$ where $\mathcal{I}$ is a set of structures (think graphs, tuples, programs etc) and $P(I)$ is a property about I. Readers familiar with basics of induction and the differences between direct proofs and inductive proofs can skip to Section 2 for examples on structures.

1.1 Direct versus Inductive proof via an example

Theorem 1.1.

$$\forall n \in \mathbb{N}, \sum_{i=1}^{n} i = n(n+1)/2.$$

We will first discuss a direct proof for this. Apparently the famous mathematician Gauss figured out the following proof when he was very small. We will describe the proof some what informally to keep things simple.

Proof. Let n be an *arbitrary* non-negative integer. Consider two cases.

Case 1: n is even and hence $n = 2k$ for some non-negative integer k. We are adding $2k$ numbers $1, 2, 3, \ldots, 2k$. We pair the numbers up as follows: $(1, 2k), (2, 2k-1), (3, 2k-3), \ldots, (k, k+1)$ and observe that sum of the numbers in each tuple is exactly $2k+1$ and there are k such tuples. Moreover each number in $\{1, 2, \ldots, 2k\}$ is in exactly one tuple. Hence the sum of the number is same as the sum of the numbers in the tuples which is $k(2k+1) = n(n+1)/2$.

Case 2: n is odd in which case we have $n = 2k+1$ for some non-negative integer. We can do the same pairing argument for $n' = 2k$ and then add the last number so we get that the sum is $k(2k+1) + 2k + 1 = (2k+1)(k+1) = n(n+1)/2$. □

Why is the above a direct proof? The *important* aspect is that we started out by assuming that n was an *arbitrary* non-negative integer and proved that the desired property holds for it, and hence the property must hold for all non-negative integers.

The inductive proof for the preceding theorem is perhaps *the* standard example so most of the readers have seen it. A proof by induction typically tries to show a statement of the form $\forall n \geq 0, P(n)$ where $P(n)$ is a statement that depends on n.

Proof. The claim is $\forall n \geq 0, P(n)$, where $P(n) : \sum_{i=1}^{n} i = \frac{n(n+1)}{2}$.
We will prove the claim by induction on n.

Base-case: When $n = 0$, $\sum_{i=0}^{0} i = 0$ and $\frac{n(n+1)}{2} = 0$. Hence the claim holds when $n = 0$.

Induction step:
Let $n > 0$ be an arbitrary positive number.
Assume the **Induction Hypothesis**: For every $0 \leq j < n$, $\sum_{i=0}^{j} i = \frac{j(j+1)}{2}$.
Now let us prove that $P(n)$ holds, i.e., let us prove that $\sum_{i=0}^{n} i = \frac{n(n+1)}{2}$.

$\sum_{i=0}^{n} i = (\sum_{i=0}^{n-1} i) + n$.

Now, by the induction hypothesis, since $n-1 < n$, we know $\sum_{i=0}^{n-1} i = \frac{(n-1)n}{2}$.

Hence $\sum_{i=0}^{n} i = \frac{(n-1)n}{2} + n = \frac{n^2-n+2n}{2} = \frac{n^2+n}{2} = \frac{n(n+1)}{2}$, which proves the claim for n.

Hence, by induction, we have proved the claim. □

Comments: The direct proof is elegant and revealing. In fact it allows us to derive the formula for the sum instead of merely verifying it as the induction proof does. On the other hand a direct proof is often more difficult to obtain; it usually requires a key global insight which in retrospect may look simple but is often not easy to see a priori. Inductive proofs may not be as revealing but tend to be also somewhat

easier and mechanical; in many settings this is a positive. They also allow us to systematically explore the structure of the problem to obtain a proof.

2 Induction on structures

The main purpose of this article is to explain and give examples on using induction for proofs on discrete structures such as graphs, tuples, and others where there are several choices on how to proceed and it is not always obvious which parts of the proof are mechanical and which parts require thinking. Consider the following three problems/statements.

Graphs: We will assume all graphs here are finite.

Theorem 2.1. *In any undirected graph G the number of nodes with odd degree is even.*

Strings: Consider a finite alphabet Σ and let Σ^* be the set of all strings over Σ. For a string $u \in \Sigma^*$ we can recursively define the reverse u^R of u as follows. If $|u| = 0$ which means that $u = \varepsilon$, $u^R = \varepsilon$. Otherwise $u = aw$ for some $a \in \Sigma$ and $w \in \Sigma^*$ and we define u^R as $w^R a$. Recall that for strings u, v we use uv to denote their concatenation. Here is a simple statement.

Theorem 2.2. *For all $u, v \in \Sigma^*$, $(uv)^R = v^R u^R$.*

Chocolate bar problem: Suppose you have a rectangular bar of chocolate, which has been scored into an $n \times m$ grid of squares. Consider breaking the chocolate into squares in the following way. In each round you take one of the available pieces of chocolate and break it along one of the grid lines into two smaller rectangles. Thus, at all times, each piece of chocolate is an $a \times b$ rectangle for some positive integers a and b; in particular, a 1×1 piece cannot be broken into smaller pieces. The process ends when all the pieces are individual squares.

Theorem 2.3. *Given any rectangular $n \times m$ chocolate bar, no matter the strategy, the number of rounds to break the chocolate bar into unit squares is $n \times m - 1$.*

The main point: How do we prove the above statements via induction. In each of these problems there is some interesting work needed to transform the given statement into a statement of the form $\forall n, P(n)$.

Consider the graph problem first. It is common to see inductive proofs on graphs start with the statement "By induction on m the number of edges" or "By induction on n the number of nodes". What does this mean? The key point is the following. Let Q be the statement of theorem. Consider the statement $\forall m \geq 0, P(m)$

where $P(m)$ is the statement "*every graph G with m edges has the property that the number of odd degree nodes in G is even*". It is easy to see that Q is equivalent to $\forall m \geq 0, P(m)$. Note that $P(m)$ is itself a universally quantified statement! We will get back to this point below. Thus we have translated proving Q to proving an equivalent statement which is in the standard induction template over integers. Consider the statement $\forall n \geq 0, P'(n)$ where $P'(n)$ is the statement "*every graph G with n nodes has the property that number of odd degree nodes in G is even*". Again it is easy to see that Q is equivalent to $\forall n \geq 0, P'(n)$.

More generally suppose we want to prove a statement of the form $\forall G \in \mathcal{G}, Q(G)$ where $\mathcal{G}$ is the set of all graphs (or say the set of all planar graphs) and $Q(G)$ is some property of graphs. Let $f : \mathcal{G} \to \mathbb{N}$ be any function that maps the set of graphs $\mathcal{G}$ to the non-negative integers. The statement $\forall G \in \mathcal{G}, Q(G)$ is equivalent to proving $\forall n \geq 0, P(n)$ where $P(n)$ is the statement "*forall $G \in \mathcal{G}$ with $f(G) = n$, $Q(G)$*". In general there are many different ways to map graphs to natural numbers. Some simple ones are by using number of nodes, number of edges or number of edges plus number of nodes. However there are non-trivial examples where one can use much more complex mappings. It is not a priori clear which mapping to use. It depends on the statement one is trying to prove. In general one tries the standard mappings to see which one may yield a correct proof. This is a trial and error process.

In the second and third problems we are dealing with two-dimensional problems. In the string problem the input is a pair of strings u, v. In the chocolate bar problem the input consists of n, m. In the string problem you may be able to do induction on $|u|$. What this corresponds to is a mapping f that maps (u, v) to $|u|$. You could also do induction on $|u| + |v|$. In the chocolate bar problem trying to do induction on n (or m) does not quite work but induction on nm or on the lexicographic order on tuples works. Thus, the choice of the mapping depends on the problem.

Finally we observe that the mapping f frequently collapses many objects to the same natural number n. Thus $P(n)$ becomes a universal statement over all objects that map to n. Thus, in the induction step when we wish to prove $P(1) \wedge P(2) \wedge \ldots \wedge P(n-1) \Rightarrow P(n)$ we need to establish this simultaneously for all objects that map to n. Typically one proves this via a direct argument and hence one starts with the statement: "let X be an *arbitrary* object such that $f(X)$ maps to n". For example in the setting of graphs when doing induction on number of edges we will say "let G be an arbitrary graph with m edges".

We now give the proofs.

2.1 Graph problem

There is a simple direct proof for this which is given after the proofs based on induction. Now we will describe two different proofs by induction. We restate the theorem.

Theorem. *In any undirected graph G the number of nodes with odd degree is even.*

Proof. Proof by induction on m the number of edges of the graph. Equivalently we wish to prove, $\forall m \geq 0$, every graph G with m edges has an even number of odd degree nodes.

Base-case: When $m = 0$, the graph has no edges and hence all degrees are 0. Therefore the number of odd degree nodes is 0 which is an even number.

Induction step:
Let $m > 0$ be an *arbitrary* positive number.
Assume the **Induction Hypothesis**: For every $0 \leq k < m$, in any graph with k edges the number of odd degree nodes is even.
Now let us prove that the desired claim holds for m. Let $G = (V, E)$ be an *arbitrary* graph with m edges. Let $e = uv$ be any edge in G; an edge exists since $m > 0$. Consider the graph $H = (V, E - \{e\})$ obtained from G by removing e. Since H has $m - 1$ edges, via the induction hypothesis, the number of odd degree nodes in H is even. Let $S \subseteq V$ be the set of odd degree nodes in H. We consider several cases based on whether u, v are in S or not. Note that the degree of nodes in $V \setminus \{u, v\}$ are the same in G and H.

Case 1: $u, v \in S$. This implies that in G, u and v have even degree. Thus the set of odd degree nodes in G is $S \setminus \{u, v\}$. Since $|S|$ is even, $|S| - 2$ is also even.

Case 2: $u \in S$, $v \notin S$. In G, u has even degree and v has odd degree. Thus the odd degree nodes in G is $S \cup \{v\} - \{u\}$ and hence the number of odd degree nodes is the same in G and H and $|S|$ is even.

Case 3: $v \in S$, $u \notin S$. Similar to previous case.

Case 4: $u, v \notin S$. This implies that in G, u and v have odd degree and hence the set of odd degree nodes is $S \uplus \{u, v\}$. Since $|S|$ is even $|S| + 2$ is also even.
Hence, by induction, we have proved the claim. □

Let us see another proof by induction.

Proof. Proof by induction on n the number of nodes of the graph. Equivalent we wish to prove, $\forall n \geq 0$, every graph G with n nodes has an even number of odd degree nodes.

Base-case: When $n = 1$, the graph has no edges and hence all degrees are 0. Therefore the number of odd degree nodes is 0 which is an even number.

Induction step:
Let $n > 1$ be an arbitrary positive number.
Assume the **Induction Hypothesis**: For every $0 \leq k < n$, in any graph with n nodes the number of odd degree nodes is even.
Now let us prove that the desired claim holds for n. Let $G = (V, E)$ be an *arbitrary* graph with n nodes. Let v be any node in G. Consider the graph $H = (V - \{v\}, E')$ obtained by removing v and its incident edges from G. Since H has $n - 1$ nodes, via the induction hypothesis, the number of odd degree nodes in H is even. Let $S \subseteq V - \{v\}$ be the set of odd degree nodes in H.

Let $A \subset V - \{v\}$ be the set of neighbors of v in G, that is those nodes that are connected to v by an edge. Note that $deg(v) = |A|$. Let $A_1 = A \cap S$ be those nodes that are neighbors of v in G and have odd degree in H. Let $A_2 = A \setminus S$ be the neighbors of v in G that have even degree in H.

Case 1: $deg(v)$ is odd. In G the nodes with odd degree are $S' = \{v\} \uplus (S \setminus A_1) \uplus A_2$. Therefore

$$\begin{aligned} |S'| &= 1 + |S \setminus A_1| + |A_2| \\ &= 1 + |S| - |A_1| + |A_2| = 1 + |S| + |A_1| + deg(v) - |A_1| \\ &= 1 + |S| + deg(v) - 2|A_1|. \end{aligned}$$

$1 + deg(v)$, $|S|$ and $2|A_1|$ are all even and hence $|S'|$ is even as desired.

Case 2: $deg(v)$ is even. In G the nodes with odd degree are $S' = |S \setminus A_1| \uplus A_2$. Arguing as above we see that

$$|S'| = |S| + deg(v) - 2|A_1|$$

which is even.
Hence, by induction, we have proved the claim. □

Here is a simple direct proof.

Proof. Let $G = (V, E)$ be an *arbitrary* graph. Let $S \subseteq V$ be the nodes in G with odd degree and hence $V \setminus S$ is the set of nodes in G that have even degree. Letting $deg(v)$ denote the degree of v we observe that

$$\sum_{v \in V} deg(v) = \sum_{e \in E} 2 = 2|E|$$

since each edge $e = uv$ contributes once to $deg(u)$. Thus the sum of the degrees is an even number. We write $\sum_{v \in V} deg(v) = \sum_{v \in S} deg(v) + \sum_{v \in V \setminus S} deg(v)$. Therefore

$$\sum_{v \in S} deg(v) = 2|E| - \sum_{v \in V \setminus S} deg(v).$$

Since $\sum_{v \in V \setminus S} deg(v)$ is an even number (each term in the sum is even) we have that $\sum_{v \in S} deg(v)$ is also an even number. For each $v \in S$, $deg(v)$ is odd. Thus, if the number of nodes in S is odd, we have that the sum of an odd number of odd numbers is even which is impossible. Thus $|S|$ is even. □

The two induction proofs are longer and appear less insightful than the direct proof but however it is again the case that the proofs require less magic and one can follow one's nose to work out the details.

2.2 String problem

Recall what we wished to prove.

Theorem. *For all* $u, v \in \Sigma^*$, $(uv)^R = v^R u^R$.

Proof. Proof by induction on $|u|$. Equivalently for all $n \geq 0$, for every string u of length n and every string v, $(uv)^R = v^R u^R$.

Base Case: $n = 0$. $u = \varepsilon$ and hence $(uv)^R = (\varepsilon v)^R = v^R$ as desired.

Induction step: Let u be an arbitrary string of length $n > 0$ and v be an arbitrary string. Assume the **Induction Hypothesis:** For all $0 \leq j < n$, for any string u of length j and any string v $(uv)^R = v^R u^R$.

Now we prove the claim for u, v. Since $|u| > 0$, $u = aw$ for some $a \in \Sigma$ and $w \in \Sigma^*$. Hence $uv = awv$. Therefore $(uv)^R = (a(wv))^R$. By definition of reversal $(a(wv))^R = (wv)^R a$. Now applying induction hypothesis to $(wv)^R$ since $|w| < |u|$ we have $(wv)^R = v^R w^R$. Thus

$$(uv)^R = v^R w^R a = v^R (aw)^R = v^R u^R$$

as desired. Note that we used the definition of reversal in the third step to replace $w^R a$ by $(aw)^R$. □

2.3 Chocolate bar problem

In this problem, the fact that the candy bar is made up of $n \times m$ squares is irrelevant, and it turns out to be cleaner to simply focus on the quantity $A = n \times m$, rather than on n or m. We will refer to A as area. We show that any area-A rectangular candy bar requires exactly $A - 1$ breaks until all pieces are size 1, where $A \geq 1$ is a natural number.

Proof. The proof is by induction on $A = n \times m$. Equivalently we wish to prove that $\forall A \geq 1$ any strategy to break up a rectangular chocloate bar with area A into unit square requires exactly $A - 1$ breaks.
Base Case: $A = 1$. Then the only way to break it into pieces of area 1 is to do nothing, so the strategy requires exactly $0 = A - 1$ breaks.

Inductive Step. Now consider an arbitrary rectangular candy bar of area exactly $A > 1$. Assume the **Inductive Hypothesis:** For all natural numbers $1 \leq k < A$, no matter how an area k rectangular chocolate bar is broken, exactly $k - 1$ breaks are required until all pieces have area 1.

Now consider any method of breaking the area A rectangular bar into squares of size 1. Since $A > 1$, at least one break is required, so there must be a first break, which divides the bar into two rectangular bars of size k and $A - k$, for some number k with $1 \leq k \leq A - 1$.

But then each of the smaller bars has area less than A, and so no matter how each is ultimately broken up into squares of size 1, they will require exactly $k - 1$ and $A - k - 1$ breaks, respectively. (We should note here that the steps to break one up do not affect the other.) Thus the total number of breaks including the first break is exactly $1 + (k - 1) + ((A - k) - 1) = A - 1$. □

We note that a very similar proof can also done by induction on $n + m$. Why does a proof by induction on n not work so easily? That is because if the first break may leave n the same but reduce m. Thus n may not be smaller in the two bars created by the first break. However, each break strictly reduces either n or m while keeping the other one the same. Thus any ordering of the tuples (n, m) which makes $(n - k, m)$ and $(n, m - k)$ "smaller" than (n, m) for each $k > 1$ would work. Examples include ordering by $n + m$, nm, or n^2m^2, and lexicographic ordering. Ordering based on $\max\{n, m\}$ does not work for the same reason that ordering based on n or m does not work.

3 Strengthening the induction hypothesis

As we discussed briefly induction is typically less insightful than a direct proof but it is also its strength in that it allows for easier proofs since one needs to know less about a problem to start working out a proof. However, often a direct approach to solve a problem via induction does not work because the statement one wants to prove, say Q, does not tell us much about the structure. In those cases one often needs to *strengthen* the hyptohesis. What this means is that we try to prove another statement Q' which is stronger than Q; that is Q' is a statement that implies Q but Q may not imply Q. The statement Q' captures some additional structure of the problem which enables an inductive proof to go through.

First, let us give a simple and classical example of a mathematical statement where this strengthening helps. Then we will discuss some common issues that arise in proving things about automata or algorithms.

Theorem 3.1. *For any integer* $n \geq 1$, $\sum_{i=1}^{n} \frac{1}{i^2} = 1 + \frac{1}{4} + \frac{1}{9} + \ldots + \frac{1}{n^2} < 2$.

This is hard to prove via induction. However consider the following statement which is clearly stronger than the preceding one. It is quite straightforward to prove it by induction! Try it.

Theorem 3.2. *For any integer* $n \geq 1$, $\sum_{i=1}^{n} \frac{1}{i^2} = 1 + \frac{1}{4} + \frac{1}{9} + \ldots + \frac{1}{n^2} < 2 - \frac{1}{n}$.

There are many such instances, some extremely famous, where the stronger induction hypothesis makes it very easy to prove by induction while the original statement does not admit an inductive proof.

I would like to demystify this process of strengthening the induction hypothesis by considering more mundane examples. These arise frequently in proving correctness of basic constructions in automata theory and algorithms. Suppose we come up with an algorithm $\mathcal{A}$ for sorting an array of integers. Consider the following statement which is what we desired to prove.

Theorem 3.3. *On all inputs* I *the algorithm* $\mathcal{A}$ *correctly computes its output.*

We note that the statement gives absolutely no information about the properties of the problem or the algorithm! Thus trying to prove the statement by induction (say on $|I|$ the length of the input or some other parameter) is unlikely to succeed. Strengthening here simply means that adding some specific properties of the actual algorithm into the statement so that one can prove the desired goal of showing that it behaves as it should.

For example consider a simple sorting algorithm such as Insertion Sort; see a description on Wikipedia if you are not familiar with it. (`https://en.wikipedia.`

org/wiki/Insertion_sort) The algorithm has an outer loop and an inner loop. Here is an informal statement that one can try to prove by induction after making it more formal: "after i iterations of the outer loop the last $n - i$ elements of the original array are not touched and the first i elements of the original array are sorted". This in fact exploits some property about how the algorithm was designed and one can try to prove it by induction.

Consider another example, namely the product construction on two DFAs $M_1 = (Q_1, \Sigma, \delta_1, s_1, F_1)$ and $M_2 = (Q_2, \Sigma, \delta_2, s_2, F_2)$. We design a new machine $M = (Q, \Sigma, \delta, (s_1, s_2), F)$. We assume the reader is familiar with the details product construction. Suppose we want to obtain a machince M where $L(M) = L(M_1) \cap L(M_2)$. We set $F = F_1 \times F_2$. How do we prove that $L(M) = L(M_1) \cap L(M_2)$? We need to show some useful property about M by exploiting how and why it was constructed the way it was. Otherwise we have very little handle on a proof. It was built to *simulate* the behaviour of M_1 and M_2 concurrently by keeping track of where M_1 and M_2 would be on any given input w. The following claim captures the behaviour of M and we can try to prove this by induction.

Theorem 3.4. *For any string $w \in \Sigma^*$, $\delta^*((s_1, s_2), w) = (\delta_1^*(s_1, w), \delta_2^*(s_2, w))$.*

In the preceding theorem δ^* is the closure of δ that extends it from one character transition function to a transition function over strings. We prove the above by induction on $|w|$ and it precisely captures what M was designed for. Once we have the above theorem proving correctness of the behaviour of M becomes rather easy.

Acknowledgments: These notes grew out of discussions with several people in the UIUC theory group. Thanks to Madhu for discussion and proofreading.