COMP9020

Foundations of Computer Science

Lecture 10: Induction

Administrivia

- Assignment 1 due on Friday at 12 noon
  - Late penalty: 5 marks per 24 hours or part thereof for maximum 5 days.
  - Contact me if you need an extension – include zId and length of requested extension
- Assignment 2 released next week
- Feedback:
  - More in-class quizzes
  - More offline consultations
  - Quiz solutions

Outline

Outline

Recursive datatypes

Describe arbitrarily large objects in a finite way

Recursive functions

Define behaviour for these objects in a finite way

Induction

Reason about these objects in a finite way

Example

Recall the recursive program:

Example

Summing the first n natural numbers:

$$\begin{aligned}&\texttt{sum}(n)\texttt{:}\\&\quad \texttt{if}(n = 0)\texttt{: } 0\\&\quad \texttt{else: } n + \texttt{sum}(n-1)\end{aligned}$$

Another attempt:

Example

$$\begin{aligned}&\texttt{sum2}(n)\texttt{:}\\&\quad \texttt{return } n * (n+1)/2\end{aligned}$$

Induction proof **guarantees** that these programs will behave the same.

Inductive Reasoning

Suppose we would like to reach a conclusion of the form

$$P(x) \textit{ for all } x \text{ (of some type)}$$

Inductive reasoning (as understood in philosophy) proceeds from examples.

E.g. From "This swan is white, that swan is white, in fact every swan I have seen so far is white"

Conclude: "Every Swan is white"

NB

This may be a good way to discover hypotheses.
But it is not a valid principle of reasoning!

Mathematical induction *is a variant that is valid.*

Mathematical Induction

Mathematical Induction is based not just on a set of examples, but also a rule for deriving new cases of $P(x)$ from cases for which P is known to hold.
General structure of reasoning by mathematical induction:

Base Case [B]: $P(a_1), P(a_2), \ldots, P(a_n)$ for some small set of examples $a_1 \ldots a_n$ (often $n = 1$)
Inductive Step [I]: A general rule showing that if $P(x)$ holds for some cases $x = x_1, \ldots, x_k$ then $P(y)$ holds for some new case y, constructed in some way from $x_1, \ldots, x_k$.

Conclusion: Starting with $a_1 \ldots a_n$ and repeatedly applying the construction of y from existing values, we can eventually construct all values in the domain of interest.

Induction proof structure

Let $P(x)$ be the proposition that ...
We will show that $P(x)$ holds for all x by induction on x.

Base case: $x = ...$**:**

- $P(x)$: ...
- ....
- so $P(x)$ holds.

[Repeat for all base cases]

Inductive case: $P(x)$ **implies** $P(y)$

- Assume $P(x)$ holds. That is, ...
- We will show $P(y)$ holds.
- ...
- So $P(x)$ implies $P(y)$.

[Repeat for all inductive cases]

Conclusion

We have shown $P(...)$, and for all x, $P(x)$ implies $P(y)$. Therefore, by induction, $P(x)$ holds for all x.

Outline

Basic induction

Basic induction is the general principle applied to the natural numbers.

Goal: Show $P(n)$ holds for all $n \in \mathbb{N}$.

Approach: Show that:

Base case (B): $P(0)$ holds; and

Inductive case (I): If $P(k)$ holds then $P(k+1)$ holds.

Conclusion (C): $P(n)$ holds for all $n \in \mathbb{N}$.

Example

Recall the recursive program:

Example

Summing the first n natural numbers:

$$\begin{aligned}&\texttt{sum}(n)\texttt{:}\\&\quad\texttt{if}(n = 0)\texttt{: } 0\\&\quad\texttt{else: } n + \texttt{sum}(n-1)\end{aligned}$$

Another attempt:

Example

$$\begin{aligned}&\texttt{sum2}(n)\texttt{:}\\&\quad\texttt{return } n * (n+1)/2\end{aligned}$$

Induction proof **guarantees** that these programs will behave the same.

Example

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

We will show that $P(n)$ holds for all $n \in \mathbb{N}$ by induction on n.

[B] Base case: $n = 0$:

$$\left(\sum_{i=0}^{0} i\right) = 0 = \frac{0(0+1)}{2}.$$

So $P(0)$ holds.

[I] Inductive case: $P(k) \Rightarrow P(k+1)$:

That is,

$$\sum_{i=0}^{k} i = \frac{k(k+1)}{2} \quad \Rightarrow \quad \sum_{i=0}^{k+1} i = \frac{(k+1)(k+2)}{2}$$

(proof?)

[C] Conclusion: We have $P(0)$ is true, and $P(k)$ implies $P(k+1)$. Therefore, by induction, $P(n)$ holds for all $n \in \mathbb{N}$.

Proof of inductive step.

Assume $P(k)$ holds. That is:

$$\left(\sum_{i=0}^{k} i\right) = \frac{k(k+1)}{2} + (k+1). \qquad \text{(IH)}$$

Then:

$$\begin{aligned}
\sum_{i=0}^{k+1} i &= \left(\sum_{i=0}^{k} i\right) + (k+1) \\
&= \frac{k(k+1)}{2} + (k+1) \qquad \text{(by the inductive hypothesis)} \\
&= \frac{k(k+1) + 2(k+1)}{2} \\
&= \frac{(k+1)(k+2)}{2}
\end{aligned}$$

Therefore $P(k)$ implies $P(k+1)$. □

Outline

Variations

There are many variants of basic induction that may be more useful in certain circumstances. For example:

1. Induction from m upwards
2. Induction steps >1
3. Strong induction
4. Backward induction
5. Forward-backward induction
6. Structural induction

Induction From m Upwards

If

[B] $P(m)$

[I] $\forall k \geq m\,(P(k) \rightarrow P(k+1))$

then

[C] $\forall n \geq m\,(P(n))$

Example

Theorem. For all $n \geq 1$, the number $8^n - 2^n$ is divisible by 6.

[B] $8^1 - 2^1$ is divisible by 6

[I] if $8^k - 2^k$ is divisible by 6, then so is $8^{k+1} - 2^{k+1}$, for all $k \geq 1$

Prove [I] using the "trick" to rewrite 8^{k+1} as $8 \cdot (8^k - 2^k + 2^k)$ which allows you to apply the IH on $8^k - 2^k$

Induction Steps $\ell > 1$

If

[B] $P(m)$

[I] $P(k) \to P(k+\ell)$ for all $k \geq m$

then

[C] $P(n)$ for every ℓ'th $n \geq m$

Example

Every 4th Fibonacci number is divisible by 3.

[B] $F_4 = 3$ is divisible by 3
[I] if $3 \mid F_k$, then $3 \mid F_{k+4}$, for all $k \geq 4$

Prove [I] by rewriting F_{k+4} in such a way that you can apply the IH on F_k

Strong Induction

This is a version in which the inductive hypothesis is stronger. Rather than using the fact that $P(k)$ holds for a single value, we use *all* values up to k.

If

[B] $P(m)$

[I] $[P(m) \wedge P(m+1) \wedge \ldots \wedge P(k)] \rightarrow P(k+1)$ for all $k \geq m$

then

[C] $P(n)$, for all $n \geq m$

Example

Claim: All integers ≥ 2 can be written as a product of primes.

[B] 2 is a product of primes

[I] If all x with $2 \leq x \leq k$ can be written as a product of primes, then $k+1$ can be written as a product of primes, for all $k \geq 2$

Proof for [I]?

Negative Integers, Backward Induction

NB

Induction can be conducted over any subset of $\mathbb{Z}$ with least element. Thus m can be negative; eg. base case $m = -10^6$.

NB

One can apply induction in the 'opposite' direction $p(m) \to p(m-1)$. It means considering the integers with the opposite ordering where the next number after n is $n-1$. Such induction would be used to prove some $p(n)$ for all $n \leq m$.

NB

Sometimes one needs to reason about all integers $\mathbb{Z}$. This requires two separate simple induction proofs: one for $\mathbb{N}$, another for $-\mathbb{N}$. They both would start form some initial values, which could be the same, e.g. zero. Then the first proof would proceed through positive integers; the second proof through negative integers.

Forward-Backward Induction

Idea

To prove $P(n)$ *for all* $n \geq k_0$

- *verify* $P(k_0)$
- *prove* $P(k_i)$ *for infinitely many* $k_0 < k_1 < k_2 < k_3 < \ldots$
- *fill the gaps*
  $P(k_1) \to P(k_1 - 1) \to P(k_1 - 2) \to \ldots \to P(k_0 + 1)$
  $P(k_2) \to P(k_2 - 1) \to P(k_2 - 2) \to \ldots \to P(k_1 + 1)$
  $\ldots\ldots\ldots$

NB

This form of induction is extremely important for the analysis of algorithms.

Outline

Structural Induction

Basic induction allows us to assert properties over **all natural numbers**. The induction scheme (layout) uses the recursive definition of $\mathbb{N}$.

The induction scheme can be applied not only to natural numbers (and integers) but to any partially ordered set in general – especially those defined recursively.

The basic approach is always the same — we need to verify that

- **[B]** the property holds for all minimal objects — objects that have no predecessors; they are usually very simple objects allowing immediate verification
- **[I]** for any given object, if the property in question holds for *all* its predecessors ('smaller' objects) then it holds for the object itself

Example: Induction on Σ^*

Recall definition of Σ^*:

$\lambda \in \Sigma^*$

If $w \in \Sigma^*$ then $aw \in \Sigma^*$ for all $a \in \Sigma$

Structural induction on Σ^*:

Goal: Show $P(w)$ holds for all $w \in \Sigma^*$.

Approach: Show that:

Base case (B): $P(\lambda)$ holds; and

Inductive case (I): If $P(w)$ holds then $P(aw)$ holds for all $a \in \Sigma$.

Example: Induction on Σ^*

Recall:

Formal definition of Σ^*:

$\lambda \in \Sigma^*$

If $w \in \Sigma^*$ then $aw \in \Sigma^*$ for all $a \in \Sigma$

Formal definition of concatenation:

(concat.B) $\lambda v = v$

(concat.I) $(aw)v = a(wv)$

Formal definition of length:

(length.B) $\text{length}(\lambda) = 0$

(length.I) $\text{length}(aw) = 1 + \text{length}(w)$

Prove:

$$\text{length}(wv) = \text{length}(w) + \text{length}(v)$$

Example: Induction on Σ^*

Let $P(w)$ be the proposition that, for all $v \in \Sigma^*$:

$$\text{length}(wv) = \text{length}(w) + \text{length}(v).$$

We will show that $P(w)$ holds for all $w \in \Sigma^*$ by **structural induction on w.**

Proof:
Base case ($w = \lambda$):

$$\begin{aligned} \text{length}(\lambda v) &= \text{length}(v) && \text{(concat.B)} \\ &= 0 + \text{length}(v) \\ &= \text{length}(w) + \text{length}(v) && \text{(length.B)} \end{aligned}$$

Example: Induction on Σ^*

Proof cont'd:
Inductive case ($w = aw'$): Assume that $P(w')$ holds. That is, for all $v \in \Sigma^*$:

$$\text{(IH):} \qquad \text{length}(w'v) = \text{length}(w') + \text{length}(v).$$

Then, for all $a \in \Sigma$, we have:

$$\begin{aligned}
\text{length}((aw')v) &= \text{length}(a(w'v)) && \text{(concat.I)} \\
&= 1 + \text{length}(w'v) && \text{(length.I)} \\
&= 1 + \text{length}(w') + \text{length}(v) && \text{(IH)} \\
&= \text{length}(aw') + \text{length}(v) && \text{(length.I)}
\end{aligned}$$

So $P(aw')$ holds.

We have $P(\lambda)$ and for all $w' \in \Sigma^*$ and $a \in \Sigma$: $P(w') \to P(aw')$.
Hence $P(w)$ holds for all $w \in \Sigma^*$.

Example 2: Induction on Σ^*

Recall append : $\Sigma^* \times \Sigma \to \Sigma^*$ defined as:

- $\text{append}(\lambda, x) = x$
- $\text{append}(aw, x) = a\,(\text{append}(w, x))$

Prove:

For all $w, v \in \Sigma^*$ and $x \in \Sigma$:

$$\text{append}(wv, x) = w(\text{append}(v, x))$$

Example 2: Induction on Σ^*

Theorem

For all $w, v \in \Sigma^*$ and $x \in \Sigma$: $append(wv, x) = w(append(v, x))$.

Proof: By induction on w...

[B]	$\text{append}(\lambda v, x) = \text{append}(v, x)$	(concat.B)
[I]	$\text{append}((aw)v, x) = \text{append}(a(wv), x)$	(concat.I)
	$= a\,\text{append}(wv, x)$	(append.I)
	$= a\,(w\,\text{append}(v, x))$	(IH)
	$= (aw)\,\text{append}(v, x)$	(concat.I)

Example 3: Induction on Σ^*

Define $\mathrm{rev} : \Sigma^* \to \Sigma^*$:

(rev.B) $\mathrm{rev}(\lambda) = \lambda$,

(rev.I) $\mathrm{rev}(a \cdot w) = \mathrm{append}(\mathrm{reverse}(w), a)$

Example 3: Induction on Σ^*

Theorem

For all $w, v \in \Sigma^*$, $reverse(wv) = reverse(v) \cdot reverse(w)$.

Proof: By induction on w...

[B]

$$\begin{aligned} \mathrm{rev}(\lambda v) &= \mathrm{rev}(v) && \text{(concat.B)} \\ &= \mathrm{rev}(v)\lambda && (*) \\ &= \mathrm{rev}(v)\mathrm{rev}(\lambda) && \text{(rev.B)} \end{aligned}$$

[I]

$$\begin{aligned} \mathrm{rev}((aw')v) &= \mathrm{rev}(a(w'v)) && \text{(concat.I)} \\ &= \mathrm{append}(\mathrm{rev}(w'v), a) && \text{(rev.I)} \\ &= \mathrm{append}(\mathrm{rev}(v)\mathrm{rev}(w'), a) && \text{(IH)} \\ &= \mathrm{rev}(v)\mathrm{append}(\mathrm{rev}(w'), a) && \text{(Example 2)} \\ &= \mathrm{rev}(v)\mathrm{rev}(aw') && \text{(rev.I)} \end{aligned}$$

Example 4: Induction on more complex structures

Recall expressions in the Proof assistant:

- (B) $A, B, \ldots, Z, a, b, \ldots z$ are expressions
- (B) $\emptyset$ and $\mathcal{U}$ are expressions
- (R) If E is an expression then so is (E) and E^c
- (R) If E_1 and E_2 are expressions then:
  - $(E_1 \cup E_2)$,
  - $(E_1 \cap E_2)$,
  - $(E_1 \setminus E_2)$, and
  - $(E_1 \oplus E_2)$ are expressions.

Theorem

In any valid expression, the number of (*equals the number of*)

Proof: By induction on the structure of E...

Exercise

Exercise

RW: 4.4.2 Define $s_1 = 1$ and $s_{n+1} = \frac{1}{1+s_n}$ for $n \geq 1$

Then $s_1 = 1,\ s_2 = \frac{1}{2},\ s_3 = \frac{2}{3},\ s_4 = \frac{3}{5},\ s_5 = \frac{5}{8}, \ldots$

The numbers in numerator and denominator remind one of the Fibonacci sequence.

Prove by induction that

$$s_n = \frac{\mathrm{FIB}(n)}{\mathrm{FIB}(n+1)}$$

Outline

Weekly Feedback

I would appreciate any comments/suggestions/requests you have on this week's lectures.

https://forms.office.com/r/xKKrxYMRn9