How to Use Structural Induction

Guidelines and Examples for Writing Proofs

CS 6520, Spring 2000

Detecting the Need for Structural Induction

Structural induction applies when proving a claim about a recursively-defined set. The set is well-founded, of course, only if there are atomic elements in the definition; those elements constitute the base case for induction proofs. The self-referential elements of the definition constitute the induction cases.

For example, suppose we have the following definition:

$$\begin{array}{rcl} L & = & \alpha \\ & | & (\beta \otimes L) \\ & | & (L \odot L) \end{array}$$

The above BNF grammar defines a set of symbol patterns. Example members of the set include α, $(\beta \otimes \alpha)$, $(\alpha \odot \alpha)$, and $((\beta \otimes (\beta \otimes \alpha)) \odot (\beta \otimes \alpha))$. We use L, sometimes with primes or subscripts, to stand for a particular element of the set. (Consistent use of a prime or subscript in a particular context indentifies a single element, and different primes or subscripts identify potentially different elements.)

Here is a claim about elements of the set:

Theorem 1: For any L, L contains an equal number of βs and $\otimes$s.

The truth of this claim is obvious. But since Theorem 1 makes a claim about all possible instances of L, and since the set of Ls is recursively defined, it should be proved formally by structural induction.

To prove a claim about all elements of a recursively-defined set, use structural induction.

The key feature of an induction proof is that it is guaranteed to cover an entire set, such as the set of Ls. Here is an example proof:

Proof for Theorem 1: By induction on the structure of L.

- Base cases:
  - **Case** α
    α contains 0 βs and 0 $\otimes$s, so the claim holds.
- Inductive cases:
  - **Case** $(\beta \otimes L)$
    By induction, since L is a substructure of $(\beta \otimes L)$, L contains an equal number — say, n — of βs and $\otimes$s. Therefore, $(\beta \otimes L)$ contains $n + 1$ βs and $\otimes$s, and the claim holds.
  - **Case** $(L_1 \odot L_2)$
    By induction, L_1 contains an equal number — say, n_1 — of βs and $\otimes$s. Similarly, L_2 contains n_2 βs and $\otimes$s. Therefore, $(L_1 \odot L_2)$ contains $n_1 + n_2$ βs and $\otimes$s, and the claim holds.

The above proof has a standard shape. The introduction, "by induction on the structure of L" encapsulates a boilerplate argument, which runs as follows:

> The claim must hold for any L. So assume that an arbitrary L is provided. If L has the shape of a base case (i.e., no substructures that are Ls) then we show how we can deduce the claim immediately. Otherwise, we rely on induction and assume the claim for substructures within L, and then deduce the claim. The claim then holds for all L by the principle of induction.

The proof for Theorem 1 contains a single base case because the definition of L contains a single case that is not self-referential. The proof contains two inductive cases because the definition of L contains two self-referential cases.

For a structural induction proof, use the standard format. The section for base cases should contain one case for each base case in the set's definition. The section for induction cases should contain one case for each remaining case in the set's definition.

Occasionally, within the section for base cases or induction cases in a proof, the case split uses some criteria other than the structure used for induction. This is ok as long as the case-split merely collapses deeply-nested cases, as compared to a standard-format proof. A non-standard case-split must cover all of the base or inductive cases in an obvious way.

All proofs over the structure of L, including Theorem 1, have the same outline:

Proof for Theorem $\cdots$: By induction on the structure of L.

- Base cases:
  - **Case** α
    ...
- Inductive cases:
  - **Case** $(\beta{\otimes}L)$
    $\cdots$ By induction, [claim about L] ...
  - **Case** $(L_1{\odot}L_2)$
    $\cdots$ By induction, [claim about L_1] and, by induction, [claim about L_2] $\cdots$

Only the details hidden by "$\cdots$" depend on the claim being proved. The following claim is also about all Ls, so it will have the same outline as above.

Theorem 2: For any L, L contains at least one α.

The proof of Theorem 2 is an exercise for the reader.

Definitions with Ellipses

Beware of definitions that contain ellipses (or asterisks), especially when "..." (or "*") means "*zero* or more". Such shorthands often conceal a combination of atomic and self-referential elements. For example,

$$\begin{array}{rcl} W & = & \alpha \\ & | & (\beta W \ldots) \end{array}$$

can be re-written as

$$
\begin{array}{rcl}
W & = & \alpha \\
 & | & (\beta) \\
 & | & (\beta W W \ldots)
\end{array}
$$

to expose the fact that W contains two atomic cases, α and β, and one self-referential case.

Theorem 3: For any W, each β in W is preceded by an open parenthesis.

The proof of Theorem 3 is an exercise for the reader.

Non-BNF Definitions

A recursively-defined set need not be expressed in BNF form. For example, suppose that we want $\triangle L$ to mean that L is "pointy", defined as follows:

$$
\begin{array}{ll}
\triangle \alpha & \text{[always]} \\
\triangle (L_1 \odot L_2) & \text{if } \triangle L_1 \text{ and } \triangle L_2
\end{array}
$$

The set of $\triangle L$s is defined as the smallest set that satisfies the above rules. The first rules defines a base case, while the second one is self-referential.

Alternatively, we might use the kind of notation that is used in type theory. The following defines "pointy derivation trees":

$$
\frac{\surd}{\triangle \alpha} \qquad \frac{\triangle L_1 \quad \triangle L_2}{\triangle (L_1 \odot L_2)}
$$

When $\triangle L$ does not appear below a line, we understand it to mean that there must be a pointy derivation tree with $\triangle L$ at the bottom. This convention is used even within the definition of pointy derivation trees, in the self-referential second rule.

The type-theory style of notation simultaneously defines two sets: the set of pointy indications $\triangle L$, and the set of pointy derivation trees. Nevertheless, both sets are simply patterns of symbols, defined recursively. Examples of pointy derivation trees include the following:

$$
\frac{\surd}{\triangle \alpha} \qquad \frac{\dfrac{\surd}{\triangle \alpha} \quad \dfrac{\dfrac{\surd}{\triangle \alpha} \quad \dfrac{\surd}{\triangle \alpha}}{\triangle (\alpha \odot \alpha)}}{\triangle (\alpha \odot (\alpha \odot \alpha))}
$$

We can now write claims about $\triangle L$ and its pointy derivation trees:

Theorem 4: If $\triangle L$, then the pointy derivation tree for $\triangle L$ contains an odd number of $\triangle$s.

The proof for this claim works just like the proof of a claim on L, by structural induction:

Proof for Theorem 4: By induction on the structure of the pointy derivation of $\triangle L$.

- Base cases:
  - **Case** $\triangle \alpha$
    The complete tree contains $\triangle \alpha$, a line, and a check mark, which is one $\triangle$ total, so the claim holds.
- Inductive cases:

– **Case** $\triangle(L_1 \odot L_2)$
  The complete tree contains $\triangle(L_1 \odot L_2)$, plus trees for $\triangle L_1$ and $\triangle L_2$. By induction, since the tree for $\triangle L_1$ is a substructure of of the tree for $\triangle(L_1 \odot L_2)$, the $\triangle L_1$ tree contains an odd number of $\triangle$s. Similarly, the $\triangle L_2$ tree contains an odd number of $\triangle$s. Two odd numbers sum to an even number, so the trees for $\triangle L_1$ and $\triangle L_2$ combined contain an even number of $\triangle$s. But the complete tree for $\triangle(L_1 \odot L_2)$ adds one more, giving an odd number of $\triangle$s total, and the claim holds.

Multiple Structures

Many useful claims about recursively-defined sets involve a connection between two different sets. For example, consider the following claim:

Theorem 5: For all $\triangle L$, L contains no βs.

Should this claim be proved by structural induction on L, or on $\triangle L$? We can find the answer by looking closely at what the theorem lets us assume. The theorem starts "for all $\triangle L$", which means the proof must consider all possible cases of $\triangle L$. The proof should therefore proceed by induction on the structure of $\triangle L$. Then, in each case of $\triangle L$, the goal is to show something about a specific L, already determined by the choice of $\triangle L$.

> A leading "for all" in a claim indicates a candidate for structural induction. An item in the claim that appears on the right-hand side of an implication is *not* a candidate.

In contrast to Theorem 5, Theorem 4 was stated "if $\triangle L$, then ... ". This "if ... then" pattern is actually serving as a form of "for all". When in doubt, try to mentally restate the theorem in terms of "for all".

Proof for Theorem 5: By induction on the structure of the proof of $\triangle L$.

- Base cases:
  – **Case** $\triangle\alpha$
    In this case, L is α, and the claim holds.
- Inductive cases:
  – **Case** $\triangle(L_1 \odot L_2)$
    By induction, L_1 and L_2 each contain no βs, so $(L_1 \odot L_2)$ contains no βs.

Here is a related, but different claim:

Theorem 6: For all L, either 1) L contains a β, or 2) $\triangle L$.

The proof of Theorem 6 is an exercise for the reader. Note that Theorem 6 must be proved over a different structure than Theorem 5.

More Definitions and More Proofs

We can define more sets and make claims about their relationships. The proofs of those claims will follow in the same way as the ones we have seen above, differing only in the structure used for induction, and the local reasoning in each case.

Here is a new set:

$$\begin{array}{lcll}
((\beta \otimes \alpha) \odot \alpha) & \diamond & (\beta \otimes \alpha) & \\
(\alpha \odot (\beta \otimes \alpha)) & \diamond & \alpha & \\
(\alpha \odot \alpha) & \diamond & \alpha & \\
(L_1 \odot L_2) & \diamond & (L_1' \odot L_2) & \text{if } L_1 \diamond L_1' \\
(L_1 \odot L_2) & \diamond & (L_1 \odot L_2') & \text{if } L_2 \diamond L_2'
\end{array}$$

Like our original definition for $\triangle L$, $L \diamond L$ is defined by the least set satisfying the above rules. Examples for this set include $((\beta \otimes \alpha) \odot \alpha) \diamond (\beta \otimes \alpha)$ and $(((\beta \otimes \alpha) \odot \alpha) \beta (\alpha \odot \alpha)) \diamond ((\beta \otimes \alpha) \beta (\alpha \odot \alpha))$. The atomic elements of the definition in this case are the first three rules, and the self-reference occurs in the last two rules.

Theorem 7: For all $L \diamond L'$, L contains more $\odot$s then L'.

Proof for Theorem 7: By induction on the structure of $L \diamond L'$.

- Base cases:
  - **Case** $((\beta \otimes \alpha) \odot \alpha) \diamond (\beta \otimes \alpha)$
    In this case, L is $((\beta \otimes \alpha) \odot \alpha)$, which has one $\odot$, and L' is $(\beta \otimes \alpha)$, which has zero $\odot$s, so the claim holds.
  - **Case** $(\alpha \odot (\beta \otimes \alpha)) \diamond \alpha$
    In this case, L is $(\alpha \odot (\beta \otimes \alpha))$, which has one $\odot$, and L' is α, which has zero $\odot$s, so the claim holds.
  - **Case** $(\alpha \odot \alpha) \diamond \alpha$
    In this case, L is $(\alpha \odot \alpha)$, which has one $\odot$, and L' is α, which has zero $\odot$s, so the claim holds.
- Inductive cases:
  - **Case** $(L_1 \odot L_2) \diamond (L_1' \odot L_2)$
    In this case, L is $(L_1 \odot L_2)$, which means it has as many $\odot$s as L_1 and L_2 combined. L' is $(L_1' \odot L_2)$, which means it has as many $\odot$s as L_1' and L_2 combined. By induction, L_1 has more $\odot$s than L_1', so L has more $\odot$s than L', and the claim holds.
  - **Case** $(L_1 \odot L_2) \diamond (L_1 \odot L_2')$
    Analogous to the previous case.

Yet another definition:

$$\begin{array}{rcl}
V & = & \alpha \\
 & \mid & (\beta \otimes V)
\end{array}$$

Theorem 8: Every V is an L.

Theorem 9: If $\triangle L$ and L is not a V, then $L \diamond L'$ for some L'

Theorem 10: If $\triangle L$ and $L \diamond L'$, then $\triangle L'$.

The proofs of Theorems 8, 9, and 10 are exercises for the reader. Note that the proof of Theorem 10 can proceed in two different ways, since the implicit “for all” applies to both $\triangle L$ and $L \diamond L'$.