

4.3 Induction and Recursion on Formulas

Section 4.2 proved statements about every natural number. This section proves statements about every formula in the same way, and with them answers the question Section 4.1 left open, whether every formula is read in only one way, and with it a second: does a definition with one clause per connective define exactly one thing? On the way it shows what goes wrong when an object can be built in two ways.

4.3.1 Structural induction

We return to formulas. The formation sequence of a formula plays the part that “0, 1, 2, ...” plays for numbers: every formula is reached from atoms and $\perp$ by finitely many uses of the clauses. So a property of formulas can be proved in the same two parts.

A **property of formulas** is a statement $P(\varphi)$ about a formula φ , true or false for each particular formula: “ φ has as many left as right parentheses”, “ φ contains the atom p ”.

If every atom and $\perp$ has a property, and each way of building a formula passes the property from the parts to the whole, then every formula has it.

Theorem 4.3.1. Structural Induction

Let $P(\varphi)$ be a property of formulas. Suppose that

- (i) $P(p)$ holds for every atom p , and $P(\perp)$ holds;
- (ii) **for every** formula φ , if $P(\varphi)$ holds, then $P(\neg\varphi)$ holds;
- (iii) **for all** formulas φ, ψ and **each** binary connective $\circ$, if $P(\varphi)$ and $P(\psi)$ hold, then $P((\varphi \circ \psi))$ holds.

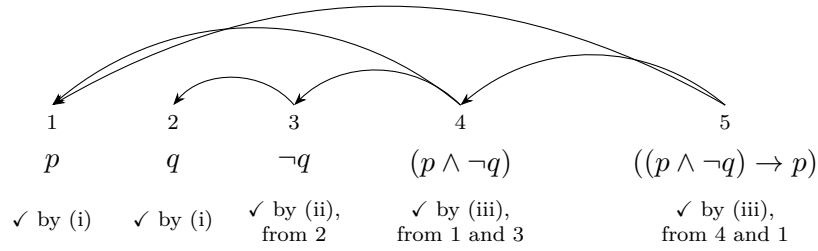
Then $P(\varphi)$ holds for every formula φ .

In words: (i) is the base case, one for each kind of formula of length one. (ii) and (iii) are the inductive steps, one for each building clause of Definition 4.1.1, and inside them “ $P(\varphi)$ and $P(\psi)$ ” is the induction hypothesis, assumed for the **immediate parts** of the formula being built. There are seven cases in all (atoms, $\perp$, $\neg$, and the four binary connectives); often the four binary ones are handled at once, with $\circ$ left unspecified.

Section 4.2’s checklist reads: objects, the formulas; property, $P(\varphi)$, with any restriction inside it; base cases, every atom and $\perp$; steps, one for $\neg$ and one for each binary connective; principle, [Theorem 4.3.1](#).

For example, take the property “ φ has as many left as right parentheses”. *Base cases*: an atom and $\perp$ have no parentheses, none of either. *Step for $\neg$* : if φ has k of each, so has $\neg\varphi$, since $\neg$ is not a parenthesis. *Step for $\circ$* : if φ has k of each and ψ has l of each, then $(\varphi \circ \psi)$ has $k + l + 1$ of each, the outer pair adding one of each. By [Theorem 4.3.1](#), every formula has as many left as right parentheses. This is part (a) of [Lemma 4.3.2](#) below; its other parts need more care.

Idea. Take any formula and a formation sequence ending in it. Walk along the sequence: each entry is an atom or $\perp$, which has the property by (i), or is built from **earlier** entries, which have it already, by (ii) or (iii). “Earlier entries, any of them” is exactly the shape of strong induction, applied to the position in the list.



The property spreads from left to right. Each entry gets its check mark from (i), or from entries to its left that already have one.

The proof, in the fold below, makes “from left to right” exact with strong induction on the position in the list.

🕒 **Second reading** · *The proof of structural induction*

Proof. Suppose (i)–(iii) hold, and let χ be a formula. By Definition 4.1.1 (F4) there is a formation sequence $\chi_1, \dots, \chi_m$ with $\chi_m = \chi$. For $i \in \mathbb{N}$, let $Q(i)$ be the property “if $1 \leq i \leq m$, then $P(\chi_i)$ holds”. We prove $Q(i)$ for every $i \in \mathbb{N}$ by Theorem 4.2.2.

Let $i \in \mathbb{N}$, and suppose $Q(k)$ holds for every $k \in \mathbb{N}$ with $k < i$. If $i = 0$ or $i > m$, then $Q(i)$ holds vacuously. So let $1 \leq i \leq m$. By the definition of a formation sequence, one of three things is true.

- χ_i is an atom or $\perp$. Then $P(\chi_i)$ holds by (i).
- $\chi_i = \neg\chi_j$ for some $j < i$. Then $1 \leq j \leq m$, and $Q(j)$ gives $P(\chi_j)$. The entry χ_j is a formula (every entry of a formation sequence is one, Section 4.1), so (ii) applies and gives $P(\chi_i)$.
- $\chi_i = (\chi_j \circ \chi_l)$ for some $j, l < i$ and a binary connective $\circ$. Then $1 \leq j, l \leq m$, and $Q(j)$ and $Q(l)$ give $P(\chi_j)$ and $P(\chi_l)$, both entries are formulas, and (iii) gives $P(\chi_i)$.

So $Q(i)$ holds. By Theorem 4.2.2, $Q(i)$ holds for every $i \in \mathbb{N}$; in particular $Q(m)$ gives $P(\chi_m)$, that is, $P(\chi)$. Since χ was an arbitrary formula, this proves the theorem. ■

The proof uses **strong** induction, because an entry of a formation sequence may be built from any earlier entries, not only from the one just before it.

Remark. A tempting alternative is induction on the **height** of a formula (the number of nodes on a longest branch of its parse tree, defined later in this section). That route is circular here. The height is read off the parse tree, and that a formula has only one parse tree is proved below from unique readability, whose proof uses structural induction. Inducting on the position in a formation sequence needs nothing but Definition 4.1.1.

Warning. A structural induction must cover **every** clause, the base case $\perp$ and the step for $\neg$ included. Try to prove “every formula contains an atom”: it holds for atoms, and the steps for $\neg$ and for $(\varphi \circ \psi)$ keep it, since the atoms of the parts are atoms of the whole. But the case $\perp$ fails, and so does the claim: $\perp$, $\neg\perp$ and $(\perp \rightarrow \perp)$ contain no atom. A proof that “forgets” $\perp$ proves something false.

Quick check. Which clause of Definition 4.1.1 makes step (ii) of [Theorem 4.3.1](#) necessary? Give a property of formulas that holds for every atom and for $\perp$, passes from φ and ψ to $(\varphi \circ \psi)$ for every binary connective, and yet fails for some formula.

Solution. Clause (F2), which builds $\neg\varphi$ from φ : step (ii) is the step for that clause. Without it, the formulas made by (F2) would be missed.

One such property: “ φ is an atom, or is $\perp$, or begins with (”. Atoms and $\perp$ have it, and every formula $(\varphi \circ \psi)$ begins with (, whatever its parts are. But $\neg p$ does not have it. So (i) and (iii) alone do not prove a property of every formula; the property fails exactly at a formula made by the forgotten clause.

🕒 **Second reading** · *Sets built from a base*

Section 4.1 called the formulas the **smallest** collection of strings closed under (F1)–(F3). Here is that sentence made exact.

Call a collection C of strings **closed** if it contains every atom and $\perp$, contains $\neg s$ whenever it contains s , and contains $(s \circ t)$ whenever it contains s and t , for each binary connective $\circ$. The collection of all strings is closed, and closed collections can be much larger than we want.

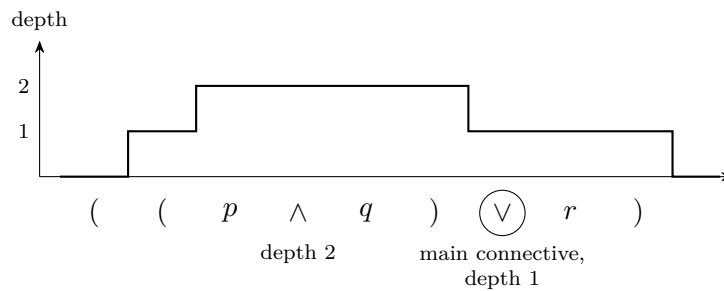
- (i) *Every closed collection C contains every formula.* Given a formula and a formation sequence $\chi_1, \dots, \chi_m$ ending in it, show “if $1 \leq i \leq m$, then $\chi_i \in C$ ” for every $i \in \mathbb{N}$ by [Theorem 4.2.2](#), case by case as in the proof of [Theorem 4.3.1](#): each entry is an atom or $\perp$, or is built by one clause from earlier entries, which are in C already. The last entry is the formula.
- (ii) *The formulas form a closed collection*, by (F1)–(F3).

So the formulas are the smallest closed collection: contained in every closed collection, and one of them. Structural induction is (i) applied to one particular collection. If a property P satisfies (i)–(iii) of [Theorem 4.3.1](#), the collection of formulas that have P is closed: it contains every atom and $\perp$ by (i), and the formulas built from members of it are formulas with P , by (ii) and (iii). So it contains every formula. The top-down description (“the smallest closed collection”) and the bottom-up one (“whatever a formation sequence reaches”) pick out the same strings. The word “collection” keeps its everyday sense here.

4.3.2 Counting parentheses: the depth profile

Section 4.1 met the string $p \wedge q \vee r$, which can be read as $(p \wedge q) \vee r$ or as $p \wedge (q \vee r)$. With p and q false and r true, the first reading is true and the second false. Section 4.4 defines the truth value of a formula one clause per connective, and the clause to use is decided by the main connective. A string with two main connectives would get two values. Formulas carry parentheses, and the parentheses are meant to settle the reading. This subsection and the next prove that they do, for every formula. The tool is a count.

Read a formula from left to right, keeping a running count: add one at each $($, subtract one at each $)$. The count just before a symbol is its **depth**: how many pairs of parentheses are open around it.



The depth profile of $((p \wedge q) \vee r)$. It starts at 0, stays at 1 or more strictly inside, comes back to 0 only after the last symbol, and the only binary connective at depth 1 is the main connective.

For every formula of the form $(\varphi \circ \psi)$ the lemma below says the same: (a) the count returns to 0, (b) it never goes below 0 and stays at 1 or more strictly inside, (c) every binary connective is at depth 1 or more. The remark after the proof adds that only the main connective is at depth exactly 1.

We count parentheses in strings, and for this some words about strings.

An **initial segment** of a string s is the string of the first k symbols of s , for some k with $0 \leq k \leq$ the length of s . For $k = 0$ it is the **empty** string, with no symbols; for k equal to the length it is s itself. An initial segment is **proper** if it is shorter than s . The **excess** of a string s is

$$\text{ex}(s) := (\text{number of } (\text{ in } s) - (\text{number of }) \text{ in } s).$$

So a left parenthesis has excess 1, a right parenthesis -1 , and any other symbol 0; and the excess of a string written as two strings one after the other is the sum of their excesses, since the counts add. For example, $\text{ex}((p \wedge (q)) = 2$: after reading $(p \wedge (q$ we are inside two pairs of parentheses that have not yet closed. That is the picture to keep: the excess of an initial segment is the **depth** we are at when we have read that far. So the depth of a symbol is the excess of the initial segment that ends just before it.

Lemma 4.3.2. Parenthesis Count

Let χ be a formula.

- (a) $\text{ex}(\chi) = 0$: χ has as many left as right parentheses.
- (b) Every initial segment u of χ has $\text{ex}(u) \geq 0$. If moreover χ has the form $(\varphi \circ \psi)$ and u is proper and non-empty, then $\text{ex}(u) \geq 1$.

- (c) For every occurrence of a binary connective in χ , the initial segment of χ that ends just before that occurrence has excess at least 1.

In words: (a) is balance. (b) says that reading a formula from the left, we never close more parentheses than we have opened, and that inside a formula $(\varphi \circ \psi)$ we stay at depth at least 1 until the very last symbol. (c) says that no binary connective lies outside all parentheses. Part (a) is the balance proved above, repeated in the proof because (b) and (c) need it for the parts; (c), with (a), is what the unique readability theorem needs, and (b), with (a), is the key to showing that no proper initial segment of a formula is a formula (Exercise C1). The steps for (b) and (c) use (a) for the parts, so the three are proved together. Part (b)'s second sentence, alone, does not pass through the step; that is why the lemma proves more, as Example (Prove more to prove less) did (fold below).

Idea. Induction on the shape of a formula, all three parts at once. For $\neg\varphi$, the extra $\neg$ changes no count. For $(\varphi \circ \psi)$, list its initial segments by where they stop: before anything, inside φ , at the connective, inside ψ , at the end. Each is “(” followed by pieces whose excess the induction hypothesis controls.

◉ **Second reading** · *The proof of the parenthesis count*

Proof. We prove, by [Theorem 4.3.1](#), that every formula χ has the property $P(\chi)$: “(a), (b), and (c) hold for χ ”.

Atoms and $\perp$. The formula is one symbol, not a parenthesis. (a) Its excess is 0. (b) Its initial segments are the empty string and the formula itself, both of excess 0; and it does not have the form $(\varphi \circ \psi)$, which has length at least 5. (c) It contains no binary connective.

Negation. Let φ be a formula with $P(\varphi)$; we show $P(\neg\varphi)$. (a) $\text{ex}(\neg\varphi) = 0 + \text{ex}(\varphi) = 0$, by (a) for φ . (b) An initial segment of $\neg\varphi$ is either empty, of excess 0, or $\neg$ followed by an initial segment u' of φ , of excess $\text{ex}(u') \geq 0$ by (b) for φ . The formula $\neg\varphi$ begins with $\neg$, so it does not have the form $(\varphi' \circ \psi')$, and the second sentence of (b) says nothing about it. (c) A binary connective in $\neg\varphi$ lies in φ , and the initial segment of $\neg\varphi$ before it is $\neg$ followed by the initial segment of φ before it, whose excess is at least 1 by (c) for φ .

Binary connectives. Let φ, ψ be formulas with $P(\varphi)$ and $P(\psi)$, let $\circ$ be a binary connective, and let $\chi = (\varphi \circ \psi)$.

(a) $\text{ex}(\chi) = 1 + \text{ex}(\varphi) + 0 + \text{ex}(\psi) - 1 = 0$, by (a) for φ and for ψ .

(b) Every initial segment of χ is of exactly one of the following five kinds, and its excess is as shown:

- the empty string: excess 0;
- (followed by an initial segment u' of φ : excess $1 + \text{ex}(u') \geq 1$, by (b) for φ ;
- $(\varphi \circ$: excess $1 + \text{ex}(\varphi) + 0 = 1$, by (a) for φ ;
- $(\varphi \circ$ followed by a non-empty initial segment v' of ψ : excess $1 + 0 + \text{ex}(v') \geq 1$, by (a) for φ and (b) for ψ ;
- χ itself: excess 0, by (a) for χ .

All five have excess at least 0. The proper non-empty ones are those of the middle three

kinds, and they have excess at least 1.

(c) An occurrence of a binary connective in χ is the connective $\circ$ between φ and ψ , or lies in φ , or lies in ψ . Before $\circ$ stands $(\varphi$, of excess 1. Before an occurrence in φ stands $($ followed by the initial segment of φ before it, of excess at least $1 + 1 = 2$ by (c) for φ . Before an occurrence in ψ stands $(\varphi \circ$ followed by the initial segment of ψ before it, of excess at least $1 + 0 + 1 = 2$ by (a) for φ and (c) for ψ . In all three cases the excess is at least 1.

So $P(\chi)$ holds. By [Theorem 4.3.1](#), every formula has the property P , which proves the lemma. ■

The three parts were proved together for a reason. The second sentence of (b), alone, cannot be proved by structural induction. For $(\neg(p \wedge q) \wedge r)$, the initial segment $(\neg(p$ is $($ followed by the initial segment $\neg(p$ of the formula $\neg(p \wedge q)$, and its excess is $1 + \text{ex}(\neg(p)$; to know that this is at least 1, we need $\text{ex}(\neg(p) \geq 0$, a fact about an initial segment of a **negation**, which the second sentence of (b) does not cover. The first sentence of (b) does. A property that is too weak to pass through the induction must be strengthened until it does, and here the strengthening is (b)'s first sentence, as the strengthening of Example (Prove more to prove less) was $S_n = \frac{n}{n+1}$.

The proof (in the fold) gives a little more than (c) states, and the next proof uses the extra. In a formula $(\varphi \circ \psi)$, the initial segment before the connective $\circ$ has excess exactly 1, and the initial segment before **any other** binary connective has excess at least 2. In the language of depth: the main connective is the only binary connective at depth one. That is the recipe of Section 4.1 for finding the root of a parse tree.

Quick check. Write down the depth of each symbol of $(p \wedge (q \vee r))$. Which binary connective is at depth 1?

Solution. $(0, p 1, \wedge 1, (1, q 2, \vee 2, r 2,) 2,) 1$, and 0 after the last symbol. The $\wedge$ is at depth 1, and it is the main connective. Compare the figure, whose main connective was the $\vee$.

4.3.3 Unique readability

We can now prove what Section 4.1 promised: a formula can be built in only one way.

Theorem 4.3.3. Unique Readability

Let χ be a formula that is neither an atom nor $\perp$. Then **exactly one** of the following holds:

- (i) $\chi = \neg\varphi$ for a formula φ ;
- (ii) $\chi = (\varphi \circ \psi)$ for formulas φ, ψ and a binary connective $\circ$.

In case (i), φ is **uniquely** determined by χ ; in case (ii), $\circ$, φ and ψ are uniquely determined by χ .

In words: every compound formula has one main connective, and one way of being split into its immediate parts. This is the precise form of Chapter 1's claim that a statement built from letters by connectives can be read in only one way. Existence is the easy half; the content is **uniqueness**. The danger it excludes is a string such as $p \wedge q \vee r$, which, if it were a formula, could be split at $\wedge$ or at $\vee$.

Idea. Some form holds, by looking at the last step (Section 4.1); the first symbol tells the two forms apart. For case (ii), suppose $\chi = (\varphi \circ \psi) = (\varphi' \circ' \psi')$. By the remark after Lemma 4.3.2, each main connective is the only binary connective at depth one; so both sit at the same position of the string, and everything else is forced.

Proof. By Section 4.1 (looking at the last step), χ is an atom, $\perp$, $\neg\varphi$ for a formula φ , or $(\varphi \circ \psi)$ for formulas φ, ψ ; by hypothesis it is one of the last two. It cannot be both, since a string of the form $\neg\varphi$ begins with $\neg$ and one of the form $(\varphi \circ \psi)$ begins with $($.

Case (i). If $\chi = \neg\varphi = \neg\varphi'$, removing the first symbol gives $\varphi = \varphi'$.

Case (ii). Let $\chi = (\varphi \circ \psi)$. Before the connective $\circ$ stands the initial segment $(\varphi$, of excess $1 + \text{ex}(\varphi) = 1$ by Lemma 4.3.2 (a). Before a binary connective inside φ stands $($ followed by an initial segment of φ of excess at least 1, by Lemma 4.3.2 (c); so the excess is at least 2. Before a binary connective inside ψ stands $(\varphi \circ$ followed by an initial segment of ψ of excess at least 1, by (c) for ψ and (a) for φ ; so again at least 2. Hence the position of $\circ$ in χ is **the only** position of a binary connective in χ whose initial segment before it has excess 1, and this description mentions only the string χ .

Now suppose also $\chi = (\varphi' \circ' \psi')$ for formulas φ', ψ' and a binary connective $\circ'$. By the same argument, the position of $\circ'$ is the only position with that description. So $\circ$ and $\circ'$ stand at the same position of χ , and $\circ = \circ'$. That position is the one just after $($ and φ , and also the one just after $($ and φ' , so φ and φ' have the same length; both are the string of symbols of χ between the first symbol and that position, so $\varphi = \varphi'$. Likewise ψ and ψ' are both the string of symbols between that position and the last symbol, so $\psi = \psi'$. This proves the theorem. ■

The theorem settles the question Section 4.1 left open about parse trees.

Every formula has exactly one parse tree. We know it has one. That it has only one is proved by structural induction. By (T4) of Definition 4.1.2, every parse tree is made by one of (T1)–(T3). An atom or $\perp$ has only the one-node tree, since a tree made by (T2) or (T3) of Definition 4.1.2 is a tree for a string beginning with $\neg$ or $($. A parse tree for $\neg\varphi$ must have been made by (T2), from a parse tree for some φ' with $\neg\varphi' = \neg\varphi$; so $\varphi' = \varphi$, and by the induction hypothesis the tree below the root is the one parse tree of φ . A parse tree for $(\varphi \circ \psi)$ must have been made by (T3), from parse trees for some φ', ψ' and a connective $\circ'$ with $(\varphi' \circ' \psi') = (\varphi \circ \psi)$; by Theorem 4.3.3, $\circ' = \circ$, $\varphi' = \varphi$ and $\psi' = \psi$, and by the induction hypothesis the two subtrees are the parse trees of φ and ψ . From now on we speak of **the** parse tree of a formula.

With one parse tree for each formula, a measurement of that tree is a measurement of the formula.

Definition 4.3.1. Height

The **height** of a formula φ , written $\text{ht}(\varphi)$, is the number of nodes on a **longest** branch of its parse tree.

Atoms and $\perp$ have height 1: their tree is one node, and its only branch has one node. (Some books count edges instead of nodes and give atoms height 0; with nodes, every formula has height at least 1.) In Example (Three parse trees), (b) has height 4, along the branch $\neg, \wedge, \neg, q$, and (c) has height 4 too, along $\vee, \leftrightarrow, \neg, q$ or along $\vee, \wedge, \rightarrow, \perp$.

Warning. The height counts the nodes on the longest **branch**, not the symbols of the formula. The formula $\neg(p \wedge \neg q)$ has seven symbols, $\neg, (, p, \wedge, \neg, q,)$, and height 4; the formula $((p \wedge q) \wedge (r \wedge p))$ has thirteen symbols and height only 3. Parentheses never count towards the height, and a wide tree can be shallow.

4.3.4 Definition by recursion

Height, the set of atoms, the number of connectives: each is naturally defined **one clause per way of building a formula**. The height of $\neg\varphi$ is one more than that of φ ; the atoms of $(\varphi \circ \psi)$ are those of φ together with those of ψ . Such a definition is called **recursive**: it defines the value at a formula through the values at its immediate parts. The question is whether it really defines something. For a string that could be split in two ways, the clauses might assign two different values, and then they would define nothing. Unique readability says this never happens.

We say **function on formulas** for a rule that assigns to each formula exactly one value; the values may be numbers, truth values, formulas, or collections of atoms. Nothing here needs more than this everyday sense of “function”.

Give a value for each atom and for $\perp$, and say how to get the value of a compound formula from its parts; then exactly one function on formulas obeys these clauses.

Theorem 4.3.4. Structural Recursion

Suppose we are given

- a value a_p for each atom p , and a value $a_\perp$;
- a rule $g_\neg$ that produces a value $g_\neg(\varphi, x)$ from any formula φ and any value x ;
- for each binary connective $\circ$, a rule $g_\circ$ that produces a value $g_\circ(\varphi, \psi, x, y)$ from any formulas φ, ψ and any values x, y .

Then there is **exactly one** function f on formulas such that, for every atom p , all formulas φ, ψ and every binary connective $\circ$,

$$\begin{aligned} f(p) &= a_p, & f(\perp) &= a_\perp, \\ f(\neg\varphi) &= g_\neg(\varphi, f(\varphi)), \\ f((\varphi \circ \psi)) &= g_\circ(\varphi, \psi, f(\varphi), f(\psi)). \end{aligned}$$

In words: the three lines are the **clauses** of a recursive definition, one for each clause of Definition 4.1.1. The rules $g_\neg$ and $g_\circ$ may look at the immediate parts themselves as well as at their values; most definitions use only the values, and the list of subformulas below uses the parts.

The theorem says the clauses define a function (**existence**) and only one (**uniqueness**).

Idea. Build the function on short formulas first. A formula with one symbol is an atom or $\perp$, and the first clauses fix its value. A longer formula has immediate parts that are shorter, whose values are already fixed, and unique readability says which clause applies and to which parts, with no choice left. So the value at each formula is forced, length by length. Formally: an ordinary induction on n , proving that there is exactly one function on the formulas of length at most n that obeys the clauses.

🕒 **Second reading** · *The proof of structural recursion*

Proof. For $n \in \mathbb{N}$ with $n \geq 1$, call a function h , defined on the formulas of length at most n , **n -good** if it obeys the three clauses at every formula of length at most n . This makes sense: the immediate parts of a formula are shorter than it, so the clauses at a formula of length at most n mention values of h only at formulas of length at most n . Let $S(n)$ be “there is exactly one n -good function”; we prove $S(n)$ for every $n \geq 1$ by Axiom 4.2.1, started at 1 (“Starting from another number”, Section 4.2).

Base case. By the first-symbol test (Section 4.1), the formulas of length 1 are the atoms and $\perp$. A 1-good function must give each atom p the value a_p and $\perp$ the value $a_\perp$, and the function that does so is 1-good, since the other two clauses concern no formula of length 1. So $S(1)$ holds.

Inductive step. Let $n \in \mathbb{N}$ with $n \geq 1$, suppose $S(n)$, and let h_n be the n -good function. Define h on the formulas of length at most $n + 1$ as follows. At a formula of length at most n , let h agree with h_n . Let χ have length $n + 1 \geq 2$; it is neither an atom nor $\perp$, so by Theorem 4.3.3 it is **either** $\neg\varphi$ for exactly one formula φ , **or** $(\varphi \circ \psi)$ for exactly one $\circ$, φ , ψ ; the parts have length at most n . Let $h(\chi) := g_\neg(\varphi, h_n(\varphi))$ in the first case and $h(\chi) := g_\circ(\varphi, \psi, h_n(\varphi), h_n(\psi))$ in the second. Unique readability leaves no choice of case or of parts, so this is one value. The function h is $(n + 1)$ -good: at formulas of length at most n it agrees with the n -good h_n , including at their parts, and at formulas of length $n + 1$ it obeys the clauses by construction.

For uniqueness, let h' be any $(n + 1)$ -good function. Its restriction to the formulas of length at most n is n -good, so it is h_n , by $S(n)$. At a formula $\chi = \neg\varphi$ of length $n + 1$, the clause for $\neg$ gives $h'(\chi) = g_\neg(\varphi, h'(\varphi)) = g_\neg(\varphi, h_n(\varphi)) = h(\chi)$; at $\chi = (\varphi \circ \psi)$ the same argument with $g_\circ$ gives $h'(\chi) = h(\chi)$. So $h' = h$, and $S(n + 1)$ holds. By Axiom 4.2.1, started at 1, $S(n)$ holds for every $n \geq 1$; let h_n denote the n -good function.

The function f . If $m \geq n$, the restriction of h_m to the formulas of length at most n is n -good, so it is h_n . For a formula χ of length n , let $f(\chi) := h_n(\chi)$; then $f(\chi) = h_m(\chi)$ for every $m \geq n$. The function f obeys the clauses: for $\chi = \neg\varphi$ of length m , $f(\chi) = h_m(\chi) = g_\neg(\varphi, h_m(\varphi)) = g_\neg(\varphi, f(\varphi))$, and likewise for the other two clauses. And it is the only one: a function f' on formulas obeying the clauses restricts, for each n , to an n -good function, which is h_n ; so $f'(\chi) = h_n(\chi) = f(\chi)$ for a formula χ of length n . This proves the theorem. ■

Unique readability was used at one place: to define $h(\chi)$ by exactly one clause, applied to exactly

one choice of parts. Without it, the clauses could demand two different values at one string, and then no function would obey them.

What goes wrong without it. The positive integers are built from 1 by two steps, “add one” and “double”; adding one alone reaches every positive integer. Try to define a function one clause per building step: $g(1) := 0$, $g(n+1) := g(n) + 1$, $g(2n) := g(n)$. The number 2 is built in two ways, as $1+1$ and as twice 1. The first clause gives $g(2) = g(1) + 1 = 1$, the second $g(2) = g(1) = 0$, and no function obeys both. The positive integers are not built **freely** by these two steps: one number can come out of two different steps, which is exactly what [Theorem 4.3.3](#) rules out for formulas.

Quick check. Do the clauses $h(1) := 1$, $h(n+1) := h(n) + 2$, $h(2n) := 2h(n)$ define a function on the positive integers?

Solution. No. At 2, the first step clause gives $h(2) = h(1) + 2 = 3$ and the second $h(2) = 2h(1) = 2$.

Example. Four recursive definitions

Each of the following is a function on formulas by [Theorem 4.3.4](#).

- **Height:** $\text{ht}(p) = \text{ht}(\perp) = 1$, $\text{ht}(\neg\varphi) = \text{ht}(\varphi) + 1$, and

$$\text{ht}((\varphi \circ \psi)) = 1 + \max\{\text{ht}(\varphi), \text{ht}(\psi)\}.$$
- **Atoms:** $\text{atoms}(p) = \{p\}$; $\text{atoms}(\perp)$ has no members; $\text{atoms}(\neg\varphi) = \text{atoms}(\varphi)$; and $\text{atoms}((\varphi \circ \psi))$ consists of the members of $\text{atoms}(\varphi)$ together with those of $\text{atoms}(\psi)$.
- **Number of connectives:** 0 for atoms and $\perp$; one more than for φ at $\neg\varphi$; one more than the sum for φ and ψ at $(\varphi \circ \psi)$.
- **Subformulas:** the list of subformulas of p is p , and of $\perp$ is $\perp$; of $\neg\varphi$, the formula $\neg\varphi$ followed by the subformulas of φ ; of $(\varphi \circ \psi)$, the formula $(\varphi \circ \psi)$ followed by the subformulas of φ and then those of ψ , each formula kept only at its first appearance.

Compute all four for $\chi = (\neg(p \wedge q) \rightarrow (q \vee \perp))$, and also its length.

Solution. Work up from the leaves. For the parts: $\text{ht}(p \wedge q) = 1 + \max\{1, 1\} = 2$, so $\text{ht}(\neg(p \wedge q)) = 3$; $\text{ht}(q \vee \perp) = 2$. Hence

$$\text{ht}(\chi) = 1 + \max\{3, 2\} = 4.$$

The atoms of $\neg(p \wedge q)$ are p and q , those of $q \vee \perp$ are just q ($\perp$ contributes none), so $\text{atoms}(\chi) = \{p, q\}$. The connectives: $\wedge$ and $\neg$ on the left, $\vee$ on the right, $\rightarrow$ at the root, four in all. The subformulas, in the order the clause lists them: χ , $\neg(p \wedge q)$, $(p \wedge q)$, p , q , $(q \vee \perp)$, $\perp$; the second q is dropped, being already listed. The length is 14: the official formula consists of the fourteen symbols

$$(\neg(p \wedge q) \rightarrow (q \vee \perp)).$$

Two checks connect these definitions with the parse trees. The **recursive height agrees with the tree height**: the parse tree of $\neg\varphi$ is that of φ with one node on top, so each branch gains one node; the tree of $(\varphi \circ \psi)$ is a new root over the trees of φ and ψ , so a longest branch is the root followed by a longest branch of the taller of the two. A structural induction whose cases are these two sentences (and the one-node tree for atoms and $\perp$) shows the two heights are equal. In the same way, the recursive list of subformulas is the list of formulas at the nodes of the parse tree (Definition 4.1.3): the nodes of the tree of $\neg\varphi$ are its root, carrying $\neg\varphi$, and the nodes of the tree of φ .

The length $\text{len}(\varphi)$ of a formula φ , its number of symbols, also satisfies clauses of this shape, by counting symbols: a formula of length one for atoms and $\perp$, one more at $\neg\varphi$, and at $(\varphi \circ \psi)$ the lengths of φ and ψ plus 3 (two parentheses and the connective). For χ above: $(p \wedge q)$ has $1 + 1 + 3 = 5$ symbols, so $\neg(p \wedge q)$ has 6; $(q \vee \perp)$ has 5; so χ has $6 + 5 + 3 = 14$, as counted.

Warning. A recursive definition must have a clause for **every** way of building a formula, and each clause may use only the **immediate parts**. “Let $f(\varphi)$ be f of the longest subformula of φ plus one” is not a recursive definition: at φ itself the longest subformula is φ , and the “definition” refers to the value it is trying to define. The theorem applies only to clauses of the form it states.

Remark (Looking ahead). Chapter 5 builds the formulas of predicate logic by clauses of the same kind, and proves things about all of them by inductions of this kind.

4.3.5 Exercises

4.3.5.1 A. Check your understanding

Exercise. A1

- (a) Which kind of induction fits “every formula contains an atom or $\perp$ ”? Write down its base cases and the statements of its steps, without proving them.
- (b) A reader proves “every formula has at least one symbol that is not a parenthesis” by structural induction. They check atoms, the step for $\neg$, and the step for $(\varphi \circ \psi)$. Which case is missing? Does the claim hold there?
- (c) Write down, without proving them, the base cases and the inductive steps of a structural induction for: every formula has exactly as many left parentheses as binary connectives.
- (d) Which parts of [Lemma 4.3.2](#) does the proof of [Theorem 4.3.3](#) use, and for which formulas?
- (e) Why does [Theorem 4.3.4](#) not apply to the positive integers built from 1 by “add one” and “double”?

Solution. (a) Structural induction. Base cases: an atom contains itself, and $\perp$ contains $\perp$. Steps: if φ contains an atom or $\perp$, so does $\neg\varphi$; if φ and ψ do, so does $(\varphi \circ \psi)$, for each binary $\circ$. (Compare the warning in the section: with “an atom” alone, the case $\perp$ fails.)

(b) The case $\perp$ is missing. The claim holds there, since $\perp$ is itself such a symbol. The proof

is incomplete, though the claim is true; compare the warning on “every formula contains an atom”, where the missing case makes the claim false.

(c) Base cases: an atom and $\perp$ have none of either. Step for $\neg$: if φ has as many of each, so has $\neg\varphi$, which adds neither. Step for $(\varphi \circ \psi)$: if each of φ, ψ has as many of each, the whole adds one left parenthesis and one binary connective.

(d) Part (a) for φ , to get excess 1 before the main connective and to get excess $1+0$ before the part ψ ; part (c) for φ and for ψ , to get excess at least 2 before every other binary connective. Here $\chi = (\varphi \circ \psi)$ is the formula being read. Part (b) is not used in that proof.

(e) Its proof needs each compound object to be built in exactly one way, which for formulas is unique readability. The number 2 is built as $1+1$ and as twice 1, so clauses can demand two values at one number. Some clauses happen to agree wherever a number is built twice: $g(1) := 1$, $g(n+1) := g(n) + 1$, $g(2n) := 2g(n)$ is obeyed by $g(n) = n$. But without unique building nothing guarantees this in advance.

Exercise. A2: Drill

The parts run from naming the induction, through computing depths, heights and recursive values, to judging claims and definitions and running the checklist.

- Which kind of induction proves “every formula has as many left as right parentheses”?
- Write down the depth of each symbol of $\neg(p \rightarrow \neg q)$. Which binary connective is at depth 1?
- What is the excess of the initial segment $((p \wedge (q \text{ of } ((p \wedge (q \vee r)) \rightarrow p)))$?
- Find the main connective of $((p \rightarrow q) \wedge (r \vee p))$ by depth.
- What is the height of $((p \wedge q) \vee \neg\neg r)$?
- True or false: every subformula of φ other than φ itself has smaller height than φ .
- Compute the recursive height and the recursive list of subformulas of $\neg(p \vee \neg p)$.
- Define by recursion the number $b(\varphi)$ of occurrences of $\perp$ in φ , and compute $b((\perp \rightarrow \neg\perp))$.
- Is “ $f(\varphi) := f(\varphi) + 1$ for every formula φ ” a definition by recursion?
- “Every formula has odd length or begins with $\neg$.” Which case of a structural induction fails, and on which formula does the claim fail?
- Do the clauses $g(1) := 1$, $g(n+1) := g(n) + 1$, $g(2n) := g(n) + 1$ define a function on the positive integers built by “add one” and “double”?
- Run the checklist on “every formula with no $\neg$ and no $\perp$ has odd length”.

Solution.

- Structural induction, or strong induction on the position in a formation sequence, which is how structural induction is proved.
- $\neg$ 0, (0, p 1, $\rightarrow$ 1, $\neg$ 1, q 1,) 1, and 0 after. The $\rightarrow$ is at depth 1: it is the main connective of the part inside the parentheses. The main connective of the whole formula is the first $\neg$.
- 3: three left parentheses, no right one.

- (d) The $\wedge$, the only binary connective at depth 1; the $\rightarrow$ and the $\vee$ are at depth 2.
- (e) $1 + \max\{2, 3\} = 4$.
- (f) True. A subformula other than φ sits at a node below the root, and the part of the tree below that node is its parse tree. A longest branch of that part, with the path from the root of φ 's tree down to the node added in front, is a branch of φ 's tree with at least one more node. So φ 's tree has a branch longer than every branch of the subformula's tree.
- (g) Height 4 (p : 1, $\neg p$: 2, $(p \vee \neg p)$: 3). List: $\neg(p \vee \neg p)$, $(p \vee \neg p)$, p , $\neg p$.
- (h) $b(p) = 0$ for every atom p , $b(\perp) = 1$, $b(\neg\varphi) = b(\varphi)$, $b((\varphi \circ \psi)) = b(\varphi) + b(\psi)$. Value $1 + 1 = 2$.
- (i) No. It gives no value at atoms or $\perp$, and it defines the value at φ from the value at φ itself, not from the immediate parts. No function obeys it, since no number equals itself plus one.
- (j) The step for $(\varphi \circ \psi)$, whose length is $\text{len}(\varphi) + \text{len}(\psi) + 3$: the hypothesis allows a part that begins with $\neg$ and has even length. The claim fails on $(p \wedge \neg q)$, of length 6, beginning with $($.
- (k) No: $g(2) = 2$ and $g(3) = 3$ by the first step clause, and then $g(4) = g(3) + 1 = 4$ by the first step clause but $g(4) = g(2) + 1 = 3$ by the second.
- (l) Objects: formulas. Property: “if φ contains no $\neg$ and no $\perp$, then its length is odd”, the restriction inside. Base cases: an atom (length 1), and $\perp$ (vacuous). Steps: $\neg\varphi$ is vacuous; for $(\varphi \circ \psi)$ without $\neg$ and $\perp$, its parts satisfy the restriction, so their lengths are odd, and odd + odd + 3 is odd. Principle: [Theorem 4.3.1](#). (Compare (j): with $\neg$ allowed the claim fails.)

4.3.5.2 B. Practice

Exercise. B1

Prove by structural induction that every formula has at most as many distinct subformulas as it has symbols. Use the recursive description of the subformulas in [Example \(Four recursive definitions\)](#) and the clauses for the length given after it.

Solution. For a formula φ , write $s(\varphi)$ for the number of distinct subformulas of φ . We prove $s(\varphi) \leq \text{len}(\varphi)$ by [Theorem 4.3.1](#).

Atoms and $\perp$. The only subformula is the formula itself: $s = 1 = \text{len}$.

Negation. Suppose $s(\varphi) \leq \text{len}(\varphi)$. The subformulas of $\neg\varphi$ are $\neg\varphi$ and those of φ , so there are at most $1 + s(\varphi)$ of them (fewer if one repeats). Hence $s(\neg\varphi) \leq 1 + s(\varphi) \leq 1 + \text{len}(\varphi) = \text{len}(\neg\varphi)$.

Binary connectives. Suppose $s(\varphi) \leq \text{len}(\varphi)$ and $s(\psi) \leq \text{len}(\psi)$. The subformulas of $(\varphi \circ \psi)$ are the formula itself and those of φ and of ψ , so

$$s((\varphi \circ \psi)) \leq 1 + s(\varphi) + s(\psi) \leq 1 + \text{len}(\varphi) + \text{len}(\psi) < \text{len}(\varphi) + \text{len}(\psi) + 3,$$

and the right side is $\text{len}((\varphi \circ \psi))$.

By [Theorem 4.3.1](#), $s(\varphi) \leq \text{len}(\varphi)$ for every formula φ . (The bound is attained by p , $\neg p$, $\neg\neg p$,

and every formula built from an atom or $\perp$ by negations alone; each binary connective adds three symbols but at most one new subformula, so the inequality is strict for every formula that contains a binary connective.)

Exercise. B2

The number of nodes of the parse tree of a formula is given by the recursion $\text{nodes}(p) = \text{nodes}(\perp) = 1$, $\text{nodes}(\neg\varphi) = \text{nodes}(\varphi) + 1$, $\text{nodes}((\varphi \circ \psi)) = \text{nodes}(\varphi) + \text{nodes}(\psi) + 1$. Prove that every formula φ of height h satisfies $\text{nodes}(\varphi) \leq 2^h - 1$, and give a formula of height 3 for which equality holds.

Solution. We prove $\text{nodes}(\varphi) \leq 2^{\text{ht}(\varphi)} - 1$ by [Theorem 4.3.1](#), using the recursive height of [Example \(Four recursive definitions\)](#).

Atoms and $\perp$. $\text{nodes} = 1 = 2^1 - 1$.

Negation. Let $h = \text{ht}(\varphi)$ and suppose $\text{nodes}(\varphi) \leq 2^h - 1$. Then $\text{ht}(\neg\varphi) = h + 1$ and

$$\text{nodes}(\neg\varphi) \leq 2^h - 1 + 1 = 2^h \leq 2^{h+1} - 1,$$

where the last step is $1 \leq 2^h$.

Binary connectives. Let $a = \text{ht}(\varphi)$, $b = \text{ht}(\psi)$, and suppose the bound for both. Then $h := \text{ht}((\varphi \circ \psi)) = 1 + \max\{a, b\}$, so $a \leq h - 1$ and $b \leq h - 1$, and $2^a, 2^b \leq 2^{h-1}$. Hence

$$\text{nodes}((\varphi \circ \psi)) \leq (2^a - 1) + (2^b - 1) + 1 \leq 2 \cdot 2^{h-1} - 1 = 2^h - 1.$$

By [Theorem 4.3.1](#), the bound holds for every formula. Equality at height 3: $((p \wedge q) \wedge (r \wedge p))$ has $7 = 2^3 - 1$ nodes (a root, two $\wedge$ nodes, and four leaves). Equality needs every node above the bottom level to have two children.

Exercise. B3: The mirror image of a formula

Define a function m on formulas by recursion: $m(p) = p$ for every atom p , $m(\perp) = \perp$, $m(\neg\varphi) = \neg m(\varphi)$, and $m((\varphi \circ \psi)) = (m(\psi) \circ m(\varphi))$ for each binary connective $\circ$.

- Compute $m((p \rightarrow (q \wedge \neg r)))$.
- Prove that $m(m(\varphi)) = \varphi$ for every formula φ .
- Prove that $\text{ht}(m(\varphi)) = \text{ht}(\varphi)$ for every formula φ .
- Give a formula φ , not an atom or $\perp$, with $m(\varphi) = \varphi$.

Solution. (a) $m((p \rightarrow (q \wedge \neg r))) = (m((q \wedge \neg r)) \rightarrow p) = ((\neg r \wedge q) \rightarrow p)$.

(b) By [Theorem 4.3.1](#), with $P(\varphi)$: “ $m(m(\varphi)) = \varphi$ ”. *Atoms and $\perp$:* m fixes them, so $m(m(p)) = p$ and $m(m(\perp)) = \perp$. *Negation:* if $m(m(\varphi)) = \varphi$, then $m(m(\neg\varphi)) = m(\neg m(\varphi)) = \neg m(m(\varphi)) = \neg\varphi$. *Binary connectives:* if $m(m(\varphi)) = \varphi$ and $m(m(\psi)) = \psi$, then

$$m(m((\varphi \circ \psi))) = m((m(\psi) \circ m(\varphi))) = (m(m(\varphi)) \circ m(m(\psi))) = (\varphi \circ \psi).$$

(c) By [Theorem 4.3.1](#), with the recursive height of [Example \(Four recursive definitions\)](#). *Atoms and $\perp$* : both sides are 1. *Negation*: $\text{ht}(m(\neg\varphi)) = \text{ht}(\neg m(\varphi)) = \text{ht}(m(\varphi)) + 1 = \text{ht}(\varphi) + 1 = \text{ht}(\neg\varphi)$. *Binary connectives*:

$$\text{ht}(m((\varphi \circ \psi))) = 1 + \max\{\text{ht}(m(\psi)), \text{ht}(m(\varphi))\} = 1 + \max\{\text{ht}(\varphi), \text{ht}(\psi)\} = \text{ht}((\varphi \circ \psi)),$$

using the hypothesis for both parts.

(d) $(p \wedge p)$, or $\neg(q \vee q)$.

Marking scheme. (8 points)

- 1: (a).
- 3: (b), all four cases (atoms, $\perp$, negation, binary).
- 3: (c), a structural induction (on the recursive height, or showing that the tree of $m(\varphi)$ mirrors that of φ) using the hypothesis for both parts.
- 1: (d).

4.3.5.3 C. Going deeper

Exercise. C1: Initial segments are not formulas

- (a) Prove that no proper initial segment of a formula is a formula.
- (b) Deduce, in a way different from the proof in the text, the uniqueness part of case (ii) of [Theorem 4.3.3](#): if $(\varphi \circ \psi) = (\varphi' \circ' \psi')$ for formulas $\varphi, \psi, \varphi', \psi'$ and binary connectives $\circ, \circ'$, then $\varphi = \varphi'$, $\circ = \circ'$ and $\psi = \psi'$.

Solution (Hint 1). Structural induction on the formula whose initial segments you consider. Which part of [Lemma 4.3.2](#) settles the case $(\varphi \circ \psi)$ at once?

Solution (Hint 2). For (b): remove the first symbol from both sides. Now both φ and φ' are initial segments of the same string.

Solution. (a) The empty string is not a formula, since every formula has at least one symbol. So it is enough to show that no proper **non-empty** initial segment of a formula χ is a formula. We prove this by [Theorem 4.3.1](#).

Atoms and $\perp$. A formula of length 1 has no proper non-empty initial segment.

Negation. Suppose no proper non-empty initial segment of φ is a formula, and let u be a proper non-empty initial segment of $\neg\varphi$. Then u is $\neg$ followed by a proper initial segment u' of φ . Suppose, for a contradiction, that u is a formula. It begins with $\neg$, so it is not an atom or $\perp$ and not of the form $(\varphi'' \circ \psi'')$; by looking at the last step (Section 4.1), $u = \neg\varphi''$ for a formula φ'' , and removing the first symbol gives $u' = \varphi''$. So u' is a formula, hence non-empty, and it is a proper non-empty initial segment of φ , contradicting the induction hypothesis.

Binary connectives. Let u be a proper non-empty initial segment of $(\varphi \circ \psi)$. By [Lemma 4.3.2](#)

(b), $\text{ex}(u) \geq 1$. If u were a formula, [Lemma 4.3.2](#) (a) would give $\text{ex}(u) = 0$. So u is not a formula. (This case needs no induction hypothesis.)

By [Theorem 4.3.1](#), no formula has a proper non-empty initial segment that is a formula, which proves (a).

(b) Suppose $(\varphi \circ \psi) = (\varphi' \circ' \psi')$. Removing the first symbol, the strings $\varphi \circ \psi$ and $\varphi' \circ' \psi'$ are equal. Both φ and φ' are initial segments of this string, so the shorter of the two is an initial segment of the longer. If their lengths differed, the shorter would be a proper initial segment of the longer, and a formula, contradicting (a). So they have the same length, and being initial segments of the same string, $\varphi = \varphi'$. The next symbol is then $\circ$ on one side and $\circ'$ on the other, so $\circ = \circ'$; what remains is $\psi) = \psi')$, and removing the last symbol gives $\psi = \psi'$.

Marking scheme. (10 points)

- 1: reduces (a) to non-empty segments (the empty string is not a formula).
- 2: sets up a structural induction (or a strong induction on the length) with the property “no proper non-empty initial segment is a formula”, with the atom and $\perp$ case.
- 3: negation case: reads the segment as $\neg$ followed by a proper initial segment of φ , and uses the last step of a formula beginning with $\neg$ to reach a contradiction with the hypothesis.
- 2: binary case: excess at least 1 by [Lemma 4.3.2](#) (b) against excess 0 by (a).
- 2: (b): compares φ and φ' as initial segments of one string, concludes equal length by (a), then reads off $\circ$ and ψ .

Fatal errors (0 for the step): using [Theorem 4.3.3](#) to prove (b); assuming the lengths of φ and φ' are equal.

Exercise. C2: Find the error

The following argument claims to prove that every formula contains an **even** number of parentheses.

- (1) We prove, by strong induction on n , that every formula with n symbols has an even number of parentheses.
 - (2) Let $n \in \mathbb{N}$, let χ be a formula with n symbols, and suppose that every formula with fewer than n symbols has an even number of parentheses.
 - (3) If χ is an atom or $\perp$, it has no parentheses, and 0 is even.
 - (4) If $\chi = \neg\varphi$, then φ is a formula with $n - 1$ symbols, so by hypothesis it has an even number of parentheses; and χ has the same parentheses as φ .
 - (5) If $\chi = (\varphi \circ \psi)$, remove the outer pair of parentheses. The string $\varphi \circ \psi$ has $n - 2$ symbols, so by hypothesis it has an even number of parentheses.
 - (6) Putting the outer pair back adds 2, so χ has an even number of parentheses.
 - (7) By Theorem 4.2.2, every formula has an even number of parentheses.
- (a) Is the conclusion true?
- (b) Which line fails, and why?
- (c) Show that the move made in that line is illegal by applying the same move to a different property that every formula has, and reaching a false statement.

(d) Repair the argument.

Solution. (a) Yes. By [Lemma 4.3.2](#) (a), a formula has as many left as right parentheses, say k of each, so $2k$ in all.

(b) Line 5. The induction hypothesis of line 2 is about **formulas** with fewer than n symbols, and line 5 applies it to the string $\varphi \circ \psi$, which is not a formula: for $\chi = (p \wedge q)$, it is $p \wedge q$, which fails the first-symbol test of Section 4.1 (length 3, beginning with an atom). The hypothesis says nothing about such strings. Every other line is correct: line 4 applies the hypothesis to the formula φ , and lines 1, 2, 3, 6 and 7 are a correct frame for a strong induction on the length.

(c) Take the property of [Lemma 4.3.2](#) (c), that no binary connective lies outside all parentheses; every formula has it. Run the same argument for it: in the case $\chi = (\varphi \circ \psi)$, line 5's move applies the hypothesis to the string $\varphi \circ \psi$ and asserts that no binary connective of $\varphi \circ \psi$ lies outside all parentheses. For $\chi = (p \wedge q)$ this asserts it of $p \wedge q$, whose $\wedge$ lies outside all parentheses. So the move leads from true hypotheses to a false statement; it is not a valid step, even though in the original argument its conclusion happens to be true.

(d) In line 5, apply the hypothesis to the formulas φ and ψ , each with fewer than n symbols: they have $2a$ and $2b$ parentheses for some integers $a, b \geq 0$, and χ has $2a + 2b + 2$, which is even. (Alternatively, prove the claim by [Theorem 4.3.1](#), with the same three cases.)

Marking scheme. (10 points)

- 2: (a), with [Lemma 4.3.2](#) (a).
- 3: (b), naming line 5 and why it fails (the hypothesis is about formulas, and $\varphi \circ \psi$ is not one).
- 3: (c), a property every formula has and a formula on which the move leads to a false statement.
- 2: (d), the repair through the formulas φ and ψ .

Fatal errors (0 for the step): locating the error in line 4.

Exercise. C3: Exit check: structural induction

Say that a formula is a **$\neg\wedge$ -formula** if its only connectives are $\neg$ and $\wedge$ and it does not contain $\perp$. Define, by [Theorem 4.3.4](#), the number of nodes of the parse tree and the number of negations:

$$\begin{aligned} \text{nodes}(p) &= 1, \\ \text{nodes}(\neg\varphi) &= \text{nodes}(\varphi) + 1, \\ \text{nodes}((\varphi \wedge \psi)) &= \text{nodes}(\varphi) + \text{nodes}(\psi) + 1, \\ \text{neg}(p) &= 0, \\ \text{neg}(\neg\varphi) &= \text{neg}(\varphi) + 1, \\ \text{neg}((\varphi \wedge \psi)) &= \text{neg}(\varphi) + \text{neg}(\psi), \end{aligned}$$

with $\text{nodes}(\perp) = 1$ and $\text{neg}(\perp) = 0$, and the clauses for the other binary connectives as for $\wedge$ (these choices play no part in what follows). Prove by structural induction that every $\neg\wedge$ -formula φ satisfies $\text{nodes}(\varphi) > \text{neg}(\varphi)$.

Solution (Hint 1). [Theorem 4.3.1](#) is about all formulas. Put the restriction to $\neg\wedge$ -formulas inside the property.

Solution (Hint 2). Let $P(\varphi)$ be “if φ is a $\neg\wedge$ -formula, then $\text{nodes}(\varphi) > \text{neg}(\varphi)$ ”. The immediate parts of a $\neg\wedge$ -formula are $\neg\wedge$ -formulas.

Solution. For a formula φ , let $P(\varphi)$ be: “if φ is a $\neg\wedge$ -formula, then $\text{nodes}(\varphi) > \text{neg}(\varphi)$ ”. We prove $P(\varphi)$ for every formula φ by [Theorem 4.3.1](#); for a $\neg\wedge$ -formula it is the claim.

The immediate parts of a $\neg\wedge$ -formula are $\neg\wedge$ -formulas, since every symbol of a part is a symbol of the whole.

Atoms. $\text{nodes}(p) = 1 > 0 = \text{neg}(p)$.

$\perp$, and the connectives $\vee, \rightarrow, \leftrightarrow$. The formulas $\perp$ and $(\varphi \circ \psi)$ with $\circ$ one of $\vee, \rightarrow, \leftrightarrow$ are not $\neg\wedge$ -formulas, so for them P holds vacuously, whatever the clauses chosen for them.

Negation. Suppose $P(\varphi)$, and let $\neg\varphi$ be a $\neg\wedge$ -formula. Then φ is one, so $\text{nodes}(\varphi) > \text{neg}(\varphi)$ by $P(\varphi)$, and adding 1 to both sides gives $\text{nodes}(\neg\varphi) > \text{neg}(\neg\varphi)$.

Conjunction. Suppose $P(\varphi)$ and $P(\psi)$, and let $(\varphi \wedge \psi)$ be a $\neg\wedge$ -formula. Then φ and ψ are, so $\text{nodes}(\varphi) > \text{neg}(\varphi)$ and $\text{nodes}(\psi) > \text{neg}(\psi)$. Adding the two inequalities and then 1,

$$\text{nodes}((\varphi \wedge \psi)) = \text{nodes}(\varphi) + \text{nodes}(\psi) + 1 > \text{neg}(\varphi) + \text{neg}(\psi) = \text{neg}((\varphi \wedge \psi)).$$

(Since the counts are integers, each hypothesis gives at least 1 to spare, so the left side exceeds the right by at least 3.)

By [Theorem 4.3.1](#), $P(\varphi)$ holds for every formula, which proves the claim.

Marking scheme. (10 points)

- 2: states the property for every formula, in a form that covers exactly the $\neg\wedge$ -formulas, with the recursive counts.
- 1: atom case.
- 2: negation case, from the hypothesis for φ , after checking that φ is a $\neg\wedge$ -formula.
- 3: conjunction case, using the hypotheses for **both** parts.
- 2: names the induction principle and says why $\perp$ and the other connectives are harmless (vacuous).

Fatal errors (0 for the step): applying the hypothesis to the formula being proved; a conjunction case that uses one part only and says “similarly” for the other.

Exercise. C4: Challenge: Inside the folded proofs

These parts are about the proofs in the second-reading folds of this section; read the folds first.

- (a) In the proof of [Theorem 4.3.1](#), on what is the strong induction performed?
- (b) Where in the proof of [Theorem 4.3.4](#) is [Theorem 4.3.3](#) used, and why is it not needed

for uniqueness?

(c) In the fold “Sets built from a base”, prove part (i).

Solution. (a) On the position i of an entry in a formation sequence $\chi_1, \dots, \chi_m$ of the given formula, with the property “if $1 \leq i \leq m$, then χ_i has P ”. Not on the height, which at that point is not yet known to be well defined.

(b) In the inductive step, to define $h(\chi)$ for a formula χ of length $n + 1$: unique readability says which clause applies and to which parts, so the clauses give exactly one value. That two functions obeying the clauses agree needs no unique readability: by [Theorem 4.3.1](#), if f and f' both obey the clauses, they agree at atoms and $\perp$; and if they agree at φ (and ψ), then at $\neg\varphi$ both equal $g_{\neg}(\varphi, f(\varphi))$, and at $(\varphi \circ \psi)$ both equal $g_{\circ}(\varphi, \psi, f(\varphi), f(\psi))$, whichever way the string may be split.

(c) Let C be a closed collection, χ a formula, and $\chi_1, \dots, \chi_m$ a formation sequence with $\chi_m = \chi$. We prove, by Theorem 4.2.2 on i , the property “if $1 \leq i \leq m$, then $\chi_i \in C$ ”. Let $i \in \mathbb{N}$ and suppose the property for every $k < i$. If $i = 0$ or $i > m$, it holds vacuously. Otherwise: an atom or $\perp$ is in C ; if $\chi_i = \neg\chi_j$ with $j < i$, then $\chi_j \in C$ by hypothesis, so $\chi_i \in C$; if $\chi_i = (\chi_j \circ \chi_l)$ with $j, l < i$, then $\chi_j, \chi_l \in C$, so $\chi_i \in C$. By Theorem 4.2.2 the property holds for every i , and at $i = m$ it gives $\chi \in C$. Since every formula is the last entry of some formation sequence, every formula is in C .

Marking scheme. (6 points)

- 2: (a).
- 2: (b).
- 2: (c): 1 for the strong induction set up on positions, 1 for the three cases.

Exercise. C5: Challenge: formulas without parentheses

Write connectives in front of their parts and drop all parentheses. The **Polish formulas** are given by the clauses: every atom and $\perp$ is a Polish formula; if φ is one, so is $N\varphi$; if φ and ψ are, so are $K\varphi\psi$, $A\varphi\psi$, $C\varphi\psi$ and $E\varphi\psi$ (for $\wedge, \vee, \rightarrow, \leftrightarrow$); and nothing else is. Give each symbol a **weight**: +1 for an atom or $\perp$, 0 for N , -1 for each of K, A, C, E . The weight $w(u)$ of a string u is the sum of the weights of its symbols.

- (a) Write $((p \wedge \neg q) \rightarrow p)$ and $\neg(p \vee (q \leftrightarrow r))$ as Polish formulas.
- (b) Prove that every Polish formula has weight 1, and that every proper initial segment of it (the empty one included) has weight at most 0.
- (c) Deduce that if $K\varphi\psi$ and $K\varphi'\psi'$ are the same string, for Polish formulas $\varphi, \psi, \varphi', \psi'$, then $\varphi = \varphi'$ and $\psi = \psi'$.
- (d) Read $CApqNr$ by the count, and write it in ordinary notation.

Solution (Hint 1).

- (b) is an induction over the Polish clauses, set up as in the fold “Sets built from a base”: the Polish formulas with the property form a collection closed under the Polish clauses. Use a strengthened property, as in [Lemma 4.3.2](#).

Solution (Hint 2). (c): if $\varphi \neq \varphi'$, one of them is a proper initial segment of the other.

Solution. (a) CKpNqp and NApEqr.

Read “nothing else is” as in the fold “Sets built from a base”: the Polish formulas are the smallest collection of strings that contains every atom and $\perp$ and is closed under the Polish clauses. So if a property holds at atoms and $\perp$ and passes from the parts to the whole at each clause, the Polish formulas with it form such a collection and include every Polish formula. In the same way every Polish formula is non-empty, and is an atom, $\perp$, $N\varphi$, or $K\varphi\psi$, $A\varphi\psi$, $C\varphi\psi$, $E\varphi\psi$ for Polish φ, ψ (the strings of these forms make such a collection); (c) and (d) use this.

(b) By induction over the clauses, the property being “weight 1, and every proper initial segment has weight ≤ 0 ”.

Atom or $\perp$. Weight 1; its only proper initial segment is the empty string, of weight 0.

$N\varphi$. Weight $0 + 1 = 1$. A proper initial segment is empty (weight 0), or N followed by a proper initial segment u of φ (weight $0 + w(u) \leq 0$).

$K\varphi\psi$ (and likewise A, C, E). Weight $-1 + 1 + 1 = 1$. A proper initial segment is of one of four kinds: empty (weight 0); K followed by an initial segment u of φ other than φ itself (weight $-1 + w(u) \leq -1$); $K\varphi$ (weight $-1 + 1 = 0$); or $K\varphi$ followed by a proper non-empty initial segment u of ψ (weight $0 + w(u) \leq 0$).

(c) Both φ and φ' begin at the second symbol. If they differ, the shorter is an initial segment of the longer (both are initial segments of the string after K), a proper one, and non-empty, since every Polish formula has a symbol. By (b) the shorter has weight 1, and as a proper initial segment of the longer it has weight ≤ 0 : impossible. So $\varphi = \varphi'$, and ψ, ψ' are both what follows $K\varphi$: equal.

(d) Read from the symbol after C : the running weight is -1 after A , 0 after p , and first reaches 1 after q . By (b), a Polish formula ends exactly where its running weight first reaches 1, so the first part of C is Apq and the second is Nr . (From the first symbol the running weight is $-1, -2, -1, 0, 0, 1$, reaching 1 only at the end.) In ordinary notation: $((p \vee q) \rightarrow \neg r)$.

Marking scheme. (10 points)

- 2: (a).
- 4: (b), with the induction set up over the clauses and all three kinds of non-empty proper initial segment of $K\varphi\psi$.
- 3: (c), with the non-emptiness of the shorter part named.
- 1: (d).

Fatal errors (0 for the step): in (b), proving only “weight 1” (then (c) has nothing to use: the strengthening is the point); in (c), assuming the two parts have equal length.