1.6 Induction over any recursively defined structures

Slide 120

Induction over any recursively defined structures

With thanks to Krysia Broda, Alexander J Summers, Tim Wood, and Rakhilya Mekhtieva

Slide 121

Induction over any recursively defined structures – Inductive Definitions lead to Inductive Principles

- Every inductively defined set gives rise to a successor relation (e.g. $+1$ for $\mathbb{N}$, or Node for Trees.
- Every inductively defined relation gives rise to a successor relation.
- Every inductively defined function gives rise to a successor relation.

Therefore,

- Every inductively defined set gives rise to an inductive principle.
- Every inductively defined relation gives rise to an inductive principle.
- Every inductively defined function gives rise to an inductive principle.

Today we shall study inductions in the more general setting.

In the previous two weeks we studied induction over inductively defined sets, and now we generalize over inductively defined functions and relations. It is not surprising that induction can be generalized in such a manner, because relations can be represented through sets, and functions can also be represented through sets.

Motivation: : The "mystery" function M

Slide 122

Motivation: The "mystery" function M

```
M :: Int -> Int
M m = M'(m, 0, 1)

M' :: (Int, Int, Int) -> Int
M' (i,cnt,acc)
    | i==cnt       = acc
    | otherwise    = M'(i,cnt+1,2*acc)
```

The value of M(3) is

$$\begin{aligned} M(3) &= M'(3,0,1) && \text{by def. M} \\ &= M'(3,1,2) && \text{by def M}' \text{ - second case} \\ &= M'(3,2,4) && \text{by def M}' \text{ - second case} \\ &= M'(3,3,8) && \text{by def M}' \text{ - second case} \\ &= 8 && \text{by def M}' \text{ - first case} \end{aligned}$$

In general, what is the value of M(m)? **Assrt_1**: $\forall m : \mathbb{N}.\ M(m) = 2^m$

Slide 123

Motivation: The "mystery" function M - 2

```
M m = M'(m, 0, 1)
M' (i,cnt,acc)
   | i==cnt       = acc
   | otherwise    = M'(i,cnt+1,2*acc)
```

M′ might feel "contrived": in general, in order to calculate M′(_, _, _) we need to call M′(_, _, _) with larger rather than smaller arguments.

Why do we care about M′?

- 1st answer Because M′ is tail-recursive
- 2nd answer Because M′ translated to imperative programming corresponds to a loop

The following Java loop corresponds to the functions M and M′.

```
1   cnt = 0;
2   acc = 1;
3   while !(m==cnt) {
4       cnt = cnt+1;
5       acc = 2*acc;
6    }
7    return acc;
```

Slide 124

Motivation: The "mystery" function M - 3

```
M = M'(m, 0, 1)
M' (i,i,acc) = acc
M' (i,cnt,acc) = M'(m,cnt+1,2*acc)
```

We want to establish
Assrt_2: $\forall m, cnt, acc, p : \mathbb{N}.\ [\ \mathsf{M}'(m, cnt, acc) = p \rightarrow p = 2^{(m-cnt)} * acc\ ]$

Challenge: M′ defined in terms of *larger*, rather than *smaller* values.
Approaches

- 1st Approach Find some *measure* which decreases with each recursive call.
- 2nd Approach Explicitly count the number of recursive calls in the execution of M′.
- 3r Approach New induction principle.

We now sketch how the first approach would work:

We can show that if $m < cnt$, then $\mathsf{M}'(m, cnt, acc)$ will not terminate. Therefore, to show **Assrt_2** is suffices to show **Assrt_2′** from below:

Assrt_2' $\forall m, cnt, acc : \mathbb{N}.\ [\ m \geq cnt \rightarrow \mathsf{M}'(m, cnt, acc) = 2^{(m-cnt)} * acc\ ]$

In order to be able to prove **Assrt_2'**, we reformulate it as follows

Assrt_3 $\forall k : \mathbb{N}.$

$$\forall m, cnt, acc : \mathbb{N}.\ (\ k = m - cnt \rightarrow \mathsf{M}'(m, cnt, acc) = 2^{(m-cnt)} * acc\ ).$$

and prove that **Assrt_3** $\rightarrow$ **Assrt_2′** .

It remains to prove **Assrt_3**. This can be done straightforwardly by induction over k. We apply the mathematical induction principle on **Assrt_3** and obtain

$\forall m, cnt, acc : \mathbb{N}.\ (\ 0 = m - cnt \rightarrow \mathsf{M}'(m, cnt, acc) = 2^{(m-cnt)} * acc\ )$
$\wedge$

$$
\begin{array}{l}
\forall k : \mathbb{N}. \\
[\quad \forall m, cnt, acc : \mathbb{N}.\ (\ k = m{-}cnt \rightarrow \mathsf{M}'(m, cnt, acc) = 2^{(m-cnt)} * acc\ ) \\
\quad\ \rightarrow \\
\quad \forall m, cnt, acc : \mathbb{N}.\ (\ k+1 = m{-}cnt \rightarrow \mathsf{M}'(m, cnt, acc) = 2^{(m-cnt)} * acc\ )\quad] \\
\longrightarrow \\
\forall k : \mathbb{N}. \forall m, cnt, acc : \mathbb{N}.\ (\ k = m{-}cnt \rightarrow \mathsf{M}'(m, cnt, acc) = 2^{(m-cnt)} * acc\ )
\end{array}
$$

We leave the rest as exercise.

2nd Approach: Explicitly count the number of recursive calls in the calculation of M' .

We encode the number of recursive calls in the calculation M'into a new function M''.

We define $\mathsf{M}'' : \mathbb{N} \times \mathbb{N} \times \mathbb{N} \rightarrow \mathbb{N} \times \mathbb{N}$ through:

$$
\begin{array}{ll}
\text{M\_4} & m = cnt \quad \rightarrow \quad \mathsf{M}''(m, cnt, acc) = (0, acc). \\
\text{M\_5} & m \neq cnt \ \wedge\ \mathsf{M}''(m, cnt{+}1, 2 * acc) = (k, n) \quad \rightarrow \mathsf{M}''(m, cnt, acc) = (k{+}1, n)
\end{array}
$$

The following assertions hold

Assrt_4 $\forall s : \mathbb{N}.\ \forall m, cnt, acc, n : \mathbb{N}.\ (\ \mathsf{M}''(m, cnt, acc) = (s, n) \rightarrow n = 2^{(m-cnt)} * acc\ ).$

Assrt_5 $\forall m, cnt, acc, n : \mathbb{N}.\ (\mathsf{M}'(m, cnt, acc) = n\ \rightarrow\ \mathsf{M}''(m, cnt, acc) = (m - cnt, n)\ )$

Proving that **Assrt_4** $\wedge$ **Assrt_5** $\rightarrow$ **Assrt_2** is easy. It remains to prove **Assrt_4** and **Assrt_5**.

We first look at **Assrt_4**. This can be proven by induction over s. Namely, application of the induction principle on **Assrt_4** gives

$$
\begin{array}{l}
\quad \forall m, cnt, acc, n.\ (\ \mathsf{M}''(m, cnt, acc) = (0, n) \rightarrow n = 2^{(m-cnt)} * acc\ ) \\
\quad \wedge \\
\forall k : \mathbb{N}. \\
[\quad \forall m, cnt, acc, n : \mathbb{N}.\ (\ \mathsf{M}''(m, cnt, acc) = (k, n) \rightarrow n = 2^{(m-cnt)} * acc\ ) \\
\quad\ \rightarrow \\
\quad \forall m, cnt, acc, n : \mathbb{N}.\ (\ \mathsf{M}''(m, cnt, acc) = (k+1, n) \rightarrow n = 2^{(m-cnt)} * acc\ )\quad] \\
\longrightarrow \\
\forall s : \mathbb{N}.\ \forall m, cnt, acc, n : \mathbb{N}.\ (\ \mathsf{M}''(m, cnt, acc) = (s, n) \rightarrow n = 2^{(m-cnt)} * acc\ ).
\end{array}
$$

We leave what remains of of the proof of **Assrt_4** as exercise.

We can prove **Assrt_5** by induction over $m{-}cnt$. That is, we will prove that $\forall k, m, cnt, acc, n : \mathbb{N}.\ [m - cnt = k\ \wedge\ \mathsf{M}'(m, cnt, acc) = n \quad \rightarrow \quad \mathsf{M}''(m, cnt, acc) = (m - cnt, n)\ ]$

Slide 125

Induction over recursive functions - motivation revisited

- It is *possible* to reason about functions by using mathematical induction.
- Such reasoning is often *indirect*. E.g., the measure `m − cnt` in 1st Approach, or the number of recursive calls in 2nd approach are extraneous to **Assrt_2**.
- We want something better.
- In *mathematical* and in *structural* induction we argue that a property is "inherited" from any elements to their "successors". E.g., 4 is a "successor" of 3, and $3 : 4 : []$ is a successor of $4 : []$. Here, "successor" means "is constructed from".
- Can we generalize the concept of "successor"?
  E.g. can the term $G'(3, 2, 4)$ be seen as a successor of $G'(3, 3, 8)$?
- Indeed, in *induction over recursively defined relations and functions*; "successor" is generalized to mean "is defined in terms of".
- Induction over the definition of functions or relations often allows for more elegant proofs.

Slide 126

Inductive definitions lead to inductive principles

- An inductive definition consists of a finite set of "primitive" cases, and a finite set "composite", derived cases.
- An inductive definition leads to an inductive principle.
- A set can be defined inductively.
- A relation may be defined inductively.
- A function may be defined inductively.

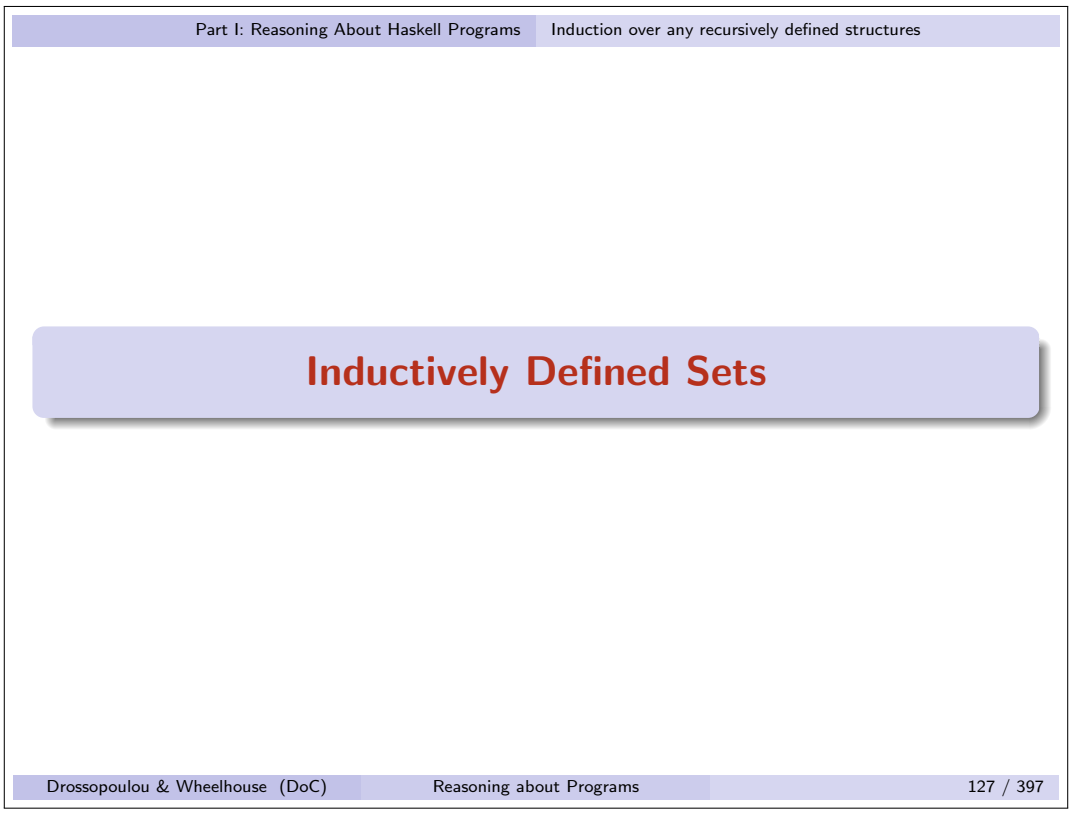
Any relation may be represented through the set of its elements,. Therefore an inductive

principle which applies to the set of the elements in this relation also applies to the relation itself.

Similarly, a function may be represented through a set of pairs. Therefore, if we can apply induction to reason about the elements in the set of pairs characterizing the function, we apply induction to the function itself.

Inductively Defined Sets

Slide 127



Slide 128

Inductively defined set

The set $\mathbb{S}_{\mathbb{N}}$ is defined over the alphabet Zero and Succ through the rules

R1 $\quad \mathsf{Zero} \in S_{\mathbb{N}}$

R2 $\quad \forall n.\, [n \in S_{\mathbb{N}} \;\rightarrow\; \mathsf{Succ}\ n \in S_{\mathbb{N}}]$

Show how we can derive that Succ (Succ (Succ Zero)) $\in S_{\mathbb{N}}$.

We obtain that Succ (Succ (Succ Zero)) $\in S_{\mathbb{N}}$ as follows:

(1)	Zero $\in S_{\mathbb{N}}$	By **R1**
(2)	Succ Zero $\in S_{\mathbb{N}}$	By (1) and **R2**
(3)	Succ (Succ Zero) $\in S_{\mathbb{N}}$	By (2) and **R2**
(4)	Succ (Succ (Succ Zero)) $\in S_{\mathbb{N}}$	By (3) and **R2**

The definition of $\mathbb{S}_{\mathbb{N}}$ is essentially the mathematical fomrulation of the definition of `Nat`-s from Haskell. Thus, it is not surprising that the inductive principle for `Nat` is essentially the same as that for $S_{\mathbb{N}}$.

Slide 129

Inductively defined set leads to inductive principle

The set $S_{\mathbb{N}}$ is defined over the alphabet Zero and Succ through the rules

R1 $\quad \text{Zero} \in S_{\mathbb{N}}$

R2 $\quad \forall n.\, [\, n \in S_{\mathbb{N}} \;\rightarrow\; \text{Succ}\ n \in S_{\mathbb{N}}\,]$

For a property $Q \subseteq S_{\mathbb{N}}$ we obtain the inductive principle

$$
\begin{array}{l}
Q(\text{Zero}) \\
\quad \wedge \\
\forall m \in S_{\mathbb{N}}.\, [\, Q(m) \;\rightarrow\; Q(\text{Succ}\ m)\,] \\
\quad \longrightarrow \\
\forall n \in S_{\mathbb{N}}.Q(n)
\end{array}
$$

Slide 130

Induct. defined set - 2

The set Tree is defined through the rules

R3 $\quad i \in \mathbb{N} \;\rightarrow\; \text{Leaf}\ i \in \text{Tree}$

R4 $\quad \forall id\, t1, t2 \in \text{Tree}.\, c \in \text{Char}.\ \text{Node}\ c\ t1\ t2 \in \text{Tree}$

Show how we can derive that Node 'a' (Leaf 5) (Node 'b' (Leaf 9)(Leaf 3)) ∈ Tree

Slide 131

Induct. defined set leads to inductive principle - 2

The set Tree is defined through the rules

R3 $\quad i \in \mathbb{N} \;\rightarrow\; \text{Leaf}\, i \in \text{Tree}$

R4 $\quad \forall id\, t1, t2 \in \text{Tree}.\, c \in \text{Char}.\, \text{Node}\, c\, t1\, t2 \in \text{Tree}$

For a property $Q \subseteq \text{Tree}$ we obtain the inductive principle

$$\begin{array}{l} \forall i \in \mathbb{N}.Q(\text{Leaf}\, i) \\ \quad \wedge \\ \forall t1, t2 \in \text{Tree}.\forall c \in \text{Char}.[\; Q(t1) \wedge Q(t2) \;\rightarrow\; Q(\text{Node}\, c\, t1\, t2)\,] \\ \quad \longrightarrow \\ \forall t \in \text{Tree}.Q(t) \end{array}$$

Slide 132

Induct. defined set - 3

The set $OL \subseteq \mathbb{N}^*$ is defined through the rules

R5 $\quad [] \in OL$

R6 $\quad \forall i \in \mathbb{N}.\; i : [] \in OL$

R7 $\quad \forall i, j \in \mathbb{N}, js \in \mathbb{N}^*.\; [\; i \leq j \wedge j : js \in OL \;\rightarrow\; i : j : js \in OL\;]$

We use [] for empty sequence, and : for sequence concatenation.

Show how we can derive that $5 : 7 : 7 : 12 : [] \in OL$

Induct. defined set leads to inductive principle - 3

The set $OL \subseteq \mathbb{N}^*$ is defined through the rules

R5 $[] \in OL$

R6 $\forall i \in \mathbb{N}.\ i : [] \in OL$

R7 $\forall i, j \in \mathbb{N}, js \in \mathbb{N}^*.\ [\ i \leq j \wedge j : js \in OL \ \rightarrow\ i : j : js \in OL\ ]$

We use [] for empty sequence, and : for sequence concatenation.

For property $Q \subseteq \mathbb{N}^*$, the definition of OL gives the inductive principle

$$
\begin{array}{l}
Q([]) \\
\quad \wedge \\
\forall i \in \mathbb{N}. Q(i : []) \\
\quad \wedge \\
\forall i, j \in \mathbb{N}, js \in \mathbb{N}^*.[\ i \leq j \wedge j : js \in OL \wedge Q(j : js) \ \rightarrow\ Q(i : j : js)\ ] \\
\quad \longrightarrow \\
\forall ns \in OL. Q(ns)
\end{array}
$$

Slide 134

Inductively Defined Relations

Slide 135

Inductively defined relation - 1

The predicate $SL \subseteq \mathbb{N} \times \mathbb{N}$, describing the ordering "strictly less than":

R8 $\forall k \in \mathbb{N}.\ SL(0, k+1)$

R9 $\forall m, n \in \mathbb{N}.[\ SL(m,n) \ \rightarrow \ SL(m+1, n+1)\ ]$

Show how we can derive that $SL(2,5)$

Slide 136

Inductively defined relation - 1

The predicate $SL \subseteq \mathbb{N} \times \mathbb{N}$, describing the ordering "strictly less than":

R8 $\quad \forall k \in \mathbb{N}.\ SL(0, k+1)$

R9 $\quad \forall m, n \in \mathbb{N}.[\ SL(m,n) \ \rightarrow\ SL(m+1, n+1)\ ]$

For property $Q \subseteq \mathbb{N} \times \mathbb{N}$, the definition of SL gives the inductive principle

$$\begin{array}{l} \forall k \in \mathbb{N}.\ Q(0, k+1) \\ \quad \wedge \\ \forall m, n \in \mathbb{N}.[\ SL(m,n) \wedge Q(m,n) \ \rightarrow\ Q(m+1, n+1)\ ] \\ \quad \longrightarrow \\ \forall m, n \in \mathbb{N}.[\ SL(m,n) \ \rightarrow\ Q(m,n)\ ] \end{array}$$

Slide 137

Induct. defined predicate - 2

The predicate $Even \subseteq S_{\mathbb{N}}$

R12 $\quad Even(\text{Zero})$

R13 $\quad \forall n \in S_{\mathbb{N}}.[\ Even(n) \ \rightarrow\ Even(\text{Succ (Succ } n))\ ]$

Show how we can derive that $Even(\text{Succ (Succ (Succ (Succ Zero))))}$

Slide 138

Induct. defined predicate to inductive principle - 2

The predicate $Even \subseteq S_{\mathbb{N}}$

R12 $Even(\text{Zero})$

R13 $\forall n \in S_{\mathbb{N}}.\, [\, Even(n) \;\rightarrow\; Even(\text{Succ (Succ } n)) \,]$

For property $Q \subseteq S_{\mathbb{N}}$, the definition of *Even* gives the inductive principle

$$
\begin{array}{l}
Q(\text{Zero}) \\
\quad \wedge \\
\forall n \in S_{\mathbb{N}}.\, [\, Even(n) \wedge Q(n) \;\rightarrow\; Q(\text{Succ (Succ } n)) \,] \\
\quad \longrightarrow \\
\forall n \in S_{\mathbb{N}}.\, [\, Even(n) \;\rightarrow\; Q(n) \,]
\end{array}
$$

Slide 139

Induct. defined predicate - 3

The predicate $Odd \subseteq S_{\mathbb{N}}$

R10 $Odd(\text{Succ Zero})$

R11 $\forall n \in S_{\mathbb{N}}.\, [\, Odd(n) \;\rightarrow\; Odd(\text{Succ (Succ } n)) \,]$

Show how we can derive that $Odd(\text{Succ (Succ (Succ Zero)))}$

Slide 140

Induct. defined predicate to inductive principle - 3

The predicate $Odd \subseteq S_{\mathbb{N}}$

R10 $\quad Odd(\text{Succ Zero})$

R11 $\quad \forall n \in S_{\mathbb{N}}.\,[\,Odd(n) \;\rightarrow\; Odd(\text{Succ (Succ } n))\,]$

For property $Q \subseteq S_{\mathbb{N}}$, the definition of Odd gives the inductive principle

$$
\begin{array}{l}
Q(\text{Succ Zero}) \\
\quad \wedge \\
\forall n \in S_{\mathbb{N}}.\,[\,Odd(n) \wedge Q(n) \;\rightarrow\; Q(\text{Succ (Succ } n))\,] \\
\quad \longrightarrow \\
\forall n \in S_{\mathbb{N}}.\,[\,Odd(n) \;\rightarrow\; Q(n)\,]
\end{array}
$$

Slide 141

Is that important?

Assume we want to show $\forall n \in S_{\mathbb{N}}.\,[\,Odd(n) \;\rightarrow\; Q(n)\,]$.
Compare the inductive principle derived from the definition of Odd

$$
\begin{array}{l}
Q(\text{Succ Zero}) \\
\quad \wedge \\
\forall n \in S_{\mathbb{N}}.\,[\,Odd(n) \wedge Q(n) \;\rightarrow\; Q(\text{Succ (Succ } n))\,] \\
\quad \longrightarrow \\
\forall n \in S_{\mathbb{N}}.\,[\,Odd(n) \;\rightarrow\; Q(n)\,]
\end{array}
$$

with the inductive principle derived from the definition of $S_{\mathbb{N}}$

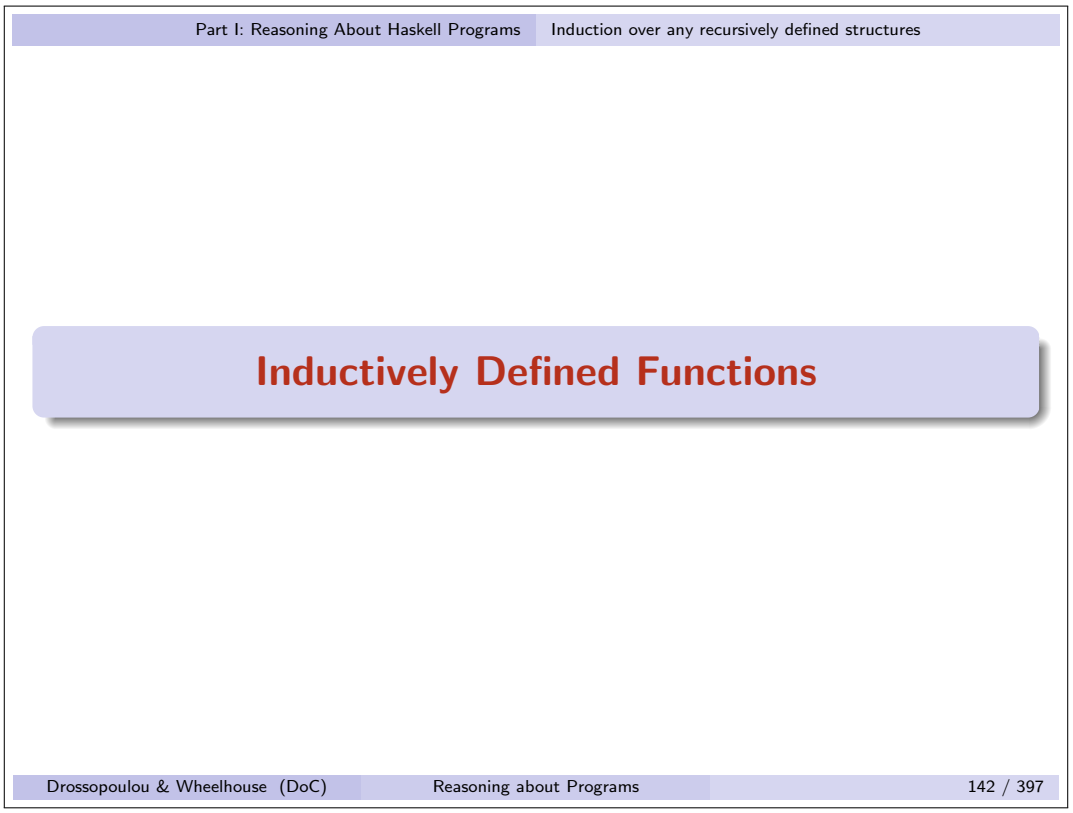
$$
\begin{array}{l}
Odd(\text{Zero}) \rightarrow\; Q(\text{Zero}) \\
\quad \wedge \\
\forall n \in S_{\mathbb{N}}.\,[\,(\,Odd(n) \rightarrow Q(n)\,) \;\rightarrow\; (\,Odd(\text{Succ } n) \rightarrow Q(\text{Succ } n)\,)\,] \\
\quad \longrightarrow \\
\forall n \in S_{\mathbb{N}}.\,[\,Odd(n) \;\rightarrow\; Q(n)\,]
\end{array}
$$

In the tutorial question we will see that sometimes it is much easier to prove according

to the inductive principle derived from the definition of $Even$, rather than with the one derived from the definition of $S_{\mathbb{N}}$.

Inductively Defined Functions

Slide 142



Slide 143

Inductiv. defined function - 1

The function F is defined as

```
F :: Int -> Int
F 0 = 0                    -- R20
F i = 1 + F (i-3)          -- R21
```

Give the calculation of F 15.

$$F\,15 = 1 + F\,12 = 2 + F\,9 = 3 + F\,6 = 4 + F\,3 = 4 + F\,0 = 5.$$

Slide 144

Inductiv. defined function - 1a

```
F :: Int -> Int
F 0 = 0                    -- R20
F i = 1 + F (i-3)          -- R21
```

R20 means that $F\,0 = 0$.

But how do we reflect the meaning of **R21**?

A $\forall \imath : \mathbb{Z}.\ F\,\imath = 1 + F\,(\imath - 3)$

B $\forall \imath : \mathbb{Z}.\ [\ \imath \neq 0\ \rightarrow\ F\,\imath = 1 + F\,(\imath - 3)\ ]$

C $\forall \imath : \mathbb{Z}.\ [\ \imath > 0 \wedge (\exists k : \mathbb{Z}. i = 3 * k)\ \rightarrow\ F\,\imath = 1 + F\,(\imath - 3)\ ]$

D $\forall \imath, \jmath : \mathbb{Z}.\ [\ \imath \neq 0 \wedge F\,\imath = \jmath\ \rightarrow\ F\,(\imath - 3) = \jmath - 1\ ]$

E $\forall \imath, \jmath : \mathbb{Z}.\ [\ \imath \neq 0 \wedge F\,(\imath - 3) = \jmath\ \rightarrow\ F\,\imath = 1 + \jmath\ ]$

E and **E** are *not* equivalent!

A is invalid; take i = 0 as a counterexample. The meaning of **B** is unclear in the presence of non-termination; eg take i = 2.

C is valid; but proving it requires a requires to reason about the function as a whole.

Slide 145

Inductiv. defined function - 1b

thank you to Michelle Zhou

```
F :: Int -> Int
F 0 = 0                    -- R20
F i = 1 + F (i-3)          -- R21
```

For **R21**, we compare

D $\quad \forall \imath, \jmath : \mathbb{Z}.\ [\ \imath \neq 0 \wedge F\,\imath = \jmath \ \rightarrow\ F\,(\imath - 3) = \jmath - 1\ ]$

E $\quad \forall \imath, \jmath : \mathbb{Z}.\ [\ \imath \neq 0 \wedge F\,(\imath - 3) = \jmath \ \rightarrow\ F\,\imath = 1 + \jmath\ ]$

where $F_{\mathbf{D}}$ is short for F according to **D**, and $F_{\mathbf{E}}$ is short for F according to **E**:

$F_{\mathbf{D}}\ 0$	=	0	$F_{\mathbf{E}}\ 0$	=	i
$F_{\mathbf{D}}\ -3$	=	-1	$F_{\mathbf{E}}\ -3$	=	?
$F_{\mathbf{D}}\ 3$	=	???	$F_{\mathbf{E}}\ 3$	=	1

D says that the value of F i is used to calculate the value of F (i − 3).
E says that the value of F (i − 3) is used to calculate the value of F i.

We can show that for i $\neq 0$, the term F i terminates is and only if F(i -3) terminates. Therefore, **D** and **E** are mathematically equivalent. However, the equation in **R21** describes the value of the LHS (ie F i) in terms of the RHS (ie $1 +$ F (i -3)). It says that if F (i -3) terminates and returns some value i, then F (i -3) terminates too and returns i + 1.

Therefore, for our purposes, where we are interested in reflecting the number of recursive calls involved, we chose **E**, which says that the value of F i − 3 is used to calculate the value of F i.

Slide 146

Inductiv. defined function to inductive principle - 1

Given that

R20 $\mathsf{F}\ 0 = 0$

R21 $\forall j, k : \mathbb{Z}.\ [\ j \neq 0 \wedge \mathsf{F}\,(j-3) = k\ \rightarrow\ \mathsf{F}\,j = 1 + k\ ]$

and a predicate $Q \subseteq \mathbb{Z} \times \mathbb{Z}$, we obtain the following inductive principle to prove that $\forall j, k : \mathbb{Z}.\ [\ \mathsf{F}j = k\ \rightarrow\ Q(j,k)\ ]$.

$$
\begin{array}{l}
Q(0,0) \\
\quad \wedge \\
\forall j, k : \mathbb{Z}.\ [\ j \neq 0\ \wedge \mathsf{F}\,(j-3) = k\ \wedge\ Q(j-3,k)\ \rightarrow\ Q(j,1+k)\ ] \\
\quad \longrightarrow \\
\forall j, k : \mathbb{Z}.\ [\ \mathsf{F}\,j = k\ \rightarrow\ Q(j,k)\ ]
\end{array}
$$

Note that we do not aim to show that $(*)\quad \forall j : \mathbb{Z}.\ , Q(j, Fj)$. Namely, the call of the function F need not always terminate, therefore in general, the assertion $(*)$ is far too strong. Instead, the assertion $\forall j, k : \mathbb{Z}.\ [\ \mathsf{F}\,j = k\ \rightarrow\ Q(j,k)\ ]$ says that *if* the call $\mathsf{F}\,j$ terminates, then it will return a value k which satisfies $Q(j,k)$.

Slide 147

Inductiv. defined function - 2

```
G :: (Natural,Natural) -> Natural
G(i,j) = G'(i,j,0,0)

G' :: (Natural,Natural,Natural,Natural) -> Natural
G'(i,j,cnt,acc)
     | i==cnt       = acc
     | otherwise    = G'(i,j,cnt+1,acc+j)
```

Show the calculation of $\mathsf{G}(3,7)$

$\mathsf{G}(3,7) = \mathsf{G}'(3,7,0,0) = \mathsf{G}'(3,7,1,7) = \mathsf{G}'(3,7,2,14) = \mathsf{G}'(3,7,3,21) = 21.$

In the above, the calculation of the term $\mathsf{G}'(3,7,0,0)$ results in the term $\mathsf{G}'(3,7,1,7)$; the callee contains larger arguments than the caller. Nevertheless, the callee requires one less execution steps than the caller.

Slide 148

Inductiv. defined function - 2a

```
G(i,j) = G'(i,j,0,0)
G'(i,j,cnt,acc)
      | i==cnt       = acc
      | otherwise    = G'(i,j,cnt+1,acc+j)
```

means that

R22 $\forall i, j : \mathbb{N}.\ \ \mathsf{G}(i,j) = \mathsf{G}'(i,j,0,0)$

R23 $\forall i, j, acc : \mathbb{N}.\ \ \mathsf{G}'(i,j,i,acc) = acc$

R24 $\forall i, j, cnt, acc, r : \mathbb{N}.$
$[\ i \neq cnt \wedge \mathsf{G}'(i,j,cnt+1,acc+j) = r$
$\rightarrow\ \mathsf{G}'(i,j,cnt,acc) = r\ ]$

The assertion

R22 $\forall i, j : \mathbb{N}.\ \ \mathsf{G}(i,j) = \mathsf{G}'(i,j,0,0)$

means that $\mathsf{G}'(i,j,0,0)$ terminates, and its value is equal to $\mathsf{G}(i,j)$.
That is, $\forall i, j : \mathbb{N}. \exists k : \mathbb{N}.\ [\ \mathsf{G}'(i,j,0,0) = k \wedge \mathsf{G}(i,j) = k\ ]$.

Slide 149

Inductiv. defined function to inductive principle - 2

Given that

R23 $\forall i, j, acc : \mathbb{N}.\ \ \mathsf{G}'(i, j, i, acc) = acc$

R24 $\forall i, j, cnt, acc, r : \mathbb{N}.$
$[\ i \neq cnt \ \wedge\ \mathsf{G}'(i, j, cnt + 1, acc + j) = r$
$\rightarrow\ \mathsf{G}'(i, j, cnt, acc) = r\ ]$

the inductive principle applied to predicate $Q \subseteq \mathbb{N} \times \mathbb{N} \times \mathbb{N} \times \mathbb{N} \times \mathbb{N}$ gives

$\forall i, j, acc : \mathbb{N}.\ Q(i, j, i, acc, acc)$
$\wedge$
$\forall i, j, acc, cnt, r : \mathbb{N}.$
$[\ \ i \neq cnt \ \wedge\ \mathsf{G}'(i, j, cnt + 1, acc + j) = r \ \wedge\ Q(i, j, cnt+1, acc+j, r)$
$\rightarrow\ \ Q(i, j, cnt, acc, r)\ \ ]$
$\longrightarrow$
$\forall i, j, acc, cnt, r : \mathbb{N}.\ [\ \mathsf{G}'(i, j, cnt, acc) = r\ \rightarrow\ Q(i, j, cnt, acc, r)\ ]$

The assertion $\forall i, j, acc, cntr : \mathbb{Z}.\ [\ \mathsf{G}'(i, j, cnt, acc) = r\ \rightarrow\ Q(i, j, cnt, acc, r)\ ]$ says that if the execution of the term $\mathsf{G}'(i, j, cnt, acc)$ terminates, then the value of this term (here r) satisfies the property in $Q(i, j, cnt, acc, r)$.

Slide 150

Proving that $\forall i, j : \mathbb{N}.\ \mathsf{G}(i,j) = i * j$

We want to prove $(*)$ $\forall i, j : \mathbb{N}.\ \mathsf{G}(i,j) = i * j$.

To prove this, we will show

(A) $\forall i, j : \mathbb{N}. \exists r : \mathbb{N}.\ [\ \mathsf{G}'(i,j,0,0) = r\ \wedge\ r = i * j\ ]$

To prove (A), it suffices to show:

(B) $\forall i, j : \mathbb{N}. \exists r : \mathbb{N}.\ [\ \mathsf{G}'(i,j,0,0) = r\ ]$

and

(C) $\forall i, j, r \in \mathbb{N}.\ [\ \mathsf{G}'(i,j,0,0) = r\ \rightarrow\ r = i * j\ ]$

To prove (B), it suffices to show:

(B') $\forall i, j, cnt, acc, n : \mathbb{N}.[\ i - cnt = n\ \rightarrow\ \exists r : \mathbb{N}. \mathsf{G}'(i,j,cnt,acc) = r\ ]$

To prove (C), it suffices to show:

(C') $\forall i, j, cnt, acc, r : \mathbb{N}.[\ \mathsf{G}'(i,j,cnt,acc) = r\ \rightarrow r = (i - cnt) * j + acc\ ]$

Note that (A) implies (*). Also (B)∧(C) implies (A). And also, (B') implies (B) and (C') implies (C).

Slide 151

Proving $\forall i, j, cnt, acc : \mathbb{N}.[\ G'(i, j, cnt, acc) = r \ \rightarrow r = (i - cnt) * j + acc\ ]$

We first look at the proof schema.
We take $Q(i, j, cnt, acc, r)$ from previous slide to mean $r = (i - cnt) * j + acc$.

Base Case

To Show $\forall i, j, acc : \mathbb{N}.\ acc = (i - i) * j + acc$

...

Inductive Step

Take i, j, cnt, acc, r:$\mathbb{N}$ arbitrary.
Assume that $i \neq cnt$, and $G'(i, j, cnt + 1, acc + j) = r$.
Inductive Hypothesis: $r = (i - (cnt + 1)) * j + (acc + j)$
To Show: $r = (i - cnt) * j + acc$

...

Note that the proof of both the Base Case and the Inductive Step are very simple. This is so, because all the hard work went into identifying the property (C') to prove.

Slide 152

Inductiv. defined function - 3

We define DM : $\mathbb{Z} \times \mathbb{Z} \to \mathbb{Z} \times \mathbb{Z}$ and DM′ : $\mathbb{Z} \times \mathbb{Z} \times \mathbb{Z} \times \mathbb{Z} \to \mathbb{Z} \times \mathbb{Z}$ through:

```
DM :: (Int,Int) -> (Int,Int)
DM (i,j) = DM' (i,j,0,0)

DM' :: (Int,Int,Int,Int) -> (Int,Int)
DM' (i,j,cnt,acc)
    | acc+j > i = (cnt,i-acc)
    | otherwise = DM' (i,j,cnt+1,acc+j)
```

Show the calculation of DM(16,5)

Slide 153

Inductiv. defined function - 3a

Given DM and DM′ below

```
DM (i,j) = DM' (i,j,0,0)
DM' (i,j,cnt,acc)
    | acc+j > i = (cnt,i-acc)
    | otherwise = DM' (i,j,cnt+1,acc+j)
```

We obtain the following mathematical definition:

R31 $\forall i, j : \mathbb{Z}\ \mathrm{DM}(i,j) = \mathrm{DM}'(i,j,0,0)$

R32 $\forall i, j, cnt, acc : \mathbb{Z}.$
$[\ acc + j > i \ \to\ \mathrm{DM}'(i,j,cnt,acc) \ =\ (cnt, i - acc)\ ]$

R33 $\forall i, j, cnt, acc, k1, k2 : \mathbb{Z}.$
$[\ acc + j \leq i \wedge \mathrm{DM}'(i,j,cnt{+}1,acc{+}j) = (k1,k2)$
$\to\ \mathrm{DM}'(i,j,cnt,acc) \ =\ (k1,k2)\ ]$

Slide 154

Inductiv. defined function to inductive principle - 3

R32 $\forall i, j, cnt, acc : \mathbb{Z}.\ [\ acc + j > i \ \rightarrow\ \mathrm{DM}'(i, j, cnt, acc) \ =\ (cnt, i - acc)\ ]$
R33 $\forall i, j, cnt, acc, k1, k2 : \mathbb{Z}.$
$[\ acc + j \leq i \wedge \mathrm{DM}'(i, j, cnt{+}1, acc{+}j) = (k1, k2)$
$\rightarrow\ \mathrm{DM}'(i, j, cnt, acc) \ =\ (k1, k2)\ ]$

For predicate $Q \subseteq \mathbb{Z} \times \mathbb{Z} \times \mathbb{Z} \times \mathbb{Z} \times \mathbb{Z} \times \mathbb{Z}$ we obtain inductive principle for DM':

$\forall i, j, cnt, acc : \mathbb{Z}.\ [\ \ acc + j > i \ \rightarrow Q(i, j, cnt, acc, cnt, i - acc)\ ]$
$\wedge$
$\forall i, j, cnt, acc, k1, k2 : \mathbb{Z}.$
$[\ acc{+}j \leq i \wedge \mathrm{DM}'(i, j, cnt{+}1, acc{+}j) = (k1, k2) \wedge Q(i, j, cnt{+}1, acc{+}j, k1, k2)$
$\rightarrow\ \ Q(i, j, cnt, acc, k1, k2)\ ]$
$\longrightarrow$
$\forall i, j, cnt, acc, k1, k2 : \mathbb{Z}.$
$[\ \mathrm{DM}'(i, j, cnt, acc) = (k1, k2) \ \rightarrow\ Q(i, j, cnt, acc, k1, k2)\ ]$

Show $\forall i, j \in \mathbb{Z}.\ [\ \mathrm{DM}(i, j) = (k1, k2) \ \rightarrow i = k1 * j + k2 \wedge k2 < j\ ]$ Coursework

Slide 155

Inductiv. defined function - 4

Remember the "mystery" function $\mathrm{M}' : \mathbb{N} \times \mathbb{N} \times \mathbb{N} \rightarrow \mathbb{N}$ defined through:

```
M m = M'(m, 0, 1)
M' (i,cnt,acc)
   | i==cnt       = acc
   | otherwise    = M'(i,cnt+1,2*acc)
```

This leads to the following equations

M_1 $\forall i : \mathbb{N}.\ \mathrm{M}(i) = \mathrm{M}'(i, 0, 1)$

M_2 $\forall i, acc : \mathbb{N}.\ \mathrm{M}'(i, i, acc) = acc$

M_3 $\forall i, cnt, acc, k : \mathbb{N}.$
$[\ i \neq cnt \wedge \mathrm{M}'(i, cnt{+}1, 2{*}acc) = k \ \rightarrow\ \mathrm{M}'(i, cnt, acc) = k\ ]$

The assertion

M_2 $\quad \forall i : \mathbb{N}.\, \mathsf{M}(i) = \mathsf{M}'(i, 0, 1)$

means that $\mathsf{G}'(i, j, 0, 0)$ terminates, and its value is equal to $\mathsf{M}(i)$.
Namely, $\quad \forall i : \mathbb{N}. \exists k : \mathbb{N}.\ [\ \mathsf{M}'(i, 0, 1) = k \ \wedge\ \mathsf{M}(i)\ ]$.

Slide 156

Inductiv. defined function to inductive principle - 4

Given

M_2 $\quad \forall i, acc : \mathbb{N}.\, \mathsf{M}'(i, i, acc) = acc$

M_3 $\quad \forall i, cnt, acc, k : \mathbb{N}.$
$[\ i \neq cnt \ \wedge\ \mathsf{M}'(i, cnt{+}1, 2{*}acc) = k \ \rightarrow\ \mathsf{M}'(i, cnt, acc) = k\ ]$

This gives rise to the inductive principle:

$$\begin{array}{l} \forall i, acc : \mathbb{N}.\ Q(i, i, acc, acc) \\ \wedge \\ \forall i, cnt, acc, k : \mathbb{N}. \\ \quad [\ i \neq cnt \ \wedge\ \mathsf{M}'(i, cnt{+}1, 2{*}acc) = k \ \wedge\ Q(i, cnt{+}1, 2{*}acc, k) \\ \qquad\qquad \rightarrow\ Q(m, cnt, acc, k)\ ] \\ \longrightarrow \\ \forall i, cnt, acc, k : \mathbb{N}.[\ \mathsf{M}'(i, cnt, ac) = k \ \rightarrow\ Q(m, cnt, acc, k)\ ] \end{array}$$

Slide 157

Proof schema for M is a power function

Proving
$\forall i, cnt, acc, k : \mathbb{N}.\ [\ \mathrm{M}'(i, cnt, acc) = k\ \rightarrow\ (k = 2^{(i-cnt)} * acc\ )\ ]$

Base Case

To Show $\forall i, acc : \mathbb{N}.\ acc = 2^{(i-cnt)} * acc$

...

Inductive Step

Take i, cnt, acc, k, arbitrary.
Assume that $i \neq cnt$ and that $\mathrm{M}'(i, cnt{+}1, 2{*}acc) = k$.
Inductive Hypothesis: $k = 2^{(i-(cnt+1))} * 2 * acc$
To Show: $k = 2^{(i-cnt)} * acc$

...

Again, the proofs are very easy. All the hard work went into setting up the proof schema.

Slide 158

What about termination?

- Can we reason about termination of a function through reasoning about its inductive definition?
- For example, remember

  M_2 $\forall i, acc : \mathbb{N}.\, \mathrm{M}'(i, i, acc) = acc$
  M_3 $\forall i, cnt, acc, k : \mathbb{N}.$
  $[\ i \neq cnt \wedge \mathrm{M}'(i, cnt{+}1, 2{*}acc) = k\ \rightarrow\ \mathrm{M}'(i, cnt, acc) = k\ ]$

  Can we argue over the definition of the function M′ to argue that it terminates for all values of i, cnt and acc?
- Such an argument would not be valid. Namely, when we reason over the inductive definition of the function, we implicitly assume that the function terminated in a finite number of steps.
- The only valid way to argue termination by induction is by applying on the arguments.

Conclusions – Inductive Definitions lead to Inductive Principles

- Every inductively defined set/relation/function gives rise to a successor relation - structural induction.
- Inductive definitions of relations and functions give rise to an inductive principle.

[Extra] Multiple and Well-Founded Induction

Our examples so far have involved induction on one entity. For example:

- $\forall n : \mathbb{N}.P(n)$
- $\forall \texttt{xs} : [\texttt{a}].Q(\texttt{xs})$
- $\forall \texttt{e} : \texttt{BoolExpr}.P(\texttt{e})$

where $P \subseteq \mathbb{N}$, $Q \subseteq$ `[a]`, and $R \subseteq$ `BoolExp`.

In general, we may have statements involving several universal quantifiers For example:

- $\forall a : A.\forall b : B.P(a, b)$

where A and B are sets, and where $P \subseteq A \times B$.

In such cases, one *may* need to introduce (and prove) an auxiliary lemma, which in its turn may be proven by induction. For exaple:

- the proof of $\forall$`m:Nat`.$\forall$`n:Nat`. $varaddmn =$ `add n m`

Some statements with multiple universal quantifiers can be proven by single induction. For example:

- the proof of $\forall$`xs:[a]`.$\forall$`ys:[a]`.`reverse(xs ++ ys)` = `reverse(ys) ++ reverse(xs)`

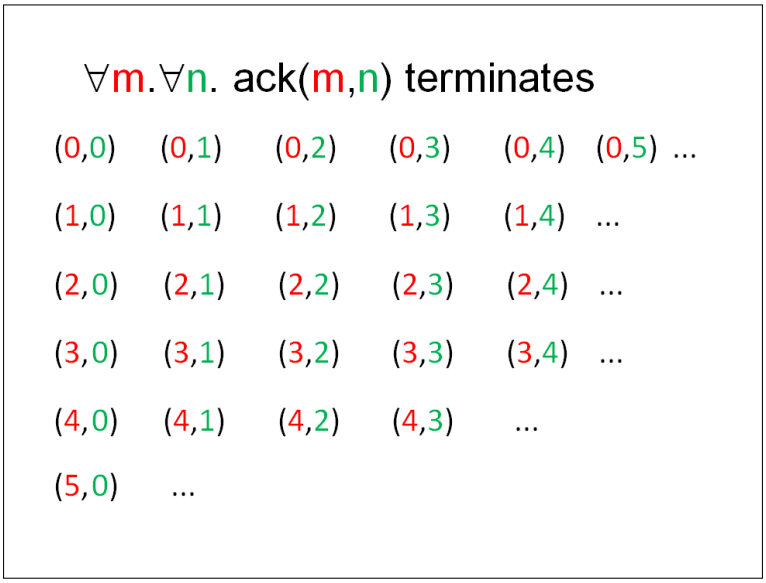
We are now going to take a look at the proof of a statement which requires auxiliary lemmas, also proven by induction. This technique as known as *multiple induction*.

We shall then introduce a *new* induction principle, known as *well-founded induction*, which allows us to prove the same properties but more elegantly. Well-founded induction is even more general than structural induction.

Consider the Ackermann function defined as follows:

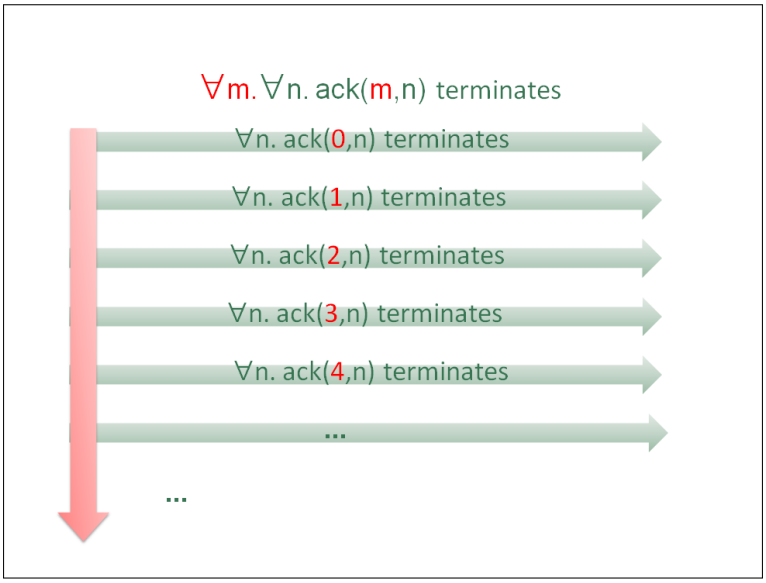
```
ack :: Int -> Int -> Int
-- Pre-condition: m >= 0, n >= 0
ack 0 n = n + 1
ack m 0 = ack (m - 1) 1
ack m n = ack (m - 1) (ack m (n - 1))
```

We want to prove that `ack m n` terminates for all `m`, `n` ≥ 0. Let us consider this property graphically:



The assertion "$\forall m.\forall n.$`ack m n` terminates" corresponds to the conjunction of infinitely many simpler assertions.

Each simpler assertion guarantees termination for all pairs in a row



[Extra] Multiple Induction

We want to prove $\forall \mathtt{m} : \mathbb{N}.\forall \mathtt{n} : \mathbb{N}.$ `ack m n` terminates.

We define $Q(\mathtt{m})$ as $Q(\mathtt{m}) \equiv \forall \mathtt{n} : \mathbb{N}.$ `ack m n` terminates.

Therefore, it suffices to prove $\forall \mathtt{m} : \mathbb{N}.Q(\mathtt{m})$.

Remember, the induction principle says,for any predicate $P \subseteq \mathbb{N}$:

$$[\ P(0)\ \wedge\ \forall k : \mathbb{N}.(\ P(k)\ \rightarrow\ P(k+1)\ )\ ]\ \rightarrow\ \forall m : \mathbb{N}.\ P(m)$$

In our case, the induction principle guarantees: [$\forall$`n` : $\mathbb{N}$. `ack 0 n` term. $\wedge$
$\forall$`k` : $\mathbb{N}$.($\forall$`n` : $\mathbb{N}$. `ack k n` term. $\rightarrow$ $\forall$`n` : $\mathbb{N}$. `ack (k + 1) n` term.)]
$\rightarrow$
$\forall$`m` : $\mathbb{N}$.$\forall$`n` : $\mathbb{N}$. `ack m n` term.

Therefore, the proof will have the following architecture:

Base Case:

To Show: $\forall$`n` : $\mathbb{N}$. `ack 0 n` terminates

...

Inductive Step:

Take `k` : $\mathbb{N}$ arbitrary:

Inductive Hypothesis:$\forall$`n` : $\mathbb{N}$. `ack k n` terminates

To Show: $\forall$`n` : $\mathbb{N}$. `ack (k + 1) n` terminates

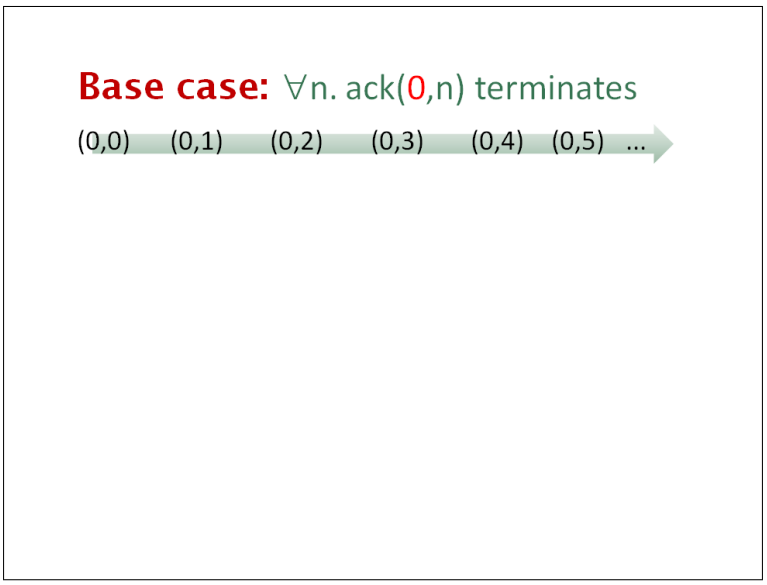
...

Proving the base case

Base Case:

To Show : $\forall$`n` : $\mathbb{N}$. `ack 0 n` terminates
The calculation of the term `n + 1` terminates.
Therefore, by definition of `ack`, the term `ack 0 n` terminates too.

The base case guarantees termination for all pairs from the top row, i.e. termination for all $(0, n) \in \{\ 0\ \} \times \mathbb{N}$.



The proof of the base case was also nice and easy :-)

Proving the inductive step

Inductive Step:

Take `k` $\in \mathbb{N}$ arbitrary:

Inductive Hypothesis: $\forall$`n` : $\mathbb{N}$. `ack k n` terminates

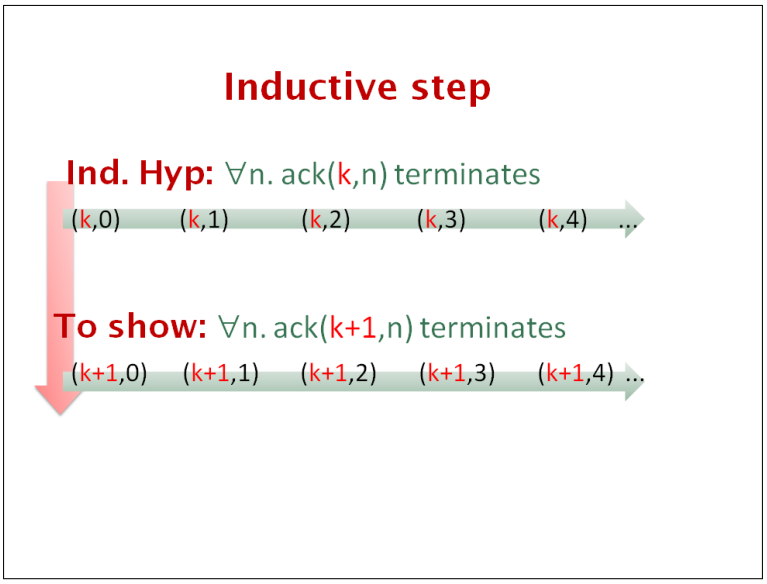
To Show: $\forall$`n` : $\mathbb{N}$. `ack (k +1 ) n` terminates

Take `n` $\in \mathbb{N}$ arbitrary.

Consider the case where `n` > 0.

Then, `ack (k + 1) n` terminates only if
(1) `ack (k + 1) (n - 1)` terminates, giving, say `q`, and
(2) `ack k q` terminates.
(2) follows from the inductive hypothesis.
But *how do we establish (1)*?
The induction hypothesis is *not* applicable on the term `ack (k + 1) (n - 1)`!

The inductive step "goes" from a row of pairs $(k, n) \in \{\ k\ \} \times \mathbb{N}$, to the "next" row of pairs $(k+1, n) \in \{\ k+1\ \} \times \mathbb{N}$, as illustared by:



We also had difficulty in completing the proof of the inductive step :-(

So, it seems that we are stuck, but are we really?

Notice that termination of `ack (k + 1) n` requires termination of `ack k ...` and of `ack ... (n - 1)`.

- In `ack k` ... the *first* argument became smaller.

- In `ack ... (n - 1)` the *second* argument became smaller.

Therefore, it seems that an inductive proof *should* be possible.

Let's revist the inductive step.

We were stuck while trying to prove:
$(*)$ $\forall$`k` : $\mathbb{N}$. ($\forall$`n` : $\mathbb{N}$.`ack k n` term. $\rightarrow$ $\forall$`n` : $\mathbb{N}$.`ack (k + 1) n` term.)

Because
$\forall$`n` : $\mathbb{N}$. `ack (k + 1) n` term.
is a universally quantified formula, the inductive principle is applicable.

So, let's formulate $(*)$ as an auxiliary lemma, and prove the conclusion by induction over `n`. If the proof of this auxiliary lemma is successful, then we can use it to prove the inductive step of the original lemma.

The Auxiliary Lemma $(*)$:

For all `k` : $\mathbb{N}$,
$\forall$`n` : $\mathbb{N}$.`ack k n` term. implies $\forall$`n` : $\mathbb{N}$.`ack (k + 1) n` term.

Proof: later...

Lets revisit the inductive step of the proof from earlier:

Inductive Step:

Take `k` $\in \mathbb{N}$ arbitrary:

Inductive Hypothesis: $\forall$`n` : $\mathbb{N}$. `ack k n` term.

To Show: $\forall$`n` : $\mathbb{N}$. `ack (k + 1) n` term.

This follows by direct application of $(*)$.

Now let's go back to the proof of this auxiliary lemma. First we give the proof schema

Auxiliary Lemma – Proof Schema

For all `k` : $\mathbb{N}$,
$\forall$`n` : $\mathbb{N}$.`ack k n` term. implies $\forall$`n` : $\mathbb{N}$.`ack (k + 1) n` term.

Proof:

Take an arbitrary `k` : $\mathbb{N}$.

Assume that
(A) $\forall$`n` : $\mathbb{N}$.`ack k n` term.

We shall show that $\forall$`n` : $\mathbb{N}$.`ack (k + 1) n` term. by induction over `n`.

BaseCase:

To Show: `ack (k + 1)` `0` term.

a proof here

Inductive Step :

Take a `j` : ℕ, arbitrary.

Inductive Hypothesis : `ack (k + 1)` `j` term.

To Show : `ack (k + 1)` `j + 1` term.

another proof here

Proving the Auxiliary Lemma - Base Case

Take an arbitrary `k` : ℕ.

Assume that
(A) ∀`n` : ℕ.`ack k n` term.

BaseCase:

To Show: `ack (k + 1) 0` terminates.

(B) `ack k 1` terminates by (A)
(C) `ack (k + 1) 0` terminates by (B), and def. of `a`

Proving the Auxiliary Lemma - Inductive Step

Take an arbitrary `k` : ℕ.

Assume that
(A) ∀`n` : ℕ.`ack k n` term.

Inductive Step :

Take an arbitrary `j` : ℕ.

Inductive Hypothesis : `ack (k + 1) j` term.

To Show : `ack (k + 1) j + 1` term.

(B): `ack (k+1) j` terminates by **Inductiv**
(C): `ack k (ack (k+1) j)` terminates by (B), and (
(D): `ack (k+1) (j+1)` terminates by (C), and (

This completes the proof of the auxiliary lemma. We have already seen that the auxiliary lemma directly proves the inductive step of the original lemma. So, we have completed the proof of termination of `ack`.

Multiple Induction or *Double Induction* are terms used to describe inductive proofs whose base case and/or inductive step require auxiliary lemmas of a similar shape to the original property, which themselves are proven by induction.

Multiple induction is an *unsurprising* application of the induction principle.

Multiple induction can be used to prove statements involving several quantifiers. For example:

$\forall a : A. \forall b : B. \forall c : C.\, P(a, b, c)$ for sets A, B, C and predicate P such that $P \subseteq A \times B \times C$.

Thus, multiple induction is a very powerful proof technique. However, as we have seen, the proofs are a bit "fiddly" and often obscure the intuition of why the property holds. We can do better

[Extra] Well-Founded Induction

The key idea behind well-founded induction is to generalise the concept of an element's "predecessor" This relies on the concpet of a *well-founded ordering*.

Well-Founded Orderings

A *well-founded ordering* over a set X is a binary relation $\prec \subseteq X \times X$ such that

- there is no infinitely decreasing chain $x_1 \mathsf{Succ} x_2 \mathsf{Succ} x_3 \mathsf{Succ} \ldots$ of elements of X

An equivalent formulation of the requirement for no infinitely decreasing chains is that every non-empty $S \subseteq X$ has a least element l with respect to $\prec$

i.e. $\emptyset \neq S \subseteq X \quad \longrightarrow \quad \exists l \in S. \forall x \in S. x \not\prec l$

Well-Founded Orderings – Examples

- X is $\mathbb{N}$ and $\prec$ is $<$ on $\mathbb{N}$
- X is the set of Haskell lists `[a]` and
  `l1` $\prec$ `l2` iff $\exists$`l3`, s.t `l2` $=$ `l1++l3` and `l3` $\neq$ `[]`
- X is the set of Haskell lists `[a]` and
  `l1` $\prec$ `l2` iff `length l1` $<$ `length l2`
- X is the set of binary trees `Tree a` and
  `t1` $\prec$ `t2` iff `height t1` $<$ `height t2`
  where `height` is defined as `height Empty` $= 0$ and `height (Node lhs x rhs)` $= 1 + max($`(height lhs)`$,$`(height rhs)`$)$
- Can you define a well-founded ordering for integers?
- X is $\mathbb{N} \times \mathbb{N}$ and
  $$(x_1, y_1) \prec (x_2, y_2) \quad \text{iff} \quad x_1 < x_2 \text{ or } (x_1 = x_2 \ \wedge \ y_1 < y_2)$$
  (the *lexicographic* ordering, referred to as $\prec_{lex}$)
  - $(0, 0)$ is the least element

- an example: $(0,0) \prec_{lex} (0,1) \prec_{lex} (1,3) \prec_{lex} (1,5) \prec_{lex} (2,6) \prec_{lex} (3,4) \prec_{lex} (3,7) \ldots$

- X is $\mathbb{N} \times \mathbb{N}$ and

$$(x_1, y_1) \prec (x_2, y_2) \quad \text{iff} \quad x_1 + y_1 < x_2 + y_2$$

  - $(0,0)$ is the least element
  - an example: $(0,0) \prec (0,1) \prec (1,1) \prec (3,0) \prec (1,5)$

We shall see later that all of the orderings above correspond to versions of multiple induction.

Well-Founded Induction – Principle

For a set S and $\prec \subseteq S \times S$ a well-founded ordering on S, the *well-founded induction principle* states:

$$[\, \forall z{:}S.\, [\, \forall y{:}S.(y \prec z \rightarrow P(y))\,] \;\rightarrow\; P(z)\,] \;\longrightarrow\; \forall x{:}S.P(x)$$

We can compare this with strong induction:

$$[\;\; P(0) \;\wedge\; \forall k{:}\mathbb{N}.\, [\forall j{:}\mathbb{N}.j \leq k \rightarrow P(j)\,] \;\rightarrow\; P(k+1)\,] \;\rightarrow\; \forall n{:}\mathbb{N}.P(n)$$

Well-Founded Induction – Proof Outline

Let $\prec$ be a well-founded ordering on a set X.

The structure of a proof of $\forall x : X.P(x)$ by well-founded induction on x is:

Inductive Step :

Take a $z{:}X$, arbitrary

Induction Hypothesis: $\forall y{:}X.\, y \prec z \rightarrow P(y)$

To show: $P(z)$

TODO - what happened to the base case?

Termination of the Ackerman Function – Revisited

Recall the Ackermann function defined above:

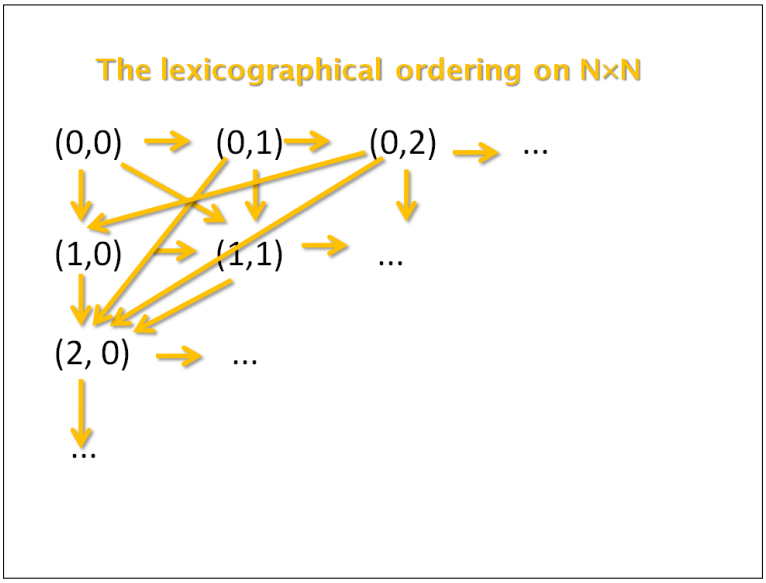
```
ack :: Int -> Int -> Int
-- REQUIRES: m>= 0, n >= 0
ack 0 n = n + 1
ack m 0 = ack (m - 1) 1
ack m n = ack (m - 1) (ack m (n - 1))
```

Prove that $\forall \mathtt{m}{:}\mathbb{N}.\forall \mathtt{n} : \mathbb{N}.\ \mathtt{ack\ m\ n}$ terminates.

We shall find an appropriate well-founded ordering $\prec \subseteq \mathbb{N} \times \mathbb{N}$, and prove $\forall(\mathtt{m},\mathtt{n}) : \mathbb{N} \times \mathbb{N}.\ \mathtt{ack\ m\ n}$ terminates.

Consider $\prec_{lex}$, the *lexicographic ordering* for $\mathbb{N} \times \mathbb{N}$

$$(m, n) \prec_{lex} (m', n') \ \Leftrightarrow\ (m < m') \vee (m = m' \wedge n < n')$$



The well-founded induction principle, applied to the termination of `ack` using the lexicographic ordering, can be written as:

$$
\begin{array}{l}
[\\
\ \forall(\mathtt{m},\mathtt{n}){:}\mathbb{N} \times \mathbb{N}.\\
\qquad [\ \forall(\mathtt{m}',\mathtt{n}'){:}\mathbb{N} \times \mathbb{N}.((\mathtt{m}',\mathtt{n}') \prec_{lex} (\mathtt{m},\mathtt{n}) \rightarrow \mathtt{ack\ m'\ n'}\ \text{terminates}))\ ]\\
\qquad\qquad \rightarrow\\
\qquad \mathtt{ack\ m\ n}\ \text{terminates}\\
]\\
\quad \longrightarrow\\
\forall(\mathtt{m},\mathtt{n}){:}\mathbb{N} \times \mathbb{N}.\ \mathtt{ack\ m\ n}\ \text{terminates}
\end{array}
$$

The schema for a proof of termination of `ack` by well-founded induction can be given as:

Inductive Step:

Take $(\mathtt{m},\mathtt{n}){:}\mathbb{N} \times \mathbb{N}$, arbitrary

Ind. Hyp.: $\forall(\mathtt{m'},\mathtt{n'}){:}\mathbb{N}\times\mathbb{N}.\ (\mathtt{m'},\mathtt{n'}) \prec_{lex} (\mathtt{m},\mathtt{n}) \rightarrow \mathtt{ack\ m'\ n'}$ terminates

To show: `ack m n` terminates

The full proof the proceeds as follows:

Inductive Step :

Take $(\texttt{m}, \texttt{n}) : \mathbb{N} \times \mathbb{N}$, arbitrary

Ind. Hyp.: $\forall(\texttt{m'},\texttt{n'}) : \mathbb{N}\times\mathbb{N}.$ `(m',n')` $\prec_{lex}$ `(m,n)` $\rightarrow$ `ack m' n'` terminates

To show: `ack m n` terminates

There are three cases to consider:

1st Case: `m` $= 0$

To show `ack 0 n` terminates

(A)	`n + 1` terminates	trivially
(B)	`ack 0 n` terminates	by (A), and definition of `acc`

2nd Case: `m` ¿ 0, `n` $= 0$

To show `ack m 0` terminates

(A)	$((\texttt{m}-1), 1) \prec_{lex} (\texttt{m}, 0)$	by definition of $\prec_{lex}$
(B)	`ack (m-1) 1` terminates	by (A) and ind. hyp.
(C)	`ack m 0` terminates	by (B) and definition of `acc`

3rd Case: `m` ¿ 0, `n` ¿ 0

To show: `ack m n` terminates

(A)	$(\texttt{m}, (\texttt{n}-1)) \prec_{lex} (\texttt{m}, \texttt{n})$	by definition of $\prec_{lex}$
(B)	`ack m (n-1)` terminates	by (A), and ind. hyp.
(C)	$\forall \texttt{q} : \mathbb{N}.\ ((\texttt{m}-1), \texttt{q}) \prec_{lex} (\texttt{m}, \texttt{n})$	by definition of $\prec_{lex}$
(D)	$\forall \texttt{q} : \mathbb{N}.$ `ack (m-1) q` terminates	by (C), and ind. hyp.
(E)	`ack (m-1) (ack m (n-1))` terminates	by (B) and (D)
(F)	`ack m n` terminates	by (E) and def. of `ack`

I think you'll agree that this was easier than with the previous approach.

TODO - Why did we chose these three cases?

[Extra] Comparing Well-Founded with Strong Induction

Does the function `g`

```
g :: Int -> Int
-- SPEC ∀n : ℕ. g n = 3^n − 2^n
g 0 = 0
g 1 = 1
g n = ( 5 * g(n-1) )  -  ( 6*g(n-2) )
```

satisfy its specification?

Define and use an appropriate well-founded ordering on $\mathbb{N}$.

Remember: A *well-founded ordering* over a set X is a binary relation $\prec \subseteq X \times X$ such that there is no infinitely decreasing chain

$$x_1 \mathsf{Succ} x_2 \mathsf{Succ} x_3 \mathsf{Succ} \ldots .$$

For any well-founded ordering, $\prec$, the well-founded induction principle says

$$[\ \forall z.\ [\ \forall y.(y \prec z \to P(y))\ ]\ \to\ P(z)\ ]\ \longrightarrow\ \forall x.P(x)$$

- Why are infinite descending chains forbidden?
- Are infinite ascending chains allowed?
- Are infinitely many chains allowed?
- Would it be allowed to have a $x \in X$, so that $\forall x' \in X.\ x \not\prec x'$?
- Would it be allowed to have a $x \in X$, so that $\forall x' \in X.\ x' \not\prec x$?

The Proof of termination of `ack` using well-founded induction is more succinct than that using double induction. Every inductively defined set has a well-founded ordering, namely based on the "is constructed from" relation. Therefore, well-founded induction is strictly more general than structural induction. Well-founded induction allows a more general concept of "predecessor". In particular elements may have infinitely many predecessors. For example, in $\prec_{lex}$-ordering, $(m, 0)$ has infinitely many, but not an "immediate" predecessor.

Can a well-founded ordering be reflexive, transitive, symmetric, or antisymmetric?

Well-Founded Induction $\implies$ Strong Induction

The principle of well-founded induction says that for a set S, and $\prec \subseteq S \times S$ a well-founded ordering on S:

$$(WFI) : [\ \forall z{:}S.\ [\ \forall y{:}S.(y \prec z \to P(y))\ ]\ \to\ P(z)]\ \longrightarrow\ \forall x{:}S.P(x)$$

The principle of strong induction says:

$$(SI) : [\ \ P(0)\ \ \wedge\ \ \forall k{:}\mathbb{N}.\ [\forall j \leq k.P(j)\ ]\ \to\ P(k+1)\ \ \ ]\ \to\ \ \forall n{:}\mathbb{N}.P(n)$$

We can show that (WFI) implies (SI).

Another cautionary tale

Given:

```
f :: Int -> Int -> Int
f 0 n = n
f m n = f (m-1) (f m n)
```

Show that $\forall$`n,m` $\in \mathbb{N}$. `f m n` terminates, using well-founded ind. with $\prec_{lex}$:

Inductive step take arbitrary `m`, `n`

> **Induction hypothesis:** `f m' n'` terminates for all `(m',n')` $\prec_{lex}$ `(m,n)`

To show: `f m n` terminates

1	`f m n = f (m-1) (f m n)`	by definition of `f`
2	`(m-1) < m`	by arithmetic
3	`((m-1),(f m n))` $\prec_{lex}$ `(m,n)`	by 2, and def. of $\prec_{lex}$
4	`f (m-1) (f m n)` terminates	by 3 and induction hypothesis
5	`f m n` terminates	by 4 and 1

Is the above proof correct?

Problem: in the inductive step we must show that *all* recursive calls to `f` stop. `f m n` has not been proven to be an integer...only if `f m n` terminates it is...

Summary of Part I

Slide 160

Part I – Conclusion