Notes for Lecture 14 (Fall 2022 week 6, part 3): Proofs about programs

Jana Dunfield

October 20, 2022

> *Reporter:* What kind of proof do you need?
> *Jean Chrétien:* I don't know. A proof is a proof. What kind of a proof? It's a proof. A proof is a proof, and when you have a good proof, it's because it's proven.
> *Reporter:* But what kind of proof do you want?
> *Chrétien:* Thank you very much.
>
> September 2002
> https://www.cbc.ca/player/play/2647901981
> (starts around 3:26)

1 Stepping proofs

A proof is *evidence of something*—but that definition is barely more useful than Jean Chrétien's. There are many kinds of evi

We will consider one ki

defined *stepping*, by "be ly, a sequence of steps.

To prove that the Haskell expression `(\x -> (\y -> y - x)) 1 6` ces the result 5, we use stepping rules and write out the steps:

	`(\x -> (\y -> y - x)) 1 6`	
$\Rightarrow$	`(\y -> y - 1) 6`	by function application, with substitution `1` for `x`
$\Rightarrow$	`6 - 1`	by function application, with substitution `6` for `y`
$\Rightarrow$	`5`	by arithmetic

So we have been doing proofs in CISC 360 already. You might think these proofs are "not interesting":

(1) if we want to know the result—the value produced at the end of the sequence of steps—we could enter the expression into GHCi, and see what it produces;

(2) we might not think the intermediate results, like `(\y -> y - 1) 6` and `6 - 1`, are interesting.

Regarding (1), this is a fair point for many programs (including the example above). But recall that GHCi will not print functions, so if we have the program

```
(\x -> (\y -> y - x)) 1
```

it will step to `(\y -> y - 1)`, which is a lambda, so GHCi will not print it.

Regarding (2), maybe we want to understand how the expression works, or how many steps it takes.

But I think most mathematicians would consider the stepping proofs we have done to be “trivial”. An interesting proof should make a *general* statement, like “any integer greater than 1 is equal to the product of prime numbers” or “for all a and b, if a and b are the lengths of the sides of a right triangle, the length of the diagonal is $\sqrt{a^2 + b^2}$”. Our stepping proofs are more like proofs of “the integer 6 is equal to the product of 3 and 2” or “if 2 and 5 are the lengths of the sides of a right triangle, then the length of the diagonal is $\sqrt{29}$”.

The purpose of this lecture is to make our stepping proofs “interesting”, or at least “nontrivial”, by adding *quantifiers*.

2 Doing "for all" in stepping proofs

Recall our old friend (?) `diag`:

```
diag m = if m == 0 then 0 else m + diag (m - 1)
```

(I changed `n` to `m` for reasons that may become clear.) If we try calling `diag` with various natural numbers, we can guess there is a relationship between the input and output:

$$
\begin{aligned}
\texttt{diag 1} &\Rightarrow^* 1\\
\texttt{diag 2} &\Rightarrow^* 3\\
\texttt{diag 3} &\Rightarrow^* 6\\
\texttt{diag 4} &\Rightarrow^* 10\\
\texttt{diag 5} &\Rightarrow^* 15
\end{aligned}
$$

(The symbol $\Rightarrow^*$ means *takes zero or more steps*. For example, `(\x -> (\y -> y - x)) 1` $\Rightarrow^*$ 5.) There is a relationship between `diag n` and (approximately) $\frac{n^2}{2}$. To be exact, the result of `diag n` is $\frac{n\cdot(n+1)}{2}$. For example, `diag 5` produces 15, and $\frac{5\cdot(5+1)}{2} = \frac{5\cdot6}{2} = \frac{30}{2} = 15$.

With the stepping proofs we have used before, we are limited to verifying the relationship

only for *specific* n, such

Let's prove a *general* statement:

For all $n \geq 0$,

$$\texttt{diag}\ n \;\Rightarrow^*\; \frac{}{2}$$

Some might regard this as sloppy, because my intent is that n is an integer. So it might be better to write it as

For all natural numbers n,

$$\texttt{diag}\ n \;\Rightarrow^*\; \frac{n\cdot(n+1)}{2}.$$

or, with more symbols:

For all $n \in \mathbb{N}$,

$$\texttt{diag}\ n \;\Rightarrow^*\; \frac{n\cdot(n+1)}{2}.$$

For those who like such things, we could also write "for all" using symbols:

$$\forall(n \mid n \geq 0)\big(\texttt{diag}\ n \;\Rightarrow^*\; \frac{n\cdot(n+1)}{2}\big)$$

or

$$\forall(n \in \mathbb{Z} \mid n \geq 0)\big(\texttt{diag}\ n \;\Rightarrow^*\; \frac{n\cdot(n+1)}{2}\big)$$

or

$$\forall(n \in \mathbb{N})\big(\texttt{diag}\ n \;\Rightarrow^*\; \frac{n\cdot(n+1)}{2}\big)$$

or

$$\forall n\Big((n \in \mathbb{N}) \to \big(\texttt{diag}\ n\ \Rightarrow^*\ \frac{n\cdot(n+1)}{2}\big)\Big)$$

or

$$\forall n\Big(\big((n \in \mathbb{Z}) \wedge (n \geq 0)\big) \to \big(\texttt{diag}\ n\ \Rightarrow^*\ \frac{n\cdot(n+1)}{2}\big)\Big)$$

(The last two are fairly close to CISC 204's notation.)

We'll stick to the "less symbols" version:

For all natural numbers n,
$\texttt{diag}\ n\ \Rightarrow^*\ \frac{n\cdot(n+1)}{2}$.

2.1 A simpler proof

Doing this proof about `diag` will require using induction, so we'll build up to that by first doing a proof that doesn't need induction:

For all natural numbers n,
$\texttt{(\textbackslash y -> y - 0)}\ n\ \Rightarrow^*\ n$.

How do we prove this statement?

To prove a "for all", we

We'll try to use our usua

First, write the expres ction application:

$\quad\texttt{(\textbackslash y -> y - 0)}\ n$

$\Rightarrow\ n\ \texttt{-}\ \texttt{0}$ by function application, with the subs

Applying the rule works the same way it always has ing n rather than a single, known number like 6.

$\quad\texttt{(\textbackslash y -> y - 0)}\ n$

$\Rightarrow\ n\ \texttt{-}\ \texttt{0}$ by function application, with substitution n for `y`

$\Rightarrow\ n$ by arithmetic

We have shown that $\texttt{(\textbackslash y -> y - 0)}\ n$ takes two steps to n. Therefore, it takes zero or more steps to n, which we can write as:

$$\texttt{(\textbackslash y -> y - 0)}\ n\ \Rightarrow^*\ n$$

which was to be proved.

2.2 The proof about `diag`

Let's return to the statement about `diag`.

For all natural numbers n,
$\texttt{diag}\ n\ \Rightarrow^*\ \frac{n\cdot(n+1)}{2}$.

We are proving a "for all" statement, so we first assume that we have some natural number n.

Now we will do a proof by induction. There are (perhaps unfortunately) many ways to "phrase" inductive proofs, so the one I use probably doesn't look exactly like what you've seen in other courses. In particular, you may have been told that in a proof by induction on natural numbers, you must have "a base case" and "an inductive case" (or "an inductive step"), perhaps something like:

> To prove 'For all natural numbers n, we have $\mathsf{Prop}(n)$' by induction, where Prop is some proposition involving n, *thou shalt write the following*:
>
> - **Base case ($n = 0$)**:
>   [prove that $\mathsf{Prop}(0)$ holds]
> - **Inductive step**:
>   Assume $\mathsf{Prop}(k)$ holds.
>   [prove that $\mathsf{Prop}(k + 1)$ holds]

However, the above is just one way of structuring an inductive proof. It has one advantage that I know of: it looks like an inductive *definition* of the natural numbers, which may make it easier to accept that proof by induction makes sense.

(**Aside:** An inductive definition of the natural numbers looks like this:

1. 0 is a natural numb
2. If k is a natural nu

I have no problem with this kind of definition, but I don't like insisti *proof* has to have the same structure as the definition.)

The above approach, with an explicit "base case" and an explicit " disadvantages:

- It forces you to structure your proof by considering cases of n. This is not always necessary or desirable. Sometimes the proof is easier if you structure it in some other way.
- It is not very general. We can do proofs by induction on things that aren't natural numbers, including:
  - pairs of natural numbers
  - lists
  - trees
  - other data structures
  - measures of data structures

  We can also do "complete induction" on natural numbers, which is more powerful than "simple induction" on natural numbers.

For this lecture, though, we'll stick to simple induction—but presented differently from what you're (probably) used to.

In an inductive proof, we assume the *inductive hypothesis* (IH). The IH, in English, will say:

If I am proving $\mathrm{Prop}(n)$ for $n > 0$, I can assume $\mathrm{Prop}(n-1)$.

For the result about `diag`, that means:

$$\text{If } n > 0 \text{ then}$$
$$\texttt{diag}\ (n-1) \Rightarrow^* \frac{(n-1)\cdot((n-1)+1)}{2}$$

(If you're interested, I got this by taking the statement we're trying to prove, replacing n with $n-1$, and adding the condition that n is *strictly* greater than zero: otherwise $n-1$ would not be a natural number.)

Because we are *assuming* the IH, we can "use" it at any time.

What we have to prove will be different, depending on whether n is 0 or greater than 0. So I will begin the proof by "case analysis":

Either $n = 0$, **or** $n > 0$.

1. **First case:** $n = 0$.

   (In this case, we are *assuming* $n = 0$, in addition to the IH.)

   Our goal is to prove

$$\texttt{diag}\ n \Rightarrow^* \frac{n\cdot(n+1)}{2}$$

   (*Do not confuse the go* e trying to prove: it is an "extra" assump about `diag` n, *n*

   Since we are assuming $n = 0$, we actually need to prov

$$\texttt{diag}\ 0 \Rightarrow^*$$

   This has eliminated the variable n, so we can further simplify our goal by observing that $\frac{0\cdot(0+1)}{2} = \frac{0\cdot 1}{2} = \frac{0}{2} = 0$.

$$\texttt{diag}\ 0 \Rightarrow^* 0$$

   Proving this doesn't require anything new! This is a stepping problem with no n, so we can solve it using techniques from weeks ago:

	`diag` 0	
$\Rightarrow$	`if 0 == 0 then 0 else 0 + diag (0 - 1)`	by function application, with subst. 0 for `m`
$\Rightarrow$	`if True then 0 else 0 + diag (0 - 1)`	by equality rule
$\Rightarrow$	`0`	by if-then-else rule

   Since `diag` 0 takes zero or more steps to `0`, we have proved

$$\texttt{diag}\ 0 \Rightarrow^* 0$$

   which is our goal.

   (This is the end of the first case, for when $n = 0$.)

2. **Second case:** $n > 0$.

   (In this case, we are *assuming* $n > 0$, in addition to the IH.)

   (It's very easy to get lost in the details. While doing a proof, keep asking yourself two questions:

   - What do I know?
   - What am I trying to prove?

   Right now, we know that n is a natural number (assumption from "for all" introduction), the IH, and that $n > 0$.

   We are trying to prove `diag` $n \Rightarrow^* \frac{n \cdot (n+1)}{2}$.)

```
  diag n
⇒ if n == 0 then 0 else n + diag (n - 1)    by function application, with subst. n for m
```

   At this point, we might have a problem: don't the rules for stepping `n == 0` need two actual integers? Not really: there is one rule for when the two integers are equal, and one for when they aren't equal. *We have the assumption* $n > 0$. If $n > 0$, then $n \neq 0$. Therefore, `n == 0` must step to `False`. After we take that step, we can take another step, using the if-then-else rule.

```
  diag n
⇒ if n == 0 then 0 else n + diag (n - 1)    by function application, with subst. n for m
⇒ if False then 0 else n + diag (n - 1)     by equality rule
⇒ n + diag (n - 1)
```

   What do we do now?

   We *could* use the rule for stepping function application. (Exercise: Try that.) However, we would find ourselves in a situation where we would need to know whether $n - 1 > 0$. We *don't* know that—we know that $n > 0$, but that means n could be 1, and $1 - 1 \not> 0$. If we split into two cases, one for $n - 1 = 0$ and one for $n - 1 > 0$, we would go through the same process we went through to get here, and would then have to consider cases $n - 2 = 0$ and $n - 2 > 0$. That process would never end! (It's somewhat similar to what happens in Prolog if it starts trying to prove a subgoal that is identical to the original goal.)

   So, again: what do we know, and what are we trying to prove? We know that n is a natural number, that $n > 0$, and that the IH is true. We haven't used the IH yet. So let's try to use the IH and see where that goes.

   The IH is:

$$\text{If } n > 0, \quad \texttt{diag}\ (n-1) \Rightarrow^* \frac{(n-1)\cdot((n-1)+1)}{2}$$

   This looks promising, because we also have the assumption $n > 0$. Therefore, we actually know (by modus ponens, if you care about that sort of thing)

$$\texttt{diag}\ (n-1) \Rightarrow^* \frac{(n-1)\cdot((n-1)+1)}{2}$$

Now this looks *very* promising: we got stuck on the expression `n + diag (n - 1)`, and the IH is telling us what `diag (n - 1)` steps to! So we can make some progress.

$$
\begin{array}{lll}
 & \texttt{diag}\ n & \\
 & \vdots & \\
\Rightarrow & n\ \texttt{+}\ \texttt{diag}\ \texttt{(}n\ \texttt{-}\ \texttt{1)} & \text{by not-equals rule} \\
\Rightarrow^{*} & n\ \texttt{+}\ \dfrac{(n-1)\cdot((n-1)+1)}{2} & \text{by IH}
\end{array}
$$

What are we trying to prove? That $\texttt{diag}\ n \Rightarrow^{*} \frac{n\cdot(n+1)}{2}$. Time to do arithmetic. This will look a little strange, because I will only replace a `+` with a $+$. These are in different fonts: the `+` is the Haskell addition operator, and the $+$ is the mathematical addition operator.

$$
\begin{array}{lll}
 & \texttt{diag}\ n & \\
 & \vdots & \\
\Rightarrow^{*} & n\ \texttt{+}\ \dfrac{(n-1)\cdot((n-1)+1)}{2} & \text{by IH} \\
\Rightarrow & n + \dfrac{(n-1)\cdot((n-1)+1)}{2} & \text{by arithmetic}
\end{array}
$$

Now we have a plain algebraic expression, which we need to show is equal to $\frac{n\cdot(n+1)}{2}$.

$$\frac{\text{[illegible]}}{2} \qquad \frac{\text{[illegible]}}{\text{[illegible]}}$$

[illegible]

$$
\begin{aligned}
&= \frac{\text{[illegible]}}{2} \\
&= \frac{2\cdot n + n\cdot n - 1\cdot n}{2} \\
&= \frac{2\cdot n + n^{2} - n}{2} \\
&= \frac{n + n^{2}}{2} = \frac{n\cdot(n+1)}{2}
\end{aligned}
$$

$$
\begin{array}{lll}
 & \texttt{diag}\ n & \\
 & \vdots & \\
\Rightarrow^{*} & n + \dfrac{(n-1)\cdot((n-1)+1)}{2} & \\
= & \dfrac{n\cdot(n+1)}{2} & \text{by above equations}
\end{array}
$$

Therefore, $\texttt{diag}\ n \Rightarrow^{*} \frac{n\cdot(n+1)}{2}$, which is our goal.

(This is the end of the second case, for when $n > 0$.)

2.3 The proof about `diag`, without explanations

I'm including this to help you understand what I think counts as a complete and correct proof.

Theorem 1. For all natural numbers n, $\texttt{diag}\ n \Rightarrow^* \frac{n\cdot(n+1)}{2}$.

Proof. By induction on n.

Assume the IH: If $n > 0$ then $\texttt{diag}\ (n-1) \Rightarrow^* \frac{(n-1)\cdot((n-1)+1)}{2}$.
Either $n = 0$, or $n > 0$.

- **First case**: $n = 0$

$\texttt{diag}\ 0$

$\Rightarrow$ `if 0 == 0 then 0 else 0 + diag (0 - 1)` by function application, with subst. 0 for `m`

$\Rightarrow$ `if True then 0 else 0 + diag (0 - 1)` by equality rule

$\Rightarrow$ 0 by if-then-else rule

$= \frac{0\cdot(0+1)}{2}$ by algebra

$= \frac{n\cdot(n+1)}{2}$ by assumption $n = 0$

- **Second case**: $n > 0$

$\texttt{diag}\ n$

$\Rightarrow$ `if n == 0 then 0 el` tion, with subst. n for `m`

$\Rightarrow$ `if False then 0 els` g assumption $n > 0$

$\Rightarrow$ `n + diag (n - 1)`

$s \Rightarrow^*\ n + \frac{(n-1)\cdot((n-1)+1)}{2}$

$\Rightarrow^*\ n + \frac{(n-1)\cdot((n-1)+1)}{2}$ by IH

$\Rightarrow\ n + \frac{(n-1)\cdot((n-1)+1)}{2}$ by arithmetic

$= n + \frac{(n-1)\cdot n}{2}$ by algebra...

$= \frac{2\cdot n}{2} + \frac{(n-1)\cdot n}{2}$

$= \frac{2\cdot n + (n-1)\cdot n}{2}$

$= \frac{2\cdot n + n\cdot n - 1\cdot n}{2}$

$= \frac{2\cdot n + n^2 - n}{2}$

$= \frac{n+n^2}{2} = \frac{n\cdot(n+1)}{2}$

□