Mathematics for Computer Scientists 2 (G52MC2)

L08 : Peano arithmetic

Thorsten Altenkirch

School of Computer Science
University of Nottingham

November 5, 2009

What are the natural numbers?

Guiseppe Peano (1858-1932)

- Peano codified the theory of the natural numbers ($\mathbb{N} = \{0, 1, 2, 3, \ldots\}$).
- All Peano numbers are constructed from 0 and successor S. E.g. $1 = \mathrm{S}\,0$, $2 = \mathrm{S}\,(\mathrm{S}\,0)$, $3 = \mathrm{S}\,(\mathrm{S}\,(\mathrm{S}\,0))$.
- Peano presented a system of axioms in predicate logic stating fundamental properties of the natural numbers.
- We refer to this system as *Peano Arithmetic*.

Peano arithmetic in Coq

In Coq we can define the natural numbers following Peano:

```
Inductive nat : Set :=
  | O : nat
  | S : nat -> nat.
```

Verifying Peano's axioms

- There is no natural number whose successor is 0.

$$\forall n : \mathbb{N}, \mathrm{S}\, n \neq 0$$

- If the successors of two numbers are the same, then the numbers must be the same.

$$\forall m\, n : \mathbb{N}, \mathrm{S}\, m = \mathrm{S}\, n \to m = n$$

Induction

One of Peano's most important axioms is:

The principle of induction

If a property is true for 0 and closed under successor (i.e. if it holds for n then also holds for $\mathrm{S\,n}$), then it holds for all natural numbers.

Given $P : \mathbb{N} \to \mathbf{Prop}$:

$$P\,0 \to (\forall i : \mathbb{N}, P\,i \to P\,(\mathrm{S}\,i)) \to \forall n : \mathbb{N}, P\,n$$

- In Coq we use the induction tactic.
- induction is similar to case.

Fixpoints

- In Coq (and Mathematics) definitions are not allowed to be recursive.
- Coq will reject the following *definition*

```
Definition is_even (n : nat) : bool :=
  match n with
  | 0 => true
  | S n' => negb (is_even n')
  end.
```

- Instead we have to use `Fixpoint`:

```
Fixpoint is_even (n : nat) : bool :=
  match n with
  | 0 => true
  | S n' => negb (is_even n')
  end.
```

- The fixpoint of a function $f : A \to A$ is an element $a : A$ such that $f\,a = a$.

Fixpoints

- Indeed `is_even` is the unique fixpoint of:

```
Definition
  f_is_even : (nat -> bool) ->(nat -> bool) :=
  fun (h : nat -> bool) => fun (n:nat) =>
    match n with
    | 0 => true
    | S n' => negb (h n')
    end.
```

- Not every function has a fixpoint, e.g.

```
Definition
   f_no_fix : (nat -> bool) -> (nat -> bool) :=
  fun (h : nat -> bool) => fun (n:nat) => negb (h n)
```

  hence the following `fixpoint` is rejected by Coq:

```
Fixpoint no_fix (n:nat) : nat :=
  negb (no_fix n).
```

- Other functions have infinitely many fixpoints (Can you think of an example?).

Structural recursion

- Coq only accepts fixpoints, which are structurally recursive.
- This is the recursive call has to be applied to a substructure of the original argument.
- Hence `is_even` is structurally recursive but also `half` (see l08.v)
- The functions related to structurally recursive definitions always have a unique fixpoint.
- For functions with several arguments, the structurally recursive position has to be indicated using `struct`.

Addition and multiplication

Examples are addition and multiplication:

```
Fixpoint plus (n m:nat) {struct n} : nat :=
  match n with
  | O => m
  | S n' => S (plus n' m)
  end.
```

```
Fixpoint mult (n m:nat) {struct n} : nat :=
  match n with
  | O => 0
  | S n' => m + mult n' m
  end.
```

- In Coq both are predefined using $+$ and `*`.
- Peano only defined addition and multiplication.
- All other structural recursive functions are *definable* from those.
- Arithmetic with addition only is called *Pressburger Arithmetic*. Unlike Peano Arithmetic it is decidable!

Algebraic properties

Using induction we can establish the usual algebraic properties for $+$ and $\times$:

$$\begin{array}{rcll} m+n & = & n+m & \text{commutativity of addition} \\ m+(n+p) & = & (m+n)+p & \text{associativity of addition} \\ i\times(j+k) & = & i\times j+i\times k & \text{commutativity of multiplication} \end{array}$$

What other properties can you think of?