# Proving with Computer Assistance
# Lecture 12

## Inversion

Herman Geuvers

# The natural number type

Recall the definition of natural numbers:

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

Meaning of this definition:

- Every number has one of two forms:
  - it is the constructor `O` **or**
  - it is built by applying the constructor `S` to another number.
- But there is more to say, which is implicit in the definition:
  - The constructor `S` is injective: If `S n = S m`, then `n = m`
  - The constructors `O` and `S` are distinct: `O` is not equal to `S n` for any `n`.

# General inductive types

Principles similar to `nat` apply to all inductively defined types:

- injectivity: the constructors are injective
- no overlap: the values built from distinct constructors are never equal.

For lists:

- `cons` is injective
- `nil` $\neq$ `cons a l` for every `a`, `l`

For booleans: `true` $\neq$ `false`

# Inversion tactic

The inversion tactic is used to exploit injectivity and no overlap
Suppose

$$H : c\, a_1\, a_2 \ldots a_n = d\, b_1\, b_2 \ldots b_m$$

for constructors $c$ and $d$ and arguments $a_1 a_2 \ldots a_n$ and $b_1 b_2 \ldots b_m$. Then

$$\texttt{inversion}\ H$$

looks at the possible ways that this equation can arise:

- If $c$ and $d$ are the same constructor, then (by the injectivity) $a_1 = b_1$, $a_2 = b_2, \ldots$. These facts are added to the context, and can be use to rewrite the goal.
- If $c$ and $d$ are different constructors, then $H$ is contradictory (the equality is false). So, the goal is provable! `inversion` $H$ completes the goal.

See the examples in the Coq files.