Induction heuristics

Basic heuristics

Theorems about recursive functions are proved by induction

Basic heuristics

Theorems about recursive functions are proved by induction

Induction on argument number i of f
if f is defined by recursion on argument number i

A tail recursive reverse

consts *itrev :: 'a list* $\Rightarrow$ *'a list* $\Rightarrow$ *'a list*

A tail recursive reverse

consts $itrev :: 'a\ list \Rightarrow 'a\ list \Rightarrow 'a\ list$

primrec

$itrev\ []\qquad ys = ys$

$itrev\ (x\#xs)\quad ys =$

A tail recursive reverse

consts $itrev :: 'a\ list \Rightarrow 'a\ list \Rightarrow 'a\ list$

primrec

$itrev\ []\ ys = ys$

$itrev\ (x\#xs)\ ys = itrev\ xs\ (x\#ys)$

A tail recursive reverse

consts *itrev :: 'a list* $\Rightarrow$ *'a list* $\Rightarrow$ *'a list*

primrec

itrev [] *ys = ys*

itrev (x#xs) *ys = itrev xs (x#ys)*

lemma *itrev xs [] = rev xs*

A tail recursive reverse

consts *itrev :: 'a list* $\Rightarrow$ *'a list* $\Rightarrow$ *'a list*
primrec
itrev [] ys = ys
itrev (x#xs) ys = itrev xs (x#ys)

lemma *itrev xs [] = rev xs*

Why in this direction?

A tail recursive reverse

consts *itrev :: 'a list* $\Rightarrow$ *'a list* $\Rightarrow$ *'a list*
primrec
itrev [] ys = ys
itrev (x#xs) ys = itrev xs (x#ys)

lemma *itrev xs [] = rev xs*

Why in this direction?

Because the lhs is “more complex” than the rhs.

Demo: first proof attempt

Generalisation (1)

Replace constants by variables

lemma *itrev xs ys = rev xs @ ys*

Demo: second proof attempt

Generalisation (2)

Quantify free variables by $\forall$
(except the induction variable)

lemma $\forall$ *ys. itrev xs ys = rev xs @ ys*