

1 map concat

$(.) :: (b \rightarrow c) \rightarrow (a \rightarrow b) \rightarrow a \rightarrow c$
 $f \cdot g = \lambda x \rightarrow f (g x)$

$\text{concat} :: [[a]] \rightarrow [a]$
 $\text{concat } [] = []$
 $\text{concat } (x:xs) = x ++ (\text{concat } xs)$

$\text{map} :: (a \rightarrow b) \rightarrow [a] \rightarrow [b]$
 $\text{map } f [] = []$
 $\text{map } f (x:xs) = [f x] ++ (\text{map } f xs)$

Claim 1.1.

$\text{map } f . \text{concat} = \text{concat} . \text{map } (\text{map } f)$

Proof. structural induction on xs
 $xs = []$

$(\text{map } f . \text{concat}) \ xs = (\text{map } f . \text{concat}) \ []$
 $\quad = \text{map } f (\text{concat } [])$
 $\quad = \text{map } f \ []$
 $\quad = []$
 $\quad = \text{concat } []$
 $\quad = \text{concat } (\text{map } (\text{map } f) \ [])$
 $\quad = (\text{concat} . \text{map } (\text{map } f)) \ []$
 $\quad = (\text{concat} . \text{map } (\text{map } f)) \ xs$

$xs = (x:xs')$ structural induction on x
 $x = []$

$(\text{map } f . \text{concat}) \ xs = (\text{map } f . \text{concat}) \ (x:xs')$
 $\quad = \text{map } f (\text{concat } ([]:xs'))$
 $\quad = \text{map } f ([] ++ (\text{concat } xs'))$
 $\quad = \text{map } f (\text{concat } xs')$
 $\quad = (\text{concat} . \text{map } (\text{map } f)) \ xs'$
 $\quad = \text{concat } (\text{map } (\text{map } f) \ xs')$
 $\quad = [] ++ (\text{concat } (\text{map } (\text{map } f) \ xs'))$
 $\quad = \text{concat } [[]] ++ (\text{map } (\text{map } f) \ xs')$
 $\quad = \text{concat } [\text{map } f \ []] ++ (\text{map } (\text{map } f) \ xs')$
 $\quad = \text{concat } (\text{map } (\text{map } f) \ ([]:xs'))$
 $\quad = \text{concat } (\text{map } (\text{map } f) \ xs)$
 $\quad = (\text{concat} . \text{map } (\text{map } f)) \ xs$

$x = (y:ys)$

```

(map f.concat) xs = (map f.concat) (x:xs')
                  = map f (concat ((y:ys):xs'))
                  = map f ((y:ys)++(concat xs'))
                  = [f y]++(map f (ys++(concat xs')))
                  = [f y]++(map f (concat (ys:xs')))
                  = [f y]++((concat.map (map f)) (ys:xs'))
                  = [f y]++(concat (map (map f) (ys:xs')))
                  = [f y]++(concat [map f ys]++(map (map f) xs'))
                  = [f y]++(map f ys)++(concat (map (map f) xs'))
                  = (map f (y:ys))++(concat (map (map f) xs'))
                  = concat [map f (y:ys)]++(map (map f) xs')
                  = concat (map (map f) ((y:ys):xs'))
                  = concat (map (map f) (x:xs'))
                  = concat (map (map f) xs)
                  = (concat.map (map f)) xs

```

□

alternative proof

Proof. we show the equivalent claim

`map f (concat xs) = concat (map (map f) xs)`

by structural induction on `xs`

`xs=[]`

```
map f (concat xs) = map f (concat [])
                  = map f []
                  = []
                  = concat []
                  = concat (map (map f) [])
                  = concat (map (map f) xs)
```

`xs=(x:xs')` structural induction on `x`

`x=[]`

```
map f (concat xs) = map f (concat ([]:xs'))
                  = map f ([]++(concat xs'))
                  = map f (concat xs')
                  = concat (map (map f) xs')
                  = []++(concat (map (map f) xs'))
                  = concat [[]]++(map (map f) xs')
                  = concat [map f []]++(map (map f) xs')
                  = concat (map (map f) ([]:xs'))
                  = concat (map (map f) xs)
```

`x=(y:ys)`

```
map f (concat xs) = map f (concat ((y:ys):xs'))
                  = map f ((y:ys)++(concat xs'))
                  = [f y]++(map f (ys++(concat xs')))
                  = [f y]++(map f (concat (ys:xs')))
                  = [f y]++(concat (map (map f) (ys:xs')))
                  = [f y]++(concat [map f ys]++(map (map f) xs'))
                  = [f y]++(map f ys)++(concat (map (map f) xs'))
                  = (map f (y:ys))++(concat (map (map f) xs'))
                  = concat [map f (y:ys)]++(map (map f) xs')
                  = concat (map (map f) ((y:ys):xs'))
                  = concat (map (map f) (x:xs'))
                  = concat (map (map f) xs)
```

□