COMP-105, Higher-Order Functions Assignment

Benjamin Tanen

October 21, 2015

Ramsey, p.181 #31

Problem: Prove that (o ((curry map) f) ((curry map) g)) = ((curry map) (o f g))

In order to complete this proof, we will use two helper proofs (see below) for the mapping of an empty list and the mapping of the cons of an element and a list. Additionally, for this proof, we can use the following given laws:
((o f g) x) = (f (g x)), apply-compose law
(((curry f) x) y) = (f x y), apply-curried law

Helper Proof #1: map-cons law, (map f (cons x xs)) = (cons (f x)(map f xs))

(map f (cons x xs))
={substitute actual parameters in definition of map function}
(if (null? (cons x xs))
'()
(cons (f (car (cons x xs))) (map f (cdr (cons x xs)))))
={car-cons law}
(if (null? (cons x xs))
'()
(cons (f x) (map f (cdr (cons x xs)))))
={cdr-cons law}
(if (null? (cons x xs))
'()
(cons (f x) (map f x)))
={null?-cons law}
(if #f
'()
(cons (f x) (map f x)))
={if-false law}
(cons (f x) (map f x)))

Thus we can see that (map f (cons x xs)) = (cons (f x)(map f xs))

Helper Proof #2: map-empty law, (map f '()) = '()

(map f '())
={substitute actual parameters in definition of map function}
(if (null? '())
  '()
  (cons (f (car '())) (map f (cdr '()))))
={null?-empty law}
(if #t
  '()
  (cons (f (car '())) (map f (cdr '()))))
={if-true law}
'()

Thus we can see that (map f '()) = '()

With these two helper functions defined, we can now start the main proof. We will do this with the base case where each function is applied to an empty list. We will first use the left side of the proof.

((o ((curry map) f) ((curry map) g)) '())
={apply-compose law}
(((curry map) f) (((curry map) g) '()))
={apply-curried law}
(((curry map) f) (map g '()))
={map-empty law}
(((curry map) f) '())
={apply-curried law}
(map f '())
={map-empty law}
'()

Next we will do the right side of the proof
(((curry map) (o f g)) '())
={apply-curried law}
(map (o f g) '())
={map-empty law}
'()

Thus, since when given the same argument / list each side returns the same thing, the base case has been proven on the empty list. We know that (((curry map) (o f g)) '()) = '() = ((o ((curry map) f) ((curry map) g)) '())

Next, we can inductively show the two are equal when applied to a non-empty list l = (cons x xs). We will again start from the left side.

((o ((curry map) f) ((curry map) g)) l)

={replace l with (cons x xs)}

((o ((curry map) f) ((curry map) g)) (cons x xs))

={apply-compose law}

(((curry map) f) (((curry map) g) (cons x xs)))

={apply-curried law}

(((curry map) f) (map g (cons x xs)))

={map-cons law}

(((curry map) f) (cons (g x) (map g xs)))

={apply-curried law}

(map f (cons (g x) (map g xs)))

={map-cons law}

(cons (f (g x)) (map f (map g xs)))

={apply-curried law}

(cons (f (g x)) (map f (((curry map) g) xs)))

={apply-curried law}

(cons (f (g x)) (((curry map) f) (((curry map) g) xs)))

={apply-compose law}

(cons (f (g x)) ((o ((curry map) f) ((curry map) g)) xs))

={induction hypothesis}

(cons (f (g x)) (((curry map) (o f g)) xs))

={apply-compose law}

(cons ((o f g) x) (((curry map) (o f g)) xs))

={apply-curried law}

(cons ((o f g) x) (map (o f g) xs))

={map-cons law}

(map (o f g) (cons x xs))

={replace (cons x xs) with l}

(map (o f g) l)

={apply-curried law}

(((curry map) (o f g)) l)

Thus, since ((o ((curry map) f) ((curry map) g)) l) = (((curry map) (o f g)) l) for any l (we proved for the empty list '() and inductively for any list l), we can say:

(o ((curry map) f) ((curry map) g)) = ((curry map) (o f g)) is true