Constructive Logic (15-317), Spring 2023
Recitation 11

Clogic Staff

April 19, 2023

1 Lambda Box

The goal of this recitation is to get more practice with the lambda-box language. We will start with a function that computes the inner product of two vectors, and then write a new function that takes advantage of quotation.

Original function:

```
ip := λn.λv.λw.case n
        |0 → 0
        |s n' → (v[n] × w[n]) + ip n' v w
```

Solution:
The type of this function should be

$$nat \to \square(vec \to \square(vec \to nat))$$

```
ip' := λn.case n
        |0 → quote(λv.quote(λw.0))
        |s n' → let n□' = lift n' in
                                unquote(n□',nv'.
                                unquote(ip' nv',f.quote(λv.
                                unquote(f v,f'.quote(λw.(f' w) +
                                (v[s nv'] × w[s nv']))))))
```

Example Walkthrough We first define the following:

```
eval x:= unquote(x, u.u)
```

Now we define the following and see how it steps through:

```
ip1 := ip' 1 = unquote ((ip' 0),f.quote(λv.
                                unquote(f v,f'.quote(λw.f' w +
                                (v[s 0]× w[s 0]))))) ↦
```

```
            unquote (quote(λv'.quote(λw'.0)),f.quote(λv.
                          unquote(f v,f'.quote(λw.f' w +
                          (v[s 0] × w[s 0]))))) ↦

            quote(λv.unquote((λv'.quote(λw'.0)) v,f'.
                          quote(λw.(f' w) + (v[s 0] × w[s 0]))))

eval (ip1) [42] = unquote((λv'.quote(λw'.0)) [42], f'.
                          quote(λw. (f' w) +
                          ([42][s 0] × w[s 0]))) ↦

                  unquote(quote(λw'.0),f'.quote(λw.
                          (f' w) + ([42][s 0] × w[s 0]))) ↦
                  quote((λw.(λw'.0) w) + ([42][s 0] × w[s 0]))
```