WORK LOG OF JUNE 12 2025

DANIEL R. BARRERO R.

1. TODOs and more

- Include Operad composition picture.
- Define Haskell listing manually. `<-- couldn't.`
- `splitForest` is a *pseudoinverse* to `Cons`. Can't say it's an inverse because types don't match. But maybe that can be worked around later.

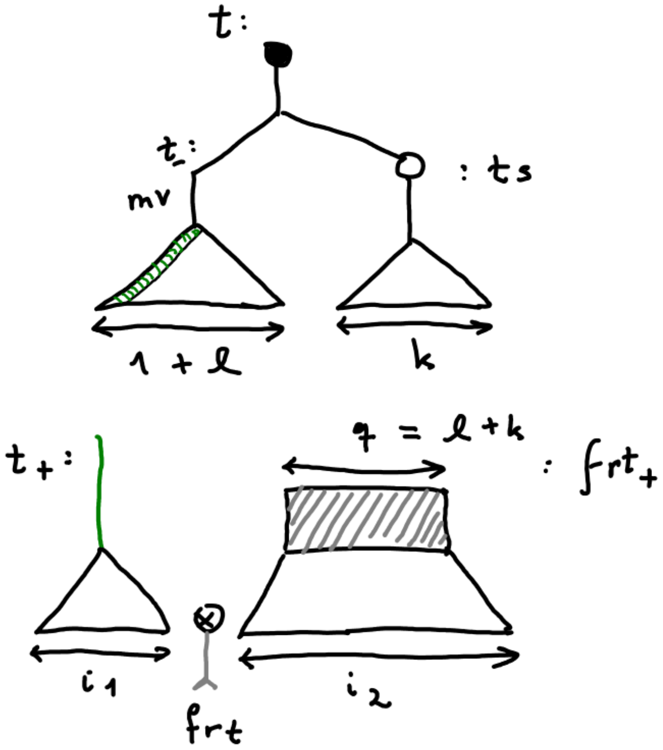
```
data MoveTree n where
  Leaf :: MoveTree One
  Fan :: Trees n -> MoveTree n

data Trees n where
  NilT :: Trees Z
  (:+) :: (Move, MoveTree a) -> Trees b -> Trees (a+b)

class (Graded f) Operad where
  ident :: f One
  compose :: f n -> Forest f m n -> fm

data Forest f m n where
  Nil :: Forest f Z Z
  Cons :: f i1 -> Forest f i2 n -> Forest f (i1+i2) (S n)

splitForest :: forall f a b z q. SNat a -> SNat b ->
  Forest f q (a+b) ->
  (
   forall j j'. (j+j') ~ q =>
   (Forest f j a, Forest f j' b) -> z
  ) -> z
splitForest (SS (sl :: SNat l))
            (sk :: SNat k)
            (Cons (t :: f j1) (frt :: Forest f j2 (l+k)))
            c =
  splitForest sl sk frt $
    (
     \((lrdr :: Forest f j2' l),(krdr :: Forest f j2'' k)) ->
        case plusAssoc (j1 :: Proxy j1)
                       (j2' :: Proxy j2')
                       (j2'' :: Proxy j2'') of
        Dict -> c (Cons t lrdr , krdr)
    )
```


t:
t_-:
mv
: ts
1 + ℓ
k
t_+:
q = ℓ + k
: frt_+
i_1
frt
i_2