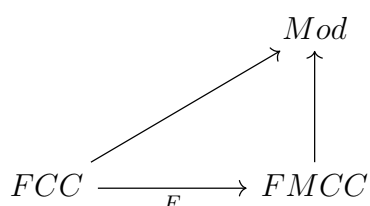


In other words, given a morphism  $f$  in a free monoidal closed category ( $FMCC$  below) between two objects that are in the image of the map  $F$  from the free closed category ( $FCC$  below), we want to compute a morphism in  $FCC$  such that it maps to the same morphism in the category of Modules.



```
inductive type (cT : Type) : Type
| const : cT → type
| arrow : type → type → type
```

```

inductive term2 (ct : type cT → Type) :  $\Pi$  (A : type cT), Type
| const {T : type cT} (t : ct T) : term2 T
| app {T1 T2 : type cT} (f : term2 (T1.arrow T2)) (x : term2 T1) : term2 T2
| id {T1 : type cT} : term2 (T1.arrow T1)
| comp {T1 T2 T3 : type cT} : term2 ((T1.arrow T2).arrow ((T2.arrow T3).arrow (T1.arrow T3)))
| swap {T1 T2 T3 : type cT} : term2 ((T1.arrow (T2.arrow T3)).arrow (T2.arrow (T1.arrow T3)))

```

```
inductive type3 (cT : Type) : Type
| const : cT → type3
| arrow : type3 → type3 → type3
| id {} : type3
| tensor : type3 → type3 → type3
```

1

## 2 SOLVING THE PROBLEM

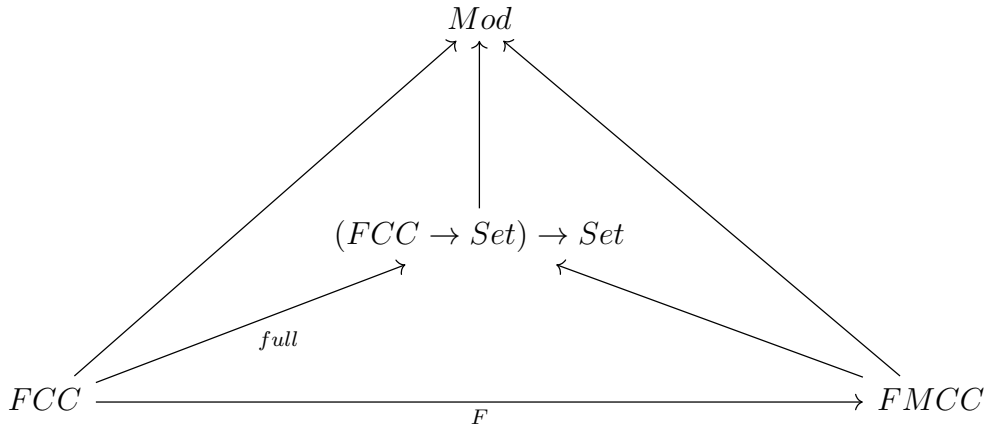
```

inductive term3 (const_term : type cT → Type) :  $\Pi$  (A : type3 cT), Type
| const {T : type cT} (t : const_term T) : term3 (type3.of_type T)
| id (T : type3 cT) : term3 (T.arrow T)
| curry {T1 T2 T3 : type3 cT} :
  term3 (((T1.tensor T2).arrow T3)).arrow (T1.arrow (T2.arrow T3))
| uncurry {T1 T2 T3 : type3 cT} :
  term3 ((T1.arrow (T2.arrow T3)).arrow ((T1.tensor T2).arrow T3))
| tensor_map {T1 T2 T3 T4 : type3 cT} (f1 : term3 (T1.arrow T3))
  (f2 : term3 (T2.arrow T4)) : term3 ((T1.tensor T2).arrow (T3.tensor T4))
| tensor_symm {T1 T2 : type3 cT} : term3 ((T1.tensor T2).arrow (T2.tensor T1))
| lid1 {T : type3 cT} : term3 ((type3.id.tensor T).arrow T)
| lid2 {T : type3 cT} : term3 (T.arrow (type3.id.tensor T))
| app {T1 T2 : type3 cT} (f : term3 (T1.arrow T2)) (x : term3 T1) : term3 T2
| comp {T1 T2 T3 : type3 cT} (f : term3 (T1.arrow T2)) :
  term3 ((T2.arrow T3).arrow (T1.arrow T3))

```

## 2 Solving the problem

We solve the problem by defining a new category  $(FCC \rightarrow Set) \rightarrow Set$ , and a functor  $FMCC$  to  $(FCC \rightarrow Set) \rightarrow Set$  such that the following commutes. By fullness of the functor  $FCC$  to  $(FCC \rightarrow Set) \rightarrow Set$ , we can compute everything we need, and the commutativity of the diagram will tell us we defined the right thing. We can define the functor  $FMCC$  to  $(FCC \rightarrow Set) \rightarrow Set$  by defining a closed monoidal structure on  $(FCC \rightarrow Set) \rightarrow Set$ . The map from  $(FCC \rightarrow Set) \rightarrow Set$  to  $Mod$  and its commutativity can be proven using the universal property of the presheaf category, and the continuity of the yoneda embedding.



Universal Property of  $(FCC \rightarrow Set) \rightarrow Set$ .

In the following diagram each functor is labelled *con* if it is continuous and *cocon* if it is cocontinuous. The category  $(FCC \rightarrow Set)^{op}$  is the completion of  $FCC$ , so by its universal property there is a continuous map into  $Set$ . It is also true that the yoneda embedding

### 3 BICOMPLETIONS

from  $FCC$  is cocontinuous. The category  $(FCC \rightarrow Set) \rightarrow Set$  is the cocompletion of  $(FCC \rightarrow Set)^{op}$  so there is a cocontinuous map into  $Mod$ , and it is also true that the embedding from  $(FCC \rightarrow Set)^{op}$  is continuous.

$$\begin{array}{ccccc}
 & & & & Mod \\
 & & & \nearrow & \uparrow \\
 & & & con & cocon \\
 FCC & \xrightarrow{cocon} & (FCC \rightarrow Set)^{op} & \xrightarrow{con} & (FCC \rightarrow Set) \rightarrow Set
 \end{array}$$

The functor  $(FCC \rightarrow Set) \rightarrow Set$  to  $Mod$  does not preserve all limits, however it does preserve all of the limits of diagrams in  $FCC$ , because these limits exist in the category  $(FCC \rightarrow Set)^{op}$ .

The category  $(FCC \rightarrow Set) \rightarrow Set$  can be thought of as freely adjoining all colimits of limits to the category  $FCC$ . This is necessary to define a monoidal closed structure on it because the definition of a hom object in this category is a colimit of a limit (although it is not actually defined like that in the code). This means that the map  $(FCC \rightarrow Set) \rightarrow Set$  will preserve hom objects and tensor objects.

We can now show why the following commutes

$$\begin{array}{ccccc}
 & & Mod & & \\
 & \nearrow & \uparrow & \nwarrow & \\
 & & (FCC \rightarrow Set) \rightarrow Set & & \\
 & \nearrow & \uparrow & \nwarrow & \\
 FCC & \xrightarrow{full} & & \xrightarrow{F} & FMCC
 \end{array}$$

The large triangle containing  $FCC$ ,  $FMCC$  and  $Mod$  commutes more or less by definition. The triangle on the left commutes because the map  $(FCC \rightarrow Set) \rightarrow Set$  is defined using the universal property of the free cocompletion and free completion. The triangle on the right commutes because the functor  $(FCC \rightarrow Set) \rightarrow Set$  preserves colimits of limits, which means it preserves hom objects and tensor products, so it preserves everything in  $FMCC$ . (This is not that precise) Therefore the entire diagram commutes.

### 3 Bicompletions

The idea of computing with presheaves to simplify expressions or prove equalities can be taken much further with bicompletions. Given a category  $\mathcal{C}$  it is hopefully possible to

computably define a bicompletion of  $\mathcal{C}$ , called  $\Lambda\mathcal{C}$ , a full faithful map  $i : \mathcal{C} \rightarrow \Lambda\mathcal{C}$ , with the following universal property. Given a bicomplete category  $\mathcal{D}$  and a functor  $F : \mathcal{C} \rightarrow \mathcal{D}$ , there is a unique bicontinuous functor  $F' : \Lambda\mathcal{C} \rightarrow \mathcal{D}$ , such that  $F' \circ i = F$ . (Maybe equalities of functors should have been isomorphisms there)

The construction of bicompletions is probably complicated but doable. You probably have to keep on adjoining limits and then colimits using presheaves lots of times taking care not to adjoin limits that have already been adjoined.

The above example with monoidal closed categories could also be done using bicompletions - the bicompletion of a symmetric closed category is symmetric monoidal closed.

### 3.1 Finite Limits and Colimits

Fullness of the functor into the bicompletion will mean that if the homsets of  $\mathcal{C}$  have decidable equality then so do many of the homsets of  $\Lambda\mathcal{C}$ , I think in the closure of  $\mathcal{C}$  inside  $\Lambda\mathcal{C}$  under finite products and coproducts. If  $A, B, C : \mathcal{C}$  are objects, then  $\text{Hom}_{\Lambda\mathcal{C}}(i(A) \amalg i(B), i(C)) \cong \text{Hom}_{\mathcal{C}}(A, C) \times \text{Hom}_{\mathcal{C}}(B, C)$ , so it has decidable equality, and equality in  $\text{Hom}_{\Lambda\mathcal{C}}(i(A) \amalg i(B), i(C))$  should be able to be decided by adjoining an object  $X$  and maps  $f : X \rightarrow A$  and  $g : X \rightarrow B$  to  $\mathcal{C}$  to make a new category  $\mathcal{D}$ . The maps  $f$  and  $g$  are both epic in  $\mathcal{D}$ . Therefore  $\text{Hom}_{\Lambda\mathcal{C}}(i(A) \amalg i(B), i(C)) \cong \text{Hom}_{\Lambda\mathcal{D}}(i(X), i(C)) \cong \text{Hom}_{\mathcal{D}}(X, C)$ . (This is not a complete proof).

### 3.2 Adjunctions

Suppose  $\mathcal{C}$  and  $\mathcal{D}$  are bicomplete categories and  $F$  is a forgetful functor between them with a left adjoint. (eg.  $\mathcal{C} = \text{Ab}$ ,  $\mathcal{D} = \text{Grp}$ ). Suppose  $\mathcal{C}'$  and  $\mathcal{D}'$  are categories and there are functors  $\mathcal{C}' \rightarrow \mathcal{C}$  and  $\mathcal{D}' \rightarrow \mathcal{D}$ , and  $F' : \mathcal{C}' \rightarrow \mathcal{D}'$  is a restriction of  $F$ . Morally  $\mathcal{C}'$  and  $\mathcal{D}'$  might be small finite diagrams generated by all the homs and objects in a context for example. Then  $F'$  can be lifted to a bicontinuous map  $\Lambda F' : \Lambda\mathcal{C}' \rightarrow \Lambda\mathcal{D}'$ . The functor  $\Lambda F'$  will have a left adjoint by the adjoint functor theorem (Maybe there are size issues here). This should commute with the adjoint of  $F$  (because the functors are bicontinuous and the formula for the adjoint is some sort of colimit). Therefore we can eliminate this adjunctions in a similar way to how we eliminated tensor products in the monoidal category example.

### 3.3 Conclusion

The bicompletion should hopefully be useful to provide some tools to prove equality of morphisms using universal properties, or to partially prove them, e.g. in the associativity of polynomials example it could hopefully reduce it to proving associativity of natural number addition. It should be able to handle expressions containing finite limits/colimits, adjunctions, tensor products etc. all at once.