1 The problem

A term in the linear lambda calculus can be interpreted as a morphism in any symmetric monoidal closed category. The category of Modules is a symmetric monoidal closed category so we can interpret it as a morphism there. The resulting interpreted morphism will use the tensor products in the monoidal closed category. We would like to interpret it without using the tensor products.

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.

$$\begin{tikzcd}
 & Mod \\
FCC \arrow[ur] \arrow[r, "F"'] & FMCC \arrow[u]
\end{tikzcd}$$

The objects in FCC are given by the following inductive type (this is not quite a free closed category). We have a set of type constants cT.

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

The morphisms in FCC are given by the following inductive type. For each `type`, there is a set of constants of that type given by `ct`.

```
inductive term2 (ct : type cT → Type) : Π (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 T
    2
| 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)))
```

The objects in $FMCC$ are given by the following inductive type. There is an obvious map `type` to `type3`.

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

The morphisms in $FMCC$ are given by

```
inductive term3 (const_term : type cT → Type) : Π (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 $(FCC \to Set) \to Set$, and a functor $FMCC$ to $(FCC \to Set) \to Set$ such that the following commutes. By fullness of the functor FCC to $(FCC \to Set) \to Set$, we can compute everything we need, and the commutavity of the diagram will tell us we defined the right thing. We can define the functor $FMCC$ to $(FCC \to Set) \to Set$ by defining a closed monoidal structure on $(FCC \to Set) \to Set$. The map from $(FCC \to Set) \to Set$ to Mod and its commutativity can be proven using the universal property of the presheaf category, and the continuity of the yoneda embedding.

$$\begin{tikzcd}
 & & Mod & & \\
 & & (FCC \to Set) \to Set \arrow[u] & & \\
FCC \arrow[rruu] \arrow[rru, "full"'] \arrow[rrrr, "F"'] & & & & FMCC \arrow[lluu] \arrow[llu]
\end{tikzcd}$$

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

In the following diagram each functor is labelled *con* if it is continuous and *cocon* if it is cocontinuous. The category $(FCC \to Set)^{op}$ 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 from FCC is cocontinuous. The category $(FCC \to Set) \to Set$ is the cocompletion of $(FCC \to Set)^{op}$ so there is a cocontinuous map into Mod, and it is also true that the embedding from $(FCC \to Set)^{op}$ is continuous.

$$\begin{tikzcd}
 & & Mod \\
FCC \arrow[r, "cocon"'] \arrow[rru] & (FCC \to Set)^{op} \arrow[r, "con"'] \arrow[ru, "con"'] & (FCC \to Set) \to Set \arrow[u, "cocon"']
\end{tikzcd}$$

The functor $(FCC \to Set) \to Set$ 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 $(FCC \to Set)^{op}$.

The category $(FCC \to Set) \to Set$ 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 $(FCC \to Set) \to Set$ will preserve hom objects and tensor objects.

We can now show why the following commutes

$$\begin{tikzcd}
 & Mod & \\
 & (FCC \to Set) \to Set \arrow[u] & \\
FCC \arrow[rr, "F"'] \arrow[ur, "full"'] \arrow[uur] & & FMCC \arrow[ul] \arrow[uul]
\end{tikzcd}$$

The large triangle containing FCC, $FMCC$ and Mod commutes more or less by definition. The triangle on the left commutes because the map $(FCC \to Set) \to Set$ is defined using the universal property of the free cocompletion and free completion. The triangle on the right commutes because the functor $(FCC \to Set) \to Set$ 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 $i : \mathcal{C} \to \Lambda\mathcal{C}$, with the following universal property. Given a bicomplete category $\mathcal{D}$ and a functor $F : \mathcal{C} \to \mathcal{D}$, there is a unique bicontinuous functor $F' : \Lambda\mathcal{C} \to \mathcal{D}$, 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 $\mathrm{Hom}_{\Lambda\mathcal{C}}(i(A) \amalg i(B), i(C)) \cong \mathrm{Hom}_{\mathcal{C}}(A, C) \times \mathrm{Hom}_{\mathcal{C}}(B, C)$, so it has decidable equality, and equality in $\mathrm{Hom}_{\Lambda\mathcal{C}}(i(A) \Pi\, i(B), i(C))$ should be able to be decided by adjoining an object X and maps $f : X \to A$ and $g : X \to B$ to $\mathcal{C}$ to make a new category $\mathcal{D}$. The maps f and g are both epic in $\mathcal{D}$. Therefore $\mathrm{Hom}_{\Lambda\mathcal{C}}(i(A)\ \Pi\ i(B), i(C)) \cong \mathrm{Hom}_{\Lambda\mathcal{D}}(i(X), i(C)) \cong \mathrm{Hom}_{D}(X, C)$. (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. $\mathcal{C} = Ab$, $\mathcal{D} = Grp$). Suppose $\mathcal{C}'$ and $\mathcal{D}'$ are categories and there are functors $\mathcal{C}' \to \mathcal{C}$ and $\mathcal{D}' \to \mathcal{D}$, and $F' : \mathcal{C}' \to \mathcal{D}'$ 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 $\Lambda F' : \Lambda\mathcal{C}' \to \Lambda\mathcal{D}'$. 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.