# Mac Lane’s Comparison Theorem for the (co)Kleisli construction in Coq

Burak Ekici

University of Innsbruck, Austria

August 13, 2018

# The statement

$$\begin{tikzcd} \mathscr{C} \arrow[r, bend left, "F"] \arrow[r, phantom, "\perp"] & \mathscr{D} \arrow[l, bend left, "G"] \end{tikzcd}$$

# The statement

$$\begin{tikzcd}
\mathscr{C} \arrow[loop above, "T"] \arrow[dr, bend left, "F"] \arrow[dr, phantom, "\perp"] \\
& \mathscr{D} \arrow[ul, bend left, "G"]
\end{tikzcd}$$

# The statement

$$
\begin{tikzcd}
\mathscr{C} \arrow[loop above, "T"] \arrow[dr, bend left, "F"] \arrow[dr, phantom, "\perp"] & \mathscr{C}_T \\
& \mathscr{D} \arrow[ul, bend left, "G"]
\end{tikzcd}
$$

# The statement

$$
\begin{tikzcd}
\mathscr{C} \arrow[loop above, "T"] \arrow[rr, bend left, "F_T"] \arrow[rr, phantom, "\perp"] \arrow[drr, bend left, "F"] \arrow[drr, phantom, "\perp"] & & \mathscr{C}_T \arrow[ll, bend left, "G_T"] \\
& & \mathscr{D} \arrow[ull, bend left, "G"]
\end{tikzcd}
$$

# The statement

$$
\begin{tikzcd}
\mathscr{C} \arrow[loop above, "T"] \arrow[rr, bend left, "F_T"] \arrow[rr, phantom, "\perp"] \arrow[drr, bend left, "F"] \arrow[drr, phantom, "\perp"] & & \mathscr{C}_T \arrow[ll, bend left, "G_T"] \arrow[d, "L"] \\
& & \mathscr{D} \arrow[ull, bend left, "G"]
\end{tikzcd}
$$

# The statement

$$
\begin{tikzcd}
\mathscr{C} \arrow[loop above, "T"] \arrow[rr, bend left, "F_T"] \arrow[rr, phantom, "\perp"] \arrow[drr, bend left=15, "F"] \arrow[drr, phantom, "\dashv"] & & \mathscr{C}_T \arrow[ll, bend left, "G_T"] \arrow[d, "!L"] \arrow[dl, phantom, "="] \\
& & \mathscr{D} \arrow[ull, bend left=15, "G"]
\end{tikzcd}
$$

$$\exists ! L \colon \mathscr{C}_T \to \mathscr{D}, L \circ F_T = F \wedge G \circ L = F_T$$

# Adjunctions

$$\begin{tikzcd} \mathscr{C} \arrow[rr, bend left, "F"] \arrow[rr, phantom, "\perp"] & & \mathscr{D} \arrow[ll, bend left, "G"] \end{tikzcd}$$

$\eta : Id \Rightarrow GF$ $\quad F \dashv G \quad$ $\varepsilon : FG \Rightarrow Id$

Definition

Let $\mathscr{C}$ and $\mathscr{D}$ be two categories. The functors $F : \mathscr{C} \to \mathscr{D}$ and $G : \mathscr{D} \to \mathscr{C}$ form an adjunction $F \dashv G : \mathscr{D} \to \mathscr{C}$ iff there exists *natural transformations* $\eta : Id_{\mathscr{C}} \Rightarrow GF$ and $\varepsilon : FG \Rightarrow Id_{\mathscr{D}}$ such that:

$$\varepsilon_{FX} \circ F\eta_X = id_{FX} \text{ for each } X \text{ in } \mathscr{C} \quad (1)$$

$$G\varepsilon_X \circ \eta_{GX} = id_{GX} \text{ for each } X \text{ in } \mathscr{D} \quad (2)$$

# Adjunctions

$$\varepsilon_{FX} \circ F\eta_X = id_{FX} \text{ for each } X \text{ in } \mathscr{C}$$

$$
\begin{tikzcd}
X \arrow[r, "Idf"] \arrow[d, "\eta_X"'] \arrow[dr, phantom, "="] & Y \arrow[d, "\eta_Y"] \\
GFX \arrow[r, "GFf"'] & GFY
\end{tikzcd}
\quad \overset{F}{\frown\!\!\!\to} \quad
\begin{tikzcd}
FX \arrow[r, "F(Idf)"] \arrow[d, "F\eta_X"] \arrow[dd, bend right=60, "Id(FX)"'] \arrow[dr, phantom, "="] & FY \arrow[d, "F\eta_Y"'] \arrow[dd, bend left=60, "Id(FY)"] \\
FGFX \arrow[r, "FGFf"'] \arrow[d, "\varepsilon_{FX}"] \arrow[dr, phantom, "="] & FGFY \arrow[d, "\varepsilon_{FY}"'] \\
FX \arrow[r, "Ff"'] & FY
\end{tikzcd}
$$

# Adjunctions: an example

In CIC, the logical $\wedge$ and $\Longrightarrow$ are adjoint operations when Coq's *Prop* universe is defined as a category.

The proof is in the library comes with this talk.

# Adjunctions: an example

In CIC, the logical $\wedge$ and $\Longrightarrow$ are adjoint operations when Coq's *Prop* universe is defined as a category.

The proof is in the library comes with this talk.

# Monads

**Definition**

A *monad* $T = (T, \eta, \mu)$ in a category $\mathscr{C}$ consists of an endofunctor $T\colon \mathscr{C} \to \mathscr{C}$ with two natural transformations

$$\eta\colon Id_{\mathscr{C}} \Rightarrow T \qquad \mu\colon T^2 \Rightarrow T \tag{3}$$

such that the following diagrams commute:

$$\begin{tikzcd}
T^3 \arrow[r, "\mu T"] \arrow[d, "T\mu"'] \arrow[dr, phantom, "="] & T^2 \arrow[d, "\mu"] \\
T^2 \arrow[r, "\mu"'] & T
\end{tikzcd}
\qquad
\begin{tikzcd}
T \arrow[r, "\eta T"] \arrow[d, "T\eta"'] \arrow[dr, "id_T" description] & T^2 \arrow[d, "\mu"] \arrow[dl, phantom, "=" near start] \\
T^2 \arrow[r, "\mu"'] \arrow[ur, phantom, "=" near start] & T
\end{tikzcd}$$

# Monad: quick examples

- A monoid is a monad in the category of endo-functors
- In CIC, the triple (fmapM, etaM, muM) forms a monad when Coq's *Type* universe is defined as a category.

```
Inductive maybe (A: Type) ≜  just: A → maybe A  | nothing: maybe A.

Definition fmapM {A B: Type} (f: A → B) (i: maybe A): maybe B ≜
  match i with
    | just _ a  ⇒ just _ (f a)
    | nothing _ ⇒ nothing _
  end.

Definition etaM {A: Type} (a: A): maybe A ≜ just A a.

Definition muM {A : Type} (i: maybe (maybe A)): maybe A ≜
  match i with
    | just _ a ⇒ a
    | nothing _ ⇒ nothing _
  end.
```

# Every adjunction gives a monad

$$\begin{tikzcd} \mathscr{C} \arrow[rr, bend left, "F"] \arrow[rr, phantom, "\perp"] & & \mathscr{D} \arrow[ll, bend left, "G"] \end{tikzcd}$$

$\eta : Id \Rightarrow GF$ $\quad F \dashv G \quad$ $\varepsilon : FG \Rightarrow Id$

# Every adjunction gives a monad

$$
\begin{tikzcd}
\mathscr{C} \arrow[loop above, "T"] \arrow[rr, bend left=10, "F"] \arrow[rr, phantom, "\perp"] & & \mathscr{D} \arrow[loop above, "D"] \arrow[ll, bend left=10, "G"]
\end{tikzcd}
$$

$\eta : Id \Rightarrow T$ $\quad F \dashv G \quad$ $\varepsilon : D \Rightarrow Id$

**Proposition**

*An adjunction $F \dashv G : \mathscr{D} \to \mathscr{C}$ determines a monad on $\mathscr{C}$ and a comonad on $\mathscr{D}$ as follows:*

- *The monad $(T, \eta, \mu)$ on $\mathscr{C}$ $T = GF : \mathscr{C} \to \mathscr{C}$, $\mu_X = G(\varepsilon_{FX})$.*
- *The comonad $(D, \varepsilon, \delta)$ on $\mathscr{D}$ $D = FG : \mathscr{D} \to \mathscr{D}$, $\delta_A = F(\eta_{GA})$.*

# Every (co)monad gives a (co)Kleisli adjunction

$$\begin{tikzcd} \mathscr{C} \arrow[loop above, "T"] \end{tikzcd}$$

# Every (co)monad gives a (co)Kleisli adjunction

$$\begin{tikzcd} \mathscr{C} \arrow[loop above, "T"] & & \mathscr{C}_T \end{tikzcd}$$

**Proposition**

*Each monad* $(T, \eta, \mu)$ *on a category* $\mathscr{C}$ *determines a Kleisli category* $\mathscr{C}_T$*,*

# Every (co)monad gives a (co)Kleisli adjunction

$$\begin{tikzcd} \mathscr{C} \arrow[loop above, "T"] \arrow[rr, bend left, "F_T"] \arrow[rr, phantom, "\perp"] & & \mathscr{C}_T \arrow[ll, bend left, "G_T"] \end{tikzcd}$$

Proposition

*Each monad* $(T, \eta, \mu)$ *on a category* $\mathscr{C}$ *determines a Kleisli category* $\mathscr{C}_T$*, and an associated adjunction* $F_T \dashv G_T : \mathscr{C}_T \to \mathscr{C}$.

# Every (co)monad gives a (co)Kleisli adjunction

$$\begin{tikzcd} \mathscr{C} \arrow[loop above, "T"] & & & \mathscr{C}_T \end{tikzcd}$$

- The categories $\mathscr{C}$ and $\mathscr{C}_T$ have the same objects and there is a morphism $f^\flat\colon X \to Y$ in $\mathscr{C}_T$ for each $f\colon X \to TY$ in $\mathscr{C}$.
- For each object $X$ in $\mathscr{C}_T$, the identity arrow is $id_X = h^\flat\colon X \to X$ in $\mathscr{C}_T$ where

$$h = \eta_X\colon X \to TX \text{ in } \mathscr{C}.$$

- The composition of a pair of morphisms $f^\flat\colon X \to Y$ and $g^\flat\colon Y \to Z$ in $\mathscr{C}_T$ is given by the Kleisli composition:

$$g^\flat \circ f^\flat = h^\flat\colon X \to Z \text{ where } h = \mu_Z \circ Tg \circ f\colon X \to TZ \text{ in } \mathscr{C}.$$

# Every (co)monad gives a (co)Kleisli adjunction

$$\begin{tikzcd} \mathscr{C} \arrow[loop above, "T"] \arrow[rr, bend left, "F_T"] \arrow[rr, phantom, "\perp"] & & \mathscr{C}_T \arrow[ll, bend left, "G_T"] \end{tikzcd}$$

- The functor $F_T \colon \mathscr{C} \to \mathscr{C}_T$ is the identity on objects. On morphisms,

$$F_T f = (\eta_Y \circ f)^\flat, \text{ for each } f \colon X \to Y \text{ in } \mathscr{C}. \tag{4}$$

- The functor $G_T \colon \mathscr{C}_T \to \mathscr{C}$ maps each object $X$ in $\mathscr{C}_T$ to $TX$ in $\mathscr{C}$. On morphisms,

$$G_T(g^\flat) = \mu_Y \circ Tg, \text{ for each } g^\flat \colon X \to Y \text{ in } \mathscr{C}_T. \tag{5}$$

# Sketch of the proof

$$
\begin{tikzcd}
\mathscr{C} \arrow[loop above, "T"] \arrow[rr, bend left, "F_T"] \arrow[rr, phantom, "\perp"] \arrow[drr, bend right=10, "F"'] \arrow[drr, phantom, "\perp"] & & \mathscr{C}_T \arrow[ll, bend left, "G_T"] \arrow[d, "!L"] \arrow[dl, phantom, "="] \\
& & \mathscr{D} \arrow[ull, bend left=10, "G"]
\end{tikzcd}
$$

1. Characterize some map $L\colon \mathscr{C}_T \to \mathscr{D}$ by

$$
\begin{cases}
LX & = & FX \\
Lf^\flat & = & \varepsilon_{FY} \circ Ff, \text{ for each } f^\flat\colon X \to Y \text{ in } \mathscr{C}_T
\end{cases}
$$

2. Prove that L is a functor satisfying $GL = G_T$ and $LF_T = F$.
3. Show that L is unique

# Proof: $L$ is a functor

- For each $X$ in $\mathscr{C}_T$, $id_X = (\eta_X)^\flat$ in $\mathscr{C}_T$, we have $L(id_X) = L(\eta_X)^\flat = \varepsilon_{FX} \circ F\eta_X$. We get

$$\varepsilon_{FX} \circ F\eta_X = id_{FX} = id_{LX}.$$

- For each pair of morphisms $f^\flat \colon X \to Y$ and $g^\flat \colon Y \to Z$ in $\mathscr{C}_T$, by Kleisli composition, we obtain

$$L(g^\flat \circ f^\flat) = \varepsilon_{FZ} \circ FG\varepsilon_{FZ} \circ FGFg \circ Ff.$$

  Since $\varepsilon$ is natural, we have $\varepsilon_{FZ} \circ Fg \circ \varepsilon_{FY} \circ Ff$ which is $L(g^\flat) \circ L(f^\flat)$ in $\mathscr{D}$.

Hence $L \colon \mathscr{C}_T \to \mathscr{D}$ is a functor.

# Proof: $L$ satisfies ...

1. For each object $X$ in $\mathscr{C}_T$, $LX = FX$ in $\mathscr{D}$ and $GLX = GFX = TX = G_T X$ in $\mathscr{C}$. For each morphism $f^\flat : X \to Y$ in $\mathscr{C}_T$, $Lf^\flat = \varepsilon_{FY} \circ Ff$ in $D$ by definition. Hence,

$$GLf^\flat = G\varepsilon_{FY} \circ GFf.$$

   Similarly, by definition $G_T f^\flat = \mu_Y \circ Tf$. Since $\mu_Y = G(\varepsilon_{FY})$, we get

$$G_T f^\flat = G\varepsilon_{FY} \circ GFf.$$

   We get $GLf^\flat = G_T f^\flat$ for each mapping $f^\flat$. Thus $GL = G_T$.

2. $F_T$ is the identity on objects, thus $LF_T X = LX = FX$. For each morphism $f : X \to Y$ in $\mathscr{C}$, we have $F_T f = (\eta_Y \circ f)^\flat$ in $\mathscr{C}_T$, by definition. So that

$$LF_T f = L(\eta_Y \circ f)^\flat = \varepsilon_{FY} \circ F\eta_Y \circ Ff.$$

   Due to $\varepsilon$ and $\eta$ being natural, we have $\varepsilon_{FY} \circ F\eta_Y = id_{FY}$ yielding $LF_T f = Ff$ for each mapping $f$. Therefore $LF_T = F$.

# Proof: $L$ is unique

We need to show that $\forall R\colon \mathscr{C}_T \to \mathscr{D}$ such that $R \circ F_T = F$ and $G \circ R = G_T, L = R$.

- forall $X$ in $\mathscr{C}_T, LX = RX$

$$
\begin{aligned}
LX &= FX && (\textit{by definition}) \\
&= RF_TX && (\textit{by axiom}) \\
&= RX && (F_T \textit{ is the identity})
\end{aligned}
$$

Therefore, $LX = RX$

# Proof: $L$ is unique

**Lemma**

*Let $F \dashv G \colon \mathscr{D} \to \mathscr{C}$ be an adjunction. For each $f \colon X \to GY$ in $\mathscr{C}$, and $g,\ h \colon FX \to Y$ in $\mathscr{D}$, if $f = Gg \circ \eta_X$ and $f = Gh \circ \eta_X$ then $g = h$.*

- forall $f^\flat \colon X \to Y$ in $\mathscr{C}_T, Lf^\flat = Rf^\flat$

$$
\begin{aligned}
G(Rf^\flat) \circ \eta_X &= G_T f^\flat \circ \eta_X && \text{(by axiom)}\\
&= \mu_Y \circ GFf \circ \eta_X && \text{(by definition of } G_T\text{)}\\
&= \mu_Y \circ \eta_{GFY} \circ f && \text{(by naturality of } \eta\text{)}\\
&= f && (GF \text{ is a monad})\\
G(Lf^\flat) \circ \eta_X &= G\varepsilon_{FY} \circ GFf \circ \eta_X && \text{(by definition of } L\text{)}\\
&= G\varepsilon_{FY} \circ \eta_{GFY} \circ f && \text{(by naturality of } \eta\text{)}\\
&= f && (F \dashv G \text{ is an adjunction})
\end{aligned}
$$

Therefore, $\mathscr{C}_T, Lf^\flat = Rf^\flat$

# A Use Case

The *full image factorization* (or decomposition) of $D$ is given by $L \circ G_D \circ F_{D,T}$.

$$
\begin{tikzcd}
\mathscr{C}_D \arrow[loop above, "T"] \arrow[r, bend left, "F_{D,\,T}"] \arrow[r, phantom, "\perp"] \arrow[dr, bend left, "F_D"] \arrow[dr, phantom, "\perp"] & \mathscr{C}_{D,\,T} \arrow[l, bend left, "G_{D,\,T}"] \arrow[d, "!L"] \arrow[d, phantom, "=", swap] \\
 & \mathscr{C} \arrow[ul, bend left, "G_D"] \arrow[loop below, "D"]
\end{tikzcd}
\qquad
\begin{tikzcd}
\mathscr{C} \arrow[r, "G_D"] \arrow[rrd, "D", swap] \arrow[rd, phantom, "="] & \mathscr{C}_D \arrow[r, "F_{D,\,T}"] \arrow[rd, "F_D", swap] \arrow[r, phantom, "=", shift right=6ex] & \mathscr{C}_{D,\,T} \arrow[d, "!L"] \\
 & & \mathscr{C}
\end{tikzcd}
$$

# A Use Case

The *full image factorization* (or decomposition) of $D$ is given by $L \circ G_D \circ F_{D,T}$.

$$\begin{tikzcd}
\mathscr{C}_D \arrow[loop above, "T \overset{\text{def}}{=} - \times S"] \arrow[r, bend left, "F_{D,T}"] \arrow[r, phantom, "\perp"] \arrow[dr, bend left, "F_D"] \arrow[dr, phantom, "=", near start, shift left=6] & \mathscr{C}_{D,T} \arrow[l, bend left, "G_{D,T}"] \arrow[d, "!L"] \\
& \mathscr{C} \arrow[loop below, "D \overset{\text{def}}{=} - \times S"] \arrow[ul, bend left, "G_D"] \arrow[ul, phantom, "\perp"]
\end{tikzcd}
\qquad
\begin{tikzcd}
\mathscr{C} \arrow[r, "G_D"] \arrow[drr, bend right, "D"'] \arrow[dr, phantom, "="] & \mathscr{C}_D \arrow[r, "F_{D,T}"] \arrow[dr, bend right, "F_D"'] \arrow[dr, phantom, "=", shift left=6] & \mathscr{C}_{D,T} \arrow[d, "!L"] \\
& & \mathscr{C}
\end{tikzcd}$$

# A Use Case

The *full image factorization* (or decomposition) of $D^2$ is given by $K \circ F_{D,T,D,T} \circ G_{D,T,D} \circ F_{D,\,T} \circ G_D$.

$$\begin{tikzcd}
\mathscr{C} \arrow[r, "G_D"] \arrow[drr, bend right, "D^2"'] & \mathscr{C}_D \arrow[r, "F_{D,\,T}"] & \mathscr{C}_{D,\,T} \arrow[r, "G_{D,T,D}"] \arrow[d, phantom, "="] & \mathscr{C}_{D,T,D} \arrow[r, "F_{D,T,D,T}"] & \mathscr{C}_{D,T,D,T} \arrow[dll, bend left, "!K"] \\
 & & \mathscr{C} & &
\end{tikzcd}$$

# Category theory in Coq

Some Category theory formalized in Coq:

- J. Gross, A. Chlipala, and D. I. Spivak. Experience implementing a performant category-theory library in Coq.
- A. Timany and B. Jacobs. Category theory in Coq 8.5.
- J. Wiegley's library on github.
- As a part of UniMath library.
- ...

All (except the one in UniMath) represent category theoretical objects (i.e., categories, functors, etc.) with Coq type classes.

# Category theory in Coq

Our approach is no different.

```
Class Functor (C D: Category): Type ≜
  mk_Functor
  {
    fobj          : @obj C → @obj D;
    fmap          : ∀ {a b: @obj C} (f: arrow b a), (arrow (fobj b) (fobj a));
    preserve_id   : ∀ {a: @obj C}, fmap (@identity C a) = (@identity D (fobj a));
    preserve_comp : ∀ {a b c: @obj C} (g : @arrow C c b) (f: @arrow C b a),
                    fmap (g o f) = (fmap g) o (fmap f)
  }.
```

# A difficulty

Might be difficult to prove equalities of two class instances.

```
Lemma F_split: ∀ (C D  : Category) (F G  : Functor C D),
               fobj F = fobj G → JMeq (fmap F) (fmap G) → F = G.
Proof.
...
Qed.
```

Requires to struggle with explicit coercions hidden behind the heterogenous equality.

# A difficulty

One way to deal with that: converting the goal into an equality on dependent pairs:

```
 ...
______________________________________(1/1)
{p : (∀ a b : obj, arrow b a → arrow (fobj F b) (fobj F a)) =
     (∀ a b : obj, arrow b a → arrow (fobj G b) (fobj G a)) &
match p in (_ = y) return y with
| eq_refl ⇒ @fmap _ _ F
end = @fmap _ _ G}
```

# The statement in Coq

$$
\begin{tikzcd}
\mathscr{C} \arrow[r, bend left, "F"] \arrow[r, phantom, "\perp"] & \mathscr{D} \arrow[l, bend left, "G"]
\end{tikzcd}
$$

# The statement in Coq

$$\begin{tikzcd}
\mathscr{C} \arrow[loop above, "T"] \arrow[rr, bend left, "F"] \arrow[rr, phantom, "\perp"] & & \mathscr{D} \arrow[ll, bend left, "G"]
\end{tikzcd}$$

```
Theorem adj_mon: ∀ {C D: Category} (F : Functor C D) (G: Functor D C),
  Adjunction F G → Monad C (Compose_Functors F G).
Proof.
 ...
Qed.
```

# The statement in Coq

$$\begin{tikzcd}
\mathscr{C} \arrow[loop above, "T"] \arrow[dr, bend left, "F"] \arrow[dr, phantom, "\perp"] & \mathscr{C}_T \\
 & \mathscr{D} \arrow[ul, bend left, "G"]
\end{tikzcd}$$

```
Definition Kleisli_Category
  (C: Category) (T: Functor C C) (M: Monad C T): Category.
Proof. unshelve econstructor.
       - exact (@obj C).
       - intros a b. exact (@arrow C (fobj T a) b).
       ...
Defined.
```

# The statement in Coq

$$\begin{tikzcd} \mathscr{C} \arrow[loop above, "T"] \arrow[rr, "F_T"] \arrow[drr, bend left=15, "F"] & & \mathscr{C}_T \\ & & \mathscr{D} \arrow[ull, bend left=15, "G"] \arrow[ull, phantom, "\perp"] \end{tikzcd}$$

```
Definition FT {C D: Category} (F: Functor C D) (G: Functor D C)
  (T ≜ Compose_Functors F G) (M: Monad C T)
  (CT ≜ (Kleisli_Category C T M)): Functor C CT.
Proof. intros; unshelve econstructor; simpl.
       - exact id.
       - intros a b f. exact (trans eta b o f).
       ...
Defined.
```

# The statement in Coq

$$
\begin{tikzcd}
\mathscr{C} \arrow[loop above, "T"] \arrow[rr, bend left, "F_T"] \arrow[drr, bend left, "F"] \arrow[drr, phantom, "\perp"] & & \mathscr{C}_T \arrow[ll, bend left, "G_T"] \\
& & \mathscr{D} \arrow[ull, bend left, "G"]
\end{tikzcd}
$$

```
Definition GT {C D: Category} (F: Functor C D) (G: Functor D C)
  (T ≜ Compose_Functors F G) (M: Monad C T)
  (CT ≜ (Kleisli_Category C T M)): Functor CT C.
Proof. intros; unshelve econstructor; simpl.
       - exact (fobj T).
       - intros a b g. exact (trans mu b o fmap T g).
       ...
Defined.
```

# The statement in Coq

$$
\begin{tikzcd}
\mathscr{C} \arrow[loop above, "T"] \arrow[rr, bend left, "F_T"] \arrow[rr, phantom, "\perp"] \arrow[drr, bend left, "F"] \arrow[drr, phantom, "\perp"] & & \mathscr{C}_T \arrow[ll, bend left, "G_T"] \\
& & \mathscr{D} \arrow[ull, bend left, "G"]
\end{tikzcd}
$$

```
Theorem mon_kladj: ∀ {C D: Category} (F: Functor C D) (G: Functor D C)
  (T ≜ Compose_Functors F G) (M: Monad C T)
  (FT ≜ FT F G M) (GT ≜ GT F G M), Adjunction FT GT.
Proof.
 ...
Qed.
```

# The statement in Coq

$$
\begin{tikzcd}
\mathscr{C} \arrow[loop above, "T"] \arrow[rr, bend left=10, "F_T"] \arrow[rr, phantom, "\perp"] \arrow[drr, bend left=10, "F"] & & \mathscr{C}_T \arrow[ll, bend left=10, "G_T"] \arrow[d, "L"] \\
& & \mathscr{D} \arrow[ull, bend left=10, "G"] \arrow[ull, phantom, "\perp"]
\end{tikzcd}
$$

```
Definition L:
     ∀ {C D: Category} (F: Functor C D) (G: Functor D C) (A1: Adjunction F G),
  let M  ≜ (adj_mon F G A1) in let CM ≜ (adj_comon F G A1) in
  let CT ≜ (Kleisli_Category C (Compose_Functors F G) M) in
  let FT ≜ (FT F G M) in let GT ≜ (GT F G M) in
  let A2 ≜ (mon_kladj F G M) in Functor CT D.
Proof. intros. cbn in *.
       unshelve econstructor.
       - exact (fobj F).
       - intros a b f. exact (trans eps (fobj F b) o fmap F f).
       ...
Defined.
```

# The statement in Coq

$$
\begin{tikzcd}[column sep=huge, row sep=huge]
\mathscr{C} \arrow[loop above, "T"] \arrow[r, bend left=15, "F_T"] \arrow[r, phantom, "\perp"] \arrow[dr, bend left=15, "F"] \arrow[dr, phantom, "\perp"] & \mathscr{C}_T \arrow[l, bend left=15, "G_T"] \arrow[d, "L"] \arrow[dl, phantom, "="] \\
& \mathscr{D} \arrow[ul, bend left=15, "G"]
\end{tikzcd}
$$

```
Lemma commL:
     ∀ {C D: Category} (F: Functor C D)  (G: Functor D C)  (A1: Adjunction F G),
              let M  ≜ (@adj_mon   C D F G A1) in
              let CT ≜ (Kleisli_Category C (Compose_Functors F G) M) in
              let FT ≜ (FT F G M) in let GT ≜ (GT F G M) in
              let A2 ≜ (mon_kladj F G M) in
                (Compose_Functors FT (L F G A1)) = F ∧
                (Compose_Functors (L F G A1) G) = GT.
Proof.
 ...
Qed.
```

# The statement in Coq

$$
\begin{tikzcd}
\mathscr{C} \arrow[loop above, "T"] \arrow[rr, bend left, "F_T"] \arrow[rr, phantom, "\perp"] \arrow[rrd, bend left, "F"] \arrow[rrd, phantom, "\perp"] & & \mathscr{C}_T \arrow[ll, bend left, "G_T"] \arrow[d, "!L"] \arrow[d, phantom, "=" left] \\
& & \mathscr{D} \arrow[llu, bend left, "G"]
\end{tikzcd}
$$

```
Lemma uniqueL:
     ∀ {C D: Category} (F: Functor C D) (G: Functor D C) (A1: Adjunction F G),
   let  M ≜ (adj_mon   F G A1) in
   let CK ≜ (Kleisli_Category C (Compose_Functors F G) M) in
   let FT ≜ (FT F G M) in let GT ≜ (GT F G M) in
   let A2 ≜ (mon_kladj F G M) in
    ∀ R : Functor CK D, Compose_Functors FT R = F ∧ Compose_Functors R G = GT
     → (L F G A1) = R.
Proof.
 ...
Qed.
```

# The statement in Coq

$$
\begin{tikzcd}[column sep=huge, row sep=huge]
\mathscr{C} \arrow[loop above, "T"] \arrow[rr, bend left=15, "F_T"] \arrow[rr, phantom, "\perp"] \arrow[drr, bend left=10, "F"] \arrow[drr, phantom, "\perp"] & & \mathscr{C}_T \arrow[ll, bend left=15, "G_T"] \arrow[d, "!L"] \arrow[dl, phantom, "="] \\
 & & \mathscr{D} \arrow[ull, bend left=10, "G"]
\end{tikzcd}
$$

```
Lemma ComparisonMacLane:
    ∀ {C D: Category} (F: Functor C D) (G: Functor D C) (A1: Adjunction F G),
  let  M ≜ (adj_mon   F G A1) in
  let CK ≜ (Kleisli_Category C (Compose_Functors F G) M) in
  let FT ≜ (FT F G M) in let GT ≜ (GT F G M) in
  let A2 ≜ (mon_kladj F G M) in
    ∃ !L, (Compose_Functors FT L) = F ∧ (Compose_Functors L G) = GT.
Proof. intros C D F G A1 M CT FT GT A2.
       ∃ (L F G A1). split.
       - apply commL.
       - apply uniqueL.
Qed.
```

# Conclusion

1. We have formalized in Coq the proof of Mac Lane's comparison theorem for the Kleisli construction.
2. Formalization is available at https://github.com/ekiciburak/ComparisonTheorem-MacLane/tree/submissionLPAR18

# Thank you for your attention!

&

# Questions?