${}^{-1-1}2\ {}^{op op}2$
$\mathcal{U}\mathcal{U}1\ \nVdash\nVdash$
$:=:=2$

Pullback Carrier Spaces in Mathlib4

A Companion to `PullbackCarrier.lean`

1 Introduction

This document provides a natural language companion to the `PullbackCarrier.lean` file in Mathlib4. The file describes the underlying topological space of fiber products (pullbacks) of schemes, giving an explicit bijective correspondence between points of $X \times_S Y$ and certain algebraic data involving residue field tensor products.

2 The Triplet Structure

2.1 Definition of Triplets

```
structure Triplet {X Y S : Scheme.{u}} (f : X      S) (g : Y      S) where
  x : X
  y : Y
  s : S
  hx : f.base x = s
  hy : g.base y = s
```

Natural Language: A triplet over morphisms $f : X \to S$ and $g : Y \to S$ consists of points $x \in X$, $y \in Y$, and $s \in S$ such that $f(x) = s = g(y)$. This captures the idea that x and y "meet" at the point s in the base scheme.

2.2 Convenient Constructor

```
def mk' (x : X) (y : Y) (h : f.base x = g.base y) : Triplet f g where
  x := x
  y := y
  s := g.base y
  hx := h
  hy := rfl
```

Natural Language: Given points $x \in X$ and $y \in Y$ such that $f(x) = g(y)$, we can construct the corresponding triplet with $s = g(y)$ as the common image point.

3 Tensor Product Construction

3.1 Residue Field Tensor Product

```
def tensor (T : Triplet f g) : CommRingCat :=
  pushout ((S.residueFieldCongr T.hx).inv      f.residueFieldMap T.x)
    ((S.residueFieldCongr T.hy).inv      g.residueFieldMap T.y)
```

Natural Language: For a triplet (x, y, s) with $f(x) = s = g(y)$, we define the tensor product $\kappa(x) \otimes_{\kappa(s)} \kappa(y)$ as the pushout of the residue field maps. This captures the algebraic structure of the "intersection" of x and y over s.

3.2 Nontriviality

```
instance (T : Triplet f g) : Nontrivial T.tensor
```

Natural Language: The tensor product $\kappa(x) \otimes_{\kappa(s)} \kappa(y)$ is always nontrivial, reflecting the fact that residue fields are fields and their tensor products over fields are nontrivial.

4 Canonical Maps

4.1 Inclusion Maps

```
def tensorInl (T : Triplet f g) : X.residueField T.x      T.tensor := pushout.inl
    _ _
def tensorInr (T : Triplet f g) : Y.residueField T.y      T.tensor := pushout.inr
    _ _
```

Natural Language: These are the canonical inclusions $\kappa(x) \to \kappa(x) \otimes_{\kappa(s)} \kappa(y)$ and $\kappa(y) \to \kappa(x) \otimes_{\kappa(s)} \kappa(y)$ into the tensor product.

4.2 Spec Map to Pullback

```
def SpecTensorTo : Spec T.tensor      pullback f g :=
  pullback.lift (Spec.map T.tensorInl      X.fromSpecResidueField T.x)
    (Spec.map T.tensorInr      Y.fromSpecResidueField T.y)
    (Spec_map_tensorInl_fromSpecResidueField _)
```

Natural Language: This constructs the natural morphism from $\mathrm{Spec}(\kappa(x) \otimes_{\kappa(s)} \kappa(y))$ to the pullback $X \times_S Y$, using the universal property of pullbacks.

5 Point Correspondence

5.1 From Pullback Points to Triplets

```
def ofPoint (t :     (pullback f g)) : Triplet f g :=
     (pullback.fst f g).base t, (pullback.snd f g).base t, _, rfl,
    congr((Scheme.Hom.toLRSHom $(pullback.condition (f := f) (g := g))).base t).
        symm
```

Natural Language: Given a point t in the pullback $X \times_S Y$, we obtain a triplet by taking its projections to X and Y, along with the common image in S.

5.2 Tensor Product Map

```
def ofPointTensor (t :     (pullback f g)) :
    (Triplet.ofPoint t).tensor      (pullback f g).residueField t
```

Natural Language: For each point t in the pullback, there's a canonical map from the tensor product of residue fields to the residue field at t.

6 Main Bijection Theorem

6.1 The Carrier Equivalence

```
def carrierEquiv :    (pullback f g)         T : Triplet f g, Spec T.tensor where
  toFun t :=    .ofPoint t, SpecOfPoint  t
  invFun T := T.1.SpecTensorTo.base T.2
  left_inv := SpecTensorTo_SpecOfPoint
  right_inv := by
    intro  T , p
    apply carrierEquiv_eq_iff.mpr
    use T.ofPoint_SpecTensorTo p
    ...
```

Natural Language: This is the main theorem: points of $X \times_S Y$ bijectively correspond to pairs (T, p) where $T = (x, y, s)$ is a triplet with $f(x) = s = g(y)$, and p is a prime ideal of $\kappa(x) \otimes_{\kappa(s)} \kappa(y)$.

6.2 Geometric Interpretation

The bijection has the following geometric interpretation:

- A point in $X \times_S Y$ determines points $x \in X$, $y \in Y$ mapping to the same $s \in S$
- The "local behavior" at this point is captured by a prime ideal in the tensor product of residue fields
- This tensor product encodes how the local rings at x and y interact over the local ring at s

7 Existence of Preimages

7.1 Main Existence Theorem

```
lemma exists_preimage_pullback (x : X) (y : Y) (h : f.base x = g.base y) :
       z :    (pullback f g),
    (pullback.fst f g).base z = x     (pullback.snd f g).base z = y
```

Natural Language: For any points $x \in X$ and $y \in Y$ such that $f(x) = g(y)$, there exists a point in the pullback $X \times_S Y$ that maps to x and y respectively. This shows that the map from the pullback to the fiber product of underlying topological spaces is surjective.

7.2 Characterization of Empty Pullbacks

```
lemma _root_.AlgebraicGeometry.Scheme.isEmpty_pullback_iff {f : X     S} {g : Y
       S} :
    IsEmpty    (Limits.pullback f g)     Disjoint (Set.range f.base) (Set.range g
       .base)
```

Natural Language: The pullback $X \times_S Y$ is empty if and only if the images of f and g in S are disjoint. This provides a topological criterion for when fibered products are empty.

8 Range Characterizations

8.1 Range of Projections

```
lemma range_fst : Set.range (pullback.fst f g).base = f.base      ' Set.range g.
    base
lemma range_snd : Set.range (pullback.snd f g).base = g.base      ' Set.range f.
    base
```

Natural Language: The first projection has image equal to the preimage under f of the image of g, and vice versa. This characterizes exactly which points in X and Y appear in the pullback.

8.2 Composition Ranges

```
lemma range_fst_comp :
    Set.range (pullback.fst f g     f).base = Set.range f.base     Set.range g.
        base
```

Natural Language: The image of the composed map through f (or g) equals the intersection of the images of f and g in S.

9 Pullback Maps

9.1 Range Under Pullback Maps

```
lemma range_map {X' Y' S' : Scheme.{u}} (f' : X'     S') (g' : Y'     S') ( i   :
     X     X')
    ( i   : Y     Y') ( i   : S     S') ( e   : f      i   =  i       f')
    ( e   : g      i   =  i       g') [Mono  i  ] :
    Set.range (pullback.map f g f' g'  i    i    i    e    e   ).base =
      (pullback.fst f' g').base      ' Set.range  i  .base
        (pullback.snd f' g').base      ' Set.range  i  .base
```

Natural Language: For compatible morphisms forming a map between pullbacks, the image of the pullback map has a precise description in terms of preimages under the component morphisms.

10 Stability Properties

10.1 Surjectivity is Stable Under Base Change

```
instance : MorphismProperty.IsStableUnderBaseChange @Surjective
```

Natural Language: If $g : Y \to S$ is surjective, then for any $f : X \to S$, the first projection $X \times_S Y \to X$ is also surjective. This is a fundamental stability property in algebraic geometry.

11 Applications and Importance

The explicit description of pullback carrier spaces provided in this file has several important applications:

11.1 Geometric Understanding

- Provides a concrete way to understand points in fiber products
- Shows how local algebraic data (residue fields) determines global geometric properties
- Gives explicit criteria for when fiber products are empty or have certain properties

11.2 Computational Aspects

- Allows explicit computation of points in pullbacks
- Provides algorithms for checking surjectivity and other properties
- Gives concrete descriptions of ranges and preimages

11.3 Theoretical Foundation

- Establishes the connection between scheme-theoretic and topological fiber products
- Provides the foundation for more advanced constructions in algebraic geometry
- Shows how categorical pullbacks relate to concrete geometric objects

12 Conclusion

The `PullbackCarrier.lean` file provides a complete description of the underlying topological spaces of scheme-theoretic pullbacks. The main achievement is the bijective correspondence between points of $X \times_S Y$ and pairs of triplets (x, y, s) with prime ideals in tensor products of residue fields. This bridges the abstract categorical definition of pullbacks with concrete, computable descriptions of their points, making the theory of fiber products in algebraic geometry both rigorous and practical.