A CATEGORICAL THEORY OF PATCHES

SAMUEL MIMRAM
&
CINZIA DI GIUSTO

MFPS
24 JUNE 2013

DVCS

Distributed Version Control Systems are used when working collaboratively on files

SUBVERSION git darcs . . .

Those feature:

- easy import of modifications from others
- storing history of files
- maintaining different flavors (branches) of a same software
- no centralized architecture
- etc.

SOME TERMINOLOGY

A **patch** is a file coding difference between two files (i.e. the list of inserted and deleted lines).

Users can perform two actions:

- **commit** the difference between the current version and the last committed version as a patch to a server
- **update** its current version by importing all the new patches on the server

USING DVCS

b

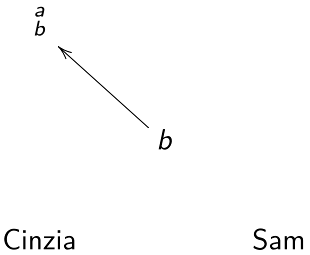
Cinzia Sam

USING DVCS

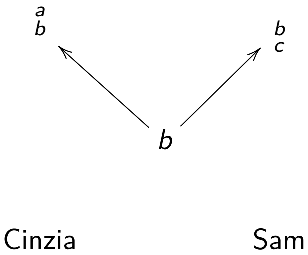
b

Cinzia Sam

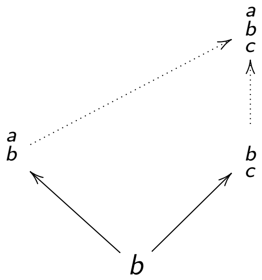
USING DVCS



USING DVCS



USING DVCS



Cinzia Sam

USING DVCS

$$\begin{tikzcd} & & \begin{matrix} a \\ b \\ c \end{matrix} \\ \begin{matrix} a \\ b \end{matrix} \arrow[urr, dotted, "f_c/f_a"] & & \begin{matrix} b \\ c \end{matrix} \arrow[u, dotted, "f_a/f_c"'] \\ & b \arrow[ul, "f_a"] \arrow[ur, "f_c"'] & \end{tikzcd}$$

Cinzia Sam

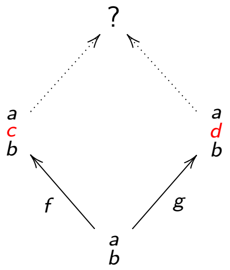
Merging modifications is naturally modeled by pushouts.

CONFLICTS

However, not every pair of coinitial morphisms has a pushout!

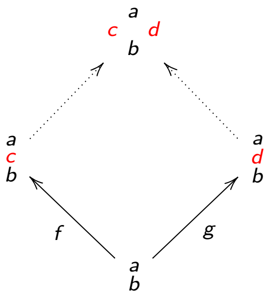
CONFLICTS

However, not every pair of coinitial morphisms has a pushout!



CONFLICTS

However, not every pair of coinitial morphisms has a pushout!




```
a
<<<<<<< HEAD
c
=======
d
>>>>>>> 5c55f7c1c4ad1e02be6d0474e858bd8ad712e22b
b
```

HANDLING CONFLICTS

We should extend our model to account for “files with conflicts” and their handling.

There were many proposals for modeling DVCS:

- Darcs: a theory based on patch commutation [Roundy,...]
- operational transformations [Ellis,Gibbs,...]
- inverse semigroups [Jacobson09]
- the Kleisli category of the exception monad [Houston12]
- ...

- Which one is the good one?
- We should start from a universal characterization!

THE STARTING POINT

- Starting from the category of files, the right model for files with conflicts can be obtained by freely adding pushouts.

THE STARTING POINT

- Starting from the category of files, the right model for files with conflicts can be obtained by freely adding pushouts.
- Since we also want an initial object (the empty file), we actually want to add all finite colimits, i.e. the

  free finite cocompletion

  of the category of files and patches.

PLAN

1. Define the category $\mathcal{L}$ of files.
2. Define abstractly the its free finite cocompletion $\mathcal{P}$.
3. Provide a concrete description of the category $\mathcal{P}$.
4. Study some examples.
5. Sketch the proof of the concrete description.

THE CATEGORY $\mathcal{L}$

We suppose fixed a set L of *lines* and write $[n] = \{0, \ldots, n-1\}$.

Definition

The category $\mathcal{L}$ has

- **files** as objects, i.e. pairs (n, ℓ) with

$$\begin{tikzcd} {[n]} \arrow[d, "\ell"'] \\ L \end{tikzcd}$$

THE CATEGORY $\mathcal{L}$

We suppose fixed a set L of *lines* and write $[n] = \{0, \ldots, n-1\}$.

Definition

The category $\mathcal{L}$ has

- **files** as objects, i.e. pairs (n, ℓ) with

$$\begin{tikzcd}
{[n]} \arrow[d, "\ell"'] \\
L
\end{tikzcd}$$

- a morphism $f : (n, \ell) \to (n', \ell')$ is a partial injective increasing function $f : [n] \to [n']$ such that

$$\begin{tikzcd}
{[n]} \arrow[rr, dotted, "f"] \arrow[dr, "\ell"'] & & {[n']} \arrow[dl, "\ell'"] \\
& L &
\end{tikzcd}$$

THE CATEGORY $\mathcal{L}$

For instance, a morphism $f : (3, \ell) \to (5, \ell')$ is

$$\begin{tikzcd}
a_0 \arrow[dr, dash] & d_0 \\
b_1 \arrow[dddr, dash] & a_1 \\
c_2 & d_2 \\
 & d_3 \\
 & b_4
\end{tikzcd}$$

which corresponds to deleting the line c and adding lines d.

(thus partial injective increasing functions)

HANDLING LABELS

Here I will focus on the case without labels, i.e. the category $\mathcal{L}$ has

- objects: integers
- morphisms: partial injective increasing functions

(the labeled case can be recovered by a slice category construction)

Proposition

The category $\mathcal{L}$ is the free category generated by

$$s_i^n : n \to n+1 \qquad \textit{and} \qquad d_i^n : n+1 \to n$$

$$s_2^4 = \begin{tikzcd}[row sep=small] 0 \arrow[r, dash] & 0 \\ 1 \arrow[r, dash] & 1 \\ 2 \arrow[dr, dash] & 2 \\ 3 \arrow[dr, dash] & 3 \\ & 4 \end{tikzcd} \qquad\qquad d_2^4 = \begin{tikzcd}[row sep=small] 0 \arrow[r, dash] & 0 \\ 1 \arrow[r, dash] & 1 \\ 2 & 2 \\ 3 \arrow[ur, dash] & 3 \\ 4 \arrow[ur, dash] & \end{tikzcd}$$

(insertion) *(deletion)*

subject to the relations

$$s_i^{n+1} s_j^n = s_{j+1}^{n+1} s_i^n \qquad d_i^n s_i^n = \mathrm{id}_n \qquad d_i^n d_j^{n+1} = d_j^n d_{i+1}^{n+1}$$

THE CATEGORY $\mathcal{L}$

Proposition

The category $\mathcal{L}$ is the free category generated by

$$s_i^n : n \to n+1 \qquad \textit{and} \qquad d_i^n : n+1 \to n$$

subject to the relations

$$s_i^{n+1} s_j^n = s_{j+1}^{n+1} s_i^n \qquad d_i^n s_i^n = \mathrm{id}_n \qquad d_i^n d_j^{n+1} = d_j^n d_{i+1}^{n+1}$$

Remark

If we restrict to *total* functions, we get patches with insertions only. We will handle this case in the following.

A SIMPLER CASE

In this talk, we will consider the case

- without labels
- without deletions

(see the article for the general case). So,

Definition

The category $\mathcal{L}$ has

- objects: $\mathbb{N}$
- morphisms $f : m \to n$ are injective increasing functions $f : [m] \to [n]$

(also known as the *augmented presimiplicial category* Δ).

What is the category $\mathcal{P}$ obtained by freely adding all finite colimits to $\mathcal{L}$?

A FREE COCOMPLETION OF $\mathcal{L}$

Our main contribution:

Theorem

The free finite conservative cocompletion $\mathcal{P}$ of $\mathcal{L}$ is the category:

- *objects $(A, \leq)$ are finite sets equipped with a transitive relation*
- *a morphism $f : A \to B$ is a function respecting the relation*

A FREE COCOMPLETION OF $\mathcal{L}$

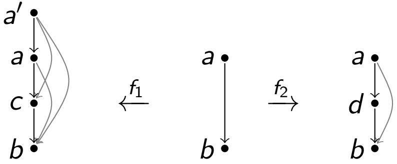
We have an embedding $\mathcal{L} \hookrightarrow \mathcal{P}$:

$$3 \qquad \rightsquigarrow \qquad ([3], <)$$

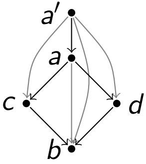
$$\begin{array}{c} a \\ b \\ c \end{array} \qquad \rightsquigarrow \qquad \begin{tikzcd} a \arrow[d] \arrow[dd, bend left] \\ b \arrow[d] \\ c \end{tikzcd}$$

A FREE COCOMPLETION OF $\mathcal{L}$

We have all pushouts, e.g. the pushout of



is



A FREE COCOMPLETION OF $\mathcal{L}$

Every object in $\mathcal{P}$ can be obtained as a colimit of objects in $\mathcal{L}$.
For instance, consider the morphisms

$$\bullet \xrightarrow{s} \begin{array}{c}\bullet\\ \downarrow \\ \bullet\end{array} \qquad \text{and} \qquad \bullet \xrightarrow{t} \begin{array}{c}\bullet\\ \downarrow \\ \bullet\end{array}$$

A FREE COCOMPLETION OF $\mathcal{L}$

Every object in $\mathcal{P}$ can be obtained as a colimit of objects in $\mathcal{L}$.
For instance, consider the morphisms

$$\begin{array}{c}\bullet\\ \\ \end{array} \xrightarrow{s} \begin{array}{c}\bullet\\ \downarrow\\ \bullet\end{array} \qquad \text{and} \qquad \begin{array}{c}\\ \\ \bullet\end{array} \xrightarrow{t} \begin{array}{c}\bullet\\ \downarrow\\ \bullet\end{array}$$

By coproduct, we get a "sequentialization" morphism

$$\bullet \quad \bullet$$

A FREE COCOMPLETION OF $\mathcal{L}$

Every object in $\mathcal{P}$ can be obtained as a colimit of objects in $\mathcal{L}$.
For instance, consider the morphisms

$$\begin{array}{c} \bullet \\ \end{array} \xrightarrow{s} \begin{tikzcd} \bullet \arrow[d] \\ \bullet \end{tikzcd} \qquad \text{and} \qquad \begin{array}{c} \\ \bullet \end{array} \xrightarrow{t} \begin{tikzcd} \bullet \arrow[d] \\ \bullet \end{tikzcd}$$

By coproduct, we get a "sequentialization" morphism

$$\begin{tikzcd} \bullet \arrow[r] & \bullet & \bullet & \bullet \arrow[l] \end{tikzcd}$$

A FREE COCOMPLETION OF $\mathcal{L}$

Every object in $\mathcal{P}$ can be obtained as a colimit of objects in $\mathcal{L}$.
For instance, consider the morphisms

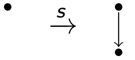
$$\begin{tikzcd} \bullet & \xrightarrow{s} & \bullet \arrow[d] \\ & & \bullet \end{tikzcd} \qquad \text{and} \qquad \begin{tikzcd} & \xrightarrow{t} & \bullet \arrow[d] \\ \bullet & & \bullet \end{tikzcd}$$

By coproduct, we get a "sequentialization" morphism

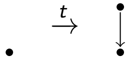
$$\begin{tikzcd} & \bullet \arrow[r] & \bullet & \\ \bullet \arrow[ur, "s"] \arrow[r] & \bullet & \bullet & \bullet \arrow[l] \arrow[ul, "t"'] \end{tikzcd}$$

A FREE COCOMPLETION OF $\mathcal{L}$

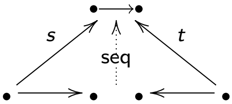
Every object in $\mathcal{P}$ can be obtained as a colimit of objects in $\mathcal{L}$.
For instance, consider the morphisms



and

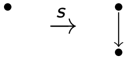


By coproduct, we get a “sequentialization” morphism

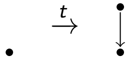


A FREE COCOMPLETION OF $\mathcal{L}$

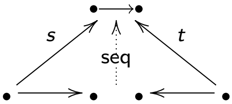
Every object in $\mathcal{P}$ can be obtained as a colimit of objects in $\mathcal{L}$.
For instance, consider the morphisms



and

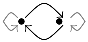


By coproduct, we get a "sequentialization" morphism



The pushout of

is



A FREE COCOMPLETION OF $\mathcal{L}$

Notice that we get a way of identifying two independent lines, which can be used to solve a conflict.

A FREE COCOMPLETION OF $\mathcal{L}$

Notice that we get a way of identifying two independent lines, which can be used to solve a conflict.

$$\bullet \longrightarrow \bullet \qquad \bullet \longleftarrow \bullet$$

A FREE COCOMPLETION OF $\mathcal{L}$

Notice that we get a way of identifying two independent lines, which can be used to solve a conflict.

$$
\begin{tikzcd}
 & & \bullet & & \\
\bullet \arrow[urr, "\mathrm{id}_\bullet"] \arrow[r] & \bullet & & \bullet & \bullet \arrow[ull, "\mathrm{id}_\bullet"'] \arrow[l]
\end{tikzcd}
$$

A FREE COCOMPLETION OF $\mathcal{L}$

Notice that we get a way of identifying two independent lines, which can be used to solve a conflict.

$$\begin{tikzcd}
 & & \bullet & & \\
\bullet \arrow[r] \arrow[urr, "\mathsf{id}_\bullet"] & \bullet & {} \arrow[u, dotted, "\mathrm{merge}"] & \bullet & \bullet \arrow[l] \arrow[ull, "\mathsf{id}_\bullet"']
\end{tikzcd}$$

A FREE COCOMPLETION

Definition

The **free cocompletion** $\mathcal{P}$ of a category $\mathcal{L}$ is the category with $y : \mathcal{L} \to \mathcal{P}$ such that for every cocomplete category $\mathcal{C}$ and functor $F : \mathcal{L} \to \mathcal{C}$, there exists $\tilde{F} : \mathcal{P} \to \mathcal{C}$ cocontinuous such that

$$\begin{tikzcd}
\mathcal{L} \arrow[r, "F"] \arrow[d, "y"'] & \mathcal{C} \\
\mathcal{P} \arrow[ur, dotted, "\tilde{F}"'] &
\end{tikzcd}$$

A FREE COCOMPLETION

Definition

The **free cocompletion** $\mathcal{P}$ of a category $\mathcal{L}$ is the category with $y : \mathcal{L} \to \mathcal{P}$ such that for every cocomplete category $\mathcal{C}$ and functor $F : \mathcal{L} \to \mathcal{C}$, there exists $\tilde{F} : \mathcal{P} \to \mathcal{C}$ cocontinuous such that

$$\begin{tikzcd} \mathcal{L} \arrow[r, "F"] \arrow[d, "y"'] & \mathcal{C} \\ \mathcal{P} \arrow[ur, dotted, "\tilde{F}"'] & \end{tikzcd}$$

Theorem (folklore)

*The free cocompletion of $\mathcal{L}$ is the category $\hat{\mathcal{L}}$ of **presheaves** over $\mathcal{L}$: functors $\mathcal{L}^{\mathrm{op}} \to \mathbf{Set}$ and natural transformations (and the embedding $y : \mathcal{L} \to \hat{\mathcal{L}}$ is given by Yoneda).*

PRESHEAVES – GRAPHS

Example

The category of graphs is the category of presheaves over the category

$$\mathcal{G} \quad = \quad V \underset{t}{\overset{s}{\rightrightarrows}} E$$

i.e. **Graph** $\cong \hat{\mathcal{G}} = [\mathcal{G}^{\mathrm{op}}, \mathbf{Set}]$. Namely, given $P \in \hat{\mathcal{G}}$ we have a diagram in **Set**

$$P(V) \underset{P(t)}{\overset{P(s)}{\leftleftarrows}} P(E)$$

i.e. a graph.

For instance,

$$\begin{tikzcd} & b \arrow[dr, "g"] & \\ a \arrow[ur, "f"] \arrow[rr, "h"'] & & c \end{tikzcd} \quad \rightsquigarrow \quad \{a, b, c\} \underset{P(t)}{\overset{P(s)}{\leftleftarrows}} \{f, g, h\}$$

PRESHEAVES – PRESIMPLICIAL SETS

Similarly, presheaves in the free cocompletion $\hat{\mathcal{L}}$ of $\mathcal{L}$ are *(augmented) presimplicial sets*:

$\in P(1)$ $\qquad$ $\in P(2)$ $\qquad$ $\in P(3)$ $\qquad$ $\in P(4)$

For instance,

$$\begin{tikzcd} b \arrow[r, "g"] & c \\ a \arrow[u, "f"] \arrow[ur, "j"] \arrow[r, "i"'] \arrow[r, phantom, shift left=3, "\alpha"] & d \arrow[u, "h"'] \end{tikzcd}$$

corresponds to $P \in \hat{\mathcal{L}}$ with

$$P(1) = \{a, b, c, d\} \qquad P(2) = \{f, g, h, i, j\} \qquad P(3) = \{\alpha\}$$

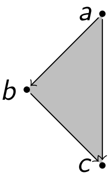
PRESHEAVES – PRESIMPLICIAL SETS

Similarly, presheaves in the free cocompletion $\hat{\mathcal{L}}$ of $\mathcal{L}$ are *(augmented) presimplicial sets*:

$\in P(1)$ $\qquad$ $\in P(2)$ $\qquad$ $\in P(3)$ $\qquad$ $\in P(4)$

In terms of files,

$$\begin{matrix} a \\ b \\ c \end{matrix} \quad \rightsquigarrow$$



PRESHEAVES – PRESIMPLICIAL SETS

Similarly, presheaves in the free cocompletion $\hat{\mathcal{L}}$ of $\mathcal{L}$ are *(augmented) presimplicial sets*:

$\in P(1)$ $\in P(2)$ $\in P(3)$ $\in P(4)$

Remark

Notice that every such presheaf has an *underlying graph*: $\mathcal{G} \hookrightarrow \mathcal{L}$.
Namely, we have the following full subcategory of $\mathcal{L}$

$$\begin{tikzcd} 0 \arrow[r] & 1 \arrow[r, shift left, "s_1^1"] \arrow[r, shift right, "s_0^1"'] & 2 \arrow[r, shift left] \arrow[r] \arrow[r, shift right] & 3 \arrow[r, shift left=2] \arrow[r, shift left=0.7] \arrow[r, shift right=0.7] \arrow[r, shift right=2] & \ldots \end{tikzcd}$$

- Why do we get such a complicated category for conflicting files?
- This is not the right completion, because we are adding again colimits which were already present in $\mathcal{L}$!

YONEDA DOES NOT PRESERVE COLIMITS

We have the following pushout in $\mathcal{L}$:

$$
\begin{tikzcd}
 & 3 & \\
2 \arrow[ur, dotted, "s_2^2"] & & 2 \arrow[ul, dotted, "s_0^2"'] \\
 & 1 \arrow[ul, "s_0^1"] \arrow[ur, "s_1^1"'] &
\end{tikzcd}
\qquad \text{for instance:} \qquad
\begin{tikzcd}
 & \begin{matrix} a \\ b \\ c \end{matrix} & \\
\begin{matrix} a \\ b \end{matrix} \arrow[ur, dotted] & & \begin{matrix} b \\ c \end{matrix} \arrow[ul, dotted] \\
 & b \arrow[ul] \arrow[ur] &
\end{tikzcd}
$$

YONEDA DOES NOT PRESERVE COLIMITS

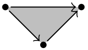
We have the following pushout in $\mathcal{L}$:

$$
\begin{tikzcd}
 & 3 & \\
2 \arrow[ur, dotted, "s_2^2"] & & 2 \arrow[ul, dotted, "s_0^2"'] \\
 & 1 \arrow[ul, "s_0^1"] \arrow[ur, "s_1^1"'] &
\end{tikzcd}
\qquad \text{for instance:} \qquad
\begin{tikzcd}
 & \begin{smallmatrix} a \\ b \\ c \end{smallmatrix} & \\
\begin{smallmatrix} a \\ b \end{smallmatrix} \arrow[ur, dotted] & & \begin{smallmatrix} b \\ c \end{smallmatrix} \arrow[ul, dotted] \\
 & b \arrow[ul] \arrow[ur] &
\end{tikzcd}
$$

Yoneda does not commute with pushouts:

$$
\begin{tikzcd}
 & ? & \\
y(2) \arrow[ur, dotted] & & y(2) \arrow[ul, dotted] \\
 & y(1) \arrow[ul, "s_0^1"] \arrow[ur, "s_1^1"'] &
\end{tikzcd}
\quad = \quad
\begin{tikzcd}
 & \bullet \longrightarrow \bullet \longrightarrow \bullet & \\
\bullet \longrightarrow \bullet \arrow[ur, dotted] & & \bullet \longrightarrow \bullet \arrow[ul, dotted] \\
 & \bullet \arrow[ul, "s_0^1"] \arrow[ur, "s_1^1"'] &
\end{tikzcd}
$$

and

$\bullet \longrightarrow \bullet \longrightarrow \bullet \quad \neq$  $= \quad y(3)$

A FREE COCOMPLETION

Definition
The **free conservative cocompletion** $\mathcal{P}$ of a category $\mathcal{L}$ is the category with cocontinuous $y : \mathcal{L} \to \mathcal{P}$ such that for every cocomplete category $\mathcal{C}$ and cocontinuous functor $F : \mathcal{L} \to \mathcal{C}$, there exists $\tilde{F} : \mathcal{P} \to \mathcal{C}$ cocontinuous such that

$$\begin{tikzcd} \mathcal{L} \arrow[r, "F"] \arrow[d, "y"'] & \mathcal{C} \\ \mathcal{P} \arrow[ur, dotted, "\tilde{F}"'] & \end{tikzcd}$$

A FREE COCOMPLETION

Definition

The **free conservative cocompletion** $\mathcal{P}$ of a category $\mathcal{L}$ is the category with cocontinuous $y : \mathcal{L} \to \mathcal{P}$ such that for every cocomplete category $\mathcal{C}$ and cocontinuous functor $F : \mathcal{L} \to \mathcal{C}$, there exists $\tilde{F} : \mathcal{P} \to \mathcal{C}$ cocontinuous such that

$$\begin{tikzcd} \mathcal{L} \arrow[r, "F"] \arrow[d, "y"'] & \mathcal{C} \\ \mathcal{P} \arrow[ur, dotted, "\tilde{F}"'] & \end{tikzcd}$$

Theorem (Kelly)

The free cocompletion of $\mathcal{L}$ is the full subcategory of $\hat{\mathcal{L}}$ whose objects are continuous presheaves.

A FREE COCOMPLETION

Definition

The **free conservative cocompletion** $\mathcal{P}$ of a category $\mathcal{L}$ is the category with cocontinuous $y : \mathcal{L} \to \mathcal{P}$ such that for every cocomplete category $\mathcal{C}$ and cocontinuous functor $F : \mathcal{L} \to \mathcal{C}$, there exists $\tilde{F} : \mathcal{P} \to \mathcal{C}$ cocontinuous such that

$$\begin{tikzcd} \mathcal{L} \arrow[r, "F"] \arrow[d, "y"'] & \mathcal{C} \\ \mathcal{P} \arrow[ur, dotted, "\tilde{F}"'] & \end{tikzcd}$$

Remark

The *finite* conservative cocompletion can be obtained by further restricting to "finite" presheaves.

Theorem (Kelly)

The free cocompletion of $\mathcal{L}$ is the full subcategory of $\hat{\mathcal{L}}$ whose objects are continuous presheaves P:

$$P(\operatorname{colim} D) \quad \cong \quad \lim(P \circ D)$$

whenever D is a diagram in $\mathcal{L}$ admitting a colimit.

So we have to

1. find properties satisfied by continuous presheaves
2. characterize all diagrams which admits a colimit in $\mathcal{L}$
3. show that presheaves satisfying 1. are the continuous ones

CONTINUOUS PRESHEAVES IN $\hat{\mathcal{L}}$

We have the following pushout in $\mathcal{L}$:

$$
\begin{tikzcd}
 & 3 & \\
2 \arrow[ur, dotted, "s_2^2"] & & 2 \arrow[ul, dotted, "s_0^2"'] \\
 & 1 \arrow[ul, "s_0^1"] \arrow[ur, "s_1^1"'] &
\end{tikzcd}
\qquad \text{for instance:} \qquad
\begin{tikzcd}
 & \begin{smallmatrix} a \\ b \\ c \end{smallmatrix} & \\
\begin{smallmatrix} a \\ b \end{smallmatrix} \arrow[ur, dotted] & & \begin{smallmatrix} b \\ c \end{smallmatrix} \arrow[ul, dotted] \\
 & b \arrow[ul] \arrow[ur] &
\end{tikzcd}
$$

Given a continuous $P \in \hat{\mathcal{L}}$, we should have a pullback in **Set**

$$
\begin{tikzcd}
 & P(3) \arrow[dl, dotted, "P(s_2^2)"'] \arrow[dr, dotted, "P(s_0^2)"] & \\
P(2) \arrow[dr, "P(s_0^1)"'] & & P(2) \arrow[dl, "P(s_1^1)"] \\
 & P(1) &
\end{tikzcd}
$$

i.e. $P(3) \cong P(2) \times_{P(1)} P(2)$:

$P(3)$ is the set of paths of length 2 in the underlying graph of P.

CONTINUOUS PRESHEAVES IN $\hat{\mathcal{L}}$

By elaborating on this idea:

Proposition

A continuous presheaf $P \in \hat{\mathcal{L}}$ satisfies

1. *for each non-empty path $x \twoheadrightarrow y$ there exists exactly one edge $x \to y$:*

$$
\begin{tikzcd}
 & x_1 \arrow[r] & x_2 \arrow[r] & \dots \arrow[r] & x_n \arrow[dr] & \\
x \arrow[ur] \arrow[rrrrr, dotted] & & & & & y
\end{tikzcd}
$$

   (in particular there is at most one edge between two vertices),

2. *$P(n+1)$ is the set of paths of length n in the underlying graph of P, and $P(0)$ is reduced to one element.*

CONTINUOUS PRESHEAVES IN $\hat{\mathcal{L}}$

By elaborating on this idea:

Proposition

A continuous presheaf $P \in \hat{\mathcal{L}}$ satisfies

1. *for each non-empty path $x \twoheadrightarrow y$ there exists exactly one edge $x \to y$:*

$$\begin{tikzcd} & x_1 \arrow[r] & x_2 \arrow[r] & \dots \arrow[r] & x_n \arrow[dr] & \\ x \arrow[ur] \arrow[rrrrr, dotted] & & & & & y \end{tikzcd}$$

   (in particular there is at most one edge between two vertices),

2. *$P(n+1)$ is the set of paths of length n in the underlying graph of P, and $P(0)$ is reduced to one element.*

Remark

Such a presheaf is characterized by its underlying graph, whose edges form transitive relation on its set of vertices.

We want to show that this is
a characterization
of continuous presheaves.

$$P(\operatorname{colim} D) \quad \cong \quad \lim(P \circ D)$$

whenever D is a diagram in $\mathcal{L}$ admitting a colimit

COLIMIT DIAGRAMS IN $\mathcal{L}$

We saw that we have the following pushout in $\mathcal{L}$

$$\begin{tikzcd}
 & 3 & \\
2 \arrow[ur, dotted, "s_2^2"] & & 2 \arrow[ul, dotted, "s_0^2"'] \\
 & 1 \arrow[ul, "s_0^1"] \arrow[ur, "s_1^1"'] &
\end{tikzcd}$$

More generally, every object $n \in \mathcal{L}$ is a colimit of objects 1 and 2 (the inclusion functor $\mathcal{G} \hookrightarrow \mathcal{L}$ is dense)

COLIMIT DIAGRAMS IN $\mathcal{L}$

We saw that we have the following pushout in $\mathcal{L}$

$$\begin{tikzcd} & 3 & \\ 2 \arrow[ur, dotted, "s_2^2"] & & 2 \arrow[ul, dotted, "s_0^2"'] \\ & 1 \arrow[ul, "s_0^1"] \arrow[ur, "s_1^1"'] & \end{tikzcd}$$

More generally, every object $n \in \mathcal{L}$ is a colimit of objects 1 and 2 (the inclusion functor $\mathcal{G} \hookrightarrow \mathcal{L}$ is dense)

In order to test that $P \in \hat{\mathcal{L}}$ sends the colimit of every diagram D to a limit, we can restrict to those where

- the objects are 1 and 2
- the morphisms are

$$s_0^1 : 1 \to 2 \quad \text{and} \quad s_1^1 : 1 \to 2$$

COLIMIT DIAGRAMS IN $\mathcal{L}$

Notice that these two diagrams always admit the same colimits:

$$
\begin{tikzcd}
{} & & B & & {} \\
& A \arrow[ul, dotted] \arrow[ur, "s"] & & A \arrow[ul, "s"'] \arrow[ur, dotted] &
\end{tikzcd}
\qquad
\begin{tikzcd}
{} & B & {} \\
& A \arrow[ul, dotted] \arrow[u, "s"] \arrow[ur, dotted] &
\end{tikzcd}
$$

COLIMIT DIAGRAMS IN $\mathcal{L}$

Notice that these two diagrams always admit the same colimits:

$$
\begin{tikzcd}
 & & \color{red}{C} & & \\
 {} & & B \arrow[u, red, "f"'] & & {} \\
 & A \arrow[ul, dotted] \arrow[ur, "s"] \arrow[uur, red, bend left, "f \circ s"] & & A \arrow[ul, "s"'] \arrow[uul, red, bend right, "f \circ s"'] \arrow[ur, dotted] &
\end{tikzcd}
\qquad\qquad
\begin{tikzcd}
 & \color{red}{C} & \\
 {} & B \arrow[u, red, "f"] & {} \\
 & A \arrow[ul, dotted] \arrow[u, "s"] \arrow[uu, red, bend left, "f \circ s"] \arrow[ur, dotted] &
\end{tikzcd}
$$

COLIMIT DIAGRAMS IN $\mathcal{L}$

Notice that these two diagrams always admit the same colimits:

$$
\begin{tikzcd}
 & & \textcolor{red}{C} & & \\
 {} & & B \arrow[u, red, "f"] & & {} \\
 & A \arrow[ul, dotted] \arrow[ur, "s"] \arrow[uur, red, bend left, "f\circ s"] & & A \arrow[ul, "s"'] \arrow[ur, dotted] \arrow[uul, red, bend right, "f\circ s"'] &
\end{tikzcd}
\qquad
\begin{tikzcd}
 & \textcolor{red}{C} & \\
 {} & B \arrow[u, red, "f"] & {} \\
 & A \arrow[ul, dotted] \arrow[u, "s"] \arrow[uu, red, bend left=40, "f\circ s"] \arrow[ur, dotted] &
\end{tikzcd}
$$

By elaborating on this idea, we can restrict to diagrams in which every object 2 is the target of

- one morphism $s_0^1 : 1 \to 2$
- and one morphism $s_1^1 : 1 \to 2$

COLIMIT DIAGRAMS IN $\mathcal{L}$

Those diagrams are of the form

$$\mathrm{El}(G) \xrightarrow{\pi} \mathcal{L}$$

for some graph $G \in \hat{\mathcal{G}}$.

For instance the diagram

$$\begin{tikzcd}
& 2 & & 2 & & 2 & \\
1 \arrow[ur, "s_1^1"] & & 1 \arrow[ul, "s_0^1"'] \arrow[ur, "s_1^1"] \arrow[drr, "s_1^1"'] & & 1 \arrow[ul, "s_0^1"'] \arrow[ur, "s_1^1"] & & 1 \arrow[ul, "s_0^1"'] \arrow[dll, "s_0^1"] \\
& & & & 2 & &
\end{tikzcd}$$

is "described" by the graph

$$\begin{tikzcd}
\bullet \arrow[r] & \bullet \arrow[r] \arrow[rr, bend right] & \bullet \arrow[r] & \bullet
\end{tikzcd}$$

COLIMIT DIAGRAMS IN $\mathcal{L}$

Those diagrams are of the form

$$\mathsf{El}(G) \xrightarrow{\pi} \mathcal{L}$$

for some graph $G \in \hat{\mathcal{G}}$.

Theorem (Paré'73,Street,Walters'73)

Any functor $F : \mathcal{C} \to \mathcal{D}$ factorizes in an essentially unique way into

- *a final functor (= does not changes colimit)*
- *followed by a discrete fibration (= "described" by a presheaf)*

COLIMIT DIAGRAMS IN $\mathcal{L}$

Those diagrams are of the form

$$\mathsf{El}(G) \xrightarrow{\pi} \mathcal{L}$$

for some graph $G \in \hat{\mathcal{G}}$.

From there and $R_F \dashv N_F$ associated to $F : \mathcal{G} \hookrightarrow \mathcal{L}$, we can

- characterize the graphs G such that the diagram

  $$\mathsf{El}(G) \xrightarrow{\pi} \mathcal{L}$$

  admits a colimit in $\mathcal{L}$,
- show that those diagrams are preserved by presheaves satisfying the previous properties.

THE FREE FINITE COCOMPLETION

The properties we have shown earlier actually characterize presheafs in $\hat{\mathcal{L}}$ which are continuous. Thus,

Theorem

The free finite conservative cocompletion $\mathcal{P}$ of $\mathcal{L}$ is the category:

- *objects $(A, \leq)$ are finite sets equipped with a transitive relation*
- *a morphism $f : A \to B$ is a function respecting the relation*

WHAT WE HAVE

- A characterization of the category of files with conflicts, starting from a universal property.
- We have shown the case of patches with insertions, but we can handle deletions and labels too.
- Pushouts can be computed concretely.
- Interestingly we recover Houston's category (up to op)!

FUTURE WORKS

- A presentation of the free cocompletion:
  what are "atomic patches" and their relations?
- Extend to more complex data structures:
  multiples files, structured files (XML), etc.
- Links with event structures in order to handle common operations: branches, cherry-picking, etc.
  (this is some form of game semantics!)