A tutorial on (generalized) fibrations for logic, automata and language theory

Noam Zeilberger[1]

Ecole Polytechnique (Laboratoire d'Informatique de l'X)

Dagstuhl, 2 April 2025
Workshop on "Categories for Automata and Language Theory"

[1]Based on joint work with Paul-André Melliès. Main references: "Functors are type refinement systems" (POPL 2015) + "The categorical contours of the Chomsky-Schützenberger representation theorem" (LMCS, to appear, arXiv:2405.14703).

Some topological intuitions

Motivation

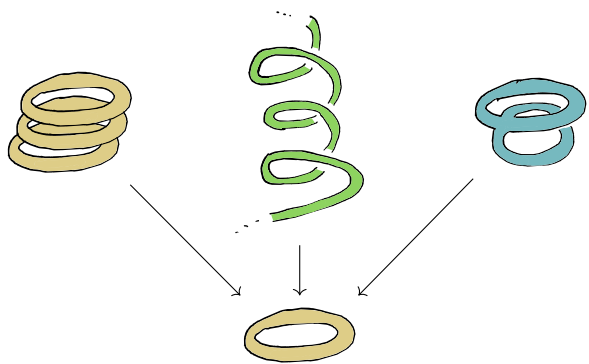
Imagine we are analyzing different spaces, more or less complicated.

Motivation

Imagine we are analyzing different spaces, more or less complicated. Rather than trying to study these spaces directly, it can be helpful to first project them down to some "simpler" space.

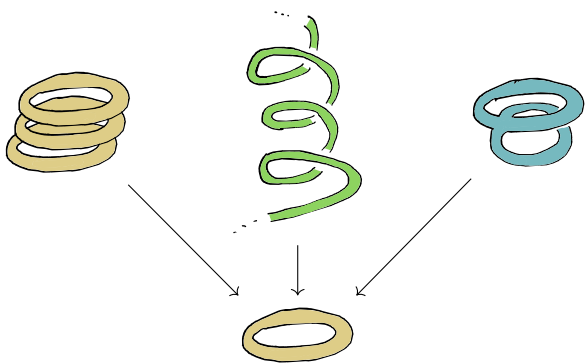
Motivation

Imagine we are analyzing different spaces, more or less complicated. Rather than trying to study these spaces directly, it can be helpful to first project them down to some "simpler" space.



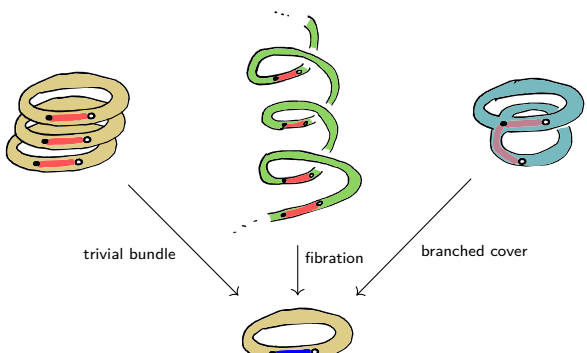
Motivation

Imagine we are analyzing different spaces, more or less complicated. Rather than trying to study these spaces directly, it can be helpful to first project them down to some "simpler" space. We can learn something about our original spaces by the nature of these projections, and even try to reconstruct them from their fibers...



Motivation

Imagine we are analyzing different spaces, more or less complicated. Rather than trying to study these spaces directly, it can be helpful to first project them down to some "simpler" space. We can learn something about our original spaces by the nature of these projections, and even try to reconstruct them from their fibers...



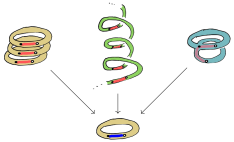
Basic terminology

A space E mapping down to a space B is sometimes called a "bundle" $p : E \to B$.

E is called the **total space**, B the **base space**, p the **projection**.

For any point $a \in B$, its inverse image $p^{-1}(a)$ is called the **fiber** over a. In general it is not just a set of isolated points, but a subspace of E.

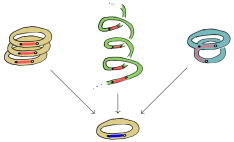
Path lifting problems



Given a bundle $p : E \to B$, we may consider different kinds of "lifting" problems...

$$\begin{array}{c} E \\ \downarrow \\ B \end{array}$$

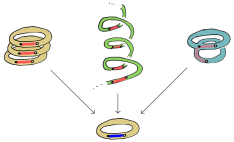
Path lifting problems



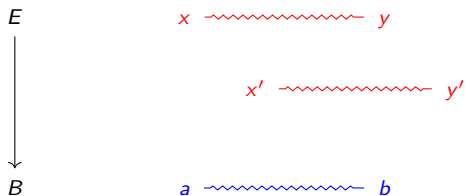
Given a bundle $p : E \to B$, we may consider different kinds of "lifting" problems...

(lift a path from the base space to the total space)

Path lifting problems

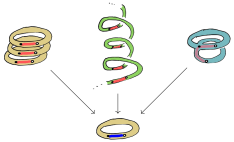


Given a bundle $p : E \to B$, we may consider different kinds of "lifting" problems...



(lift a path from the base space to the total space)

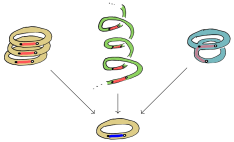
Path lifting problems



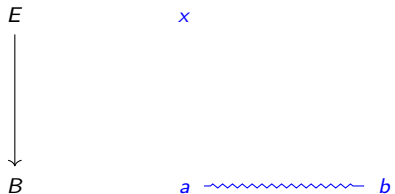
Given a bundle $p : E \to B$, we may consider different kinds of "lifting" problems...

$$\begin{array}{c} E \\ \downarrow \\ B \end{array}$$

Path lifting problems

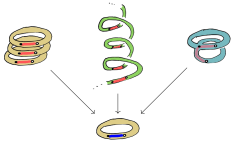


Given a bundle $p : E \to B$, we may consider different kinds of "lifting" problems...

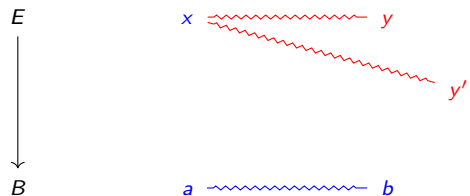


(lift a path from the base space to a given point in the total space)

Path lifting problems

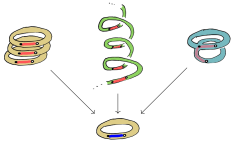


Given a bundle $p : E \to B$, we may consider different kinds of "lifting" problems...



(lift a path from the base space to a given point in the total space)

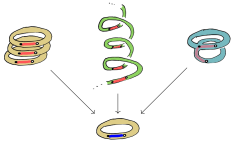
Path lifting problems



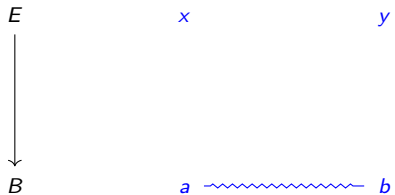
Given a bundle $p : E \to B$, we may consider different kinds of "lifting" problems...

$$\begin{array}{c} E \\ \downarrow \\ B \end{array}$$

Path lifting problems

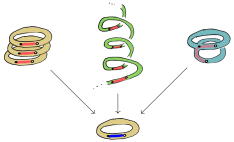


Given a bundle $p : E \to B$, we may consider different kinds of "lifting" problems...

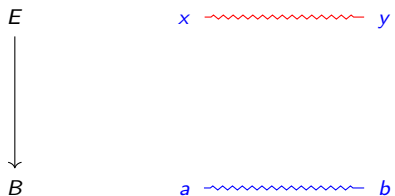


(lift a path from the base space between two given points of the total space)

Path lifting problems



Given a bundle $p : E \to B$, we may consider different kinds of "lifting" problems...



(lift a path from the base space between two given points of the total space)

Fibrational perspectives on deductive systems

Old idea (Lawvere 1969): the structure of predicate logic and the nature of quantifiers may be clarified by organizing proofs into bundles lying over spaces of formulas.

This idea may be extended to a variety of computational and deductive systems, including program logics, finite-state automata and context-free grammars.

Proof, recognition, and parsing may be thus reduced to lifting problems.

Plan

1. Functors as type refinement systems (aka, bundles of categories)
2. Finite-state automata as bundles I: determinism and codeterminism
3. Finite-state automata as bundles II: ULF and finitary functors
4. Finite-state automata as bundles III: NFAs over categories

Functors as type refinement systems (aka, bundles of categories)

Some terminology and notation

We like to think of a category $\mathcal{D}$ equipped with a functor $p : \mathcal{D} \to \mathcal{C}$ as defining a kind of abstract type system, or **type refinement system**. The idea is that the bundle $p : \mathcal{D} \to \mathcal{C}$ provides a source of *additional typing information* for the arrows of $\mathcal{C}$.

Some terminology and notation

We like to think of a category $\mathcal{D}$ equipped with a functor $p : \mathcal{D} \to \mathcal{C}$ as defining a kind of abstract type system, or **type refinement system**. The idea is that the bundle $p : \mathcal{D} \to \mathcal{C}$ provides a source of *additional typing information* for the arrows of $\mathcal{C}$.

Let $R \in \mathcal{D}$ and $A \in \mathcal{C}$ be objects s.t. $p(R) = A$. We write $R \sqsubset A$ and say R *refines* A.

Some terminology and notation

We like to think of a category $\mathcal{D}$ equipped with a functor $p : \mathcal{D} \to \mathcal{C}$ as defining a kind of abstract type system, or **type refinement system**. The idea is that the bundle $p : \mathcal{D} \to \mathcal{C}$ provides a source of *additional typing information* for the arrows of $\mathcal{C}$.

Let $R \in \mathcal{D}$ and $A \in \mathcal{C}$ be objects s.t. $p(R) = A$. We write $R \sqsubset A$ and say R *refines* A.

A *typing judgment* is a triple (S, f, T) of an arrow $f : A \to B$ of $\mathcal{C}$ and a pair of objects $S \sqsubset A$ and $T \sqsubset B$ of $\mathcal{D}$ refining its domain and codomain.

Some terminology and notation

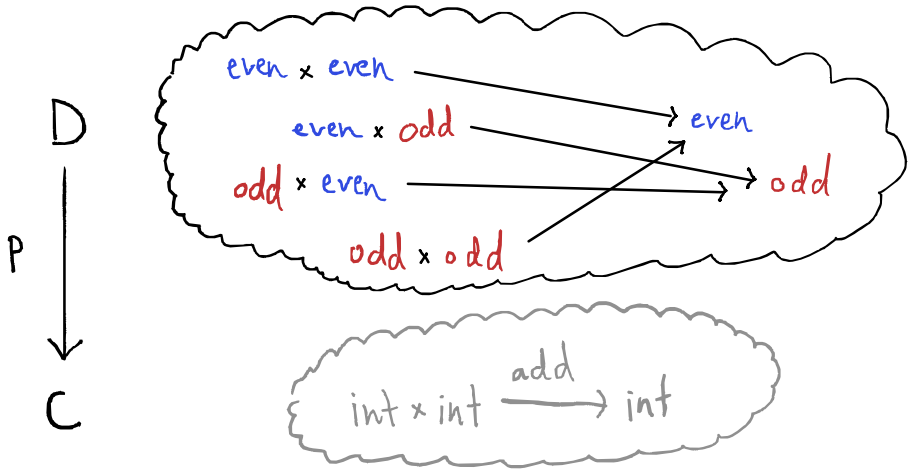
We like to think of a category $\mathcal{D}$ equipped with a functor $p : \mathcal{D} \to \mathcal{C}$ as defining a kind of abstract type system, or **type refinement system**. The idea is that the bundle $p : \mathcal{D} \to \mathcal{C}$ provides a source of *additional typing information* for the arrows of $\mathcal{C}$.

Let $R \in \mathcal{D}$ and $A \in \mathcal{C}$ be objects s.t. $p(R) = A$. We write $R \sqsubset A$ and say R *refines* A.

A *typing judgment* is a triple (S, f, T) of an arrow $f : A \to B$ of $\mathcal{C}$ and a pair of objects $S \sqsubset A$ and $T \sqsubset B$ of $\mathcal{D}$ refining its domain and codomain.

A *derivation* of a typing judgment (S, f, T) is an arrow $\alpha : S \to T$ of $\mathcal{D}$ s.t. $p(\alpha) = f$.

A picture to have in mind



Sequent calculus

We use sequent notation $S \Longrightarrow_f T$ for judgments, and write inference rules to stand for transformations from derivations of the premises to a derivation of the conclusion.

Proposition: the following rules are valid for any type refinement system $p : \mathcal{D} \to \mathcal{C}$.

$$\frac{S \underset{f}{\Longrightarrow} R \qquad R \underset{g}{\Longrightarrow} T}{S \underset{fg}{\Longrightarrow} T}\ \text{COMP} \qquad\qquad \frac{R \sqsubset A}{R \underset{id_A}{\Longrightarrow} R}\ \text{ID}$$

Sequent calculus

We use sequent notation $S \Longrightarrow_f T$ for judgments, and write inference rules to stand for transformations from derivations of the premises to a derivation of the conclusion.

Proposition: the following rules are valid for any type refinement system $p : \mathcal{D} \to \mathcal{C}$.

$$\frac{S \underset{f}{\Longrightarrow} R \qquad R \underset{g}{\Longrightarrow} T}{S \underset{fg}{\Longrightarrow} T}\ \text{COMP} \qquad\qquad \frac{R \sqsubset A}{R \underset{id_A}{\Longrightarrow} R}\ \text{ID}$$

Proof:

(COMP): Let α and β be derivations of the premises, meaning that there are arrows $\alpha : S \to R$ and $\beta : R \to T$ in $\mathcal{D}$ such that $p(\alpha) = f$ and $p(\beta) = g$. Then the arrow $\alpha\beta : S \to T$ is a derivation of the conclusion, since $p(\alpha\beta) = p(\alpha)p(\beta) = fg$.

Sequent calculus

We use sequent notation $S \Longrightarrow_f T$ for judgments, and write inference rules to stand for transformations from derivations of the premises to a derivation of the conclusion.

Proposition: the following rules are valid for any type refinement system $p : \mathcal{D} \to \mathcal{C}$.

$$\frac{S \underset{f}{\Longrightarrow} R \qquad R \underset{g}{\Longrightarrow} T}{S \underset{fg}{\Longrightarrow} T}\ \text{COMP} \qquad\qquad \frac{R \sqsubset A}{R \underset{id_A}{\Longrightarrow} R}\ \text{ID}$$

Proof:

(COMP): Let α and β be derivations of the premises, meaning that there are arrows $\alpha : S \to R$ and $\beta : R \to T$ in $\mathcal{D}$ such that $p(\alpha) = f$ and $p(\beta) = g$. Then the arrow $\alpha\beta : S \to T$ is a derivation of the conclusion, since $p(\alpha\beta) = p(\alpha)p(\beta) = fg$.

(ID): $p(id_R : R \to R) = id_{p(R)} = id_A$.

Subtyping

We define subtyping as a special case of typing:

$$R \leqslant_A R' \quad := \quad R \underset{id_A}{\Longrightarrow} R'$$

Note that subtyping is only meaningful between refinements of the same type.

Proposition: the following rule is valid for any type refinement system $p : \mathcal{D} \to \mathcal{C}$.

$$\frac{S \leqslant S' \qquad S' \underset{f}{\Longrightarrow} T' \qquad T' \leqslant T}{S \underset{f}{\Longrightarrow} T}\ \text{CONS}$$

Subtyping

We define subtyping as a special case of typing:

$$R \leqslant_A R' \quad := \quad R \underset{id_A}{\Longrightarrow} R'$$

Note that subtyping is only meaningful between refinements of the same type.

Proposition: the following rule is valid for any type refinement system $p : \mathcal{D} \to \mathcal{C}$.

$$\frac{S \leqslant S' \qquad S' \underset{f}{\Longrightarrow} T' \qquad T' \leqslant T}{S \underset{f}{\Longrightarrow} T} \text{ CONS}$$

Proof:

$$\begin{array}{ccccccccc} \mathcal{D} & & S & \xrightarrow{\alpha} & S' & \xrightarrow{\beta} & T' & \xrightarrow{\gamma} & T \\ {\scriptstyle p}\downarrow & & & & & & & & \\ \mathcal{C} & & A & = & A & \xrightarrow{f} & B & = & B \end{array}$$

Subtyping

Restricting to subtyping derivations defines a family of *fiber categories*

$$\mathcal{D}_A = p^{-1}(id_A) = \{\, \alpha : R \to R' \mid p(\alpha) = id_A \,\} \subset \mathcal{D}.$$

If these contain only identity arrows (i.e., $p(\alpha) = id_A$ implies $\alpha = id_R$ for some R) then subtyping is trivial and p is said to have **discrete fibers**.

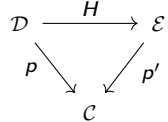
Ambiguity

A judgment $S \Longrightarrow_f T$ may in general have more than one derivation, in other words, a type refinement system may be *ambiguous*.

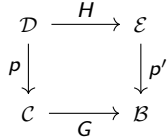
By definition, $p : \mathcal{D} \to \mathcal{C}$ is unambiguous iff p is **faithful**, i.e., for any pair of arrows $\alpha, \alpha' : S \to T$ with the same source and target, if $p(\alpha) = p(\alpha')$ then $\alpha = \alpha'$.

Morphism of type refinement systems

Two bundles over $\mathcal{C}$ may be related by a commutative triangle:



More generally, two bundles may be related by a commutative square:



Such a morphism maps p-derivations of $S \Longrightarrow_f T$ to p'-derivations of $H\,S \Longrightarrow_{G\,f} H\,T$.

General perspectives on type refinement systems

This perspective may be applied to *any* functor $p : \mathcal{D} \to \mathcal{C}$.

We are typically interested in functors with some additional properties/structure.

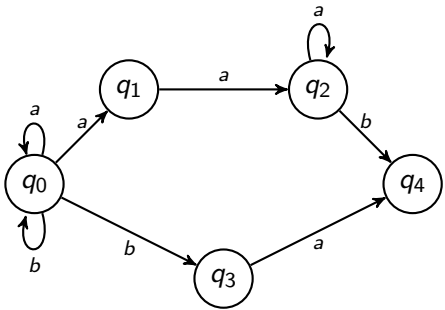
Considering different logico-computational systems as bundles, what are their fibrational characteristics?

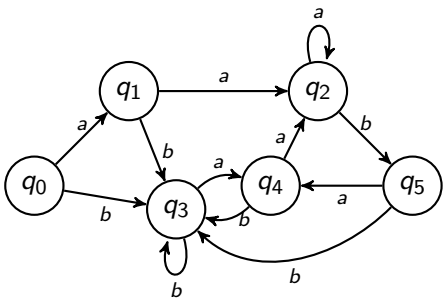
Finite-state automata as bundles I: determinism and codeterminism

Automata as graph homomorphisms

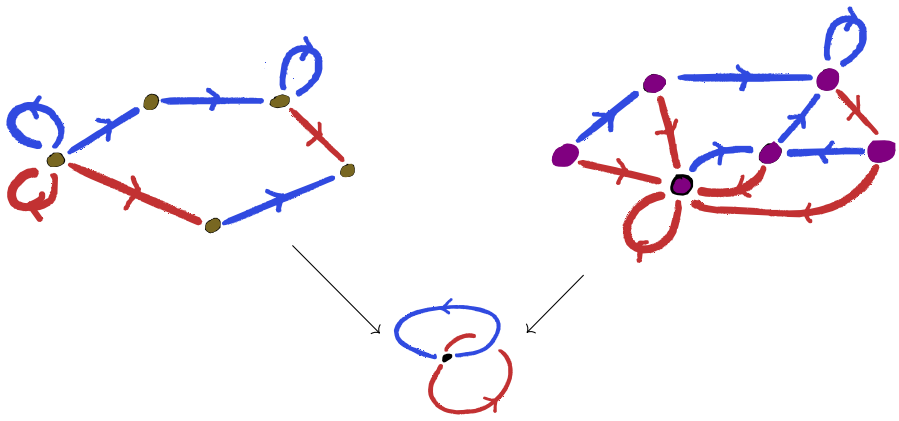
The underlying transition graph of any NFA (without ϵ-transitions) over the alphabet Σ is entirely described by a graph homomorphism $\phi : \mathbb{G} \to \mathbb{B}_\Sigma$ into the *bouquet graph* with one node $*$ and a loop $a : * \to *$ for every $a \in \Sigma$.

Automata as graph homomorphisms

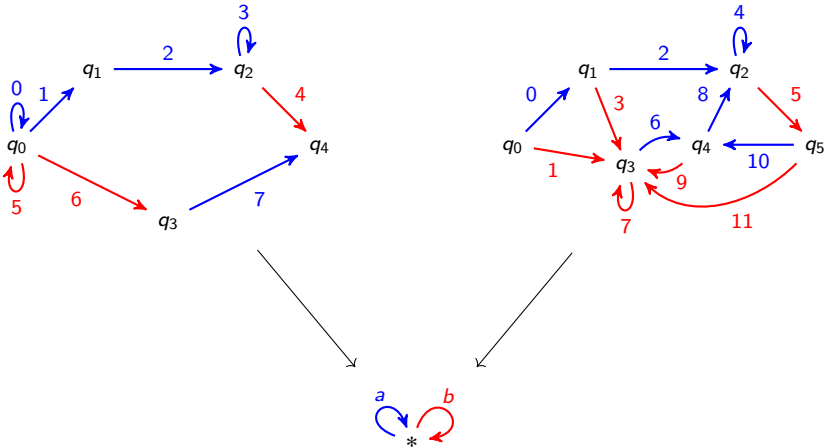




Automata as graph homomorphisms



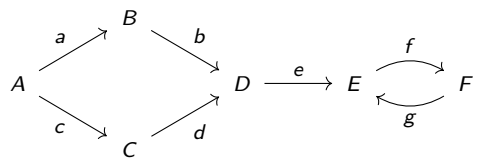
Automata as graph homomorphisms



Free categories

To any graph $\mathbb{G}$ is associated a **free category** $\mathcal{F}\mathbb{G}$ whose objects are nodes and whose arrows are paths. For example, the free category over

$\mathbb{G}$ =



has $\hom(A, D) = \{\, ab, cd \,\}$ and $\hom(E, E) = (fg)^*$.

Universal property of free categories: any functor $\mathcal{F}\mathbb{G} \to \mathcal{C}$ into a category $\mathcal{C}$ is uniquely determined by a graph homomorphism $\mathbb{G} \to \mathcal{C}$ into the underlying graph of $\mathcal{C}$.

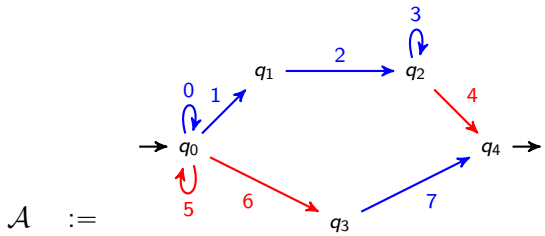
Recognition as a path lifting problem

Any *word* $w \in \Sigma^*$ corresponds to a *path* in $\mathbb{B}_\Sigma$, i.e., to an arrow $* \to *$ in $\mathcal{F}\mathbb{B}_\Sigma$.

Any graph homomorphism $\phi : \mathbb{G} \to \mathbb{H}$ induces a corresponding functor between free categories $\mathcal{F}\phi : \mathcal{F}\mathbb{G} \to \mathcal{F}\mathbb{H}$, sending paths in $\mathbb{G}$ to paths in $\mathbb{H}$ *of the same length*.

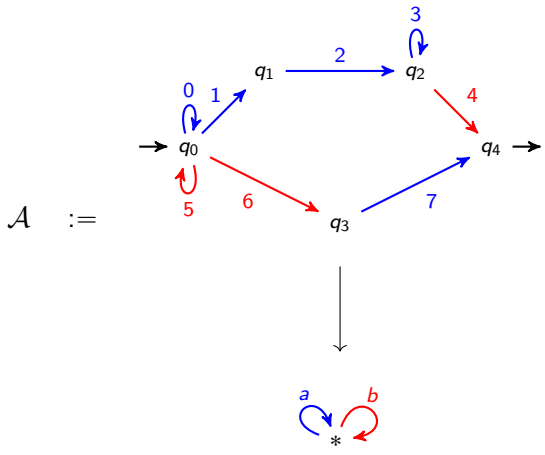
Let $\mathcal{A}$ be an NFA with transition graph $\phi : \mathbb{G} \to \mathbb{B}_\Sigma$ and associated functor $p = \mathcal{F}\phi$. Then $\mathcal{A}$ accepts $w \in \Sigma^*$ just in case there is an arrow $\alpha : q_0 \to q_f$ in $\mathcal{F}\mathbb{G}$ such that $p(\alpha) = w$, from an initial state q_0 to an accepting state q_f.

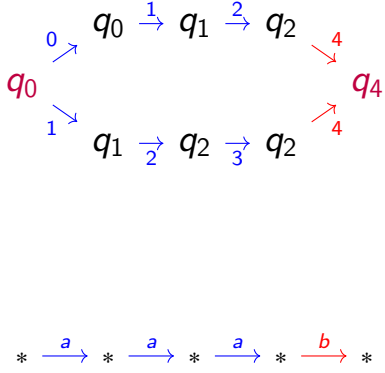
Recognition as a path lifting problem



q_0 q_4

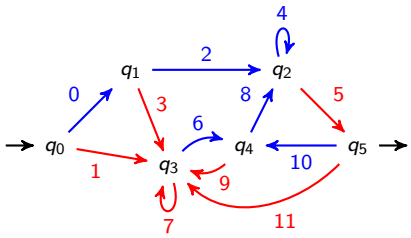
Recognition as a path lifting problem





Recognition as a path lifting problem

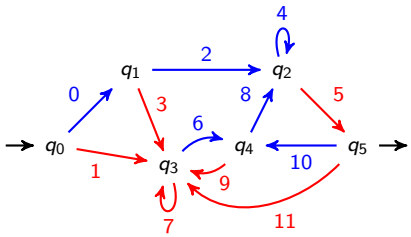
$\mathcal{A}$:=



q_0

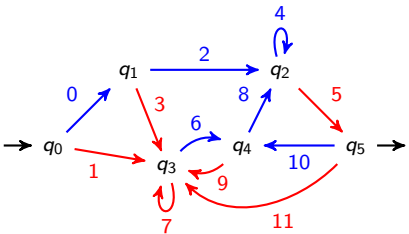
Recognition as a path lifting problem

$\mathcal{A}$:=



$q_0 \xrightarrow{0} q_1$

Recognition as a path lifting problem

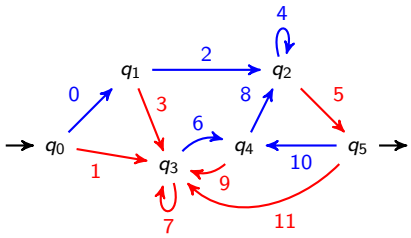


$\mathcal{A} \quad :=$

$q_0 \xrightarrow{0} q_1 \xrightarrow{2} q_2$

Recognition as a path lifting problem

$\mathcal{A}$:=

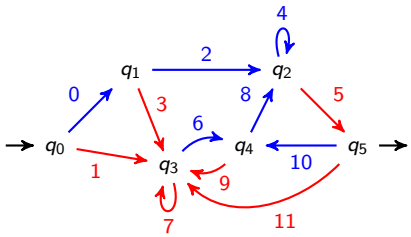


$q_0 \xrightarrow{0} q_1 \xrightarrow{2} q_2 \xrightarrow{4} q_2$

$* \xrightarrow{a} * \xrightarrow{a} * \xrightarrow{a} * \xrightarrow{b} *$

Recognition as a path lifting problem

$\mathcal{A}$:=



$q_0 \xrightarrow{0} q_1 \xrightarrow{2} q_2 \xrightarrow{4} q_2 \xrightarrow{5} q_5$

$* \xrightarrow{a} * \xrightarrow{a} * \xrightarrow{a} * \xrightarrow{b} *$

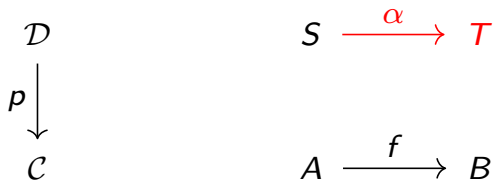
Determinism

Proposition: let $\phi : \mathbb{G} \to \mathbb{B}_\Sigma$ be a homomorphism of finite graphs. TFAE:

1. ϕ is the transition graph of a complete DFA.
2. for every node $q \in \mathbb{G}$ and every loop $a : * \to * \in \mathbb{B}_\Sigma$, there is a unique node q' and unique edge $e : q \to q'$ such that $\phi(e) = a$.
3. $\mathcal{F}\phi : \mathcal{F}\mathbb{G} \to \mathcal{F}\mathbb{B}_\Sigma$ is a *discrete opfibration*.

Definition

A functor $p : \mathcal{D} \to \mathcal{C}$ is a **discrete opfibration** if for any object $S \sqsubset A$ of $\mathcal{D}$ and any arrow $f : A \to B$ of $\mathcal{C}$ there exists a unique object $T \sqsubset B$ and arrow $\alpha : S \Longrightarrow_f T$.



Presheaves

A **covariant presheaf** on a category $\mathcal{C}$ is a functor $G : \mathcal{C} \to \mathrm{Set}$.

Explicitly, a covariant presheaf G consists of the following:

- a set G_A for every object A in $\mathcal{C}$;
- a function $G_f : G_A \to G_B$ for every arrow $f : A \to B$ in $\mathcal{C}$;
- such that $G_{fg} = G_f \,;G_g$ and $G_{id_A} = id_{G_A}$.

Category of elements

Given a presheaf $G : \mathcal{C} \to \mathrm{Set}$, the **category of elements** $\int G$ is defined like so:

- Objects are pairs (A, R) of an object A in $\mathcal{C}$ and an element $R \in G_A$.
- A morphism $(A, S) \to (B, T)$ is given by a morphism $f : A \to B$ in $\mathcal{C}$ such that the function $G_f : G_A \to G_B$ maps S to T.

This category is equipped with an evident projection functor $\pi_G : \int G \to \mathcal{C}$.

Proposition: $\pi_G : \int G \to \mathcal{C}$ is a discrete opfibration.

Example: monoid actions

Let M be a monoid, considered as a one-object category $\mathcal{B}_M =$ 

.

A presheaf $G : \mathcal{B}_M \to \mathrm{Set}$ consists of the following:

- a set $Q = G_*$;
- for every element $x \in M$, a function $G_x : Q \to Q$;
- such that $(q)G_{xy} = ((q)G_x)G_y$ and $(q)G_1 = q$.

This is equivalent to the data of a set Q equipped with a *right action* $Q \times M \to Q$.

To any such monoid action is associated a discrete opfibration $\pi_G : \int G \to \mathcal{B}_M$.

Example: monoid actions

When M is a free monoid $M = \Sigma^*$, then any any functor $\mathcal{B}_M \cong \mathcal{F}\mathbb{B}_\Sigma \to \mathrm{Set}$ is uniquely determined by a graph homomorphism $\mathbb{B}_\Sigma \to \mathrm{Set}$. (Equivalently: any monoid action $Q \times \Sigma^* \to Q$ is uniquely determined by its action on the generators $Q \times \Sigma \to Q$.)

Example: monoid actions

When M is a free monoid $M = \Sigma^*$, then any any functor $\mathcal{B}_M \cong \mathcal{F}\mathbb{B}_\Sigma \to \mathrm{Set}$ is uniquely determined by a graph homomorphism $\mathbb{B}_\Sigma \to \mathrm{Set}$. (Equivalently: any monoid action $Q \times \Sigma^* \to Q$ is uniquely determined by its action on the generators $Q \times \Sigma \to Q$.)

For example, let $\Sigma = \{ a, b \}$, and consider the functor $G : \mathcal{F}\mathbb{B}_\Sigma \to \mathrm{Set}$ defined by

$$G_* = \{ q_0, \ldots, q_5 \}$$

$$G_a = \begin{pmatrix} q_0 & q_1 & q_2 & q_3 & q_4 & q_5 \\ q_1 & q_2 & q_2 & q_4 & q_2 & q_4 \end{pmatrix} \qquad G_b = \begin{pmatrix} q_0 & q_1 & q_2 & q_3 & q_4 & q_5 \\ q_3 & q_3 & q_5 & q_3 & q_3 & q_3 \end{pmatrix}$$

Then $\pi_G : \int G \to \mathcal{F}\mathbb{B}_\Sigma$ is exactly (isomorphic to) the DFA from a few slides back.

Duality of the fibered and indexed perspectives

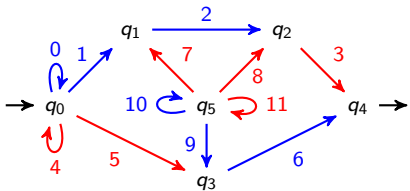
Conversely, to any discrete opfibration $p : \mathcal{D} \to \mathcal{C}$ is associated a **fiber functor** $G : \mathcal{C} \to \mathrm{Set}$ defined by taking $G_A = \{ R \in \mathcal{D} \mid R \sqsubset A \}$ and letting $G_f : G_A \to G_B$ be the function which maps any $R \sqsubset A$ to the unique $S \sqsubset B$ such that $R \Longrightarrow_f S$.

These two constructions

$$\text{discrete opfibration} \underset{\text{category of elements}}{\overset{\text{fiber functor}}{\rightleftarrows}} \text{covariant presheaf}$$

extend to an equivalence of categories between the category $\mathrm{DOpFib}(\mathcal{C})$ of discrete fibrations over $\mathcal{C}$ with commutative triangles as morphisms, and the category $[\mathcal{C}, \mathrm{Set}]$ of covariant presheaves on $\mathcal{C}$ with natural transformations as morphisms.

Running an automaton backwards

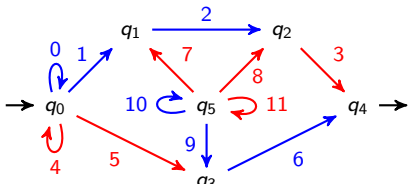


$\mathcal{A} \quad :=$

q_4

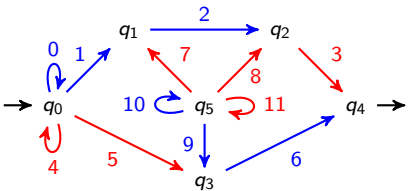
Running an automaton backwards

$\mathcal{A}$:=



$q_2 \xrightarrow{3} q_4$

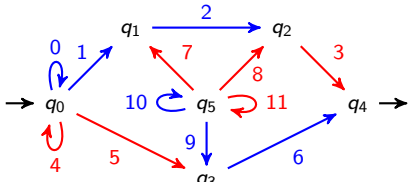
Running an automaton backwards



$\mathcal{A} \quad :=$

$q_5 \xrightarrow{8} q_2 \xrightarrow{3} q_4$

Running an automaton backwards

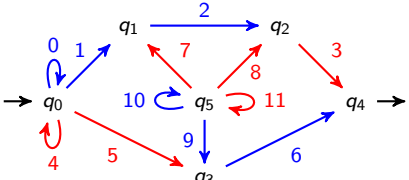


$\mathcal{A} \quad :=$

$$q_5 \xrightarrow{10} q_5 \xrightarrow{8} q_2 \xrightarrow{3} q_4$$

$$* \xrightarrow{a} * \xrightarrow{a} * \xrightarrow{b} * \xrightarrow{b} *$$

Running an automaton backwards



$\mathcal{A} \quad :=$

$$q_5 \xrightarrow{10} q_5 \xrightarrow{10} q_5 \xrightarrow{8} q_2 \xrightarrow{3} q_4$$

$$* \xrightarrow{a} * \xrightarrow{a} * \xrightarrow{b} * \xrightarrow{b} *$$

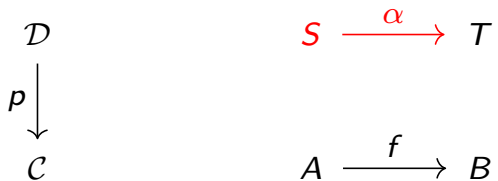
Codeterminism

Proposition: let $\phi : \mathbb{G} \to \mathbb{B}_\Sigma$ be a homomorphism of finite graphs. TFAE:

1. ϕ is the transition graph of a complete codeterministic automaton.
2. for every node $q \in \mathbb{G}$ and every loop $a : * \to * \in \mathbb{B}_\Sigma$, there is a unique node q' and unique edge $e : q' \to q$ such that $\phi(e) = a$.
3. $\mathcal{F}\phi : \mathcal{F}\mathbb{G} \to \mathcal{F}\mathbb{B}_\Sigma$ is a *discrete fibration*.

Definition

A functor $p : \mathcal{D} \to \mathcal{C}$ is a **discrete fibration** if for any object $T \sqsubset B$ of $\mathcal{D}$ and any arrow $f : A \to B$ of $\mathcal{C}$ there exists a unique object $S \sqsubset A$ and arrow $\alpha : S \Longrightarrow_f T$.



Discrete fibrations as contravariant presheaves

A **contravariant presheaf** on a category $\mathcal{C}$ is a functor $G : \mathcal{C} \to \mathrm{Set}^{\mathrm{op}}$.

Explicitly, a contravariant presheaf G consists of the following:

- a set G_A for every object A in $\mathcal{C}$;
- a function $G_f : G_B \to G_A$ for every arrow $f : A \to B$ in $\mathcal{C}$;
- such that $G_{fg} = G_f \circ G_g$ and $G_{id_A} = id_{G_A}$.

Symmetric versions of the category of elements and fiber functor constructions realize an equivalence of categories $\mathrm{DFib}(\mathcal{C}) \cong [\mathcal{C}, \mathrm{Set}^{\mathrm{op}}]^{\mathrm{op}}$.

Finite-state automata as bundles II: ULF and finitary functors

Characterizing nondeterministic finite-state automata

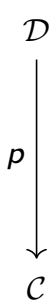
Recall that any NFA $\mathcal{A}$ over an alphabet Σ has an underlying transition graph $\phi : \mathbb{G} \to \mathbb{B}_\Sigma$ that induces an associated functor $p = \mathcal{F}\phi : \mathcal{F}\mathbb{G} \to \mathcal{F}\mathbb{B}_\Sigma$.

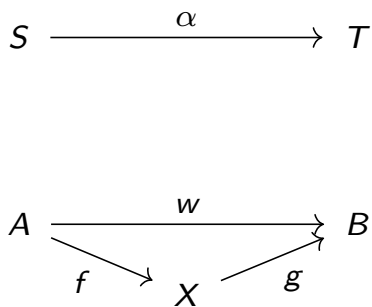
As we saw, $\mathcal{A}$ is deterministic (resp. codeterministic) iff p is a discrete opfibration (resp. discrete fibration). Either condition implies that p is faithful, i.e., unambiguous.

But what about in general? Can we characterize those functors $p : \mathcal{D} \to \mathcal{F}\mathbb{B}_\Sigma$ that are generated by the transition graph of an arbitrary (potentially ambiguous) NFA?

ULF functors (aka discrete Conduché fibrations)

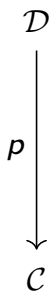
A functor $p : \mathcal{D} \to \mathcal{C}$ has **unique lifting of factorizations** if for every arrow α in $\mathcal{D}$ and pair of arrows f, g in $\mathcal{C}$ such that $p(\alpha) = fg$, there exist unique β and γ in $\mathcal{D}$ such that $\alpha = \beta\gamma$ and $p(\beta) = f$ and $p(\gamma) = g$.

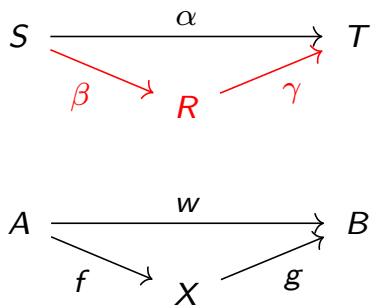




ULF functors (aka discrete Conduché fibrations)

A functor $p : \mathcal{D} \to \mathcal{C}$ has **unique lifting of factorizations** if for every arrow α in $\mathcal{D}$ and pair of arrows f, g in $\mathcal{C}$ such that $p(\alpha) = fg$, there exist unique β and γ in $\mathcal{D}$ such that $\alpha = \beta\gamma$ and $p(\beta) = f$ and $p(\gamma) = g$.





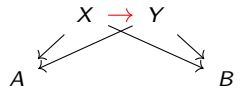
ULF $\Rightarrow$ discrete fibers

Proposition: if $p : \mathcal{D} \to \mathcal{C}$ is ULF then it has discrete fibers.

Proof: Suppose that $\alpha : R \to R'$ were a non-identity arrow such that $p(\alpha) = id_A$. Then (id_R, α) and $(\alpha, id_{R'})$ would be two distinct liftings of the factorization (id_A, id_A) of id_A, a contradiction.

(Hence ULF functors have no ϵ-transitions!)

Another equivalent view of ULF functors

Let Span be the bicategory whose objects are sets, whose 1-cells are spans $A \leftarrow X \rightarrow B$, and whose 2-cells are morphisms of spans 

.

To any functor whatsoever $p : \mathcal{D} \rightarrow \mathcal{C}$ is associated a *lax fiber functor* $G : \mathcal{C} \rightarrow \mathrm{Span}$ that sends every arrow $f : A \rightarrow B$ of $\mathcal{C}$ to the following span of sets:

$$p^{-1}(A) = \{\, S \mid S \sqsubset A \,\} \longleftarrow p^{-1}(f) = \{\, \alpha \mid \alpha : S \Longrightarrow_f T \,\} \longrightarrow p^{-1}(B) = \{\, T \mid T \sqsubset B \,\}$$

In general this fiber functor is only a lax functor, equipped with structure maps $\textsc{comp} : G_f\, G_g \Rightarrow G_{fg}$ and $\textsc{id} : id_{G_A} \Rightarrow G_{id_A}$ that are not necessarily invertible.

Proposition: p is ULF iff $\textsc{comp}$ and $\textsc{id}$ are invertible, i.e., iff G is a pseudofunctor.

ULF into free = free

Proposition (Street 1996): Let $p : \mathcal{D} \to \mathcal{C}$ be a functor into a category $\mathcal{C} = \mathcal{F}\mathbb{H}$ freely generated by some graph $\mathbb{H}$. Then p is ULF iff $\mathcal{D} = \mathcal{F}\mathbb{G}$ is free over some graph $\mathbb{G}$ and $p = \mathcal{F}\phi$ is generated by a graph homomorphism $\phi : \mathbb{G} \to \mathbb{H}$.

Proof: The direction ($\Leftarrow$) is immediate. For ($\Rightarrow$), take $\mathbb{G} = p^{-1}(\mathbb{H})$. Since the image of any arrow α in $\mathcal{D}$ uniquely decomposes as a path $p(\alpha) = e_1 \cdots e_n$ in $\mathbb{H}$, the ULF property implies that α uniquely decomposes as a composition of edges in $\mathbb{G}$, and p is generated by its restriction to the generators $\phi : \mathbb{G} \to \mathbb{H}$.

Finitary functors

We say that a functor $p : \mathcal{D} \to \mathcal{C}$ is **finitary** if the sets $p^{-1}(A)$ and $p^{-1}(f)$ are finite for every object A and arrow $f : A \to B$ of $\mathcal{C}$.

Equivalently, p is finitary iff the lax fiber functor $G : \mathcal{C} \to \mathrm{Span}$ factors via $\mathrm{FinSpan}$.

Proposition: Let $\phi : \mathbb{G} \to \mathbb{H}$ be a homomorphism into a finite graph $\mathbb{H}$. Then $\mathcal{F}\phi : \mathcal{F}\mathbb{G} \to \mathcal{F}\mathbb{H}$ is finitary iff $\mathbb{G}$ is finite.

The characterization

Corollary: Let $p : \mathcal{D} \to \mathcal{F}\mathbb{B}_\Sigma$ be a functor into the free category on a bouquet. TFAE:

1. $\mathcal{D} = \mathcal{F}\mathbb{G}$ and $p = \mathcal{F}\phi$ where $\phi : \mathbb{G} \to \mathbb{B}_\Sigma$ is the transition graph of a NFA.
2. p is finitary and ULF.

Finite-state automata as bundles III: NFAs over categories

Definition

A **NFA over a category** $\mathcal{C}$ is a tuple $\mathcal{A} = (\mathcal{Q}, p, q_0, q_f)$ consisting of

- a category $\mathcal{Q}$
- a finitary ULF functor $p : \mathcal{Q} \to \mathcal{C}$
- a pair of objects $q_0, q_f \in \mathcal{Q}$.

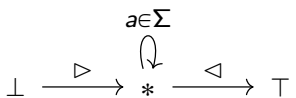
The automaton recognizes a **regular language of arrows** $\mathcal{L}_{\mathcal{A}}$ defined as the image of the homset $\mathcal{Q}(q_0, q_f)$ along p, that is,

$$\mathcal{L}_{\mathcal{A}} \stackrel{\text{def}}{=} \{\, p(\alpha) \mid \alpha : q_0 \to q_f \,\} \subseteq \mathcal{C}(A, B)$$

where $q_0 \sqsubset A$, $q_f \sqsubset B$.

Example: automata over free categories

Let $\mathbb{B}_{\Sigma}^{\triangleright\triangleleft}$ be the graph obtained from $\mathbb{B}_{\Sigma}$ by adjoining a pair of initial and final nodes:

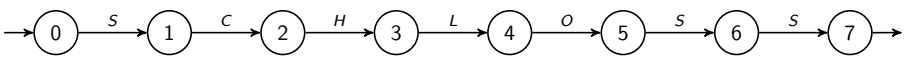


An automaton over $\mathcal{F}\mathbb{B}_{\Sigma}^{\triangleright\triangleleft}$ processes a word with explicit begin/end markers.

More generally, an automaton over the free category $\mathcal{F}\mathbb{H}$ on an arbitrary graph $\mathbb{H}$ recognizes a language of paths, which may be considered as "typed words".

Example: singleton automaton

For any $w \in \Sigma^*$ s.t. $|w| = n$, there is an $(n+1)$-state automaton recognizing $\{ w \}$:

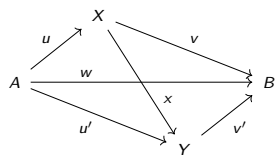


This is a special case of the following general construction...

Example: singleton automaton

Let $\mathcal{C}$ be a category and let $w : A \to B$ be an arrow of $\mathcal{C}$. We define an automaton $\mathcal{A}_w = (\mathcal{Q}, p, q_0, q_f)$ recognizing the singleton language $\{ w \} \subseteq \mathcal{C}(A, B)$ as follows:

- $\mathcal{Q} = \mathrm{Fact}_w$ is the *category of factorizations* of w.



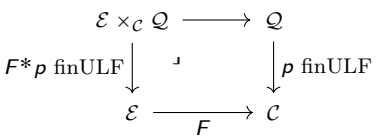
- $p = p_w : \mathrm{Fact}_w \to \mathcal{C}$ is the projection returning the middle of a factorization.
- the initial and final states are $q_0 = (id_A, w)$ and $q_f = (w, id_B)$.

Key fact: $p_w : \mathrm{Fact}_w \to \mathcal{C}$ is ULF. (But not necessarily finitary.)

If the category $\mathcal{C}$ has *finitary factorizations* then $\mathcal{A}_w$ defines a NFA.

Example: pullback automaton

Proposition: Finitary ULF functors are preserved by pullback along arbitrary functors.



(Easy way to see this: if $G : \mathcal{C} \to \mathrm{FinSpan}$ is the fiber functor of p, then $G \circ F$ is the fiber functor of F^*p.)

Corollary: Regular languages of arrows are closed under inverse image along functors, and under intersection.

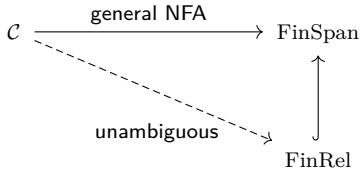
Classifying automata by their fibrational properties

We say that a categorical automaton is unambiguous/deterministic/codeterministic if p is faithful/opfibration/fibration, or equivalently, just in case its fiber functor factors as follows...

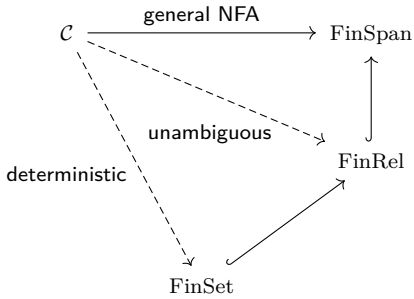
Classifying automata by their fibrational properties

$$\mathcal{C} \xrightarrow{\text{general NFA}} \mathrm{FinSpan}$$

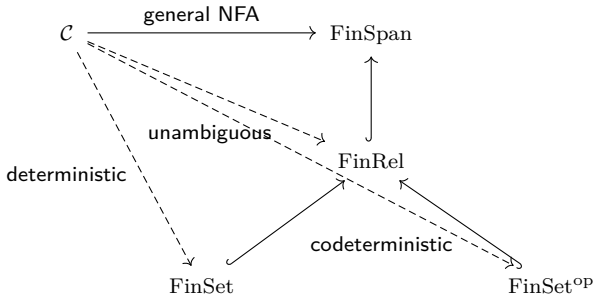
Classifying automata by their fibrational properties



Classifying automata by their fibrational properties



Classifying automata by their fibrational properties



Transducers between categories

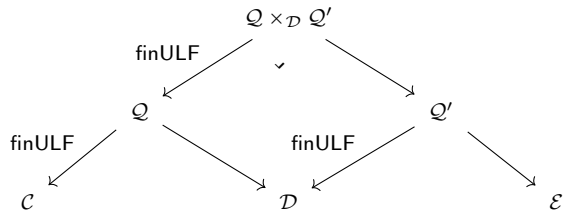
A **nondeterministic finite-state transducer** $\mathcal{M} : \mathcal{C} \nrightarrow \mathcal{D}$ is a pair $\mathcal{M} = (\mathcal{A}, O)$ consisting of an NFA $\mathcal{A} = (\mathcal{Q}, p, q_0, q_f)$ over $\mathcal{C}$ and a functor $O : \mathcal{Q} \to \mathcal{D}$. By taking the joint image of the homset $\mathcal{Q}(q_0, q_f)$, a transducer induces a relation

$$\{ (p(\alpha), O(\alpha)) \mid \alpha : q_0 \to q_f \} \subseteq \mathcal{C}(A, B) \times \mathcal{D}(X, Y)$$

where $(A, B) = (p(q_0), p(q_f))$ and $(X, Y) = (O(q_0), O(q_f))$.

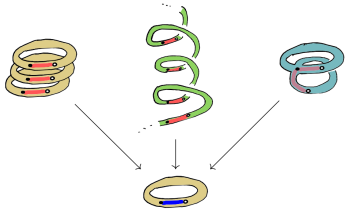
Transducers between categories

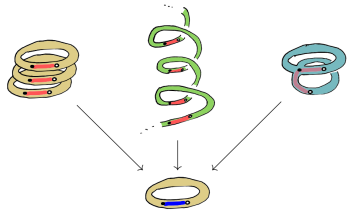
Transducers $\mathcal{M} : \mathcal{C} \multimap \mathcal{D}$ and $\mathcal{M}' : \mathcal{D} \multimap \mathcal{E}$ can be composed as follows:



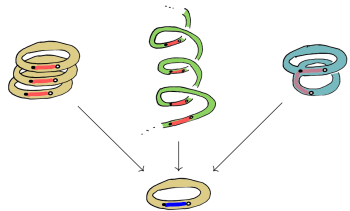
Unambiguous/deterministic/codeterministic transducers are closed under composition.

Concluding perspectives



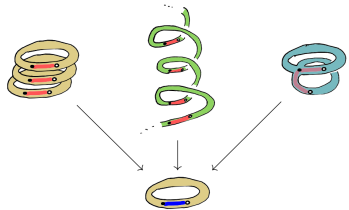


Finite-state automata may be considered as bundles of categories (without violence!), in a way that allows to treat them uniformly like other type refinement systems.



Finite-state automata may be considered as bundles of categories (without violence!), in a way that allows to treat them uniformly like other type refinement systems.

Finitary ULF functors play an important role, taking nondeterminism with potential ambiguity as a starting point. (Accounting for ϵ-transitions is work-in-progress.)



Finite-state automata may be considered as bundles of categories (without violence!), in a way that allows to treat them uniformly like other type refinement systems.

Finitary ULF functors play an important role, taking nondeterminism with potential ambiguity as a starting point. (Accounting for ϵ-transitions is work-in-progress.)

Moving from categories to operads (aka multicategories) allows to treat regular languages of trees and context-free languages of arrows, including an account of the Chomsky-Schützenberger representation theorem.

Extra slides

Traditionally (e.g. [2]) one looks at a finite-state machine as processing sequences of inputs drawn from a finite set A, the input alphabet: one then considers the free monoid A [...]. To the automaton can be associated a congruence γ of a finite index on A*. A*/γ being a finite monoid, one is then led to investigate relationships between the structure of this algebraic system and the combinatorial processing of input sequences.*

Recent research has established the possibility and the necessity of generalizing this model. For example, we are often interested in decompositions of automata. In such situations a component may receive its input from the output of some other component. This "preprocessing" imposes restrictions on the possible input sequences that need to be considered. A simple way to take into account these restrictions is to view a machine as processing input sequences that are paths in a finite directed multigraph.

D. Thérien and M. Sznajder-Glodowski (1988), "Finite categories and regular languages".

Categorical bifibrations for logic and languages

A functorial view of Hoare logic

Let $\mathcal{S}$ be the one-object category whose arrows $c : * \to *$ are sequential compositions of program commands defined over some global set of variables.

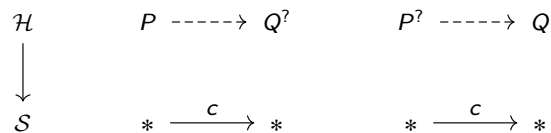
Suppose we have fixed some interpretation $[c] \subseteq S \times S$ of program commands as (potentially partial or nondeterministic) relations from input states to output states. (This interpretation should correspond to a functor $[-] : \mathcal{S} \to \mathrm{Rel}$ with $S = [*]$.)

Let $\mathcal{H}$ be the category whose objects are predicates on the global state, and whose arrows $P \to Q$ are given by commands c satisfying the Hoare triple $\{P\}c\{Q\}$, i.e., such that for all pairs of states s, s', if $P(s)$ and $(s, s') \in [c]$ then $Q(s')$.

Let $p : \mathcal{H} \to \mathcal{S}$ be the evident forgetful functor.

Two kinds of lifting problems

Given a command c and a predicate P (resp., a predicate Q), does there exists a predicate Q (resp. P) such that a $\{P\}c\{Q\}$?



In general there can be many solutions! (e.g., $Q^? = \mathit{true}$, $P^? = \mathit{false}$)

Weakest preconditions and strongest postconditions

Given a command c and a predicate Q, a *weakest precondition* is a predicate $wp(c, Q)$ such that for any predicate P, the triple $\{P\}c\{Q\}$ is valid iff $P \Rightarrow wp(c, Q)$.

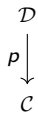
Dually, given a command c and a predicate P, a *strongest postcondition* is a predicate $sp(c, P)$ such that for any predicate Q, the triple $\{P\}c\{Q\}$ is valid iff $sp(c, P) \Rightarrow Q$.

TFAE:

1. Weakest preconditions and strongest postconditions exist for all P, c, Q.
2. $p : \mathcal{H} \to \mathcal{S}$ is a *bifibration*

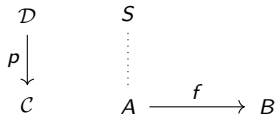
What is a bifibration?

Formally:



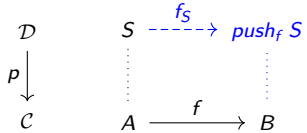
What is a bifibration?

Formally:



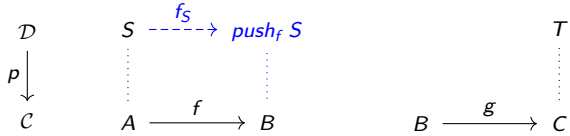
What is a bifibration?

Formally:



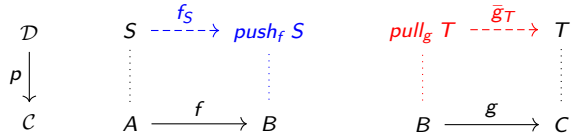
What is a bifibration?

Formally:



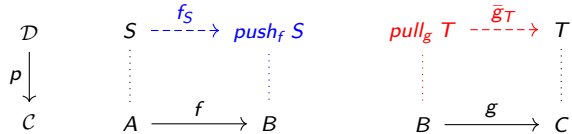
What is a bifibration?

Formally:



What is a bifibration?

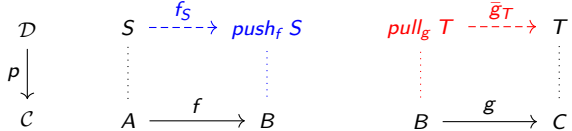
Formally:



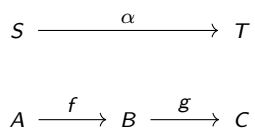
...and these liftings should be universal...

What is a bifibration?

Formally:

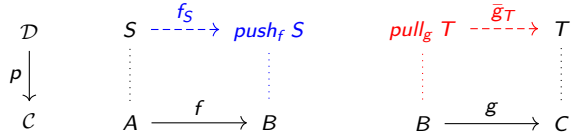


...and these liftings should be universal...



What is a bifibration?

Formally:



...and these liftings should be universal...

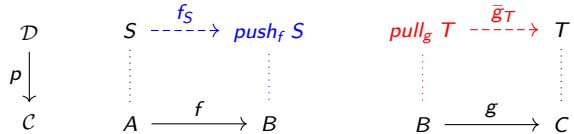
$$S \xrightarrow{f_S} push_f\ S \xrightarrow{f \setminus_g \alpha} T \qquad S \xrightarrow{\quad\alpha\quad} T$$

$$=$$

$$A \xrightarrow{f} B \xrightarrow{g} C \qquad A \xrightarrow{f} B \xrightarrow{g} C$$

What is a bifibration?

Formally:



...and these liftings should be universal...

$$S \xrightarrow{f_S} push_f\ S \xrightarrow{f \setminus_g \alpha} T \quad = \quad S \xrightarrow{\alpha} T \quad = \quad S \xrightarrow{\alpha\, f / \bar{g}} pull_g\ T \xrightarrow{\bar{g}_T} T$$

$$A \xrightarrow{f} B \xrightarrow{g} C \qquad A \xrightarrow{f} B \xrightarrow{g} C \qquad A \xrightarrow{f} B \xrightarrow{g} C$$

Definition of a bifibration in inference rules

$$\frac{S \sqsubset A \qquad f : A \to B}{push_f\ S \sqsubset B} \qquad \frac{f : A \to B \qquad T \sqsubset B}{pull_f\ T \sqsubset A}$$

$$\frac{S \underset{fg}{\Longrightarrow} T}{push_f\ S \underset{g}{\Longrightarrow} T} \qquad \frac{S' \underset{f'}{\Longrightarrow} S}{S' \underset{f'f}{\Longrightarrow} push_f\ S}$$

$$\frac{T \underset{g'}{\Longrightarrow} T'}{pull_g\ T \underset{gg'}{\Longrightarrow} T'} \qquad \frac{S \underset{fg}{\Longrightarrow} T}{S \underset{f}{\Longrightarrow} pull_g\ T}$$

(+ some equations)

Some consequences of the definitions

For all S, f, g, T of the appropriate type, we have

$$push_{fg}\, S \equiv push_g\, push_f\, S \qquad push_{id}\, S \equiv S$$

$$pull_{fg}\, T \equiv pull_f\, pull_g\, T \qquad pull_{id}\, T \equiv T$$

where "$\equiv$" denotes iso in $\mathcal{D}$ lying over an identity in $\mathcal{C}$ ("vertical isomorphism").

Some consequences of the definitions

The following subtyping rules are derivable:

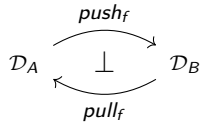
$$\frac{S \leqslant_A S'}{push_f\ S \leqslant_B push_f\ S'} \qquad \frac{T \leqslant_B T'}{pull_f\ T \leqslant_A pull_f\ T'}$$

The following judgments are interderivable:

$$S \leqslant_A pull_f\ T \quad \text{iff} \quad S \underset{f}{\Longrightarrow} T \quad \text{iff} \quad push_f\ S \leqslant_B T$$

Bifibrations as indexed adjunctions

Summarizing the last two slides, pushing and pulling along any given arrow $f : A \to B$ in the base of a bifibration[2] $p : \mathcal{D} \to \mathcal{C}$ induces an *adjunction of fiber categories*



and taking fibers lets us define a *pseudofunctor* $\mathcal{C} \to \mathrm{Adj}$ *into the category of small categories and adjunctions.* Conversely, any such pseudofunctor $G : \mathcal{C} \to \mathrm{Adj}$ induces a bifibration $\pi_G : \int G \to \mathcal{C}$ via a generalization of the category of elements construction.

[2] Technically the bifibration should be equipped with a choice of lifts, i.e., be "cloven".

Example #1: functional image and inverse image

Let SubSet be the category whose objects are pairs $(A, R \subseteq A)$, and whose arrows $(A, S) \to (B, T)$ are functions $f : A \to B$ such that $\forall a, a \in S \Rightarrow f\, a \in T$.

The projection functor $\pi : \mathrm{SubSet} \to \mathrm{Set}$ is a bifibration with

$$push_f\,(A, S) \stackrel{\mathrm{def}}{=} (B, f(S)) \qquad pull_f\,(B, T) \stackrel{\mathrm{def}}{=} (A, f^{-1}(T))$$

Example #2: existential and universal quantification along a relation

Let SubRel be the category with same objects as SubSet but whose arrows $(A, S) \to (B, T)$ are *relations* $r \subseteq A \times B$ such that $\forall ab, a \in S \wedge (a, b) \in r \Rightarrow b \in T$.

The projection functor $\pi : \mathrm{SubRel} \to \mathrm{Rel}$ is a bifibration with

$$push_r\,(A, S) \stackrel{\text{def}}{=} (B, \{\, b \mid \exists a, a \in S \wedge (a, b) \in r \,\})$$
$$pull_r\,(B, T) \stackrel{\text{def}}{=} (A, \{\, a \mid \forall b, (a, b) \in r \Rightarrow b \in T \,\})$$

Monoidal closed bifibrations

$\mathrm{SubSet} \to \mathrm{Set}$ and $\mathrm{SubRel} \to \mathrm{Rel}$ are examples of *monoidal closed bifibrations*, in the sense that the categories are monoidal closed and the projection functors strictly preserve the monoidal closed structure, in addition to being bifibrations.

Monoidal closed bifibrations

$\mathrm{SubSet} \to \mathrm{Set}$ and $\mathrm{SubRel} \to \mathrm{Rel}$ are examples of *monoidal closed bifibrations*, in the sense that the categories are monoidal closed and the projection functors strictly preserve the monoidal closed structure, in addition to being bifibrations.

Proposition: if $p : \mathcal{D} \to \mathcal{C}$ is a monoidal closed bifibration, and M is a monoid in $\mathcal{C}$, then the fiber category $\mathcal{D}_M$ is monoidal closed, with tensor and internal hom defined by

$$R \otimes_M S \stackrel{\text{def}}{=} \mathit{push}_m\,(R \otimes_{\mathcal{D}} S) \qquad R \multimap_M S \stackrel{\text{def}}{=} \mathit{pull}_{\lambda m}\,(R \multimap_{\mathcal{D}} S)$$

where $m : M \otimes_{\mathcal{C}} M \longrightarrow M$ is the monoid multiplication map and λm is its currying.