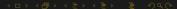
Category theory background for constraint satisfaction (Part 1)

Charlotte Aten

University of Colorado Boulder

2025 September 16



Introduction

■ These are notes on the category theory background needed to read: Maximilian Hadek, Tomáš Jakl, and Jakub Opršal. "A categorical perspective on constraint satisfaction: The wonderland of adjunctions". In: arXiv e-prints (Mar. 2025). arXiv: 2503.10353 [cs.L0]

Introduction

- PCSP in categorical terms
- Reductions as adjunctions
- Polymorphisms

Definition (Promise (decision) problem)

A promise (decision) problem over some class C of elements, called instances is a pair of subclasses $Y, N \subset C$. We refer to members of Y as YES instances and members of N as NO instances.

- We say that a promise problem is *well-defined* when Y and N are disjoint. We usually assume this.
- We may write (C, Y, N) to denote a promise problem.

Definition (Reduction)

A reduction from a promise problem (C, Y, N) to (C', Y', N') is a function $f \colon C \to C'$ such that $f(Y) \subset Y'$ and $f(N) \subset N'$.

- We will usually focus on *tractable* promise problems, which are those that have polynomial-time algorithm for deciding whether a given instance is in the YES or NO class.
- We are therefore concerned with efficient reductions, which are those that can be computed in polynomial time, as these preserve tractability.

- Given objects A and B of a category \mathscr{C} , we denote by $A \to B$ the existence of a morphism from A to B in \mathscr{C} .
- Similarly, we denote by $A \not\rightarrow B$ the absence of such a morphism.
- Note that in categorical logic we would write $A \vdash B$ rather than $A \rightarrow B$ and $A \nvdash B$ when $A \not\rightarrow B$.
- This is distinct from the material implication $A \implies B$, which is usually realized as an internal hom bifunctor.

Definition (Promise CSP)

Let A and B be objects in the category \mathscr{C} . A promise CSP for the template (A,B) is is the promise decision problem whose instances are $\mathrm{Ob}(\mathscr{C})$, whose YES instances are objects I with $I \to A$, and whose NO instances are object I with $I \not\to B$.

- We might write $\mathrm{PCSP}_{\mathscr{C}}(A,B) = (\mathrm{Ob}(\mathscr{C}),\,Y(A),N(B))$ to indicate this promise decision problem.
- We might write PCSP(A, B) as a shorthand when the category $\mathscr C$ is clear from context.

Definition (Thin category)

A thin (or posetal) category is a category $\mathscr C$ in which $|\mathscr C(A,B)|\leq 1$ for every pair of objects $A,B\in\mathscr C$.

■ Thin categories are basically preorders (partial orders without antisymmetry).

- Given a category \mathscr{C} , let $Thin(\mathscr{C})$ be the the category whose objects are those of \mathscr{C} and whose morphisms are given by setting $(Thin(\mathscr{C}))(A,B)$ to be a singleton set when $A \to B$ and setting $(Thin(\mathscr{C}))(A,B) = \varnothing$ when $A \not\to B$.
- Note that Thin(Set) is equivalent to 2, the walking arrow category.
- The category $Thin(\mathscr{C})$ is also known as the *preorder reflection* of \mathscr{C} .

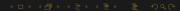
Note that for each

$$PCSP_{\mathscr{C}}(A, B) = (Ob(\mathscr{C}), Y(A), N(B))$$

we can define

$$\mathrm{PCSP}_{\mathrm{Thin}(\mathscr{C})}(A,B) = (\mathrm{Ob}(\mathrm{Thin}(\mathscr{C})),\, Y(A), N(B)).$$

■ Claim: The identity map $1_{\mathrm{Ob}(\mathscr{C})}$ is a reduction from $\mathrm{PCSP}_{\mathscr{C}}(A,B)$ to $\mathrm{PCSP}_{\mathrm{Thin}(\mathscr{C})}(A,B)$ as well as a reduction in the other direction.



- We might say that $PCSP_{\mathscr{C}}(A,B)$ is isomorphic to $PCSP_{Thin(\mathscr{C})}(A,B)$.
- Since $Thin(\mathscr{C})$ is a preorder, it looks like promise constraint isn't "really" a categorical notion.
- In practice we can't make use of this reduction since it's equivalent to being able to solve $PCSP_{\mathscr{C}}(A,B)$.

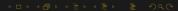
Definition (Adjunction)

Given functors $L\colon \mathscr{C}\to \mathscr{D}$ and $R\colon \mathscr{D}\to \mathscr{C}$ we say that (L,R) is an adjoint pair (with left adjoint L and right adjoint R) when

$$\mathscr{D}(L(X), Y) \cong \mathscr{C}(X, R(Y))$$

is a natural isomorphism of bifunctors $\mathscr{C}^{op} \times \mathscr{D} \to \operatorname{Set}$.

 \blacksquare A critical example is the adjunction $F\dashv U$ between the free and forgetful functors for a variety of algebras.



- Another important example is $\Sigma \dashv \Delta \dashv \Pi$ where $\Delta \colon \mathscr{C} \to \mathscr{C}^2$ is the diagonal functor, Σ is the coproduct, and Π is the product.
- More general limits and colimits may be realized as adjoints in a similar fashion.

Definition (Adjunction)

Given functors $L\colon \mathscr{C} \to \mathscr{D}$ and $R\colon \mathscr{D} \to \mathscr{C}$ we say that (L,R) is an adjoint pair when there exist natural transformations $\epsilon\colon L\circ R \to 1_\mathscr{Q}$ and $\eta\colon 1_\mathscr{C} \to R\circ L$ such that

$$1_L = \epsilon 1_L \circ 1_L \eta$$

and

$$1_R = 1_R \epsilon \circ \eta 1_R.$$

We call ϵ and η the *counit* and *unit* of the adjunction, respectively.

■ We can obtain reductions from adjunctions.

Lemma

Whenever $L\colon \mathscr{C} \to \mathscr{D}$ is a functor and $L\dashv R$ we have that L is a reduction from $\operatorname{PCSP}_{\mathscr{C}}(A,B)$ to $\operatorname{PCSP}_{\mathscr{D}}(A',B')$ if and only if $A\to R(A')$ and $R(B')\to B$.

Proof.

Suppose L is a reduction. Since $A \to A$ we have that A is a YES instance for $\mathrm{PCSP}_{\mathscr{C}}(A,B)$. This means that L(A) is a YES instance for $\mathrm{PCSP}_{\mathscr{D}}(A',B')$. It follows that $L(A) \to A'$, which implies that $A \to R(A')$.

Since $\epsilon\colon L\circ R\to 1_{\mathscr D}$ we have that $(L\circ R)(B')\to B'.$ This means that L(R(B')) is not a NO instance for $\mathrm{PCSP}_{\mathscr D}(A',B').$ It follows that R(B') is not a NO instance for $\mathrm{PCSP}_{\mathscr C}(A,B).$ That is, $R(B')\to B.$

Proof (Cont.)

Now suppose that $A \to R(A')$ and $R(B') \to B$ but we don't know that L is a reduction.

Given a YES instance I of $\mathrm{PCSP}_\mathscr{C}(A,B)$ we have $I\to A$. Since $A\to R(A')$ this implies that $I\to R(A')$. Since $L\vdash R$ we have that $L(I)\to A'$, so L(I) is a YES instance of $\mathrm{PCSP}_\mathscr{D}(A',B')$. Given a NO instance I of $\mathrm{PCSP}_\mathscr{C}(A,B)$ we have $I\not\to B$. If L failed to preserve NO instances then we would have $L(I)\to B'$ for some such I, which implies that $I\to R(B')$ and hence $I\to B$, a contradiction.

- There is a remark in the paper that the CSP literature uses the notion of "thin adjunction", which means that $L(X) \to Y$ if and only if $X \to R(Y)$.
- This appears to just be the usual notion of adjunction between $\mathrm{Thin}(\mathscr{C})$ and $\mathrm{Thin}(\mathscr{D})$, as opposed to an adjunction between \mathscr{C} and \mathscr{D} .

Polymorphisms

Definition (Polymorphism of an object)

Given an object A in a category $\mathscr C$ with finite products and some $n\in\mathbb N$, an n-ary polymorphism of A is a $\mathscr C$ -morphism $A^n\to A$.

Definition (Polymorphism of a template)

Given objects A and B in a category $\mathscr C$ with finite products and some $n\in\mathbb N$, an n-ary polymorphism of (A,B) is a $\mathscr C$ -morphism $A^n\to B.$

Polymorphisms

- Each template (A,B) in a category $\mathscr C$ has a corresponding polymorphism minion $\operatorname{Pol}(A,B)\colon \operatorname{Fin} \to \operatorname{Fin}$ given by $n \mapsto \mathscr C(A^n,B)$.
- Efficient reductions between PCSPs come from *minion* homomorphisms, which are natural transformations between minions.

Polymorphisms

- At the bottom of page 3 it is claimed that polymorphisms $A^n \to B$ do not form an algebra, but it seems like they do under generalized composition, as long as one includes the projections $A^m \to A$ and $B^m \to B$.
- It is similarly claimed that $\operatorname{Pol}(A,B)$ is not a monad (although I'm not 100% sure on what category), but that $\operatorname{Pol}(A)$ is. It looks to me like both $\operatorname{Pol}(A)$ and $\operatorname{Pol}(A,B)$ are multiply-sorted algebraic structures which correspond to monads.