Linear Logic and Co-Design Notebook

Marius Furter

March 16, 2021

Contents

1 Categories in Linear Logic **1**
1.1 The Category of Derivations . . . 1
1.2 The Category of Propositions . . . 3
1.3 The Relation between Der and Prop . . . 3

2 Linear Logic and Co-Design **5**
2.1 Basic Definitions . . . 5
2.2 Sites to Structure . . . 7
2.3 Currying Queries . . . 8
2.4 Bool-Posets . . . 10
2.5 Queries in $\mathbf{Pos}_{\mathbf{Bool}}$ . . . 12
2.5.1 Generalized Querying . . . 12
2.5.2 The Generalized Query Functor . . . 13
2.6 Queries as Universal Objects in $\mathbf{Pos}_{\mathbf{Bool}}$ . . . 13
2.6.1 Describing Other Queries . . . 14
2.7 $\mathcal{V}$-Posets . . . 15

1 Categories in Linear Logic

1.1 The Category of Derivations

Given a sequent calculus S, we form the category of derivations in S, denoted $\mathbf{Der}_S$ as follows. The objects are well-formed sequents

$$\Gamma, A \vdash B \qquad \Gamma \vdash A \multimap B \qquad \dots$$

and the morphisms are valid derivations based on the rules of S, for example,

$$\frac{\Gamma, A \vdash B}{\Gamma \vdash A \multimap B}\ (\multimap \text{Right})$$

Composition is given by the composition of derivations.

$$\begin{array}{c} \dfrac{\Gamma \vdash B}{\Gamma, A \vdash B}\ (\text{weakening Left}) \\[2ex] \dfrac{\Gamma, A \vdash B}{\Gamma \vdash A \multimap B}\ (\multimap \text{Right}) \end{array} \quad \rightsquigarrow \quad \dfrac{\dfrac{\Gamma \vdash B}{\Gamma, A \vdash B}\ (\text{weakening Left})}{\Gamma \vdash A \multimap B}\ (\multimap \text{Right})$$

It is clearly associative and has the "do nothing" sequent

$$\frac{\Gamma \vdash \Delta}{\Gamma \vdash \Delta}\ (\text{identity})$$

as identity.

This all works fine for unary rules. To accommodate rules of higher arity, we need to allow multiple sources for our morphisms. We could then think of $\mathbf{Der}_S$ as a multicategory (colored operad) or, alternatively, replace objects by lists of objects and introduce a monoidal product $\boxplus$ that allows us to concatenate these lists. The unit for this monoidal product is the empty list $[\,]$. We shall write lists of objects $[A, B, C]$ as $A \boxplus B \boxplus C$, that is we omit the brackets for lists of length one. For now, due to a current lack a familiarity with multicategories, we shall regard $\mathbf{Der}_S$ as a monoidal category. In this setting, rules of higher arity become morphisms from lists of length greater 1 to lists of length 1. For example, the cut rule

$$\frac{\Gamma, A \vdash B \qquad \Delta, B \vdash C}{\Gamma, \Delta, A \vdash C}\ (\text{Cut})$$

is a morphism $(\Gamma, A \vdash B) \boxplus (\Delta, B \vdash C) \to \Gamma, \Delta, A \vdash C$. The axiom

$$\frac{}{A \vdash A}\ (\text{Axiom})$$

is a morphism $[\,] \to A \vdash A$.

We may now view the category $\mathbf{Der}_S$ as being generated by the rules of the sequent calculus S: We fix some set of propositions $\Gamma, \Delta, \Lambda, \ldots$ and consider all sequents that can be formed from these. We then form all finite lists of such sequents and add morphisms according to the rules of S. Finally $\mathbf{Der}_S$ is obtained by closing this structure under composition.

Question 1.1. *Read about operads / multicategories in Leinster's book Chapter 2. Think about the algebras of* $\mathbf{Der}_S$*.*

1.2 The Category of Propositions

The category of proposition $\mathbf{Prop}_S$ has a objects propositions $\Gamma, \Delta, \Lambda, \ldots$ and as morphisms proofs of sequents $\Gamma \vdash \Delta$. The compostion of morphisms is given by the cut rule

$$\frac{A \vdash B \qquad B \vdash C}{A \vdash C} \text{(Cut)}$$

which turn two proofs into one. We note that cut rule depends on the sequent calculus. For example, in intuitionistic linear logic it is

$$\frac{\Gamma \vdash A \qquad \gamma_1, A, \gamma_2 \vdash B}{\gamma_1, \Gamma, \gamma_2 \vdash B} \text{(Cut)}$$

Our basic rule is obtained with empty contexts γ_1, γ_2.

1.3 The Relation between Der and Prop

To start with suppose that our calculus S consists only of unary rules. In this case $\mathbf{Der}_S$ is simply a category with a special object for the empty sequent $[\,]$. We observe that the arrows of $\mathbf{Prop}_S$ are special derivations: An arrow $\Gamma \to \Delta$ in $\mathbf{Prop}_S$ corresponds to an arrow in $\mathbf{Der}_S$ of the form $[\,] \to \Gamma \vdash \Delta$. Therefore $\mathbf{Prop}_S(\Gamma, \Delta) \cong \mathbf{Der}_S([\,], \Gamma \vdash \Delta)$. Moreover, each element of $\mathbf{Der}_S(\Gamma \vdash \Delta, A \vdash B)$ gives a function $\mathbf{Prop}_S(\Gamma, \Delta) \to \mathbf{Prop}_S(A, B)$, obtained by composing derivations. This yields a functor $F : \mathbf{Der}_S \to \mathbf{Set}$ mapping objects $(\Gamma \vdash \Delta) \mapsto \mathbf{Prop}_S(\Gamma, \Delta)$ and derivations $\Gamma, \Delta \to A, B$ to the induced function $\mathbf{Prop}_S(\Gamma, \Delta) \to \mathbf{Prop}_S(A, B)$. Indeed, functoriality is immediate.

We extend the construction above to the case where S has rules of arbitrary arity. In this case the objects in $\mathbf{Der}_S$ are lists of sequents. Because $\boxplus^n[\,] \cong [\,]$ we still have the correspondence $\mathbf{Prop}_S(\Gamma, \Delta) \cong \mathbf{Der}_S([\,], \Gamma \vdash \Delta)$. Next we note that if we have an arrow $d : A_1 \boxplus \ldots \boxplus A_n \to B_1 \boxplus \ldots \boxplus B_m$ then $n \geq m$ because the rules of S are many to one. Moreover, we can regard d as consisting of m parallel derivations $d_i : A_{i_1} \boxplus \ldots \boxplus A_{i_r} \to B_i$. Writing $A_{i_j} = \Gamma_{i_j} \vdash \Delta_{i_j}$ and $B_i = \Psi_i \vdash \Omega_i$, we see that each d_i induces a function

$$\mathbf{Prop}_S(\Gamma_{i_1}, \Delta_{i_1}) \times \ldots \times \mathbf{Prop}_S(\Gamma_{i_r}, \Delta_{i_r}) \to \mathbf{Prop}_S(\Psi_i, \Omega_i).$$

Hence d induces a function

$$\prod_i \mathbf{Prop}_S(\Gamma_{i_1}, \Delta_{i_1}) \times \ldots \times \mathbf{Prop}_S(\Gamma_{i_r}, \Delta_{i_r}) \to \prod_i \mathbf{Prop}_S(\Psi_i, \Omega_i).$$

We now face the problem that the decomposition of the A_i is not the same for every derivation with the same source and target as d. However, writing $A_j = \Gamma_j \vdash \Delta_j$, we have a natural iso

$$\prod_i \mathbf{Prop}_S(\Gamma_{i_1}, \Delta_{i_1}) \times \ldots \times \mathbf{Prop}_S(\Gamma_{i_r}, \Delta_{i_r}) \cong \prod_{j=1}^{n} \mathbf{Prop}_S(\Gamma_j, \Delta_j)$$

by reordering the big product. This suggests that we assign a list of sequents $(\Gamma_1 \vdash \Delta_1) \boxplus \ldots \boxplus (\Gamma_n \vdash \Delta_n)$ to the product $\prod_{j=1}^{n} \mathbf{Prop}_S(\Gamma_j, \Delta_j)$. A morphism

$$f : (\Gamma_1 \vdash \Delta_1) \boxplus \ldots \boxplus (\Gamma_n \vdash \Delta_n) \to (\Psi_1 \vdash \Omega_1) \boxplus \ldots \boxplus (\Psi_m \vdash \Omega_m)$$

is then assigned to the function

$$\tilde{f} : \prod_{j=1}^{n} \mathbf{Prop}_S(\Gamma_j, \Delta_j) \to \prod_i \mathbf{Prop}_S(\Psi_i, \Omega_i)$$

which it induces by the procedure described above, possibly composing with the reordering isomorphism.

Explicitly, we can describe $\tilde{f}$ as follows: Given a tuple of proofs of sequents

$$(\Gamma_1 \vdash \Delta_1, \ldots, \Gamma_n \vdash \Delta_n),$$

regard them as a derivation

$$[\,] \to (\Gamma_1 \vdash \Delta_1) \boxplus \ldots \boxplus (\Gamma_n \vdash \Delta_n)$$

and compose with f to get a derivation

$$[\,] \to (\Psi_1 \vdash \Omega_1) \boxplus \ldots \boxplus (\Psi_m \vdash \Omega_m)$$

which corresponds to a tuple of proofs of sequents

$$(\Psi_1 \vdash \Omega_1, \ldots, \Psi_m \vdash \Omega_m).$$

The functoriality of this assignment is again immediate. Hence, we also get a functor $F : \mathbf{Der}_S \to \mathbf{Set}$ in the case of a general sequent calculus S. Furthermore, F maps monoidal products in $\mathbf{Der}_S$ to cartesian products in $\mathbf{Set}$. We have $F([\ ]) = \{*\}$ by definition, which aligns with the intuition that there is exactly one trivial proof of the empty sequent. Therefore F is a strict monoidal functor into $(\mathbf{Set}, \times, \{*\})$.

Question 1.2. *Regarding* $\mathbf{Der}_S$ *as an operad, can we describe the above as an operad algebra?*

We conclude by observing that for $A_i = \Gamma_i \vdash \Delta_i$ we have

$$\mathbf{Der}_S([\,], A_1 \boxplus \ldots \boxplus A_n) \cong \mathbf{Prop}_S(\Gamma_1, \Delta_1) \times \ldots \times \mathbf{Prop}_S(\Gamma_n, \Delta_n) = F(A_1 \boxplus \ldots \boxplus A_n)$$

so in fact F is represented by $[\,]$. Furthermore, we have defined the way F acts on arrows in terms of this isomorphism, so the isomorphism is natural. Hence $F \cong \mathrm{Hom}([\,], -)$. By Yoneda we have that for any element $x \in F(A)$ there is a unique morphism $f : [\,] \to A$ such that $F(f)(*) = x$. This is saying that proofs correspond precisely to derivations starting from $[\,]$.

2 Linear Logic and Co-Design

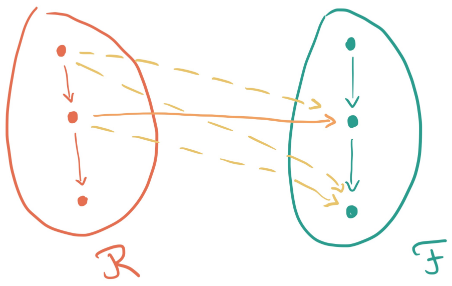
2.1 Basic Definitions

Let $\mathcal{R}$ and $\mathcal{F}$ be posets. $\mathcal{R}$ represents resources and $\mathcal{F}$ represents functionalities. For $a, b \in \mathcal{R}$, we interpret $a \leq b$ as meaning "if I have a, then I also have b". For example, $2\$ \leq 1\$$. This is the opposite convention that is currently in use, but better suits the Linear Logic interpretation. To avoid confusion, we write $a \to b$, or $a \vdash b$, instead of $a \leq b$. This also fits nicely with the fact that we will be regarding the posets as categories.

In Co-Design we consider what functionalities we can obtain from given resources.

Definition 2.1 (Feasibility Relation)**.** Let $\mathcal{R}$, $\mathcal{F}$ be posets. A *feasibility relation* $\mathcal{R} \nrightarrow \mathcal{F}$ is a monotone map $\mathcal{R}^{\mathrm{op}} \times \mathcal{F} \to \mathbf{Bool}$, where $\mathbf{Bool}$ denotes the poset generated by `false` $\to$ `true`. If we regard posets as $\mathbf{Bool}$-categories, then feasibility relations are just $\mathbf{Bool}$-profunctors.

Remark 2.1. We can draw a feasibility relation $\mathcal{R} \nrightarrow \mathcal{F}$ as an internal diagram:



The orange and yellow arrows mean that a given pair is mapped to `true`. Absence of an arrow means the pair is mapped top `false`. Additionally, we may think of a feasibility relation as being generated by certain assignments. The solid orange arrow induces the dashed yellow arrows by composition with the arrows internal to both $\mathcal{R}$ and $\mathcal{F}$.

We can now define a category in which feasibility relations live.

Definition 2.2. The category **DP** of co-design problems has posets as objects and feasibility relations as morphisms. Composition is given by profunctor compostition. Explicitly, if we have feasibility relations $\Phi : \mathcal{X} \nrightarrow \mathcal{Y}$ and $\Psi : \mathcal{Y} \nrightarrow \mathcal{Z}$, then

$$\Phi \fatsemi \Psi(x, z) = \bigvee_{y \in \mathcal{Y}} (\Phi(x, y) \wedge \Psi(y, z)).$$

Every feasibility relation $\mathcal{R} \overset{\Phi}{\nrightarrow} \mathcal{F}$ naturally gives rise to two dual optimization problems. If we fix an $f \in \mathcal{F}$, then we can ask what the minimal set of resources $r \in \mathcal{R}$ are, such that $\Phi(r, f) = \texttt{true}$, that is (r, f) is feasible. In formulae:

$$\begin{aligned} f_{\max} &= \mathrm{Max}\{r \in \mathcal{R} \mid \Phi(r, f) = \texttt{true}\} \\ &= \{r \in \mathcal{R} \mid \Phi(r, f) = \texttt{true} \text{ and } \forall r' : \Phi(r', f) = \texttt{true} \wedge r \to r' \Rightarrow r = r'\}. \end{aligned}$$

Observe that the minimality condition implies that this set is an antichain in $\mathcal{R}$. Hence we get a function $h : \mathcal{F} \to \mathsf{A}\mathcal{R}$, $f \mapsto f_{\max}$ where $\mathsf{A}\mathcal{R}$ denotes the set of antichains of $\mathcal{R}$. We can put a partial ordering on $\mathsf{A}\mathcal{R}$ by saying $A \leq B$ iff $\downarrow A \subseteq \downarrow B$, where $\downarrow$ denotes the lower closure operator. In our convention the lower closure of a set is all of the resources / functionalities that are 'more expensive' than items in the set.

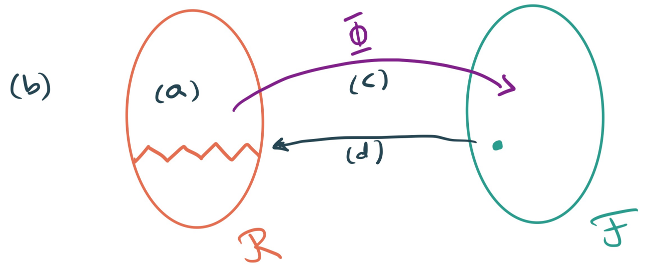
Lemma 2.1. *If all ascending chains in $\mathcal{R}$ are finite, then the map $h : \mathcal{F} \to \mathsf{A}\mathcal{R}$ is monotone. That is, if $f \to g$, then $\downarrow f_{max} \subseteq \downarrow g_{max}$.*

Proof. Let $f \to g$, and $x \in \downarrow f_{\max}$. This means $x \to r$ for some $r \in f_{\max}$. Because $\Phi(r, f) = \texttt{true}$ also $\Phi(x, g) = \texttt{true}$ using monotonicity in both arguments. Consider $\{s \in \mathcal{R} \mid x \to s \text{ and } \Phi(s, g) = \texttt{true}\}$. This set is non-empty because it contains x itself. Moreover it has to contain a maximal element, otherwise we could build an infinite ascending chain. This maximum m will be an element of $g_{\max}$, hence we have an arrow $x \to m$, which means $x \in \downarrow g_{\max}$ ⯑

Warning! The map h does not have to monotone in general (even if one uses upper closure or the opposite orderings). The ascending chain condition assures that one does not have arbitrarily cheap resources. To see that it is necessary consider $\mathcal{R} = (\mathbb{Z}, \leq)$, $\mathcal{F} = a \to b$ with feasibility relation $\Psi(r, f) =$ `true` iff $f = b$ or $r \leq 0$. Then $a_{\max} = \text{Max}\{n \in \mathbb{Z} | \Psi(r, a) = \texttt{true}\} = 0$ and $b_{\max} = \text{Max}\{n \in \mathbb{Z} | \Psi(r, b) = \texttt{true}\} = \text{Max } \mathbb{Z} = \emptyset$. Hence $\downarrow a_{\max} = \mathbb{Z}_{\leq 0} \not\subseteq \emptyset = \downarrow b_{\max}$.

2.2 Sites to Structure

The main goal of this project is to add some of the structure of Linear Logic to the co-design framework. In principle, there are several places in the theory where one could consider implementing these concepts. Here we describe four possibilities and give some thoughts on how they may be related.



One can imagine having Linear Logic (LL) structure:

(a) internal to the resource / functionality posets. This would involve restricting the objects of **DP** to posets that can be viewed as models for LL.

(b) between objects of **DP**. This would involve turning **DP** into a model of LL.

(c) within a feasibility relation. This might involve looking at monotone maps $\mathcal{R}^{\text{op}} \times \mathcal{F} \to \mathcal{V}$, where $\mathcal{V}$ is a model for LL.

(d) on the queries.

Having (b) might give (a) as well, since we can regard a poset $\mathcal{P}$ as the collage of trivial feasibility relations between singletons: If $a \to b$, then this may be seen as a feasibility relation $\Phi : \{a\} \nrightarrow \{b\}$ with $\Phi(a, b) = \texttt{true}$. If we collage these together we can reconstruct $\mathcal{P}$. For this to work we would have to describe a multi-object collage operation, but this seems possible.

The category **DP** is compact closed, which means it forms a degenerate *-autonomous category. Hence it is already a model for classical LL. However, the 4 classical LL connectives become pairwise identified. Thus the task would be to modify **DP** in such a way, that we get distinct connectives.

Next, changing the enriching category as in (c) may also affect (b). Moreover, it will alter what type of queries are possible because there would then be more states to optimize over. On the other hand, it may be possible to complexify the types of queries that can be performed, while keeping the standard boolean feasibility relation.

2.3 Currying Queries

There is an adjunction between the exponential and the cartesian product in **Cat**:

$$\mathrm{Hom}(\mathcal{C} \times \mathcal{D}, \mathcal{E}) \cong \mathrm{Hom}(\mathcal{C}, \mathcal{E}^{\mathcal{D}}).$$

If $\mathcal{P}$ and $\mathcal{Q}$ are posets, then so is the functor category $\mathcal{Q}^{\mathcal{P}}$: If $g, h : \mathcal{P} \to \mathcal{Q}$ are functors (i.e monotone maps), there is at most one natural transformation $\alpha : g \Rightarrow h$, since there is at most one morphism $\alpha_p : g(p) \to h(p)$ for each $p \in \mathcal{P}$. Therefore, the adjunction restricts to the subcategory **Pos** of posets.

Let $\mathcal{R}, \mathcal{F}$ be posets, and let $\mathcal{R} \overset{\Phi}{\nrightarrow} \mathcal{F}$ be a feasibility relation. Denote the lower sets of $\mathcal{R}$ by $\mathsf{L}\mathcal{R}$, where $A \leq B$ iff $A \subseteq B$. We claim that $\mathbf{Bool}^{\mathcal{R}^{\mathrm{op}}} \cong \mathsf{L}\mathcal{R}$. A monotone map $h : \mathcal{R}^{\mathrm{op}} \to \mathbf{Bool}$ is the same as a monotone map $h' : \mathcal{R} \to \mathbf{Bool}^{\mathrm{op}}$. The pre-image $L_h = h'^{-1}(\texttt{true})$ is now a lower set: If $y \in L$ and $x \to y$, then $h'(x) \to h'(y) = \texttt{true}$, so $h'(x) = \texttt{true}$ since $\texttt{true}$ is the bottom in $\mathbf{Bool}^{\mathrm{op}}$. Moreover, if $g, h : \mathcal{R}^{\mathrm{op}} \to \mathbf{Bool}$ and $g \Rightarrow h$, then $g(r) \to h(r)$ in $\mathbf{Bool}^{\mathrm{op}}$ for all $r \in \mathcal{R}$. Hence, if $h(r) = \texttt{true}$, then also $g(r) = \texttt{true}$, which shows $L_h \subseteq L_g$, i.e. $L_h \leq L_g$ in the lower set order. Conversely, for each lower set $L \subseteq \mathcal{R}$, we can construct an monotone map $l : \mathcal{R}^{\mathrm{op}} \to \mathbf{Bool}$ by setting $l(x) = \texttt{true}$ iff $l(x) \in L$. The result is monotone: $y \to x$ in $\mathcal{R}^{\mathrm{op}}$ means $x \to y$ in $\mathcal{R}$. If $l(x) = \texttt{false}$, then $l(y) = \texttt{false}$, because $x \notin L \Rightarrow y \notin L$ by the contrapositive of the lower set property. Similarly, if $l(y) = \texttt{true}$, then also $l(x) = \texttt{true}$. It is immediate that both operations described are inverse. Therefore, $\mathbf{Bool}^{\mathcal{R}^{\text{op}}} \cong \mathsf{L}\mathcal{R}$.

Furthermore, if $\mathcal{R}$ has no infinite ascending chains, then each lower set corresponds uniquely to an antichain by taking its maximal elements. Since we order antichains by the order induced by their lower closures, we have $\mathsf{L}\mathcal{R} \cong \mathsf{A}\mathcal{R}$.

In summary, if $\mathcal{R}$ has no infinite ascending chains, we have the following series of isomorphisms:

$$\begin{aligned}
\mathrm{Hom}(\mathcal{R}^{\text{op}} \times \mathcal{F}, \mathbf{Bool}) &\cong \mathrm{Hom}(\mathcal{F} \times \mathcal{R}^{\text{op}}, \mathbf{Bool}) && (1)\\
&\cong \mathrm{Hom}(\mathcal{F}, \mathbf{Bool}^{\mathcal{R}^{\text{op}}}) && (2)\\
&\cong \mathrm{Hom}(\mathcal{F}, \mathsf{L}\mathcal{R}) && (3)\\
&\cong \mathrm{Hom}(\mathcal{F}, \mathsf{A}\mathcal{R}) && (4)
\end{aligned}$$

It follows that our feasibility relation Φ corresponds uniquely to a monotone function $h : \mathcal{F} \to \mathsf{A}\mathcal{R}$. To see what this function does, we follow the isos above. (1) sends Φ to its partial evaluations $f \mapsto \Phi(-, f)$. For each $f \in \mathcal{F}$, the function $\Phi(-, f)$ corresponds to the lower set

$$\Phi(-, f)^{-1}(\texttt{true}) = \{r \in \mathcal{R} \mid \Phi(r, f) = \texttt{true}\}.$$

Taking maximal elements of this gives an antichain of resources, which are the cheapest resources which are feasible. But this is precisely the query $f_{\max}$ described in section 2.1. Hence we have show that querying correspond to equivalence described by the isomorphisms above, provided $\mathcal{R}$ fulfils the ascending chain condition.

We can repeat the above by swapping the roles of $\mathcal{F}$ and $\mathcal{R}$. Observe that $\mathbf{Bool}^{\mathcal{F}} \cong (\mathsf{L}\mathcal{F})^{\text{op}}$ as follows: If $h : \mathcal{F} \to \mathbf{Bool}$, then $h^{-1}(\texttt{false})$ is a lower set. If $h(y) = \texttt{false}$ and $x \to y$, then by monotonicity $h(x) \to h(y)$, so $x \in h^{-1}(\texttt{false})$. Moreover, if $g, h : \mathcal{F} \to \mathbf{Bool}$ with $g \Rightarrow h$, then for each $r \in \mathcal{F}$, we have $g(r) \to h(r)$. Hence, if $h(r) = \texttt{false}$, then $g(r) = \texttt{false}$, so $h^{-1}(\texttt{false}) \subseteq g^{-1}(\texttt{false})$. This is the opposite of the usual ordering on $\mathsf{L}\mathcal{F}$. The inverse operation is obtained by setting $l(x) = \texttt{false}$ iff x is in the lower set. Thus $\mathbf{Bool}^{\mathcal{F}} \cong (\mathsf{L}\mathcal{F})^{\text{op}}$, as claimed.

We must now change up our isos slightly:. Assuming $\mathcal{F}$ has no infinite ascending chains:

$$\begin{aligned}
\mathrm{Hom}(\mathcal{R}^{\mathrm{op}} \times \mathcal{F}, \mathbf{Bool}) &\cong \mathrm{Hom}(\mathcal{R}^{\mathrm{op}}, \mathbf{Bool}^{\mathcal{F}}) && (5)\\
&\cong \mathrm{Hom}(\mathcal{R}^{\mathrm{op}}, (\mathsf{L}\mathcal{F})^{\mathrm{op}}) && (6)\\
&\cong \mathrm{Hom}(\mathcal{R}, \mathsf{L}\mathcal{F}) && (7)\\
&\cong \mathrm{Hom}(\mathcal{R}, \mathsf{A}\mathcal{F}) && (8)
\end{aligned}$$

The composite map sends $r \in \mathcal{R}$ to the antichain of the 'cheapest' functionalities in $\mathcal{F}$ that are still infeasible.

By swapping out lower sets for upper sets in the above, we should be able to get two more querying operation, provided the target poset fulfils a descending chain condition: One sends $f \in \mathcal{F}$ to the antichain of the most 'expensive' resources that still make the functionality infeasible. The other sends a resource to the most 'expensive' antichain of functionalities that makes the pairs feasible.

2.4 Bool-Posets

We have seen above that there is a tight relation between maps into **Bool** and lower sets, upper sets, and antichains. This leads up to consider such maps as the objects of a category.

Definition 2.3. A ***Bool**-poset* is just a monotone map $\mathcal{P} \to \mathbf{Bool}$ from a poset $\mathcal{P}$ into **Bool**.

Next we define feasibility relations between **Bool**-posets.

Definition 2.4. A feasibility relation between **Bool**-posets $\rho : \mathcal{R} \to \mathbf{Bool}$ and $\varphi : \mathcal{F} \to \mathbf{Bool}$, is a feasibility realtion $\mathcal{F} \overset{\Phi}{\nrightarrow} \mathcal{R}$, where for all $r \in \mathcal{R}$ and $f \in \mathcal{F}$,

$$\Phi(r, f) = \texttt{true} \Rightarrow \rho(r) \leq \varphi(f).$$

We can express this condition in the following diagram which commutes up to a natural transformation:

$$\begin{tikzcd}
\mathcal{R} \arrow[d, "|" marking] \arrow[rr, bend left] \arrow[drr, bend left] \arrow[dr, phantom, "\Swarrow"] & & \\
\mathcal{F} \arrow[rr] & & \mathbf{Bool}
\end{tikzcd}$$

Warning! Requiring the condition $\varphi(f) \leq \rho(r)$ leads to the restriction that resources above a certain threshold must be mapped bellow a certain functionality threshold, which is the opposite of what we want.

Lemma 2.2. **Bool**-*posets and their feasibility relationships form a category, denoted* $\mathbf{Pos}_{\mathbf{Bool}}$. *Composition is given by the usual composition of feasibility relations.*

Proof. If $\Phi : X \nrightarrow Y$, $\Psi : Y \nrightarrow Z$ are feasibility relations between **Bool**-posets $\xi : X \to \mathbf{Bool}$, $\upsilon : Y \to \mathbf{Bool}$ and $\zeta : Z \to \mathbf{Bool}$, then if $\Phi \fatsemi \Psi(x, z)$, the exists $y \in Y$ such that $\Phi(x, y)$ and $\Psi(y, z)$. Hence $\xi(x) \leq \upsilon(y)$ and $\upsilon(y) \leq \zeta(z)$, whence $\xi(x) \leq \zeta(z)$. This shows that composition is well-defined. Associativity is inherited from the composition of feasibility relations. Finally, there are identities given by the unit feasibility relation $\mathrm{id} : X \nrightarrow X$, with $\mathrm{id}(x, y) = \texttt{true}$ iff $x \to y \in X$. These satisfy the condition for being a morphism in $\mathbf{Pos}_{\mathbf{Bool}}$: If $\mathrm{id}(x, y) = \texttt{true}$, then $x \to y$, so $\xi(x) \leq \xi(y)$. Moreover, they are the identities for usual feasibility relations (see Fong / Spivak Lemma 4.19), so they also act that way here. ⧠

The category $\mathbf{Pos}_{\mathbf{Bool}}$ has some interesting features, which we explore below:

Feasibility Relations are Objects Since a feasibility relation $\Phi : \mathcal{R}^{\mathrm{op}} \times \mathcal{F} \to \mathbf{Bool}$ is just a monotone map into **Bool**, it is an object of $\mathbf{Pos}_{\mathbf{Bool}}$. We can thus build feasibility relations between feasibility relations: If $\Psi : \mathcal{R}'^{\mathrm{op}} \times \mathcal{F}' \to \mathbf{Bool}$ is another feasibility relation, a second order feasibility relation $\Phi \overset{\Delta}{\nrightarrow} \Psi$ would be a monotone map

$$\Delta : (\mathcal{R}^{\mathrm{op}} \times \mathcal{F})^{\mathrm{op}} \times \mathcal{R}'^{\mathrm{op}} \times \mathcal{F}' \to \mathbf{Bool}$$

satisfying

$$\Delta((r, f), (r', f')) = \texttt{true} \Rightarrow \Phi(r, f) \leq \Psi(r', f').$$

Hence if $((r, f), (r', f'))$ is feasible, then $\Phi(r, f) \Rightarrow \Psi(r', f')$. Hence our association must preserve feasibility.

Limits and Colimits

Inclusion of Pos

Inclusion of DP

Projection to DP

2.5 Queries in $\mathbf{Pos}_{\mathbf{Bool}}$

2.5.1 Generalized Querying

We can generalize the notion of query within $\mathbf{Pos}_{\mathbf{Bool}}$. We can interpret an object $\varphi : \mathcal{F} \to \mathbf{Bool}$ as representing a poset together with a chosen lower set (resp. upper set). This is done by considering the pre-image of `false` (resp. `true`). Conversely, choosing a lower set (resp. upper set) on a poset gives an object of $\mathbf{Pos}_{\mathbf{Bool}}$.

Using this interpretation, let $\mathcal{R}, \mathcal{F}$ be posets representing resources and functionalities, respectively. Choose lower sets $L_{\mathcal{R}} \subseteq \mathcal{R}$ and $L_{\mathcal{F}} \subseteq \mathcal{F}$. Recall that lower sets represent resources / functionalities that closed under being more 'expensive'. If the posets fulfil the ascending chain condition, we can think of the lower sets as thresholds, where we include anything more 'expensive' than the elements of some antichain.

Now observe that a morphism in $\mathbf{Pos}_{\mathbf{Bool}}$ is a feasibility relation $\mathcal{R} \overset{\Phi}{\nrightarrow} \mathcal{F}$, such that if (r, f) is feasible then $\rho(r) \leq \varphi(f)$, where $\varphi(f) =$ `false` iff $f \in L_{\mathcal{F}}$, and $\rho(r) =$ `false` iff $r \in L_{\mathcal{R}}$. Hence, if (r, f) is feasible and $f \in L_{\mathcal{F}}$, then $r \in L_{\mathcal{R}}$. Conversely, if $r \in \mathcal{R} \setminus L_{\mathcal{R}}$, then $f \in f \in \mathcal{F} \setminus L_{\mathcal{F}}$. In our threshold interpretation this means the following for a feasible pair (r, f): If f lies above (or on) our threshold for functionality, then r must lie above our threshold for resources. Conversely, if r lies below our resource threshold, then f lies below the functionality threshold.

Lets first think about varying $L_{\mathcal{R}}$. On one extreme if we let $L_{\mathcal{R}} = \mathcal{R}$, this would impose no restrictions whatsoever. On the other, if we choose $L_{\mathcal{R}} = \emptyset$, then only feasible pairs are allowed where f lies below the functionality threshold. Now lets vary $L_{\mathcal{F}}$. If $L_{\mathcal{F}} = \mathcal{F}$, then only feasible pairs with r above the resource threshold are allowed. If $L_{\mathcal{F}} = \emptyset$, we impose no restrictions.

Remark 2.2. It may seem like the variances are off here. However, suppose we had a condition that said, "if you are above a certain resource threshold, then you are above a certain funcionality threshold". This can't work, since if (r, f) is feasible and r is above the threshold, it may be that f is also above the threshold, but because of monotonicity (r, f') will also be feasible for any

$f \to f'$. Such an f' could then lie below the threshold. So our threshold would have to be trivial in order to have any feasible pairs.

We can ask the following: Given some fixed feasibility relation and fixed lower set $L_{\mathcal{F}}$ in $\mathcal{F}$, i.e. some lower threshold for functionality, how low do we need to set the threshold in $\mathcal{R}$, i.e. how big do we need to make $L_{\mathcal{R}}$ until we get a feasibility relation that can satisfy the condition for a morphism in $\mathbf{Pos}_{\mathbf{Bool}}$.

To see what this amounts to, lets take some feasibility relation Φ and fix $f \in \mathcal{F}$. We then set $L_{\mathcal{F}} = \downarrow f$. Our optimization procedure above would then yield a minimal lower set $L_{\mathcal{R}}$ in $\mathcal{R}$ such that if $\Phi(r, f)$ and $g \in \downarrow f$, then $r \in L_{\mathcal{R}}$. Assuming $\mathcal{R}$ has no infinite ascending chains, we can associate to $L_{\mathcal{R}}$ an antichain whose elements r are the 'cheapest' resources making (r, f) feasible: The minimality on $L_{\mathcal{R}}$ means that all elements r in $L_{\mathcal{R}}$ do in fact make (r, f) feasible, otherwise we could have excluded them to obtain a smaller lower set. The restriction on $L_{\mathcal{R}}$ means that we have found the cheapest resources for which f is feasible. If we take any element s that lies strictly below the antichain, (s, f) will not be feasible. For if (s, f) were feasible, then $s \in L_{\mathcal{R}}$.

In summary, we have shown that we can describe the optimization of section 2.1 in this setting. However, we can more generally query antichains of functionalities, rather than just single elements.

2.5.2 The Generalized Query Functor

Generalized querying as described in section 2.5.1 yields a contravariant functor $\mathbf{DP}_{\text{asc}} \to \mathbf{Pos}$, where $\mathbf{DP}_{\text{asc}}$ is the subcategory of $\mathbf{DP}$ where all posets satisfy the ascending chain condition. It sends a feasibility relation $\mathcal{R} \overset{\Phi}{\nrightarrow} \mathcal{F}$ to the monotone map $H_\Phi : \mathsf{A}\mathcal{F} \to \mathsf{A}\mathcal{R}$ which assigns each antichain in $\mathcal{F}$ the antichain of resources in $\mathcal{R}$ obtained by querying.

Functoriality ...

2.6 Queries as Universal Objects in $\mathbf{Pos}_{\mathbf{Bool}}$

Recall the diagram expressing the condition for morphisms in $\mathbf{Pos}_{\mathbf{Bool}}$.

$$\begin{tikzcd}
\mathcal{R} \arrow[d, "|" marking] \arrow[drr, bend left] & & \\
\mathcal{F} \arrow[rr] \arrow[ur, phantom, "\Swarrow"] & & \mathbf{Bool}
\end{tikzcd}$$

We can ask whether for fixed $\mathcal{R} \overset{\Phi}{\nrightarrow} \mathcal{F}$ and fixed $\phi : \mathcal{F} \to \mathbf{Bool}$, there is a universal ρ: For any $\rho' : \mathcal{R} \to \mathbf{Bool}$ which makes Φ a morphism in $\mathbf{Pos}_{\mathbf{Bool}}$, we have $\rho'(r) \leq \rho(r)$ for all $r \in \mathcal{R}$. This means ρ is the maximal monotone map that makes Φ a morphism in $\mathbf{Pos}_{\mathbf{Bool}}$.

$$\begin{tikzcd}
\mathcal{R} \arrow[d, "\shortmid" marking] \arrow[dr, "\rho" description] \arrow[dr, bend left=60, ""{name=U}] & \\
\mathcal{F} \arrow[r, ""{name=D}] & \mathbf{Bool}
\arrow[from=U, to=1-1, phantom, "\Swarrow" near start]
\arrow[from=1-1, to=D, phantom, "\Downarrow" near end]
\end{tikzcd}$$

Lets see what this means. We can think about ρ and φ as a lower sets $L_\rho \subseteq \mathcal{R}$ and $L_\varphi \subseteq \mathcal{F}$ in and by taking the pre-images of `false`. Because ρ makes Φ a morphism in $\mathbf{Pos}_{\mathbf{Bool}}$, if (r, f) is feasible and $f \in L_\phi$, then $r \in L_\rho$. Moreover, the universality of ρ makes L_ρ the smallest lower set making Φ a morphism in $\mathbf{Pos}_{\mathbf{Bool}}$. This is exactly the result of the optimization in section 2.5.1. In summary, we have given a description of the process there in terms of a universal property.

Lemma 2.3. *We can obtain the universal ρ by setting $\rho(r) = \bigwedge_{\{f \mid \Phi(r,f)\}} \varphi(f)$*

Proof. The result is monotone: If $r \to s$ and $\rho(s) =$ `false`, then there is f with $\varphi(f) =$ `false` and $\Phi(s, f) =$ `true`. But then $\Phi(r, f) =$ `true` by monotonicity, hence $\rho(f) =$ `false` as well. On the other hand if $\rho(r) =$ `true`, then $\varphi(f) =$ `true` for all f satisfying $\Phi(r, f) =$ `true`. If $\rho(s)$ were `false`, then there is a f with $\Phi(s, f) =$ `true` and $\varphi(f) =$ `false`. But by monotonicity also $\Phi(r, f) =$ `true`, a contradiction.

Finally, we show that this ρ is universal. First, suppose $\Phi(r, f) =$ `true`. If $\varphi(f) =$ `false`, then we have a `false` in the big wedge, so $\rho(r) =$ false. Hence, ρ makes Φ a morphism in $\mathbf{Pos}_{\mathbf{Bool}}$. Suppose we have any other ρ' making Φ a morphism. It is sufficient to show $\rho'(r) \Rightarrow \rho(r)$. Suppose $\rho'(r) =$ `true`. If $\rho(r) =$ `false`, then there would be f such that (r, f) is feasible and $\varphi(f)$ is `false`. But (r, f) feasible implies that `true` $= \rho'(r) \leq \varphi(r) =$ `false`, a contradiction. Therefore $\rho(r) =$ `true` and we have proved universality.

2.6.1 Describing Other Queries

The other optimization problem we can describe in this way is shown in the following diagram

$$\begin{tikzcd}
\mathcal{R} \arrow[d, "|" marking] \arrow[dr] & \\
\mathcal{F} \arrow[r, "\phi" description] \arrow[r, bend right=50] \arrow[r, phantom, shift left=3, "\Downarrow"] \arrow[r, phantom, shift right=3, "\Downarrow"] & \mathbf{Bool}
\end{tikzcd}$$

The univesal φ would satisfy: For any φ' making Φ a morphism in $\mathbf{Pos_{Bool}}$, we have $\phi \leq \phi'$. This makes the associated lower set L_ϕ the largest lower set making a given antichain of resources feasible.

2.7 $\mathcal{V}$-Posets