Karlsruhe Institute of Technology

Turing Categories
Computability and Complexity

Bachelor's Thesis

Jonas Linßen

Institute of Theoretical Informatics (ITI)
Department of Informatics
Karlsruhe Institute of Technology

Reviewer:	Prof. Dr. D.Hofheinz
Second reviewer:	Prof. Dr. J.Müller-Quade
Advisor:	A.Ünal, M.Sc.

28. May 2019 – 27. September 2019

Ich versichere wahrheitsgemäß, die Arbeit selbstständig verfasst, alle benutzten Hilfsmittel vollständig und genau angegeben und alles kenntlich gemacht zu haben, was aus Arbeiten anderer unverändert oder mit Abänderungen entnommen wurde sowie die Satzung des KIT zur Sicherung guter wissenschaftlicher Praxis in der jeweils gültigen Fassung beachtet zu haben.

Karlsruhe, den 27. September 2019

Abstract

In dieser Arbeit stellen wir eine kategorielle Abstraktion der berechenbaren Funktionen vor. Wir geben ausführlich wieder, wie diese sogenannten Turing-Kategorien aus einfachen kombinatorischen Systemen entstehen. Zudem erweitern wir die Liste der klassischen Resultate, die sich in diesem kategoriellen Rahmen ausdrücken lassen. Um die Arbeit in sich abgeschlossen zu halten, entwickeln wir die notwendige Kategorientheorie von Grund auf.

In this work we present a categorical abstraction of computable functions. These so called Turing categories are generated from simple combinatory systems which we discuss extensively. We extend the list of classical results which can be expressed in this categorical framework. To make the work selfcontained, we develop the necessary category theory from scratch.

Contents

Introduction

In the last century, with machines becoming increasingly useful for solving numerical problems, renowned mathematicians like Kleene, Post, Turing, Gödel and Church approached the problem of formalizing the notion of *computable functions*. Most approaches, in particular those using the λ-calculus, Turing machines or partial recursive functions, turned out to be equivalent, for details see [Odi89]. This fact opened up many possibilities to generalize computability, to encompass more than just functions on natural numbers. This work will use such a generalization in the form of certain abstract combinatory structures.

In a similiar time frame, category theory was developed to encapsulate the abstract notion of mathematical structures and their structure preserving maps. Being vastly general, categorical methods quickly found application in many areas of modern mathematics. In this light it was and still is natural to ask, how category theory might relate to computability theory. Many approaches to this question have been made, ranging from the definition of a *realizability tripos* (cf. [Oos08]) to that of an *monoidal computer* in [Pav12]. In this work we seek to generalize how the ordinary category of computable functions on $\mathbb{N}$ arises from the abstract combinatory structures mentioned above.

Since computable functions are inherently partial functions, an abstract notion of partiality is essential, when discussing categories of computable functions. While purely category theoretic formulations of partiality do exist, they usually are cumbersome and rather difficult to work with. *Restriction categories*, which were introduced in [CL02], avoid this problem by introducing an external operator. This results in an algebraic calculus, with which basic properties of partial functions can be expressed easily. Discussing the generalized concepts that emerge from this calculus extensively, we try to recast them in a higher categorical framework. In particular we develop a notion of *restriction (co)limits* that slightly differs from the original formulation given in [CL07] at first glance, but turns out to be equivalent.

With a notion of partiality at hand we return to the problem of formalizing computability in categorical terms. For this purpose we introduce *applicative systems* in arbitrary categories. Their definition generalizes the fact that every partial recursive function can be computed by running the universal Turing machine on the combination of a Gödel-number of a corresponding Turing machine and the input. Following [CH08] we characterize certain well behaved applicative systems, for which *categories of computable maps* arise. These so called *partial combinatory algebras*, whose set theoretic counterpart is well studied in realizability theory, allow us to express first results of classical computability theory in the language of restriction categories.

Since the definition of these categories of computable maps seems to be quite restrictive from a categorical point of view, Robin Cockett and Pieter Hofstra envisioned *Turing categories*. Following the line of argumentation found in [CH08], we discuss how they arise as inflations of categories of computable maps with respect to a partial combinatory algebra. We also explore how they give the same notion of computability in a precise categorical sense. Under mild assumptions Turing categories admit reformulations of even more classical results. We extend the discussion of *m-reductions*, *decidability*, *recursively inseparable sets* and *index sets* found in [Coc10] by general notions of *complexity classes*, *hierarchies of sets* and *relative computability*.

The general structure of this thesis follows the outline given above. Nevertheless we now give a concise overview of the individual chapters. In the first part the categorical framework will be developed. It starts with a comprehensive introduction into category theory in section 1.1. We then state basic terminology and discuss the categorical constructions relevant for this work. The next section 1.2 is devoted to an extensive treatment of restriction categories, in particular of restriction (co)limits. We start of the second part by introducing partial combinatory algebras in section 2.1. We show how they incorporate the classical notion of computability and how first results of recursion theory can be generalized. In section 2.2 we introduce Turing categories and put much work into showing that they are equivalent to partial combinatory algebras in a certain categorical sense. The last section 2.3 offers reformulations of many classical concepts in the language of Turing categories. In the appendix A.1 we recall basic terminology of order theory, which is used in the context of restriction categories.

Acknowledgements

First and foremost I would like to express my gratitude towards my supervisors Prof. Dr. Dennis Hofheinz and Akin Ünal, who accepted my wish to write this thesis about Turing categories.

Second, but by no means less important, I want to thank Dr. Tobias Columbus, who sparked my interest for category theory and enforced it by mentoring me.

Last but not least I would like to thank Johannes Gloßner and Laurin Benz, who supported me by reading this work, correcting my mistakes and offering helpful comments and discussions.

1 Foundations

1.1 General Category Theory

In this section we will introduce a few concepts of category theory. A *category*, at its core, abstracts the well known pattern of a mathematical structure and its structure preserving maps. This is modeled in the form of vertices and edges of a directed multigraph, equipped with a notion of composition and identities.

After introducing common terminology we will turn our focus on the structure of categories themselves. Discussing structure preserving maps between categories and their properties, we will arrive at the notion of a 2-*category*. Finally we will elaborate on various constructions within a category by using the framework of 2-categories.

The tools developed in this section can be found in any standard text on category theory, like [Mac78] or [Bor94], and will be applied repeatedly throughout this work.

§ Categories

Definition 1.1.1

A **category** $\mathcal{C}$ consists of a class[1] $\mathrm{Ob}\,\mathcal{C}$ of **objects** and a class $\mathrm{Mor}\,\mathcal{C}$ of **morphisms**, which satisfy the following conditions.

- Any morphism $f \in \mathrm{Mor}\,\mathcal{C}$ comes equipped with a unique **domain** $\mathrm{dom}\, f \in \mathrm{Ob}\,\mathcal{C}$ and a unique **codomain** $\mathrm{cod}\, f \in \mathrm{Ob}\,\mathcal{C}$ and we write $f\colon \mathrm{dom}\, f \longrightarrow \mathrm{cod}\, f$. The **hom-set** of objects $X, Y \in \mathrm{Ob}\,\mathcal{C}$ is the subclass $\mathcal{C}(X,Y)$ of $\mathrm{Mor}\,\mathcal{C}$ consisting of those morphisms with domain X and codomain Y.
- For any two **composable morphisms** $f\colon X \longrightarrow Y$ and $g\colon Y \longrightarrow Z$, i.e. the codomain of f is the domain of g, there is a distinct **composite morphism** $gf = g \circ f\colon X \longrightarrow Z$.
- Composition of morphisms is **associative**, i.e. for morphisms $f\colon W \longrightarrow X$, $g\colon X \longrightarrow Y$, $h\colon Y \longrightarrow Z$ we have $h(gf) = (hg)f =: hgf$.
- For any object X there is a distinct **identity morphism** $1_X\colon X \longrightarrow X$ such that for all morphisms $f\colon W \longrightarrow X$, $g\colon X \longrightarrow Y$ we have the identities $1_X f = f$ and $g1_X = g$.

Almost all mathematical structures have an associated category, whose objects are the instances of this structure. The most commonly used categories are those of sets with additional structure and structure preserving maps. This includes the following examples.

Name	Objects	Morphisms
Set	sets	functions
Preord	sets with preorders	order preserving maps
Poset	sets with partial orders	order preserving maps
Grp	groups	group homomorphisms
Ab	abelian groups	group homomorphisms
Ring	unital rings	ring homomorphisms
$K\text{-}\mathsf{Vec}_{\mathrm{fin}}$	finite dimensional vector spaces over K	vector space homomorphisms
Top	topological spaces	continuous functions
Cat	(small) categories	functors

There are more basic examples like finite categories, which are commonly depicted as a directed graph, as seen in the following image, where $\bullet$ denotes an object and identities are omitted.

[1] For us class is just an informal notion of large set. It is used to make a distinction from the usual (small) sets and gives the opportunity to have a class of sets. We will elaborate on this a little bit more after seeing some examples.

$$\mathbb{1} = \quad \bullet \qquad\qquad \mathbb{2} = \quad \bullet \longrightarrow \bullet \qquad\qquad \mathbb{3} = \quad \begin{tikzcd} \bullet \arrow[r] \arrow[rr, bend right] & \bullet \arrow[r] & \bullet \end{tikzcd}$$

We can also express mathematical structures in categorical terms. For example a group can be identified with a category consisting of one object, where every group element corresponds to a morphism, composition is defined by multiplication and the group axioms. Similarly preorders can be identified with categories, whose objects are the elements of the underlying set and there is a unique morphism between elements, if and only if they are comparable.

Last but not least, we can build new categories out of existing ones. The following definition states two important constructions.

Definition 1.1.2

Let $\mathcal{C}$, $\mathcal{D}$ be two categories.

(i) The **dual / opposite category** $\mathcal{C}^{op}$ is the category obtained from $\mathcal{C}$ by reversing all arrows. To be specific, we have $\operatorname{Ob}\mathcal{C}^{op} := \operatorname{Ob}\mathcal{C}$ and for an ordered pair X, Y of objects the homset is given by $\mathcal{C}^{op}(X,Y) := \mathcal{C}(Y,X)$. The composition is given by $g^{op}f^{op} := (fg)^{op}$.

(ii) The **product category** $\mathcal{C} \times \mathcal{D}$ of $\mathcal{C}$ and $\mathcal{D}$ is the category, whose objects are pairs (X, X') of objects from the respective categories, whose morphisms are pairs (f, f') of morphisms of the respective categories and whose composition is given componentwise.

In the following we will see that categories have some sort of structure preserving map themselves, which allows us to construct a category of categories. To be able to state this, we used the term *class* in the definition of a category, because there cannot be a set of sets. To have finegrained control over the size of categories, one usually classifies them according to the following table. As long as no problems with size arise, it is common practice to fail to mention the size of a category to be considered.

large	$\operatorname{Ob}\mathcal{C}$, $\operatorname{Mor}\mathcal{C}$ and all homsets $\mathcal{C}(X,Y)$ are classes.
locally small	$\operatorname{Ob}\mathcal{C}$ and $\operatorname{Mor}\mathcal{C}$ are classes, but all homsets $\mathcal{C}(X,Y)$ are sets.
small	$\operatorname{Mor}\mathcal{C}$ and in particular $\operatorname{Ob}\mathcal{C}$ and all homsets $\mathcal{C}(X,Y)$ are sets.
finite	$\operatorname{Mor}\mathcal{C}$ and in particular $\operatorname{Ob}\mathcal{C}$ and all homsets $\mathcal{C}(X,Y)$ are finite sets.

Before we look into the structure preserving maps of categories, we seek to discuss various types of morphisms, which are of particular interest inside a given category.

Definition 1.1.3

Let $\mathcal{C}$ be a category. A morphism $f\colon X \longrightarrow Y$ in $\mathcal{C}$ is

- **mono**, if it is *left-cancelable*, i.e. for any two morphisms $k_1, k_2\colon W \longrightarrow X$ in $\mathcal{C}$ we have that $fk_1 = fk_2$ implies $k_1 = k_2$.
- **epi**, if it is *right-cancelable*, i.e. for any two morphisms $h_1, h_2\colon Y \longrightarrow Z$ in $\mathcal{C}$ we have that $h_1 f = h_2 f$ implies $h_1 = h_2$.
- **split mono**, if there is a morphism $h\colon Y \longrightarrow X$ in $\mathcal{C}$ with $hf = 1_X$, i.e. if it has a *left inverse.*
- **split epi**, if there is a morphism $k\colon Y \longrightarrow X$ in $\mathcal{C}$ with $fk = 1_Y$, i.e. if it has a *right inverse.*
- **iso**, if there is a morphism $f^{-1}\colon Y \longrightarrow X$ in $\mathcal{C}$ with $f^{-1}f = 1_X$ and $ff^{-1} = 1_Y$, i.e. if it has an *inverse.*
- **endo**, if its domain equals its codomain, i.e. $X = Y$.
- **auto**, if it is an iso endomorphism.

The following table shows how the notions of mono, epi and isomorphism translate in the categories $\mathsf{Set}, \mathsf{Grp}$ and Top.

Category	mono	epi	iso
Set	injective function	surjective function	bijective function
Grp	injective group homom.	surjective group homom.	group isomorphism
Top	injective map	surjective map	homeomorphism

These examples suggest that (at least for categories whose objects have underlying sets) monos have to be injective and epis surjective. However, this is false in general. The inclusion $\mathbb{Q} \hookrightarrow \mathbb{R}$ is by density of the image epi in the category Haus of Hausdorff-spaces, but clearly not surjective. Even in the case that mono means injective and epi means surjective, for a morphism being iso it does not suffice to be mono and epi. For example a bijective continuous map needs not to be a homeomorphism, i.e. an isomorphism in Top. However in Set and Grp we have that any monomorphism is split mono. It is easy to show that in any category an epimorphism, which is split mono is an isomorphism.

As split monomorphisms will be of great importance for our purposes, we fix some notations and terminologies.

Definition 1.1.4

Let $m\colon X \longrightarrow Y$ be a split monomorphism with left inverse $r\colon Y \longrightarrow X$. One says that r is a **retraction** of m and X is a **retract** of Y. We will use the terminology that (m, r) is an **embedding-retraction pair** and will often denote it by $(m, r)\colon X \lhd Y$.

An **idempotent** is a endomorphism $e\colon Y \longrightarrow Y$ with the property $ee = e$. Note that given an embedding-retraction pair $(m, r)\colon X \lhd Y$ the composite mr is an idempotent. We say that (m, r) is a **splitting** of the idempotent mr. In general, an idempotent $e\colon Y \longrightarrow Y$ **splits**, if it is of the form $e = mr$ for an embedding-retraction pair (m, r).

Now let us consider the structure preserving maps of categories promised before.

Definition 1.1.5

Let $\mathcal{C}, \mathcal{D}$ be two categories. A **functor** $\mathcal{F}\colon \mathcal{C} \longrightarrow \mathcal{D}$ consists of two assignments $\mathcal{F}\colon \operatorname{Ob}\mathcal{C} \longrightarrow \operatorname{Ob}\mathcal{D}$ and $\mathcal{F}\colon \operatorname{Mor}\mathcal{C} \longrightarrow \operatorname{Mor}\mathcal{D}$ satisfying the following conditions.

- For each morphism $f\colon X \longrightarrow Y$ in $\mathcal{C}$ we have $\mathcal{F}f\colon \mathcal{F}X \longrightarrow \mathcal{F}Y$, i.e. $\mathcal{F}$ preserves domains and codomains.
- For each composable pair of morphisms $f\colon X \longrightarrow Y$, $g\colon Y \longrightarrow Z$ in $\mathcal{C}$ we have $\mathcal{F}(gf) = \mathcal{F}g\mathcal{F}f$, i.e. $\mathcal{F}$ preserves composition.
- For any object X in $\mathcal{C}$ we have $\mathcal{F}1_X = 1_{\mathcal{F}X}$, i.e. $\mathcal{F}$ preserves identities.

Since functors compose by definition and we certainly have an identity functor $1_{\mathcal{C}}\colon \mathcal{C} \longrightarrow \mathcal{C}$ for any arbitrary category $\mathcal{C}$ we get a (locally small) category Cat of (small) categories.

There are a lot of examples of functors in day to day mathematics. Besides obvious *forgetful functors* like $\mathsf{Top} \longrightarrow \mathsf{Set}$, $\mathsf{Grp} \longrightarrow \mathsf{Set}$ and $\mathsf{Ab} \longrightarrow \mathsf{Grp}$ one of the following examples might be familiar to the reader.

(i) To every set X there is a powerset $\mathcal{P}(X)$. Moreover, by taking images a function $f\colon X \longrightarrow Y$ gives a function $\mathcal{P}(f)\colon \mathcal{P}(X) \longrightarrow \mathcal{P}(Y)$. This assignment obviously preserves identities and composition, so we have the *powerset functor* $\mathcal{P}\colon \mathsf{Set} \longrightarrow \mathsf{Set}$.

(ii) For every set S there is a *free group* $F(S)$ with a function $\iota\colon S \hookrightarrow F(S)$, which satisfies that for any group G and function $\phi\colon S \longrightarrow G$ there exists a unique group homomorphism $\Phi\colon F(S) \longrightarrow G$ such that $\Phi\iota = \phi$. This *universal property* turns the construction of a free group into the *free group functor* $F\colon \mathsf{Set} \longrightarrow \mathsf{Grp}$.

(iii) Similarly, any group G can be turned into an abelian group $G^{ab} := G/[G, G]$ by taking the quotient with its *commutator subgroup* $[G, G] := \langle ghg^{-1}h^{-1} \mid g, h \in G\rangle$. As every group homomorphism $\Phi\colon G \longrightarrow H$ turns into an (abelian) group homomorphism $\Phi^{ab}\colon G^{ab} \longrightarrow H^{ab}$ by the first isomorphism theorem this gives the *abelianization functor* $(-)^{ab}\colon \mathsf{Grp} \longrightarrow \mathsf{Ab}$

(iv) Denote the category of topological spaces with a chosen basepoint and basepoint-preserving maps by $\mathsf{Top}_\bullet$. To each such space (X, x_0) we can assign its *fundamental group* $\pi_1(X, x_0)$, which consists of homotopy classes of loops with basepoint x_0. A basepoint preserving continuous map $f\colon (X, x_0) \longrightarrow (Y, y_0)$ induces a group homomorphism between the fundamental groups $\pi_1(f)\colon \pi_1(X, x_0) \longrightarrow \pi_1(Y, y_0)$. This gives the *fundamental group functor* $\pi_1\colon \mathsf{Top}_\bullet \longrightarrow \mathsf{Grp}$.

Another abstract but trivial example can be constructed out of any object D. The *constant functor* $\Delta D\colon \mathcal{C} \longrightarrow \mathcal{D}$ sends every object of the category $\mathcal{C}$ to D and every morphism in $\mathcal{C}$ to the identity on D. This functor will appear again later.

From a categorical standpoint, a very important functor is the so called **hom-functor** given by the assignments

$$\begin{array}{rccl} \mathcal{C}(-,-): & \mathcal{C}^{op} \times \mathcal{C} & \longrightarrow & \mathsf{Set} \\ & (X, Y) & \longmapsto & \mathcal{C}(X, Y) \ni g \\ & \downarrow (f, h) & \longmapsto & \downarrow \mathcal{C}(f, h) \qquad \downarrow \\ & (W, Z) & \longmapsto & \mathcal{C}(W, Z) \ni hgf. \end{array}$$

We cannot get into details here to explain its importance. Instead we want to make a remark about which of the categorical definitions made so far are respected by functors.

By the definition of functors every categorical property, which is defined using composition and identities only, is *preserved* by functors. This means that the image of an object or morphism with this property also has the property. Examples for such properties include being split mono, split epi and split idempotent as well as being iso. Similarly, the image of a commuting diagram under a functor is again a commuting diagram.

Like morphisms functors can have various properties. The following definition states some of them, mainly to give us some terminology to work with.

Definition 1.1.6

Let $\mathcal{F}\colon \mathcal{C} \longrightarrow \mathcal{D}$ be a functor between categories $\mathcal{C}$ and $\mathcal{D}$. We say that $\mathcal{F}$ is

- **injective on objects**, if its assignment $\mathcal{F}\colon \operatorname{Ob}\mathcal{C} \longrightarrow \operatorname{Ob}\mathcal{D}$ is injective.
- **faithful**, if it is *locally injective*, i.e. if the assignment $\mathcal{F}\colon \operatorname{Mor}\mathcal{C} \longrightarrow \operatorname{Mor}\mathcal{D}$ is injective. This is equivalent to the induced assignment $\mathcal{F}\colon \mathcal{C}(X, Y) \longrightarrow \mathcal{D}(\mathcal{F}X, \mathcal{F}Y)$ being injective for all objects X, Y in $\mathcal{C}$.
- an **embedding**, if it is faithful and injective on objects.
- **full**, if it is *locally surjective*, i.e. for all objects $X, Y \in \operatorname{Ob}\mathcal{C}$ the induced assignment $\mathcal{F}\colon \mathcal{C}(X, Y) \longrightarrow \mathcal{D}(\mathcal{F}X, \mathcal{F}Y)$ is surjective.

One often wants to compare not only categories (via functors) but functors with each other. For example consider the relation between the identity functor on Grp and the abelianization functor of the examples above seen as functor $(-)^{ab}\colon \mathsf{Grp} \longrightarrow \mathsf{Grp}$. Writing $\pi_G\colon G \longrightarrow G^{ab}$ for the projection of G onto $G^{ab} = G/[G, G]$ we have for any group homomorphism $\Phi\colon G \longrightarrow H$ the following commuting diagram.

$$\begin{tikzcd} G \arrow[r, "\pi_G"] \arrow[d, "\Phi"'] & G^{ab} \arrow[d, "\Phi^{ab}"] \\ H \arrow[r, "\pi_H"'] & H^{ab} \end{tikzcd}$$

In fact, giving a precise definition of this behaviour was a key motivation for the development of category theory by Eilenberg and MacLane [Mac78, p. 18].

Definition 1.1.7

Let $\mathcal{F}, \mathcal{G} \colon \mathcal{C} \longrightarrow \mathcal{D}$ be two parallel functors between categories $\mathcal{C}$ and $\mathcal{D}$. A **natural transformation** $\alpha \colon \mathcal{F} \implies \mathcal{G}$ from $\mathcal{F}$ to $\mathcal{G}$ consists of a family $(\alpha_Z)_{Z \in \mathrm{Ob}\,\mathcal{C}}$ of morphisms in $\mathcal{D}$, called **components** of α, such that for every morphism $f \colon X \longrightarrow Y$ in $\mathcal{C}$ the diagram on the right commutes.

$$\begin{tikzcd} \mathcal{F}X \arrow[r, "\alpha_X"] \arrow[d, "\mathcal{F}f"'] & \mathcal{G}X \arrow[d, "\mathcal{G}f"] \\ \mathcal{F}Y \arrow[r, "\alpha_Y"'] & \mathcal{G}Y \end{tikzcd}$$

If all components of a natural transformation are isomorphisms, we call the natural transformation a **natural isomorphism** and denote it by $\mathcal{F} \overset{\sim}{\implies} \mathcal{G}$.

Intuitively, a natural transformation moves the image of one functor to the image of another functor within the target category. Besides relating functors with each other, natural transformations play a very important role in comparing categories. It turns out that some categories behave very much the same, but are not related with an isomorphism in Cat, i.e. with an invertible functor. The problem is that the inverses are often only defined up to natural isomorphism. Thanks to natural transformations we can give a definition of such a sort of weak isomorphism of categories.

Definition 1.1.8

A functor $\mathcal{F} \colon \mathcal{C} \longrightarrow \mathcal{D}$ establishes an **equivalence of categories**, if there is another functor $\mathcal{G} \colon \mathcal{D} \longrightarrow \mathcal{C}$ such that there are natural isomorphisms $1_{\mathcal{C}} \overset{\sim}{\implies} \mathcal{G}\mathcal{F}$ and $\mathcal{F}\mathcal{G} \overset{\sim}{\implies} 1_{\mathcal{D}}$. In this case we say $\mathcal{C}$ and $\mathcal{D}$ are **equivalent** and denote it by $\mathcal{C} \simeq \mathcal{D}$.

One last thing to mention about natural transformations is that they again compose (componentwise) and for any functor $\mathcal{F}$ there always is an identity transformation $1_{\mathcal{F}} \colon \mathcal{F} \implies \mathcal{F}$. In particular for every pair of (small) categories $\mathcal{C}$ and $\mathcal{D}$ we have a (small) category $[\![\mathcal{C}, \mathcal{D}]\!]$ of functors and natural transformations. This means that the hom-sets $\mathsf{Cat}(\mathcal{C}, \mathcal{D})$ are not just plain sets but can be equipped with the categorical structure of $[\![\mathcal{C}, \mathcal{D}]\!]$. Moreover natural transformations behave well with respect to pre- or postcomposing functors under an operation called *whiskering*. Indeed, given the following configuration of categories, functors and a natural transformation

$$\begin{tikzcd} \mathcal{W} \arrow[r, "\mathcal{H}"] & \mathcal{X} \arrow[r, bend left, "\mathcal{F}_1", ""{name=U, below}] \arrow[r, bend right, "\mathcal{F}_2"', ""{name=D, above}] & \mathcal{Y} \arrow[r, "\mathcal{K}"] & \mathcal{Z} \arrow[from=U, to=D, phantom, "\Downarrow\alpha"] \end{tikzcd}$$

we obtain a natural transformation $\mathcal{K}\alpha\mathcal{H} \colon \mathcal{K}\mathcal{F}_1\mathcal{H} \implies \mathcal{K}\mathcal{F}_2\mathcal{H}$ in the following way.

$$\begin{tikzcd} \mathcal{F}_1X \arrow[r, "\alpha_X"] \arrow[d, "\mathcal{F}_1x"'] & \mathcal{F}_2X \arrow[d, "\mathcal{F}_2x"] & \mathcal{F}_1\mathcal{H}W \arrow[r, "\alpha_{\mathcal{H}W}"] \arrow[d, "\mathcal{F}_1\mathcal{H}w"'] & \mathcal{F}_2\mathcal{H}W \arrow[d, "\mathcal{F}_2\mathcal{H}w"] \\ \mathcal{F}_1X' \arrow[r, "\alpha_{X'}"'] & \mathcal{F}_2X' & \mathcal{F}_1\mathcal{H}W' \arrow[r, "\alpha_{\mathcal{H}W'}"'] & \mathcal{F}_2\mathcal{H}W' \end{tikzcd}$$

$$\begin{tikzcd} \mathcal{K}\mathcal{F}_1X \arrow[r, "\mathcal{K}\alpha_X"] \arrow[d, "\mathcal{K}\mathcal{F}_1x"'] & \mathcal{K}\mathcal{F}_2X \arrow[d, "\mathcal{K}\mathcal{F}_2x"] & \mathcal{K}\mathcal{F}_1\mathcal{H}W \arrow[r, "\mathcal{K}\alpha_{\mathcal{H}W}"] \arrow[d, "\mathcal{K}\mathcal{F}_1\mathcal{H}w"'] & \mathcal{K}\mathcal{F}_2\mathcal{H}W \arrow[d, "\mathcal{K}\mathcal{F}_2\mathcal{H}w"] \\ \mathcal{K}\mathcal{F}_1X' \arrow[r, "\mathcal{K}\alpha_{X'}"'] & \mathcal{K}\mathcal{F}_2X' & \mathcal{K}\mathcal{F}_1\mathcal{H}W' \arrow[r, "\mathcal{K}\alpha_{\mathcal{H}W'}"'] & \mathcal{K}\mathcal{F}_2\mathcal{H}W' \end{tikzcd}$$

The upper left square commutes for every morphism $x \colon X \longrightarrow X'$ in $\mathcal{X}$ since α is a natural transformation. This in particular holds for all morphisms in the image of $\mathcal{H}$ and is depicted as the upper right square. On the other hand applying the functor $\mathcal{K}$ onto a naturality square of α preserves its commutativity. This is depicted in the bottom left square. Together we obtain a naturality square as depicted at the bottom right. In fact, by associativity of functor composition we have $\mathcal{K}(\alpha\mathcal{H}) = (\mathcal{K}\alpha)\mathcal{H}$, i.e. this construction is independent from the order of pre- and postcomposition.

Moreover one can show that it assembles into a functor

$$\begin{array}{rcl}
\mathsf{Cat}(\mathcal{H},\mathcal{K}) : [\![\mathcal{X},\mathcal{Y}]\!] & \longrightarrow & [\![\mathcal{W},\mathcal{Z}]\!] \\
\mathcal{F}_1 & \longmapsto & \mathcal{K}\mathcal{F}_1\mathcal{H} \\
\alpha \Big\Downarrow & \longmapsto & \Big\Downarrow \mathcal{K}\alpha\mathcal{H} \\
\mathcal{F}_2 & \longmapsto & \mathcal{K}\mathcal{F}_2\mathcal{H}.
\end{array}$$

The following section tries to give an abstract categorical notion of the name 2-category, which describes this behaviour of Cat in the way of having a category with morphisms between morphisms and some compatibility conditions with respect to the underlying structure.

§ 2-Categories

There are many different, but equivalent ways to define 2-categories. The most common approach is to define a *category enriched in the monoidal category* Cat. A less common approach is to define a notion of category with two types of morphisms and axiomatize Cat in this setting. Because it suits our needs best, we will use such a definition found in [Str96].

Definition 1.1.9
A **2-category** $\mathcal{C}$ consists of a (locally small) category $\mathcal{C}_0$ together with a functor

$$\mathcal{C}(-,-)\colon \mathcal{C}_0^{op} \times \mathcal{C}_0 \longrightarrow \mathsf{Cat}$$

which satisfies the following conditions.

- Postcomposing $\mathcal{C}(-,-)$ with the object functor $\mathrm{Ob}\colon \mathsf{Cat} \longrightarrow \mathsf{Set}$, which sends each (small) category to its set of objects, gives the usual hom-functor of $\mathcal{C}_0$, i.e.

$$\mathrm{Ob} \circ \mathcal{C}(-,-) = \mathcal{C}_0(-,-).$$

  With other words, the objects in the **hom-categories** are precisely the morphisms in the underlying category $\mathcal{C}_0$ and will sometimes be called **1-cells**. We will denote the class of all 1-cells in $\mathcal{C}$ by $\mathrm{Mor}(\mathcal{C})$.

  The morphisms in the hom-categories are called **2-cells** and will be denoted in the form $\alpha\colon f_1 \Rightarrow f_2\colon X \longrightarrow Y$. The class of all 2-cells in $\mathcal{C}$ will be denoted by $\mathrm{Cell}(\mathcal{C})$. We refer to the composition of 2-cells as **vertical composition**, a name suggested by the following diagram, and denote it by $\beta \cdot \alpha$.

$$\begin{tikzcd}
X \arrow[rr, bend left=60, "f_1"] \arrow[rr, "f_2" description] \arrow[rr, bend right=60, "f_3"'] \arrow[rr, phantom, shift left=3ex, "\Downarrow \alpha"] \arrow[rr, phantom, shift right=3ex, "\Downarrow \beta"] & & Y
\end{tikzcd}
=
\begin{tikzcd}
X \arrow[r, bend left, "f_1"] \arrow[r, bend right, "f_3"'] \arrow[r, phantom, "\Downarrow \beta \cdot \alpha"] & Y
\end{tikzcd}$$

  Moreover as $\mathcal{C}(-,-)$ is a functor, it sends a pair of morphisms $h\colon W \longrightarrow X, k\colon Y \longrightarrow Z$ in $\mathcal{C}_0^{op} \times \mathcal{C}_0$ to a functor

$$\begin{array}{rrcl}
\mathcal{C}(h,k) : & \mathcal{C}(X,Y) & \longrightarrow & \mathcal{C}(W,Z) \\
& [f : X \to Y] & \longmapsto & [kfh : W \to Z] \\
& [\eta : f_1 \Rightarrow f_2] & \longmapsto & [k\eta h : kf_1h \Rightarrow kf_2h]
\end{array}$$

called **whiskering**. Diagrammatically this translates to having an assignment

$$\left[\begin{tikzcd} W \arrow[r, "h"] & X \arrow[r, bend left, "f_1"] \arrow[r, bend right, "f_2"'] \arrow[r, phantom, "\Downarrow\eta"] & Y \arrow[r, "k"] & Z \end{tikzcd}\right] \longmapsto \left[\begin{tikzcd} W \arrow[rr, bend left, "kf_1h"] \arrow[rr, bend right, "kf_2h"'] \arrow[rr, phantom, "\Downarrow k\eta h"] & & Z \end{tikzcd}\right].$$

- Given 2-cells $\alpha\colon f_1 \Rightarrow f_2\colon X \longrightarrow Y$ and $\beta\colon g_1 \Rightarrow g_2\colon Y \longrightarrow Z$ we have the equation

$$\begin{tikzcd} X \arrow[r, bend left, "g_1f_1"] \arrow[r, bend right, "g_2f_2"'] \arrow[r, phantom, "\Downarrow \beta f_2 \cdot g_1\alpha"] & Z \end{tikzcd} = \begin{tikzcd} X \arrow[r, bend left=60, "g_1f_1"] \arrow[r, "g_1f_2" description] \arrow[r, bend right=60, "g_2f_2"'] \arrow[r, phantom, shift left=3, "\Downarrow g_1\alpha"] \arrow[r, phantom, shift right=3, "\Downarrow \beta f_2"] & Z \end{tikzcd} = \begin{tikzcd} X \arrow[r, bend left=60, "g_1f_1"] \arrow[r, "g_2f_1" description] \arrow[r, bend right=60, "g_2f_2"'] \arrow[r, phantom, shift left=3, "\Downarrow \beta f_1"] \arrow[r, phantom, shift right=3, "\Downarrow g_2\alpha"] & Z \end{tikzcd} = \begin{tikzcd} X \arrow[r, bend left, "g_1f_1"] \arrow[r, bend right, "g_2f_2"'] \arrow[r, phantom, "\Downarrow g_2\alpha \cdot \beta f_1"] & Z \end{tikzcd}.$$

Defining $\beta * \alpha \coloneqq \beta f_2 \cdot g_1\alpha = g_2\alpha \cdot \beta f_1$ we get the **horizontal composition** of 2-cells like in the following diagram.

$$\begin{tikzcd} X \arrow[r, bend left, "f_1"] \arrow[r, bend right, "f_2"'] \arrow[r, phantom, "\Downarrow\alpha"] & Y \arrow[r, bend left, "g_1"] \arrow[r, bend right, "g_2"'] \arrow[r, phantom, "\Downarrow\beta"] & Z \end{tikzcd} = \begin{tikzcd} X \arrow[r, bend left, "g_1f_1"] \arrow[r, bend right, "g_2f_2"'] \arrow[r, phantom, "\Downarrow \beta * \alpha"] & Z \end{tikzcd}$$

Remark 1.1.10

According to [Str96] vertical and horizontal composition of 2-cells commute. Classically, this is expressed as the *middle four interchange law*, which is given by the following equation.

$$\begin{tikzcd} X \arrow[r, bend left, "g_1f_1"] \arrow[r, bend right, "g_3f_3"'] \arrow[r, phantom, "\Downarrow (\delta\cdot\gamma)*(\beta\cdot\alpha)"] & Z \end{tikzcd} = \begin{tikzcd} X \arrow[r, bend left=60, "f_1"] \arrow[r, "f_2" description] \arrow[r, bend right=60, "f_3"'] \arrow[r, phantom, shift left=3, "\Downarrow\alpha"] \arrow[r, phantom, shift right=3, "\Downarrow\beta"] & Y \arrow[r, bend left=60, "g_1"] \arrow[r, "g_2" description] \arrow[r, bend right=60, "g_3"'] \arrow[r, phantom, shift left=3, "\Downarrow\gamma"] \arrow[r, phantom, shift right=3, "\Downarrow\delta"] & Z \end{tikzcd} = \begin{tikzcd} X \arrow[r, bend left, "g_1f_1"] \arrow[r, bend right, "g_3f_3"'] \arrow[r, phantom, "\Downarrow (\delta*\beta)\cdot(\gamma*\alpha)"] & Z \end{tikzcd}$$

In particular this means that $- * -\colon \mathcal{C}(Y,Z) \times \mathcal{C}(X,Y) \longrightarrow \mathcal{C}(X,Z)$ is a functor.

We have already shown that the category Cat consisting of small categories, functors and natural transformations forms the archetypical 2-category. Another example is the 2-category Top of topological spaces, continuous maps and homotopy classes of homotopies of maps. We are more interested in the following family of 2-categories.

Recall that any preorder or poset can be considered as a (small) category with unique arrows if there is a relation between elements / objects. Suppose that we have a preorder or poset structure on the homsets of a category $\mathcal{C}$, which is respected by pre- and postcomposition in the sense that for morphisms $f_1, f_2\colon X \longrightarrow Y$ and $h\colon W \longrightarrow X$,$k\colon Y \longrightarrow Z$ we have that $f_1 \leq f_2$ implies $kf_1h \leq kf_2h$. This then gives us a 2-category structure on $\mathcal{C}$, because all equations involving 2-cells are trivially satisfied by uniqueness of the 2-cells.

Of course 2-categories come with their own form of structure preserving map.

Definition 1.1.11

Let $\mathcal{C}, \mathcal{D}$ be 2-categories. A **2-functor** $\mathcal{F}\colon \mathcal{C} \longrightarrow \mathcal{D}$ consists of assignments $\mathcal{F}\colon \mathrm{Ob}(\mathcal{C}) \longrightarrow \mathrm{Ob}(\mathcal{D})$, $\mathcal{F}\colon \mathrm{Mor}(\mathcal{C}) \longrightarrow \mathrm{Mor}(\mathcal{D})$ and $\mathcal{F}\colon \mathrm{Cell}(\mathcal{C}) \longrightarrow \mathrm{Cell}(\mathcal{D})$ such that the following conditions are met.

- The assignments on objects and morphisms define a functor $\mathcal{F}_0\colon \mathcal{C}_0 \longrightarrow \mathcal{D}_0$, i.e. preserve domains, codomains, identities and composition of 1-cells.
- Restricted to any hom-category $\mathcal{C}(X,Y)$ the assignments of morphisms and cells give a functor $\mathcal{F}_{XY}\colon \mathcal{C}(X,Y) \longrightarrow \mathcal{D}(\mathcal{F}X, \mathcal{F}Y)$, i.e. preserve domains, codomains, identities and vertical

composition of 2-cells.

- For any two morphisms $h\colon W \longrightarrow X, k\colon Y \longrightarrow Z$ in $\mathcal{C}$ we have the identity

$$\mathcal{F}_{WZ} \circ \mathcal{C}(h,k) = \mathcal{D}(\mathcal{F}h, \mathcal{F}k) \circ \mathcal{F}_{XY},$$

  i.e. the assignments preserve whiskering and in particular horizontal composition.

With other other words a 2-functor preserves all kinds of domains, codomains, identities and compositions arising in the definition of a 2-category. Similarly there is a straightforward generalization of natural transformation into the 2-categorical setting.

Definition 1.1.12

A **2-natural transformation** $\alpha\colon \mathcal{F} \Longrightarrow \mathcal{G}$ of 2-functors $\mathcal{F}, \mathcal{G}\colon \mathcal{C} \longrightarrow \mathcal{D}$ consists of morphisms $\alpha_X\colon \mathcal{F}X \longrightarrow \mathcal{G}X$ in $\mathcal{D}$ which assemble into an ordinary natural transformation and which are 2-natural in the sense that for any 2-cell $\delta\colon f_1 \Rightarrow f_2\colon X \longrightarrow Y$ in $\mathcal{C}$ we have the following equality.

$$\begin{tikzcd} \mathcal{F}X \arrow[r, bend left, "\mathcal{F}f_1"] \arrow[r, bend right, "\mathcal{F}f_1"'] \arrow[r, phantom, "\Downarrow \mathcal{F}\delta"] & \mathcal{F}Y \arrow[r, "\alpha_Y"] & \mathcal{G}Y \end{tikzcd} = \begin{tikzcd} \mathcal{F}X \arrow[r, "\alpha_X"] & \mathcal{G}X \arrow[r, bend left, "\mathcal{G}f_1"] \arrow[r, bend right, "\mathcal{G}f_2"'] \arrow[r, phantom, "\Downarrow \mathcal{G}\delta"] & \mathcal{G}Y \end{tikzcd}$$

Yet again we get a 2-category $\mathsf{2\text{-}Cat}$ of (small) 2-categories, 2-functors and 2-natural transformations.

The presence of 2-cells gives us the opportunity to define a lax version of natural transformations between 2-categories, which will play a role for us in later definitions.

Definition 1.1.13

A **lax natural transformation** $\alpha\colon \mathcal{F} \Longrightarrow \mathcal{G}$ between given 2-functors $\mathcal{F}, \mathcal{G}\colon \mathcal{C} \longrightarrow \mathcal{D}$ consists of morphisms $\alpha_X\colon \mathcal{F}X \longrightarrow \mathcal{G}X$ in $\mathcal{D}$ together with 2-cells $\alpha_f\colon \alpha_Y \mathcal{F}f \Longrightarrow \mathcal{G}f \alpha_X$ for any morphism $f\colon X \longrightarrow Y$ in $\mathcal{C}$ (as depicted on the right) such that the following conditions hold.[2]

$$\begin{tikzcd} \mathcal{F}X \arrow[r, "\alpha_X"] \arrow[d, "\mathcal{F}f"'] & \mathcal{G}X \arrow[d, "\mathcal{G}f"] \\ \mathcal{F}Y \arrow[r, "\alpha_Y"'] \arrow[ur, phantom, "\Nearrow \alpha_f"] & \mathcal{G}Y \end{tikzcd}$$

- At the level of 2-cells α is a natural transformation of some kind. Concretely, for any 2-cell $\delta\colon f_1 \Rightarrow f_2$ in $\mathcal{C}$ the diagram on the right commutes.

$$\begin{tikzcd} \alpha_Y \mathcal{F}f_1 \arrow[r, Rightarrow, "\alpha_{f_1}"] \arrow[d, Rightarrow, "\alpha_Y \mathcal{F}\delta"'] & \mathcal{G}f_1 \alpha_X \arrow[d, Rightarrow, "\mathcal{G}\delta\alpha_X"] \\ \alpha_Y \mathcal{F}f_2 \arrow[r, Rightarrow, "\alpha_{f_2}"'] & \mathcal{G}f_2 \alpha_X \end{tikzcd}$$

- In some sense, α respects identities and composition. Specifically we assert that we have the equation

$$\alpha_{1_X} = 1_{\alpha_X}$$

  and that the diagram on the right commutes.

$$\begin{tikzcd} \alpha_Z \mathcal{F}g\mathcal{F}f \arrow[r, Rightarrow, "\alpha_g \mathcal{F}f"] \arrow[d, equal] & \mathcal{G}g\alpha_Y \mathcal{F}f \arrow[r, Rightarrow, "\mathcal{G}g\alpha_f"] & \mathcal{G}g\mathcal{G}f\alpha_X \arrow[d, equal] \\ \alpha_Z \mathcal{F}(gf) \arrow[rr, Rightarrow, "\alpha_{gf}"'] & & \mathcal{G}(gf)\alpha_X \end{tikzcd}$$

By whiskering appropriately lax natural transformations can be composed. Indeed there is a 2-category $\mathsf{2\text{-}Cat}_{lax}$ of (small) 2-categories, 2-functors and lax natural transformations.

Furthermore 2-natural transformations can be considered as lax natural transformations, where the family $(\alpha_f)_{f \in \mathrm{Mor}\,\mathcal{C}}$ is given by identity 2-cells. In this light the following definition applies to

[2]For any reader just warming up with category theory we note that the following equations, which involve 2-cells only, will not be relevant in this work and can be ignored for our purposes. This is due to the fact that occurring 2-categories arise from a poset structure on homsets, so by uniqueness of 2-cells all such equations are fulfilled automatically.

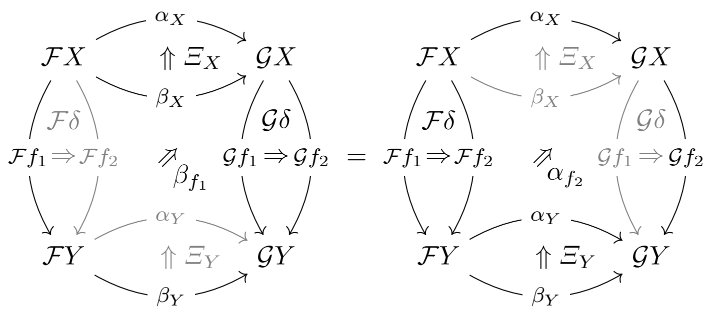
2-natural transformations as well.

Definition 1.1.14

Let $\alpha, \beta \colon \mathcal{F} \implies \mathcal{G}$ be two lax natural transformations between 2-functors $\mathcal{F}, \mathcal{G} \colon \mathcal{C} \longrightarrow \mathcal{D}$. A **modification** $\alpha \Rrightarrow \beta$ consists of 2-cells $\Xi_X \colon \alpha_X \Rightarrow \beta_X$ for each object X, such that for each 2-cell $\delta \colon f_1 \Rightarrow f_2$ in $\mathcal{C}$ the equation[2]

$$(\mathcal{G}\delta * \Xi_X) \cdot \beta_{f_1} = \alpha_{f_2} \cdot (\Xi_Y * \mathcal{F}\delta)$$

which is depicted in the following diagram, holds.



The careful reader might have noticed that we just defined a kind of morphism between 2-natural transformations. This and the notation $\alpha \Rrightarrow \beta$ suggests that we have defined a *3-cell* in some kind of *3-category*. This opens the door to the study of *higher categories*. Our purpose of making the preceding definition is, however, that it makes the category $[\![\mathcal{C}, \mathcal{D}]\!]$ of 2-functors and 2-natural transformations into a 2-category. Similarly we get the 2-category $[\![\mathcal{C}, \mathcal{D}]\!]_{lax}$ of 2-functors, lax natural transformations and modifications.

Having the notions of categories and 2-categories at hand we can complete our survey of category theory by considering a few structures one can find within categories or 2-categories.

§ Internal structures

Definition 1.1.15

Let $\mathcal{C}$ be a 2-category and $f \colon X \longrightarrow Y$, $g \colon Y \longrightarrow X$ be two morphisms in $\mathcal{C}$. The morphisms f and g establish an **equivalence** of the objects X and Y, if there are invertible 2-cells $1_X \overset{\sim}{\Rightarrow} gf$ and $fg \overset{\sim}{\Rightarrow} 1_Y$.

In the 2-category Cat we have already seen this notion as equivalence of categories. In 2-categories with 2-cells given by posets, the notion of equivalence can be a useful weakening of objects being isomorphic, as we will see in lemma 2.1.12 and theorem 2.2.8. However in 2-categories, whose 2-category structure comes from posets, reflexivity implies that there are no invertible 2-cells besides identities. This means that the notion of isomorphic and equivalent objects coincide.

One can see the following definition as further generalization of equivalences.

Definition 1.1.16

Let $f \colon X \longrightarrow Y$, $g \colon Y \longrightarrow X$ be two morphisms in a 2-category $\mathcal{C}$. We say that f is **left adjoint** to g, denoted by $f \dashv g$, (and conversely g is **right adjoint** to f) if there are 2-cells $\eta \colon 1_X \implies gf$ and $\epsilon \colon fg \implies 1_Y$ in $\mathcal{C}$, called **unit** and **counit**, making the following diagrams of 2-cells commute.

They represent the so called *triangle identities.*

$$\begin{tikzcd} g \arrow[r, Rightarrow, "\eta g"] \arrow[dr, Rightarrow, "1_g"'] & gfg \arrow[d, Rightarrow, "g\epsilon"] \\ & g \end{tikzcd} \quad \text{and} \quad \begin{tikzcd} f \arrow[r, Rightarrow, "f\eta"] \arrow[dr, Rightarrow, "1_f"'] & fgf \arrow[d, Rightarrow, "\epsilon f"] \\ & f \end{tikzcd}$$

In the category Cat adjunctions are called *adjoint functors* and those in 2-Cat are called 2-*adjoint functors.*

Adjunctions often appear, when some mathematical structure is stacked upon another in such a way that one can forget the structure on the one hand and on the other hand retrieve a free object with the additional structure. For example the *free group functor* $F\colon \mathsf{Set} \longrightarrow \mathsf{Grp}$ is left adjoint to the *forgetful functor* $U\colon \mathsf{Grp} \longrightarrow \mathsf{Set}$, where the unit is given by the inclusions $\iota_S\colon S \hookrightarrow UF(S)$ of the construction of a free group and the counit is given by the group homomorphism $FU(G) \longrightarrow G$ induced by the identity homomorphism $id_G\colon G \longrightarrow G$ considered as function. Similarly, since every category can be considered as a 2-category with trivial 2-cell structure and every 2-category $\mathcal{C}$ has an underlying 1-category $\mathcal{C}_0$, we have the following 2-adjunction.

$$\begin{tikzcd} \mathsf{Cat} \arrow[r, bend left, "\mathsf{Triv}"] \arrow[r, phantom, "\perp"] & 2\text{-}\mathsf{Cat}, \arrow[l, bend left, "(-)_0"] \end{tikzcd}$$

Moreover, as we will see, it is possible to use adjoint functors to define a global notion of the following 1-categorical structure and to lift it into other settings like 2-categories or restriction categories.

Definition 1.1.17

Let $\mathcal{C}$ be a category and $\mathcal{I}$ be a small category. The image of a functor $\mathcal{D}\colon \mathcal{I} \longrightarrow \mathcal{C}$ is a **diagram of shape** $\mathcal{I}$ in $\mathcal{C}$. If $\mathcal{I}$ is a finite category, we say that $\mathcal{D}$ gives a **finite diagram**.

We use the same terminology for the case when $\mathcal{C}$ and possibly $\mathcal{I}$ are 2-categories.

In the following constructions we will need a certain kind of object, defined as follows.

Definition 1.1.18

Let $\mathcal{C}$ be a 1-category. An object 0 in $\mathcal{C}$ is **initial**, if for every object T in $\mathcal{C}$ there is a unique morphism $?_T\colon 0 \longrightarrow T$. Dually, an object 1 is **terminal**, if for every object T there is a unique morphism $!_T\colon T \longrightarrow 1$. An object, which is both initial and terminal is called a **zero object**.[3]

It is easy to show that initial and terminal objects are unique up to unique isomorphism. In particular this means that the following constructions itself are unique up to isomorphism.

Definition 1.1.19

Let $\mathcal{I}$ be a small category and $\mathcal{D}\colon \mathcal{I} \longrightarrow \mathcal{C}$ be a diagram in a category $\mathcal{C}$.

A **cone under** $\mathcal{D}$ with nadir X is a natural transformation $\alpha\colon \mathcal{D} \Longrightarrow \Delta X$ from $\mathcal{D}$ to the constant functor $\Delta X\colon \mathcal{I} \longrightarrow \mathcal{C}$. We call its components **legs**.

A **morphism of cones** from $\alpha\colon \mathcal{D} \Longrightarrow \Delta X$ to $\beta\colon \mathcal{D} \Longrightarrow \Delta Y$ is a morphism $f\colon X \longrightarrow Y$ in $\mathcal{C}$ such that for each $I \in \operatorname{Ob}\mathcal{I}$ we have $f\alpha_I = \beta_I$. This defines a category of cones under $\mathcal{D}$.

A **colimit** of the diagram $\mathcal{D}$ is a cone under $\mathcal{D}$, denoted by $\operatorname{colim}\mathcal{D}$, which is initial in the category of cones under $\mathcal{D}$.

[3]Not to be confused with the notation 0 for an (not necessarily terminal) initial object. Category theorists are very good in using inconsistent terminology and notation.

The dual notion is that of a limit. The following makes this explicit.

Definition 1.1.20

Let $\mathcal{I}$ be a small category and $\mathcal{D}\colon \mathcal{I} \longrightarrow \mathcal{C}$ be a diagram in a category $\mathcal{C}$.

A **cone over** $\mathcal{D}$ with apex X is a natural transformation $\alpha\colon \varDelta X \implies \mathcal{D}$ from the constant functor $\varDelta X : \mathcal{I} \longrightarrow \mathcal{C}$ to $\mathcal{D}$. Its components will be called **legs**.

A **morphism of cones** from $\alpha\colon \varDelta X \implies \mathcal{D}$ to $\beta\colon \varDelta Y \implies \mathcal{D}$ is a morphism $f\colon X \longrightarrow Y$ in $\mathcal{C}$ such that for each $I \in \operatorname{Ob}\mathcal{I}$ we have $\beta_I f = \alpha_I$. This defines a category of cones over $\mathcal{D}$.

A **limit** of the diagram $\mathcal{D}$ is a cone over $\mathcal{D}$, denoted by $\lim \mathcal{D}$, which is terminal in the category of cones over $\mathcal{D}$.

The following diagrams show the defining *universal properties* of a colimit and a limit. For purposes of visualization the colimit cone is denoted as natural transformation $\gamma\colon \mathcal{D} \implies C$ and the limit cone is depicted as natural transformation $\lambda\colon L \implies \mathcal{D}$. It is important to note that the unique morphism of cones from γ to some cone α / from some cone α to λ must not be altered for different choices of objects I and J in $\mathcal{I}$. Indeed, the diagrams only depict the naturality squares (triangles) for some arbitrary morphism $I \longrightarrow J$ in $\mathcal{I}$. With this in mind the diagrams hopefully clarify the new terminology and construction.

$$\begin{tikzcd}
\mathcal{D}I \arrow[rr] \arrow[dr, "\gamma_I"] \arrow[ddr, bend right, "\alpha_I"'] & & \mathcal{D}J \arrow[dl, "\gamma_J"'] \arrow[ddl, bend left, "\alpha_J"] \\
& C \arrow[d, dashed, "\exists!"] & \\
& X &
\end{tikzcd}$$

colimit

$$\begin{tikzcd}
& X \arrow[d, dashed, "\exists!"] \arrow[ddl, bend right, "\alpha_I"'] \arrow[ddr, bend left, "\alpha_J"] & \\
& L \arrow[dl, "\lambda_I"'] \arrow[dr, "\lambda_J"] & \\
\mathcal{D}I \arrow[rr] & & \mathcal{D}J
\end{tikzcd}$$

limit

These constructions may seem a bit arbitrary at first but in fact many objects of mathematical interest can be identified as limit or colimit. We give some examples.

(i) Initial and terminal objects can be regarded as colimit and limit of the empty diagram. While this is a circular definition at this point, it might become handy when definining a notion of initial or terminal object in other settings.

In Ring we have $\mathbb{Z}$ as initial and $\{1\}$ as terminal object. In Grp the *trivial group* 0 is both initial and terminal i.e. a zero object.

(ii) In Set and Top the colimit of a diagram consisting of two sets or topological spaces X, Y is given by the disjoint union $X \sqcup Y$, while its limit is given by their product $X \times Y$.

$$\begin{tikzcd}
X \arrow[r, "\iota_X"] \arrow[dr, "f"'] & X+Y \arrow[d, dashed, "{[f,g]}"] & Y \arrow[l, "\iota_Y"'] \arrow[dl, "g"] \\
& \mathrm{T} &
\end{tikzcd}$$

coproduct

In general, a **coproduct** / **product** of objects X, Y in some category $\mathcal{C}$ is the colimit / limit of the diagram consisting of these two objects and comes with the following universal properties depicted on the left / right. The legs of the corresponding cones are called **inclusions** / **projections**. The unique morphisms $\nabla \coloneqq [1_X, 1_X]\colon X+X \longrightarrow X$ and $\varDelta \coloneqq \langle 1_X, 1_X \rangle\colon X \longrightarrow X \times X$ are called **codiagonal** and **diagonal**.

$$\begin{tikzcd}
& \mathrm{T} \arrow[dl, "f"'] \arrow[d, dashed, "{\langle f,g \rangle}"] \arrow[dr, "g"] & \\
X & X \times Y \arrow[l, "\pi_X"] \arrow[r, "\pi_Y"'] & Y
\end{tikzcd}$$

product

The product of groups G, H in Grp is defined via the product of the underlying sets. The coproduct however is not given by the disjoint union, but rather by the *free product* $G * H$. Meanwhile, in Ab products and coproducts coincide and form a *biproduct* $A \oplus B$. This shows that one has to be careful, since the same construction can take many forms, even if the category is defined on underlying sets.

(iii) In a category, which represents a preorder, the limit of a set of objects represents their meet, while the colimit of a set of objects represents their join. In particular such a category has all (small) colimits / limits, if and only if the preorder it represents has all joins / meets.

(iv) The limit of a diagram $X \overset{f}{\underset{g}{\rightrightarrows}} Y$ in a category $\mathcal{C}$ is called the **equalizer** $\mathrm{eq}(f,g)$ of f and g.

In the concrete cases of Set, Top, Grp or Ab it always exists and has $\{x \mid f(x) = g(x)\}$ as underlying set. In categories with zero morphisms, like Grp or Ab, the special case $\mathrm{eq}(f,0)$ is the **kernel** of f, often denoted as $\ker f$. Therefore, in some sense the equalizer of two morphisms f and g acts like their intersection.

These were only a tiny fraction of possible examples. What makes limits and colimits great is that they come with a *universal property*, i.e. a unique morphism satisfying some property. This universality opens the possibility to find further relations to other objects and constructions. For example the universality of the product assures that, if the corresponding products are present, there is a functor $- \times A\colon \mathcal{C} \longrightarrow \mathcal{C}$, sending any object X to $X \times A$ and a morphism $f\colon X \longrightarrow Y$ to the unique morphism $f \times A\colon X \times A \longrightarrow Y \times A$ obtained from the cone with legs $f\pi_X\colon X \times A \longrightarrow Y$ and $\pi_A\colon X \times A \longrightarrow A$ as depicted on the right.

$$
\begin{tikzcd}
 & X \times A \arrow[dl, "f\pi_X"'] \arrow[d, dashed, "f \times A"] \arrow[dr, "\pi_A"] & \\
Y & Y \times A \arrow[l, "\pi_Y"] \arrow[r, "\pi_A"'] & \mathrm{A}
\end{tikzcd}
$$

Similarly the universal properties can be used to assemble limits and colimits into a functor. This results in the following global characterization of limits and colimits in terms of an adjunction to the *constant diagram functor* $\varDelta\colon \mathcal{C} \longrightarrow [\![\mathcal{I},\mathcal{C}]\!]$. Sending each object X in $\mathcal{C}$ to its constant functor $\varDelta X\colon \mathcal{I} \longrightarrow \mathcal{C}$ in $[\![\mathcal{I},\mathcal{C}]\!]$ and a morphism $f\colon X \longrightarrow Y$ in $\mathcal{C}$ to the obvious natural transformation $\varDelta f\colon \varDelta X \implies \varDelta Y$ with components $(\varDelta f)_I = f$, we observe that $\varDelta$ is in fact a full embedding as long as $\mathcal{I}$ is not the empty category. The following is a standard result, commonly proofed using a characterization of adjoint functors, which we did not introduce. Hence we give an ad hoc proof.

Theorem 1.1.21

Let $\mathcal{C}$ and $\mathcal{I}$ be categories. Then $\mathcal{C}$ has all limits / all colimits of shape $\mathcal{I}$ if and only if the constant diagram functor $\varDelta\colon \mathcal{C} \longrightarrow [\![\mathcal{I},\mathcal{C}]\!]$ has a right adjoint / left adjoint.

$$
\begin{tikzcd}
\mathcal{C} \arrow[r, "\varDelta" description] \arrow[r, phantom, shift left=3, "\perp"] \arrow[r, phantom, shift right=3, "\perp"] & {[\![\mathcal{I},\mathcal{C}]\!]} \arrow[l, bend right=60, "\mathrm{colim}"'] \arrow[l, bend left=60, "\lim"]
\end{tikzcd}
$$

Graphically the theorem reduces to the diagram on the right.

Proof We will omit the proof of the statement about colimits, since it can be explained by invoking duality and is of less importance for this work.

" $\implies$ ":
Suppose every diagram $\mathcal{D}\colon \mathcal{I} \longrightarrow \mathcal{D}$ has a limit. We seek to construct a limit functor $\lim\colon [\![\mathcal{I},\mathcal{C}]\!] \longrightarrow \mathcal{C}$, which is right adjoint to $\varDelta\colon \mathcal{C} \longrightarrow [\![\mathcal{I},\mathcal{C}]\!]$, what means there are natural transformations $\eta\colon 1_{\mathcal{C}} \implies \lim \varDelta$ and $\varepsilon\colon \varDelta \lim \implies 1_{[\![\mathcal{I},\mathcal{C}]\!]}$ satisfying the triangle identities

$$
\begin{tikzcd}
\lim \arrow[r, Rightarrow, "\eta \lim"] \arrow[dr, equal] & \lim \varDelta \lim \arrow[d, Rightarrow, "\lim \varepsilon"] \\
 & \lim
\end{tikzcd}
\quad \text{and} \quad
\begin{tikzcd}
\varDelta \arrow[r, Rightarrow, "\varDelta\eta"] \arrow[dr, equal] & \varDelta \lim \varDelta \arrow[d, Rightarrow, "\varepsilon\varDelta"] \\
 & \varDelta
\end{tikzcd}
.
$$

First we define the functor $\lim\colon [\![\mathcal{I},\mathcal{C}]\!] \longrightarrow \mathcal{C}$ by sending a diagram $\mathcal{D}$ in $[\![\mathcal{I},\mathcal{C}]\!]$ to the limit apex $\lim \mathcal{D}$. A natural transformation $\alpha\colon \mathcal{D} \implies \mathcal{D}'$ in $[\![\mathcal{I},\mathcal{C}]\!]$ is mapped to the unique morphism of cones over $\mathcal{D}'$ given by

$$\begin{tikzcd}
\lim \mathcal{D} \arrow[rrr, dashed, "\lim \alpha"] \arrow[dr, "\lambda_I"] \arrow[ddr, "\lambda_J"'] & & & \lim \mathcal{D}' \arrow[dl, "\lambda'_I"'] \arrow[ddl, "\lambda'_J"] \\
& \mathcal{D}I \arrow[r, "\alpha_I"] \arrow[d] & \mathcal{D}'I \arrow[d] & \\
& \mathcal{D}J \arrow[r, "\alpha_J"] & \mathcal{D}'J &
\end{tikzcd}$$

writing λ and λ' for the corresponding limit cones. The universal property ensures that this construction is in fact functorial.

We define $\varepsilon\colon \Delta \lim \implies 1_{[\![\mathcal{I},\mathcal{C}]\!]}$ componentwise in such a way that $\varepsilon_{\mathcal{D}}\colon \Delta(\lim \mathcal{D}) \implies \mathcal{D}$ is given by the limit cone $\lambda\colon \Delta(\lim \mathcal{D}) \implies \mathcal{D}$ over $\mathcal{D}$. The construction of $\lim \alpha$ ensures that ε indeed is a natural transformation.

Recall that an arbitrary morphism $f\colon X \longrightarrow Y$ in $\mathcal{C}$ bijectively corresponds to a natural transformation $\Delta f\colon \Delta X \implies \Delta Y$, whose legs are all given by f. For the sake of readability we will from now on simply use f instead of writing $(\Delta f)_I$.

Finally we define $\eta\colon 1_{\mathcal{C}} \implies \lim \Delta$ componentwise with $\eta_X\colon X \longrightarrow \lim \Delta X$ being the unique morphism of cones as stated in the diagram on the right, where λ_X is the morphism with $\Delta\lambda_X = \lambda\colon \Delta(\lim X) \implies \Delta X$ being the limit cone.

$$\begin{tikzcd}
& X \arrow[d, dashed, "\eta_X"] \arrow[ddl, bend right, "1_X"'] \arrow[ddr, bend left, "1_X"] & \\
& \lim \Delta X \arrow[dl, "\lambda_X"'] \arrow[dr, "\lambda_X"] & \\
\Delta X(I) \arrow[rr, "1_X"'] & & \Delta X(J)
\end{tikzcd}$$

We wish to show that η is a natural transformation. To derive this observe that in the diagram

$$\begin{tikzcd}
X \arrow[r, "f"] \arrow[d, "\eta_X"] \arrow[dd, bend right=60, "1_X"'] & Y \arrow[d, "\eta_Y"'] \arrow[dd, bend left=60, "1_Y"] \\
\lim \Delta X \arrow[r, "\lim \Delta f"] \arrow[d, "\lambda_X"] & \lim \Delta Y \arrow[d, "\lambda_Y"'] \\
\Delta X(I) \arrow[r, "f"] & \Delta Y(I)
\end{tikzcd}$$

everything besides the upper square commutes by definition. Thus we have

$$\lambda_Y \lim \Delta f \eta_X = f = \lambda_Y \eta_Y f,$$

so by the universal property of $\lim(\Delta X)$ we have the naturality $\eta_Y f = \lim(\Delta f)\eta_X$.

Moreover, applying Δ to the defining equation of η, we immediately get the second triangle identity. Hence it is left to show that η and ε satisfy the first triangle identity. This will be done componentwise, so let $\mathcal{D}$ be a diagram in $[\![\mathcal{I},\mathcal{C}]\!]$. For any leg $\lambda_I = (\varepsilon_{\mathcal{D}})_I$ of the limit cone λ over $\mathcal{D}$ we have the commuting diagram

$$\begin{tikzcd}
\lim \mathcal{D} \arrow[d, "\eta_{\lim \mathcal{D}}"] \arrow[dd, bend right=60, "1_X"'] & \\
\lim \Delta \lim \mathcal{D} \arrow[r, "\lim \varepsilon_{\mathcal{D}}"] \arrow[d, "\lambda_{\lim \mathcal{D}}"] & \lim \mathcal{D} \arrow[d, "\lambda_I"] \\
\Delta \lim \mathcal{D}(I) \arrow[r, "\lambda_I"'] & \mathcal{D}I
\end{tikzcd}$$

which by the universal property of $\lim \mathcal{D}$ implies $\lim \varepsilon_{\mathcal{D}} \eta_{\lim \mathcal{D}} = 1_{\lim \mathcal{D}}$.

" $\Longleftarrow$ ":
Conversely, suppose we have a functor $\lim\colon [\![\mathcal{I}, \mathcal{C}]\!] \longrightarrow \mathcal{C}$, which is right adjoint to the functor $\Delta\colon \mathcal{C} \longrightarrow [\![\mathcal{I}, \mathcal{C}]\!]$ with given unit $\eta\colon 1_{\mathcal{C}} \Longrightarrow \lim \Delta$ and counit $\varepsilon\colon \Delta \lim \Longrightarrow 1_{[\![\mathcal{I},\mathcal{C}]\!]}$.

As the proof of " $\Longrightarrow$ " suggests, the component $\varepsilon_{\mathcal{D}}\colon \Delta(\lim \mathcal{D}) \Longrightarrow \mathcal{D}$ of the counit will be the limit cone over the diagram $\mathcal{D}$ in $[\![\mathcal{I}, \mathcal{C}]\!]$. Hence we have to show that it satisfies the universal property of a limit cone.

Let $\alpha\colon \Delta X \Longrightarrow \mathcal{D}$ be an arbitrary cone over $\mathcal{D}$. The diagram

$$\begin{tikzcd}
\Delta X \arrow[d, "\Delta \eta_X"'] \arrow[dr, "1_{\Delta X}"] & \\
\Delta \lim \Delta X \arrow[r, "\varepsilon_{\Delta X}"] \arrow[d, "\Delta \lim \alpha"'] & \Delta X \arrow[d, "\alpha"] \\
\Delta \lim \mathcal{D} \arrow[r, "\varepsilon_{\mathcal{D}}"] & \mathcal{D}
\end{tikzcd}$$

commutes by the second triangle identity and naturality of ε. In particular it establishes $\lim \alpha \circ \eta_X$ as morphism of cones.

We are left to show that this morphism is unique. So let $\Delta f\colon \Delta X \longrightarrow \Delta(\lim \mathcal{D})$ be an arbitrary morphism satisfying $\varepsilon_{\mathcal{D}} \Delta f = \alpha$. We get the commuting diagram

$$\begin{tikzcd}
X \arrow[r, "\eta_X"] \arrow[d, "f"'] & \lim \Delta X \arrow[d, "\lim \Delta f"'] \arrow[dr, "\lim \alpha"] & \\
\lim \mathcal{D} \arrow[r, "\eta_{\lim \mathcal{D}}"] \arrow[rr, bend right, "1_{\lim \mathcal{D}}"'] & \lim \Delta \lim \mathcal{D} \arrow[r, "\lim \varepsilon_{\mathcal{D}}"] & \lim \mathcal{D}
\end{tikzcd}$$

where the left square commutes by naturality of η, the right triangle commutes by functoriality of lim and the curved region commutes by the first triangle identity. From this we can deduce $\Delta f = \Delta(\lim \alpha \circ \eta_X)$, i.e. $f = \lim \alpha \circ \eta_X$. $\square$

This theorem now tells us how to define the right notion of limit or colimit in other settings like 2-categories. Without going into detail [4] we state

Remark 1.1.22
Let $\mathcal{C}$ and $\mathcal{I}$ be 2-categories and $[\![\mathcal{I}, \mathcal{C}]\!]$ be the 2-category of 2-functors from $\mathcal{I}$ to $\mathcal{C}$, 2-natural transformations and *modifications* between them. Then $\mathcal{C}$ has all **2-limits** / **2-colimits** of shape $\mathcal{I}$ if and only if the constant diagram 2-functor $\Delta\colon \mathcal{C} \longrightarrow [\![\mathcal{I}, \mathcal{C}]\!]$ has a right 2-adjoint / left 2-adjoint in the 2-category Cat.

Similarly we have

Remark 1.1.23
Let $\mathcal{C}$ and $\mathcal{I}$ be 2-categories and $[\![\mathcal{I}, \mathcal{C}]\!]_{lax}$ be the 2-category of 2-functors from $\mathcal{I}$ to $\mathcal{C}$, lax natural transformations and *modifications* between them. Then $\mathcal{C}$ has all **lax limits** / **lax colimits** of

[4] In fact, due to our definition of 2-category being quite uncommon, we cannot cite the following results in the form presented here. Instead we refer to [Rie14], where a similar result is given in the form of *conical $\mathcal{V}$-enriched limits*, which has to be adapted into the Cat-enriched case.

shape $\mathcal{I}$ if and only if the constant diagram 2-functor $\Delta\colon \mathcal{C} \longrightarrow [\![\mathcal{I},\mathcal{C}]\!]_{lax}$ has a right 2-adjoint / left 2-adjoint in the 2-category Cat_{lax} of 2-categories, 2-functors and lax natural transformations.

This gives a categorically nice, global notion of limit and colimit, from which a local notion (the universal property of a limit / colimit of a specific diagram) can be recovered. We will see this happening in the form of restriction limits and restriction colimits at the end of the following section.

1.2 Restriction Categories

Let us consider a modification of the category Set, where we allow functions to be undefined on some inputs. This kind of function naturally arises in the context of computability theory for example, since Turing machines do not have to halt on every input. Similarly in calculus a function does not need to be continuous or differentiable in every point.

We can formalize this kind of function by adding a dedicated point $\uparrow$ to every set and defining a *partial function* $f\colon X \rightharpoonup Y$ to be a function $f\colon X \sqcup \{\uparrow\} \longrightarrow Y \sqcup \{\uparrow\}$, which satisfies $f(\uparrow) =\uparrow$. For some x in X we say that $f(x)$ is *undefined*, if $f(x) =\uparrow$, and is *defined*, if $f(x) \neq\uparrow$, which we will denote by $f(x) =\downarrow$ as long as the value of $f(x)$ is not of interest. The *domain of definition* of f is the set of those x, on which $f(x)$ is defined. As we still have identities, composition and associativity of composition, partial functions assemble into the category Par.

There have been many attempts to define partiality in the categorical setting. The problem is how to specify the domain of definition in a suitable way. One common approach is to model a subset $A \subseteq X$ as an embedding $A \hookrightarrow X$. However, to arrive at a useful definition of category of partial morphisms this view requires rather crude constructions and the existence of certain limits. In this work we will use another approach, which Cockett and Lack worked out in [CL02]. In fact, as they show in this paper, both approaches are equivalent.

In this section we first define these so called *restriction categories* and inspect various generalizations of phenomena occurring in the category Par, such as completely undefined functions or intersections of partial functions. Then we consider, how restriction categories are related, by defining *restriction functors* and *restriction transformations*. We complete the introduction into restriction categories by dissecting the categorical notion of *restriction limits* rigorously and connecting it with other properties of restriction categories.

§ Definition and Enrichment

Definition 1.2.1

A **restriction category** is a category $\mathcal{C}$ equipped with an **restriction operator** assigning to each morphism $f\colon X \longrightarrow Y$ a **restriction domain** or **restriction idempotent** $\overline{f}\colon X \longrightarrow X$, which satisfies for arbitrary morphisms $f, f_1, f_2\colon X \longrightarrow Y$, $g\colon Y \longrightarrow Z$

$$\begin{aligned} f\overline{f} &= f && (R1)\\ \overline{f_2}\,\overline{f_1} &= \overline{f_1}\,\overline{f_2} && (R2)\\ \overline{f_2\overline{f_1}} &= \overline{f_2}\,\overline{f_1} && (R3)\\ \overline{g}\,f &= f\overline{gf} && (R4) \end{aligned}$$

A morphism f is called **total**, if $\overline{f} = 1_X$. In diagrams total morphisms will be drawn thickened.

The category Par is a restriction category. For an arbitrary partial function $f\colon X \rightharpoonup Y$ the restriction idempotent is given by

$$\overline{f}\,(x) = \begin{cases} x & \text{if } f(x) = \downarrow \\ \uparrow & \text{if } f(x) = \uparrow \end{cases}.$$

In words, the restriction idempotents are partial identity functions, which are defined precisely when the partial functions are. One can also see them as some sort of characteristic function for the domain of definition and thus identify them with subsets.

In order to avoid confusion with the words *domain* and *codomain* occurring in usual category theory, we will prefer the name *restriction idempotent* over the name *restriction domain*. The former name is not without reason as the following remark shows.

Remark 1.2.2

Let $\mathcal{C}$ be a restriction category. Then the following holds

(i) For any $f\colon X \longrightarrow Y$ the restriction idempotent really is an idempotent, as $\overline{f}\,\overline{f} \overset{(R3)}{=} \overline{\overline{f}f} \overset{(R1)}{=} \overline{f}$ shows. Thus we could redefine restriction idempotents to be endomorphisms e satisfying $\overline{e} = e$.

(ii) Every monomorphism is total, in particular isomorphisms are total.

We will write $\mathcal{O}(X)$ for the set of restriction idempotents on an object X and $\mathcal{O}(\mathcal{C})$ for the class of all restriction idempotents.

Note that for an idempotent to be a restriction idempotent is a nontrivial assumption. For example in Par a constant endofunction on a set with more than one element is an idempotent, but not a restriction idempotent.

In the following lemma we will see that the definition of the restriction operator allows us to detect more structure on the homsets of a restriction category. The intuition behind this comes from Par. Here we have a canonical order on partial functions given by the inclusion of graphs. More specific, in Par we can define a partial order on parallel partial functions f_1, f_2 by saying $f_1 \subseteq f_2$ if and only if when f_1 is defined, so is f_2 and they agree. This can be generalized to restriction categories.

Lemma 1.2.3

Any restriction category $\mathcal{C}$ has a poset-structure on its hom-sets, which is compatible with composition. In particular it is a 2-category.

Proof Given morphisms $f_1, f_2\colon X \longrightarrow Y$ define $f_1 \subseteq f_2 \;:\Longleftrightarrow\; f_2\overline{f_1} = f_1$ to obtain a preorder on the homsets. As $f_1 \subseteq f_2$ and $f_2 \subseteq f_1$ implies $\overline{f_1} = \overline{f_1}\,\overline{f_2} = \overline{f_2}$ we actually defined a poset-structure on the homsets.

Now for arbitrary morphisms $k\colon W \longrightarrow X, h\colon Y \longrightarrow Z$ we have that $f_1 \subseteq f_2$ implies $hf_1k \subseteq hf_2k$, as the following calculation shows. With

$$\overline{f_1}\,\overline{hf_1} \overset{(R2)}{=} \overline{hf_1}\,\overline{f_1} \overset{(R3)}{=} \overline{hf_1\overline{f_1}} \overset{(R1)}{=} \overline{hf_1} \qquad (\star)$$

we can deduce

$$hf_2k\overline{hf_1k} \overset{(R4)}{=} hf_2\overline{hf_1}\,k \overset{\star}{=} hf_2\overline{f_1}\,\overline{hf_1}\,k = hf_1\overline{hf_1}\,k \overset{(R1)}{=} hf_1k$$

This means the poset structure is preserved by composition and we thus obtain a 2-category. □

Clearly this additional structure is the key to understand the differences between ordinary categories and restriction categories. For example this gives us the possibility to express some facts about restriction idempotents.

Lemma 1.2.4
For the set $\mathcal{O}(X)$ of restriction idempotents on an object X the following holds.

(i) $\mathcal{O}(X)$ is a meet-semilattice and has a greatest element.

(ii) For an arbitrary morphism $f\colon X \longrightarrow Y$ its restriction idempotent $\overline{f}$ is the least restriction idempotent $e \in \mathcal{O}(X)$ satisfying $fe = f$. In particular, postcomposing with a total morphism does not alter the domain.

Proof (i) The meet of arbitrary restriction idempotents $e_1, e_2 \in \mathcal{O}(X)$ is given by e_1e_2 and the greatest element is given by 1_X.

(ii) Applying the restriction operator on $fe = f$ we get $\overline{fe} = \overline{f}\,\overline{e} = \overline{f}$ i.e. $\overline{f} \subseteq \overline{e} = e$.

If g is a total morphism which is composable with f, we have

$$\overline{f}\,\overline{gf} = \overline{gf}\,\overline{f} = \overline{gf\overline{f}} = \overline{gf}\,,$$

so $\overline{gf} \subseteq \overline{f}$. On the other hand $f\overline{gf} = \overline{g}\,f = f$, which shows $\overline{f} \subseteq \overline{gf}$. □

If we think of restriction idempotents as subsets of X, having their meet translates into having the intersection of those subsets. This is the intuition behind some of the following reformulations of concepts found in the hom-posets of Par.

For example in Par we always have partial functions, which are undefined everywhere. Their behaviour can be formulated in terms of our general restriction structure. However in arbitrary restriction categories they might not exist, as arbitrary posets might or might not have a least element.

Definition 1.2.5
A restriction category $\mathcal{C}$ has **restriction zero morphisms**, if every poset $\mathcal{C}(X, Y)$ has a least element $\emptyset_{XY}\colon X \longrightarrow Y$ which is preserved by composition, which means that for arbitrary morphisms $k\colon W \longrightarrow X$, $h\colon Y \longrightarrow Z$ we have $h\emptyset_{XY}k = \emptyset_{WZ}$, and whose domain is $\overline{\emptyset_{XY}} = \emptyset_{XX}$.

In a category, which possesses a zero object in the sense of definition 1.1.18, a morphism factoring through the zero object is commonly called a *zero morphism*. Our notion of restriction zero morphism does not make use of this kind of object, so we proposed to call it *empty morphism* to avoid confusion. However, as we will see, in the presence of restriction zero objects our notion of restriction zero morphisms coincides with the classical notion. This and the fact that it is the common name in the literature about restriction categories made us keep the name.

Another thing we can do in Par is gluing together partial functions, supposed they are sufficiently compatible. In terms of our poset structure this translates into morphisms having joins.

Definition 1.2.6
In a restriction category with restriction zero morphisms two parallel morphisms $f_1, f_2\colon X \longrightarrow Y$ are **disjoint**, if $\overline{f_1}\,\overline{f_2} = \emptyset_{XX}$. The join of disjoint morphisms $f_1, f_2\colon X \longrightarrow Y$ will be denoted by $f_1 \sqcup f_2\colon X \longrightarrow Y$, if it exists and is compatible with composition in the sense that we have $k(f_1 \sqcup f_2)h = kf_1h \sqcup kf_2h$ for arbitrary morphisms $h\colon W \longrightarrow X$ and $k\colon Y \longrightarrow Z$.

Regarding arbitrary parallel morphisms $f_1, f_2\colon X \longrightarrow Y$ we say that f_1 and f_2 are **compatible**, if they satisfy $f_1\overline{f_2} = f_2\overline{f_1}$, or equivalently if $f_1\overline{f_2} \subseteq f_2$ or $f_2\overline{f_1} \subseteq f_1$. The join of two compatible morphisms is denoted by $f_1 \cup f_2\colon X \longrightarrow Y$, if it exists and is respected by composition, i.e. it satisfies $k(f_1 \cup f_2)h = kf_1h \cup kf_2h$ like above.

A restriction category, in which any two disjoint / compatible parallel morphisms have a join, which is preserved by composition, is said to have **disjoint / compatible joins**.

The notion of disjointness can be used to define a complement of sets, regarded as restriction idempotents. This is of great importance for later applications.

Definition 1.2.7

In a restriction category with restriction zero morphisms a restriction idempotent $e \in \mathcal{O}(X)$ has a **complement**, if there exists another restriction idempotent $e' \in \mathcal{O}(X)$ such that $ee' = \emptyset_{XX}$ and $e \sqcup e' = 1_X$. If every restriction idempotent has a complement, the restriction category is said to be **complemented**.

In Par we can intersect partial functions to obtain a partial function which is defined on those inputs where they agree. Again this translates directly into morphisms having meets. However, as is pointed out in [CGH12], a simple example in Par shows that we cannot expect meets to behave well under composition. Indeed, given suitable sets, let $f_1, f_2 \colon X \longrightarrow Y$ be two distinct constant functions and $k \colon Y \longrightarrow Z$ be another constant function. Then $k(f_1 \cap f_2) = \emptyset$, but $kf_1 \cap kf_2 \neq \emptyset$.

Definition 1.2.8

In a restriction category, the meet of parallel morphisms $f_1, f_2 \colon X \longrightarrow Y$ will be denoted by $f_1 \cap f_2 \colon X \longrightarrow Y$, if it exists and is compatible with pre-composition, meaning for any morphism $h \colon W \longrightarrow X$ holds $(f_1 \cap f_2)h = f_1 h \cap f_2 h$.

A restriction category, in which any two parallel morphisms have a meet, which is preserved by pre-composition, is said to have **meets**.

Last but not least, in Par we can interleave functions by picking values from one or the other when evaluating on the intersection of their domains. In our setting this can be expressed according to the following definition.

Definition 1.2.9

Let $\mathcal{C}$ be a restriction category. An **interleaving** of parallel morphisms $f_1, f_2 \colon X \longrightarrow Y$ is a morphism $f \colon X \longrightarrow Y$, which satisfies $\overline{f_1} \subseteq \overline{f}, \overline{f_2} \subseteq \overline{f}$ and $f = (f_1 \cap f) \cup (f_2 \cap f)$.

Before we start to inspect the categorical side of restriction categories we want to state some rules regarding the different structures just defined.

Lemma 1.2.10

Let $\mathcal{C}$ be a restriction category. Assuming the corresponding joins and meets exist, the following assertions hold.

(i) Given compatible morphisms f_1, f_2 we have $\overline{f_1 \cup f_2} = \overline{f_1} \cup \overline{f_2}$.

(ii) For any endomorphism $f \colon X \longrightarrow X$, the morphism $f \cap 1_X$ is a restriction idempotent.

(iii) For arbitrary parallel morphisms f, g_1, g_2 with g_1, g_2 being compatible we have the identity

$$f \cap (g_1 \cup g_2) = (f \cap g_1) \cup (f \cap g_2).$$

(iv) Any interleaving of compatible morphisms f_1, f_2 is their join $f_1 \cup f_2$.

Proof (cf. [Coc10] Lem. 5.25, Lem. 5.30, Lem. 7.2)

(i) Since $f_1 \cup f_2$ is a join, we have $(f_1 \cup f_2)\overline{f_i} = f_i$ for $i = 1, 2$, so $\overline{f_1 \cup f_2}\,\overline{f_i} = \overline{f_i}$ for $i = 1, 2$. Thus the join $\overline{f_1} \cup \overline{f_2}$ satisfies $\overline{f_1} \cup \overline{f_2} \subseteq \overline{f_1 \cup f_2}$. In particular it is a restriction idempotent.

On the other hand $(f_1 \cup f_2)(\overline{f_1} \cup \overline{f_2}) = ((f_1 \cup f_2)\overline{f_1}) \cup ((f_1 \cup f_2)\overline{f_2}) = f_1 \cup f_2$, hence by lemma 1.2.4(ii) we have $\overline{f_1 \cup f_2} \subseteq \overline{f_1} \cup \overline{f_2}$.

(ii) As $f \cap 1_X$ is a meet we have $f\overline{f \cap 1_X} = 1_X\overline{f \cap 1_X}$, so we can make the calculation

$$f \cap 1_X = (f \cap 1_X)\overline{f \cap 1_X} = f\overline{f \cap 1_X} \cap 1_X\overline{f \cap 1_X} = 1_X\overline{f \cap 1_X} = \overline{f \cap 1_X}.$$

(iii) We calculate

$$\begin{aligned} f \cap (g_1 \cup g_2) &= (f \cap (g_1 \cup g_2))\overline{g_1 \cup g_2} = (f \cap (g_1 \cup g_2))(\overline{g_1} \cup \overline{g_2}) \\ &= (f \cap (g_1 \cup g_2))\overline{g_1} \cup (f \cap (g_1 \cup g_2))\overline{g_2} \\ &= (f\overline{g_1} \cap (g_1 \cup g_2)\overline{g_1}) \cup (f\overline{g_2} \cap (g_1 \cup g_2)\overline{g_2}) \\ &= (f\overline{g_1} \cap g_1) \cup (f\overline{g_2} \cap g_2) = (f \cap g_1)\overline{g_1} \cup (f \cap g_2)\overline{g_2} \\ &= (f \cap g_1) \cup (f \cap g_2). \end{aligned}$$

(iv) Let f be an interleaving of f_1 and f_2, i.e. $\overline{f}\,\overline{f_i} = \overline{f_i}$ for $i = 1, 2$ and $f = (f_1 \cap f) \cup (f_2 \cap f)$. Clearly $f \subseteq (f_1 \cup f_2)$, since $(f_i \cap f) \subseteq f_i \subseteq f_1 \cup f_2$ for $i = 1, 2$. On the other hand $f_1 \subseteq f$, since

$$\begin{aligned} f_1 = f_1\overline{f} &= f_1(\overline{f_1 \cap f} \cup \overline{f_2 \cap f}) = f_1\overline{f_1 \cap f} \cup f_1\overline{f_2 \cap f} \\ &= (f_1 \cap f) \cup f_1\overline{f_2}\,\overline{f_2 \cap f} = (f_1 \cap f)\overline{f_1} \cup f_2\overline{f_2 \cap f}\,\overline{f_1} \\ &= ((f_1 \cap f) \cup (f_2 \cap f))\overline{f_1} = f\overline{f_1}, \end{aligned}$$

and similarly $f_2 \subseteq f$, so $f_1 \cup f_2 \subseteq f$. □

§ Restriction Functors and Restriction Transformations

Definition 1.2.11

A functor $\mathcal{F}\colon \mathcal{C} \longrightarrow \mathcal{D}$ between restriction categories, which satisfies $\mathcal{F}\overline{f} = \overline{\mathcal{F}f}$ for any morphism $f \in \operatorname{Mor}\mathcal{F}$, is a **restriction functor**.

Any restriction functor is a 2-functor, because it preserves the poset structure induced by the restriction operator. However, not every 2-functor between restriction categories is a restriction functor. This is due to the fact that the 2-functor does not need to preserve least elements.

We will now give an example for a restriction functor, which will be very important in the following work. Recall that restriction idempotents can be thought of as the subsets of an object X, which arise as the domain of a partial function. In the case that such a restriction idempotent $e\colon X \longrightarrow X$ splits, i.e. we have $e = mr$ for $m\colon Y \longrightarrow X$, $r\colon X \longrightarrow Y$ with $rm = 1_Y$, the object Y realizes the restriction domain e as a real domain in the category in some sense. The following construction makes this precise.

Theorem 1.2.12

Let $\mathcal{C}$ be a (small) restriction category and $\mathcal{E}$ be a set of idempotents containing all identities of objects in $\mathcal{C}$. Then there is a restriction category $\operatorname{Split}_{\mathcal{E}}(\mathcal{C})$ such that $\mathcal{C}$ embeds into $\operatorname{Split}_{\mathcal{E}}(\mathcal{C})$ via a full restriction functor and all idempotents in $\mathcal{E}$ are split regarding this embedding.

We will denote the particular case of taking all idempotents in $\mathcal{C}$ by $\operatorname{Split}\mathcal{C}$. Further, a restriction category, in which all restriction idempotents split, is called a **split restriction category**.

Proof (cf. [CL02] Prop. 2.26)

The category $\operatorname{Split}_{\mathcal{E}}(\mathcal{C})$ consists of

objects: the idempotents of $\mathcal{E}$

morphisms: triples $(e_X, f, e_Y)\colon e_X \longrightarrow e_Y$, where e_X, e_Y are idempotents in $\mathcal{E}$ on objects X, Y respectively and $f\colon X \longrightarrow Y$ is a morphism in $\mathcal{C}$ satisfying $e_Y f e_X = f$

and the composition is defined via $(e_Y, g, e_Z)(e_X, f, e_Y) \coloneqq (e_X, gf, e_Z)$. It is easy to check that the composition is well defined and its associativity is inherited from the category $\mathcal{C}$. The identity on an idempotent e in $\mathcal{E}$ is given by (e, e, e). Thus $\operatorname{Split}_{\mathcal{E}}(\mathcal{C})$ indeed forms a category.

Now the restriction structure is defined to be given by $\overline{(e_X, f, e_Y)} \coloneqq (e_X, \overline{f}\,e_X, e_X)$. This is well defined, as

$$e_X\overline{f}\,e_X e_X = e_X\overline{f}\,e_X = e_X e_X\overline{f e_X} = e_X\overline{f e_X} = \overline{f}\,e_X. \tag{$\star$}$$

That this definition satisfies the restriction axioms (R1) to (R4) can be deduced from the corresponding equations (suppressing powers of idempotents)

$$f\overline{f}\,e_X = f e_X \overset{(\star\star)}{=} f \tag{1}$$

$$\overline{f_2}\,e_X\overline{f_1}\,e_X \overset{(\star)}{=} \overline{f_2}\,\overline{f_1}\,e_X = \overline{f_1}\,\overline{f_2}\,e_X = \overline{f_1}\,e_X\overline{f_2}\,e_X \tag{2}$$

$$\begin{aligned}\overline{f_2\overline{f_1}\,e_X}\,e_X &= \overline{f_2 e_X\overline{f_1 e_X}}\,e_X = \overline{f_2 e_X}\,\overline{f_1 e_X}\,e_X \overset{(\star)}{=} e_X\overline{f_2 e_X}\,\overline{f_1 e_X}\,e_X \\ &= \overline{f_2}\,e_X\overline{f_1 e_X}\,e_X = \overline{f_2}\,e_X\overline{f_1}\,e_X\end{aligned} \tag{3}$$

$$\overline{g}\,e_Y f = e_Y f\overline{g e_Y f} \overset{(\star\star)}{=} f e_X\overline{g f e_X} = f\overline{gf}\,e_X, \tag{4}$$

where we used the identity

$$e_Y f = e_Y e_Y f e_X = e_Y f e_X = f = e_Y f e_X e_X = f e_X. \tag{$\star\star$}$$

Finally the category $\mathcal{C}$ can be identified with the full subcategory of $\mathrm{Split}_{\mathcal{E}}(\mathcal{C})$ on the identities. By construction this defines a full embedding which preserves the restriction operator. Further, any idempotent $e\colon X \longrightarrow X \mathrel{\hat{=}} (1_X, e, 1_X)\colon 1_X \longrightarrow 1_X$ splits into an embedding $(1_X, e, e)\colon e \longrightarrow 1_X$ and a retraction $(e, e, 1_X)\colon 1_X \longrightarrow e$. $\square$

As usual, the construction in the preceding theorem is in fact 2-functorial, as we will show now.

Lemma 1.2.13

Let $\mathcal{C}, \mathcal{D}$ be two restriction categories and $\mathcal{E}$ and $\mathcal{I}$ be appropriate sets of idempotents in $\mathcal{C}$ respectively $\mathcal{D}$.

Any restriction functor $\mathcal{F}\colon \mathcal{C} \longrightarrow \mathcal{D}$, which satisfies $\mathcal{F}(\mathcal{E}) \subseteq \mathcal{I}$, extends to a restriction functor $\widetilde{\mathcal{F}}\colon \mathrm{Split}_{\mathcal{E}}\,\mathcal{C} \longrightarrow \mathrm{Split}_{\mathcal{I}}\,\mathcal{D}$. Similarly a natural transformation $\alpha\colon \mathcal{F} \Longrightarrow \mathcal{G}$ between restriction functors extends to a natural transformation $\widetilde{\alpha}\colon \widetilde{\mathcal{F}} \Longrightarrow \widetilde{\mathcal{G}}$ between the extensions.

Proof (cf. [Bor94] Prop. 6.5.9)

It is obvious that the assignment

$$\begin{array}{rcl} \widetilde{\mathcal{F}}:\ \mathrm{Split}_{\mathcal{E}}\,\mathcal{C} & \longrightarrow & \mathrm{Split}_{\mathcal{I}}\,\mathcal{D} \\ e & \longmapsto & \mathcal{F}e \\ (e_X, f, e_Y) & \longmapsto & (\mathcal{F}e_X, \mathcal{F}f, \mathcal{F}e_Y) \end{array}$$

defines a restriction functor, which extends $\mathcal{F}$. Similarly the natural transformation α extends to $\widetilde{\alpha}$ with components $\widetilde{\alpha}_e \coloneqq (\mathcal{F}e, (\mathcal{G}e)\alpha_X(\mathcal{F}e), \mathcal{G}e)$, where e is an idempotent on X and contained in $\mathcal{E}$. The naturality of this construction can easily be deduced by using the naturality of α and the equation $(\star\star)$ in the proof of the preceding theorem. $\square$

We will use the functoriality of idempotent splitting, as well as the following definition in theorem 2.2.8.

Definition 1.2.14

Two (restriction) categories $\mathcal{C}$ and $\mathcal{D}$ are **Morita-equivalent**, if there are sets of idempotents $\mathcal{E}$ and $\mathcal{I}$ in $\mathcal{C}$ respectively $\mathcal{D}$, such that the categories $\mathrm{Split}_{\mathcal{E}}(\mathcal{C})$ and $\mathrm{Split}_{\mathcal{I}}(\mathcal{D})$ are equivalent as categories. In the spirit of the preceding lemma, $\mathcal{C}$ and $\mathcal{D}$ are Morita-equivalent if and only if $\mathrm{Split}\,\mathcal{C}$ and $\mathrm{Split}\,\mathcal{D}$ are equivalent as categories.

Obviously Morita equivalence is an even weaker notion of categories being the same as that of equivalence or isomorphism of categories. However, we will see that it is appropriate in the setting of restriction categories.

Definition 1.2.15

Let $\mathcal{F}, \mathcal{G} \colon \mathcal{C} \longrightarrow \mathcal{D}$ be two restriction functors between restriction categories $\mathcal{C}$ and $\mathcal{D}$. A **restriction transformation** α from $\mathcal{F}$ to $\mathcal{G}$ is a natural transformation $\alpha \colon \mathcal{F} \implies \mathcal{G}$ whose components are total. Similarly, lax natural transformations with total components are called **lax restriction transformations**.

Again restriction transformations are in particular 2-natural transformations, because the required equation is fulfilled by definition.

Requiring the components of restriction transformations to be total may seem a bit arbitrary. However it gives us a clear distinction of restriction categories from ordinary categories. For example it offers a nice description, of how the 2-category rCat of (small) restriction categories, restriction functors and restriction transformations live over the 2-category Cat of categories, functors and natural transformations. We sketch the line of argumentation found in [CL02].

On the one hand any restriction category gives rise to a subcategory of total morphisms. Considering the action of restriction functors and restriction transformations on those subcategories one notices that they act precisely like ordinary functors and natural transformations. That is to say we have a 2-functor $\mathsf{Tot} \colon \mathsf{rCat} \longrightarrow \mathsf{Cat}$.

On the other hand each category can be considered as restriction category with trivial restriction structure, i.e. all restriction idempotents are identities. Again, restriction functors and restriction transformations act on trivial restriction categories like ordinary functors and natural transformations, so we have a 2-functor $\mathsf{Triv} \colon \mathsf{Cat} \longrightarrow \mathsf{rCat}$.

The connection between Cat and rCat can now be made precise by stating that we have a 2-adjunction

$$\begin{tikzcd} \mathsf{Cat} \arrow[r, bend left, "\mathsf{Triv}"] \arrow[r, phantom, "\perp"] & \mathsf{rCat}, \arrow[l, bend left, "\mathsf{Tot}"] \end{tikzcd}$$

where unit and counit are given by the corresponding identity transformations. The same observations give us a 2-adjunction

$$\begin{tikzcd} \mathsf{Cat} \arrow[r, bend left, "\mathsf{Triv}"] \arrow[r, phantom, "\perp"] & \mathsf{rCat}_{lax}, \arrow[l, bend left, "\mathsf{Tot}"] \end{tikzcd}$$

for the 2-category $\mathsf{rCat}_{\mathrm{lax}}$ of (small) restriction categories, restriction functors and lax restriction transformations.

§ Restriction Limits and Restriction Colimits

Finally we consider the notions of colimits and limits appropriate for restriction categories. As promised at the end of the previous section we will try to adapt the global characterization of limits and colimits to this setting. As the definition of a restriction category is not self dual in the sense that the opposite category of a restriction category is again a restriction category we cannot expect the definitions to dualize. Hence we will consider colimits and limits separately. We will discuss restriction colimits rather briefly, because they are not very important for this work. Then a thorough discussion of restriction limits will follow.[5]

The global characterization of limits and colimits involves the functor category $[\![\mathcal{I}, \mathcal{C}]\!]$ between categories $\mathcal{I}$ and $\mathcal{C}$. Since we want to use it in the setting of restriction categories, it is natural to ask what restriction structure the functor category $[\![\mathcal{I}, \mathcal{R}]\!]$ of a restriction category $\mathcal{R}$ should have. The trivial restriction structure turns out to be inappropriate, which is plausible, since it completely ignores that of $\mathcal{R}$. Indeed we want to have the restriction structure on $[\![\mathcal{I}, \mathcal{R}]\!]$ to be induced by the restriction structure on $\mathcal{R}$. However the canonical, componentwise definition does not work in general. Hence we have to make some adjustments.

Remark 1.2.16

Let $\mathcal{R}$ be a restriction category and $\mathcal{I}$ be some arbitrary category. A functor $\mathcal{F}\colon \mathcal{I} \longrightarrow \mathcal{R}$ is said to be **total**, if it factors through $\operatorname{Tot}\mathcal{R}$, which means that all morphisms in its image are total.

Using this terminology the functor category $[\![\mathcal{I}, \mathcal{R}]\!]_{tot}$ of total functors and natural transformations between them turns into a restriction category with restriction structure defined componentwise.

Indeed, because the restriction axioms are trivially fulfilled componentwise, the only thing to check is that given a natural transformation $\alpha\colon \mathcal{F} \implies \mathcal{G}$ between total functors $\mathcal{F}\colon \mathcal{I} \longrightarrow \mathcal{R}$ and $\mathcal{G}\colon \mathcal{I} \longrightarrow \mathcal{R}$ the family $(\overline{\alpha_I})_{I\in\mathcal{I}}$ again is a natural transformation. Given an arbitrary morphism $f\colon I \longrightarrow J$ in $\mathcal{I}$ the calculation

$$\overline{\alpha_J}\,\mathcal{F}f = \mathcal{F}f\,\overline{\alpha_J\mathcal{F}f} = \mathcal{F}f\,\overline{\mathcal{G}f\alpha_I} = \mathcal{F}f\,\overline{\alpha_I}$$

shows just that.

Now we can state

Definition 1.2.17

Given a restriction category $\mathcal{R}$ and an arbitrary category $\mathcal{I}$ then $\mathcal{R}$ has all **restriction colimits** of shape $\mathcal{I}$, if the constant diagram functor $\Delta\colon \mathcal{R} \longrightarrow [\![\mathcal{I}, \mathcal{R}]\!]_{tot}$ has a left adjoint in rCat.

We will turn this categorically nice description into a local characterization by inspecting a given total diagram $\mathcal{D}\colon \mathcal{I} \longrightarrow \mathcal{R}$. Dualizing the proof of the global characterization of limits we derive that the colimit cone under $\mathcal{D}$ is a component of the adjunction's unit $\eta\colon 1_{[\![\mathcal{I},\mathcal{R}]\!]_{tot}} \implies \Delta\,\mathrm{colim}$, which is a restriction transformation. Hence it has total components, so all legs of the colimit cone have to be total. In retrospect this gives another reason why we have to take $[\![\mathcal{I}, \mathcal{R}]\!]_{tot}$. By requiring the colimit cone γ to have total legs, the equation $\gamma_J\mathcal{D}i = \gamma_I$ forces the morphism $\mathcal{D}i$ to be total by lemma 1.2.4(ii).

$$\begin{tikzcd}
\mathcal{D}I \arrow[rr] \arrow[dr, "\gamma_I"] \arrow[ddr, bend right, "\alpha_I"'] & & \mathcal{D}J \arrow[dl, "\gamma_J"'] \arrow[ddl, bend left, "\alpha_J"] \\
 & C \arrow[d, dashed, "\exists!"] & \\
 & X &
\end{tikzcd}$$

The universal property of a restriction colimit can now be derived analogously to that of an ordinary colimit. It is depicted on the right for some cone $\alpha\colon \mathcal{D} \implies \Delta X$ under $\mathcal{D}$ with the colimit cone denoted by $\gamma\colon \mathcal{D} \implies \Delta C$.

Another point of view is that a restriction colimit of a total diagram in $\mathcal{R}$ is an ordinary colimit in the underlying category of $\mathcal{R}$ with the additional requirement that all the legs of the colimit cone need to be total.

For this work only the following restriction colimit is of some relevance.

[5] While being interesting from a categorical point of view, this discussion will not contribute much to the theory of abstract computability developed in the following chapter. In fact, in this context memorizing definition 1.2.25, lemma 1.2.26 and lemma 1.2.28 does suffice.

Definition 1.2.18

An initial object in a restriction category $\mathcal{R}$ is a **restriction initial object**, since the uniqueness of its morphisms forces them to be total.

This is just one part of the following definition, which can also be found in [CL07].

Definition 1.2.19

A **restriction zero object** in a restriction category $\mathcal{R}$ is a object 0, which is both (restriction) initial and terminal (in the 1-categorical sense) and which satisfies that for each object X the unique morphism $0_X : X \longrightarrow X$ factoring through 0 is a restriction idempotent.

The relevance of restriction zero objects may be explained by the following lemma.

Lemma 1.2.20

Let $\mathcal{R}$ be a restriction category with restriction zero morphisms and let X be an object for which the restriction zero morphism $\emptyset_X : X \longrightarrow X$ splits into an embedding-retraction pair $(m, r) : Z \lhd X$. Then the object Z is a restriction zero object.

Conversely, if $\mathcal{R}$ has a restriction zero, then it has restriction zero morphisms.[6]

Proof By assumption we have an restriction zero morphism $\emptyset_{ZT}$ for any object T. Further given an arbitrary morphism $t : Z \longrightarrow T$ we have the equation

$$tr = trmr = tr\emptyset_X = \emptyset_{XT} = \emptyset_{ZT}r.$$

Using the fact that r is split epi we derive $t = \emptyset_{ZT}$ making Z (restriction) initial.

Again we have for every object T an restriction zero morphism $\emptyset_{TZ}$. Given any other morphism $f : T \longrightarrow Z$ we get the equation

$$m\emptyset_{TZ} = \emptyset_{TX} = \emptyset_X mf = mrmf = mf,$$

so by m being split mono we have $f = \emptyset_{TZ}$.

By definition of restriction zero morphisms we have $\emptyset_T = \emptyset_{ZT}\emptyset_{TZ}$ and $\overline{\emptyset_T} = \emptyset_T$, so Z indeed is a restriction zero object.

Conversely, let 0 be the restriction zero object and X, Y be two objects in $\mathcal{R}$. We show that 0_{XY}, i.e. the unique morphism from X to Y factoring through 0, is the restriction zero morphism from X to Y.

First we note that for arbitrary morphisms $h : W \longrightarrow X$ and $k : Y \longrightarrow Z$ we have the identity $k0_{XY}h = 0_{WZ}$, because the composite morphism factorizes through 0 and thus is uniquely specified. Furthermore this shows $\overline{0_{XY}} = \overline{0_X} = 0_X$, since by lemma 1.2.4(ii) $0_X\overline{0_{XY}} = 0_X$ implies $\overline{0_X} \subseteq \overline{0_{XY}}$ and $0_{XY}\overline{0_X} = 0_{XY}$ implies $\overline{0_{XY}} \subseteq \overline{0_X}$. Finally we have for any morphism $f : X \longrightarrow Y$ that $f\overline{0_{XY}} = f0_X = 0_{XY}$, so $0_{XY} \subseteq f$. □

The situation for restriction limits is more involved. If we try to define a restriction terminal object via a right adjoint in rCat the totality of the unique morphisms $!_X : X \longrightarrow 1$ forces the restriction structure on $\mathcal{R}$ to be trivial, because for any restriction idempotent e on X the equation $!_X e = !_X$ by lemma 1.2.4(ii) implies $e = \overline{!_X} = 1_X$. Similar problems arise when defining restriction products as right adjoints in rCat as is described in [CL07]. Hence we have to use more of the restriction structure and deviate to rCat_{lax}, suggesting to consider a form of lax limit.

Another hint that lax limits may be worth considering is the fact that the functor category $[\![\mathcal{I}, \mathcal{R}]\!]_{lax}$ of functors and lax natural transformations comes equipped with the canonical restriction structure, which the functor category $[\![\mathcal{I}, \mathcal{R}]\!]$ lacks.

[6] At this point making a distinction between the restriction zero morphisms, which we proposed to call empty morphisms, and the restriction zero morphisms, which factor through the restriction zero object, might clarify the statement.

Remark 1.2.21

Given a restriction category R and an arbitrary category $\mathcal{I}$ the functor category $[\![\mathcal{I},\mathcal{R}]\!]_{lax}$ of functors and lax natural transformations is again a restriction category with restriction structure defined componentwise.

Again, because the axioms of restriction are satisfied componentwise, the only thing to check is that given a lax natural transformation $\alpha\colon \mathcal{F} \Longrightarrow \mathcal{G}$ its restriction $\overline{\alpha}: \mathcal{F} \Longrightarrow \mathcal{F}$ again is a lax natural transformation. But given an arbitrary morphism $f\colon I \longrightarrow J$ in $\mathcal{I}$ and components $\alpha_I\colon \mathcal{F}I \longrightarrow \mathcal{G}I$ and $\alpha_J\colon \mathcal{F}J \longrightarrow \mathcal{G}J$ of α the calculation

$$\overline{\alpha_J}\,\mathcal{F}f = \mathcal{F}f\overline{\alpha_J\mathcal{F}f} \subseteq \mathcal{F}f\overline{\mathcal{G}f\alpha_I} \subseteq \mathcal{F}f\overline{\alpha_I}$$

gives us the desired lax naturality diagram

$$\begin{tikzcd}
\mathcal{F}I \arrow[r, "\overline{\alpha_I}"] \arrow[d, "\mathcal{F}f"'] \arrow[dr, phantom, "\circlearrowleft"] & \mathcal{F}I \arrow[d, "\mathcal{F}f"] \\
\mathcal{F}J \arrow[r, "\overline{\alpha_J}"'] & \mathcal{F}J.
\end{tikzcd}$$

Moreover the relation $\alpha \subseteq \beta$ in this restriction category coincides with the notion of a modification between (lax) natural transformations of definition 1.1.14.

With this at hand we can make the following definition.

Definition 1.2.22

Given a restriction category $\mathcal{R}$ and an arbitrary category $\mathcal{I}$ then $\mathcal{R}$ has all **restriction limits** of shape $\mathcal{I}$, if the constant diagram functor $\Delta\colon \mathcal{R} \longrightarrow [\![\mathcal{I},\mathcal{R}]\!]_{lax}$ has a right adjoint in rCat_{lax}.

Let us dissect how the restriction limit of a specific diagram $\mathcal{D}\colon \mathcal{I} \longrightarrow \mathcal{R}$ looks like. Again, comparing with the global characterization of limits, the limit cone over $\mathcal{D}$ is a component of the adjunction's counit $\varepsilon\colon \Delta\,\mathrm{colim} \Longrightarrow 1_{[\![\mathcal{I},\mathcal{R}]\!]_{lax}}$, which is a lax restriction transformation and thus has total components. Thus the (lax) limit cone $\varepsilon_{\mathcal{D}} = \lambda\colon \Delta\lim\mathcal{D} \Longrightarrow \mathcal{D}$ has to have total legs. However given an arbitrary morphism $f\colon I \longrightarrow J$ in $\mathcal{I}$, the equation $\lambda_J = \mathcal{D}f\lambda_I\overline{\lambda_J} = \mathcal{D}f\lambda_I$ forces the limit cone to be a restriction transformation.

Let $\eta\colon 1_{\mathcal{R}} \Longrightarrow \lim\Delta$ denote the adjunction's unit and let $\alpha\colon \Delta X \Longrightarrow \mathcal{D}$ be some lax cone. We want to know how the limit cone λ relates to α, so imitating the proof of the global characterization we get the following diagram in $[\![\mathcal{I},\mathcal{D}]\!]_{lax}$

$$\begin{tikzcd}
\Delta X \arrow[d, "\Delta\eta_X"'] \arrow[dr, "1_{\Delta X}"] & \\
\Delta\lim\Delta X \arrow[r, "\varepsilon_{\Delta X}"] \arrow[d, "\Delta\lim\alpha"'] \arrow[dr, phantom, "\circlearrowleft"] & \Delta X \arrow[d, "\alpha"] \\
\Delta\lim\mathcal{D} \arrow[r, "\varepsilon_{\mathcal{D}}"] & \mathcal{D}.
\end{tikzcd}$$

Again the triangle commutes because of one triangle identity, while the square is given by lax naturality of the counit ε. Recall that the restriction structure on $[\![\mathcal{I},\mathcal{R}]\!]_{lax}$ is defined componentwise. Hence, abbreviating $l := \lim\alpha \circ \eta_X$ and $\lambda = \varepsilon_{\mathcal{D}}$, this translates to having the inequality $\lambda_I l \subseteq \alpha_I$ for any object I in $\mathcal{I}$. In particular by lemma 1.2.4(ii) we have $\overline{l} \subseteq \overline{\alpha_I}$ for all objects I in $\mathcal{I}$. Even better, $\overline{l}$ is the meet of the restriction idempotents $\overline{\alpha_I}$ in $\mathcal{O}(X)$. We will use the symbol $\wedge$ instead of $\cap$, because we do not require this meet to be stable under precomposition. Note that in the case of $\mathcal{I}$ being finite this meet always exists and is given by $\bigwedge_{I \in \operatorname{Ob}\mathcal{I}} \overline{\alpha_I} = \overline{\alpha_{I_1}} \ldots \overline{\alpha_{I_n}}$.

Indeed we will now show that for any restriction idempotent $e\colon X \longrightarrow X$ with $e \subseteq \overline{\alpha_I}$ for all I in $\mathcal{I}$ we have $e \subseteq l$. Let $e\colon X \longrightarrow X$ be a restriction idempotent such that for all I in $\mathcal{I}$ we have $e \subseteq \overline{\alpha_I}$, or equivalently $\mathit{\Delta} e \subseteq \overline{\alpha}$. By applying the restriction functor lim and precomposing with η_X we get the inequality

$$(\lim \mathit{\Delta} e)\eta_X \subseteq (\lim \overline{\alpha}\,)\eta_X = \overline{\lim \alpha}\,\eta_X = \eta_X \overline{\lim \alpha \circ \eta_X} = \eta_X \overline{l}\,.$$

Meanwhile, since η_X is a restriction transformation we have $\eta_X e \subseteq (\lim \mathit{\Delta} e)\eta_X$. Thus we have $\eta_X e \subseteq \eta_X \overline{l}$, but by totality of η_X this translates to having $e \subseteq \overline{l}$.

Further given any morphism $f\colon X \longrightarrow \lim \mathcal{D}$ satisfying $\varepsilon_{\mathcal{D}} \circ \mathit{\Delta} f \subseteq \alpha$ we have the diagram

$$\begin{tikzcd}
X \arrow[r, "\eta_X"] \arrow[d, "f"'] \arrow[dr, phantom, "\subseteq"] & \lim \mathit{\Delta} X \arrow[d, "\lim \mathit{\Delta} f"'] \arrow[dr, "\lim \alpha"] \arrow[dr, phantom, "\subseteq", shift right=3] & \\
\lim \mathcal{D} \arrow[r, "\eta_{\lim \mathcal{D}}"] \arrow[rr, bend right, "1_{\lim \mathcal{D}}"'] & \lim \mathit{\Delta} \lim \mathcal{D} \arrow[r, "\lim \varepsilon_{\mathcal{D}}"] & \lim \mathcal{D},
\end{tikzcd}$$

where we use the lax naturality of η, a triangle identity and that lim is a restriction functor. Hence we get $f \subseteq l$, which makes l the maximal morphism satisfying $\varepsilon_{\mathcal{D}} \circ \mathit{\Delta} l \subseteq \alpha$ and as such unique.

The local characterization now reads as follows.

Remark 1.2.23

Let $\mathcal{R}$ be a restriction category and let $\mathcal{D}\colon \mathcal{I} \longrightarrow \mathcal{R}$ be a functor from some category $\mathcal{I}$ to $\mathcal{D}$. The **restriction limit** of the diagram $\mathcal{D}$ consists of, if it exists, an object L in $\mathcal{R}$ and a restriction transformation $\lambda\colon \mathit{\Delta} L \implies \mathcal{D}$, which satisfies the following condition.

$$\begin{tikzcd}
& X \arrow[dl, bend right, "\alpha_I"'] \arrow[dr, bend left, "\alpha_J"] \arrow[d, phantom, "\supseteq"] & \\
\mathcal{D}I \arrow[rr] & {} & \mathcal{D}J
\end{tikzcd}$$

For each lax cone $\alpha\colon \mathit{\Delta} X \implies \mathcal{D}$ as depicted on the left there is a unique morphism $l\colon X \longrightarrow L$, which is the maximum of all morphisms $f\colon X \longrightarrow L$ satisfying $\lambda \mathit{\Delta} f \subseteq \alpha$ (as depicted on the right) and whose domain is given by $\overline{l} = \bigwedge_{I \in \operatorname{Ob}\mathcal{I}} \overline{\alpha_I}$.

$$\begin{tikzcd}
& X \arrow[d, "f"] \arrow[ddl, bend right, "\alpha_I"'] \arrow[ddr, bend left, "\alpha_J"] & \\
\supseteq & L \arrow[dl, "\lambda_I"'] \arrow[dr, "\lambda_J"] & \subseteq \\
\mathcal{D}I \arrow[rr] & & \mathcal{D}J
\end{tikzcd}$$

It is important to note that we did not state the local characterization in a minimal way. In fact it suffices to have a unique morphisms for each lax cone, which makes the limit cone fit into it and whose restriction domain is the join of the restriction domains of the lax cone's legs [CL07]. Indeed, if $\lambda\colon \mathit{\Delta} L \implies \mathcal{D}$ is a total cone satisfying this seemingly weaker universal property and $\alpha\colon \mathit{\Delta} X \implies \mathcal{D}$ is a lax cone, we get a unique arrow $l\colon X \longrightarrow L$ such that $\lambda_I l \subseteq \alpha_I$ and its restriction domain $\overline{l}$ is given by the meet of all the restriction domains $\overline{\alpha_I}$. If $f\colon X \longrightarrow L$ is an arbitrary morphism satisfying $\lambda_I f \subseteq \alpha_I$ we wish to show $f \subseteq l$ i.e. $l\overline{f} = f$. We can deduce this by noting that both f and $l\overline{f}$ are morphisms, which satisfy the seemingly weaker universal property of the lax cone $\alpha\overline{f}\colon \mathit{\Delta} X \longrightarrow \mathcal{D}$ with components $\alpha_I \overline{f}$. To be precise, we clearly have that $\lambda_I f = \lambda_I f \overline{f} \subseteq \alpha_I \overline{f}$ and $\lambda_I l \overline{f} \subseteq \alpha_I \overline{f}$. Furthermore one calculates

$$\overline{l\overline{f}} = \overline{l}\,\overline{f} = \overline{l} \wedge \overline{f} = \Big(\bigwedge_{I \in \operatorname{Ob}\mathcal{I}} \overline{\alpha_I}\Big) \wedge \overline{f} = \bigwedge_{I \in \operatorname{Ob}\mathcal{I}} (\overline{\alpha_I} \wedge \overline{f}\,) = \bigwedge_{I \in \operatorname{Ob}\mathcal{I}} (\overline{\alpha_I}\,\overline{f}\,) = \bigwedge_{I \in \operatorname{Ob}\mathcal{I}} \overline{\alpha_I \overline{f}}\,,$$

as well as

$$\overline{f} = \bigwedge_{I \in \operatorname{Ob}\mathcal{I}} \overline{f} = \bigwedge_{I \in \operatorname{Ob}\mathcal{I}} \overline{\alpha_I}\,\overline{f} = \bigwedge_{I \in \operatorname{Ob}\mathcal{I}} \overline{\alpha_I \overline{f}}\,.$$

There is an interesting example, which relates the splitting of idempotents in theorem 1.2.12 with restriction limits in the sense that it turns it into the completion of a restriction category under certain limits.

Lemma 1.2.24

Let $\mathcal{R}$ be a restriction category and $f\colon X \longrightarrow Y$. The restriction limit of the diagram on f precisely is a splitting of the restriction idempotent $\overline{f}$.

Proof (cf. [CL07] Prop. 4.9)

Let the restriction limit be given by an object E and a total morphism $e\colon E \longrightarrow X$, where fe is total.

$$\begin{tikzcd}
 & X \arrow[d, dashed, "r"] \arrow[ddl, bend right, "1_X"'] \arrow[ddr, bend left, "f"] & \\
 & E \arrow[dl, "e"'] \arrow[dr] \arrow[l, phantom, "\supseteq"] \arrow[r, phantom, "\subseteq"] & \\
X \arrow[rr, "f"'] & & Y
\end{tikzcd}
\qquad\qquad
\begin{tikzcd}
 & E \arrow[d, dashed, "s"] \arrow[ddl, bend right, "e"'] \arrow[ddr, bend left, "fe"] & \\
 & E \arrow[dl, "e"'] \arrow[dr] \arrow[l, phantom, "\supseteq"] \arrow[r, phantom, "\subseteq"] & \\
X \arrow[rr, "f"'] & & Y
\end{tikzcd}$$

The universal property of the lax cone with legs 1_X and f gives us a unique morphism $r\colon X \longrightarrow E$ and the universal property of the lax cone with legs e and fe gives us a unique morphism $s\colon E \longrightarrow E$. Further the universal properties establish the equations

$$ere = \overline{er}\,e = \overline{f}\,e = e\overline{fe} = e \qquad\text{and}\qquad es = e\overline{es} = e\overline{fe} = e,$$

so by uniqueness of s we can deduce $s = re = 1_E$ and thus (e, r) is a splitting of $\overline{f}$.

Conversely let $(e, r)\colon E \lhd X$ be a embedding-retraction pair, which is a splitting of the restriction idempotent $\overline{f}$, and let $t_X\colon T \longrightarrow X$ and $t_Y\colon T \longrightarrow Y$ be the legs of a lax cone over f satisfying $ft_X \supseteq t_Y$. As $\overline{fe} = \overline{\overline{f}\,e} = \overline{ere} = \overline{e} = 1_X$ the pair e and fe are the legs of a restriction cone over f. The morphism $rt_X\overline{t_Y}$ has as domain $\overline{t_X}\,\overline{t_Y}$ and satisfies

$$\begin{tikzcd}
 & T \arrow[d, "rt_X\overline{t_Y}"] \arrow[ddl, bend right, "t_X"'] \arrow[ddr, bend left, "t_Y"] & \\
 & E \arrow[dl, "e"'] \arrow[dr] \arrow[l, phantom, "\supseteq"] \arrow[r, phantom, "\subseteq"] & \\
X \arrow[rr, "f"'] & & Y
\end{tikzcd}$$

$$ert_X\overline{t_Y} = \overline{f}\,t_X\overline{t_Y} = t_X\overline{ft_X}\,\overline{t_Y} \subseteq t_X$$

and

$$fert_X\overline{t_Y} = f\overline{f}\,t_X\overline{t_Y} \subseteq t_Y\overline{t_Y} = t_Y.$$

Now, as e is split mono, $rt_X\overline{t_Y}$ is unique with this property. $\square$

Even more important for our purposes are the following two restriction limits.

Definition 1.2.25

The restriction limit of the empty diagram is a **restriction terminal object**. Concretely a restriction terminal object in a restriction category $\mathcal{R}$ is an object 1 together with a unique total morphism $!_X\colon X \longrightarrow 1$ for every object X in $\mathcal{R}$ such that $!_1 = 1_1$ and for any morphism $f\colon X \longrightarrow Y$ in $\mathcal{R}$ we have the following diagram.

$$\begin{tikzcd}
X \arrow[dr, "!_X"] \arrow[dd, "f"'] & \\
 \arrow[r, phantom, "\subseteq"] & 1 \\
Y \arrow[ur, "!_Y"'] &
\end{tikzcd}$$

Similarly, the restriction limit of two objects X and Y is the **restriction product** of X and Y. This amounts to having an object $X \times Y$ in $\mathcal{R}$ and total **projections** $\pi_X : X \times Y \longrightarrow X$ and $\pi_Y : X \times Y \longrightarrow Y$ satisfying that for any pair of morphism $f : T \longrightarrow X$ and $g : T \longrightarrow Y$ there is a unique morphism $\langle f, g \rangle : T \longrightarrow X \times Y$ with restriction domain $\overline{\langle f, g \rangle} = \overline{f}\,\overline{g}$ such that we have the diagram

$$
\begin{tikzcd}
 & T \arrow[dl, "f"'] \arrow[d, "\langle f,g \rangle"] \arrow[dr, "g"] \arrow[dl, phantom, "\geq", near end, shift left=3] \arrow[dr, phantom, "\leq", near end, shift right=3] & \\
X & X \times Y \arrow[l, "\pi_X"] \arrow[r, "\pi_Y"'] & Y.
\end{tikzcd}
$$

A restriction category, which has a restriction terminal object as well as all binary restriction products, is called **cartesian restriction category**.

The universal property of the restriction initial object may need some explanation. In fact it has nothing to do with the local characterization we have worked out, since there are no (lax) cones over the empty diagram. Instead we note that we have an isomorphism of categories of the functor category $[\![\emptyset, \mathcal{R}]\!]_{lax} \cong \mathbb{1}$ to the terminal category. In this context the lax naturality of the unit of the adjunction $\Delta \dashv \lim$ gives the desired universal property.

Lemma 1.2.26

Let $\mathcal{R}$ be a cartesian restriction category.

(i) For any object A mapping X to $X \times A$ induces a restriction functor $- \times A : \mathcal{R} \longrightarrow \mathcal{R}$.

(ii) For any object X we have a unique isomorphism $X \cong 1 \times X$ respectively $X \cong X \times 1$.

(iii) If 0 is a restriction initial object in $\mathcal{R}$, then for any object X we have a unique isomorphism $0 \times X \cong 0$.

(iv) If $\mathcal{R}$ has restriction zero morphisms and there is an object X such that the restriction zero morphism $\emptyset_{1X}$ is total, then $\mathcal{R}$ is equivalent to the trivial category $\mathbb{1}$.

Proof (i) Let $f : X \longrightarrow Y$ be a morphism in $\mathcal{R}$. Then the universal property

$$
\begin{tikzcd}
 & X \times A \arrow[dl, "f\pi_X"'] \arrow[d, "f \times A"] \arrow[dr, "\pi_A"] \arrow[dl, phantom, "\geq", near end, shift left=3] \arrow[dr, phantom, "\leq", near end, shift right=3] & \\
Y & Y \times A \arrow[l, "\pi_Y"] \arrow[r, "\pi_A"'] & A
\end{tikzcd}
$$

gives us a unique morphism $f \times A := \langle f\pi_X, \pi_A \rangle$. In the case of $f = 1_X$ the universal property of the restriction product ensures that $- \times A$ respects identities. Furthermore for any other morphism $g : Y \longrightarrow Z$ the diagram

$$
\begin{tikzcd}
 & X \times A \arrow[dl, "f\pi_X"'] \arrow[d, "f \times A"] \arrow[dr, "\pi_A"] \arrow[dl, phantom, "\geq", near end, shift left=3] \arrow[dr, phantom, "\leq", near end, shift right=3] & \\
Y \arrow[d, "g"'] & Y \times A \arrow[l, "\pi_Y"'] \arrow[r, "\pi_A"] \arrow[dl, "g\pi_Y"'] \arrow[d, "g \times A"] \arrow[dr, "\pi_A"] \arrow[dl, phantom, "\geq", near end, shift left=3] \arrow[dr, phantom, "\leq", near end, shift right=3] & A \arrow[d, equal] \\
Z & Z \times A \arrow[l, "\pi_Z"'] \arrow[r, "\pi_A"] & A
\end{tikzcd}
$$

assembles into the universal property of the composite morphism gf, since

$$\begin{aligned}\overline{(g\times A)(f\times A)} &= \overline{\overline{g\times A}\, f\times A} = \overline{\overline{g\pi_Y}\, f\times A} = \overline{f\times A\,\overline{g\pi_Y f\times A}} \\ &= \overline{f\times A}\,\overline{g\pi_Y f\times A} = \overline{f\times A}\,\overline{gf\pi_X\overline{f\times A}} \\ &= \overline{gf\pi_X}\,\overline{f\times A} = \overline{gf\pi_X}\,\overline{f\pi_X} = \overline{gf\pi_X}\,.\end{aligned}$$

Thus by uniqueness $(g\times A)(f\times A) = (gf\times A)$ and $-\times A$ is functorial.

Similarly one shows $\overline{f}\times A = \overline{f\times A}\,.$

(ii) Taking the morphisms 1_X and $!_X$ as total cone over the product diagram with X and 1 we get a unique and total morphism $u\colon X\longrightarrow X\times 1$, which has π_X as a left inverse. On the other hand, taking π_X and $!_{X\times 1}$ as total cone over the same diagram we note that both $u\pi_X$ and $1_{X\times 1}$ satisfy the universal property, so by uniqueness they are equal and π_X is a right inverse.

(iii) By definition of a restriction zero there are unique total morphisms $?_X\colon 0\longrightarrow X$ and $?_{0\times X}\colon 0\longrightarrow 0\times X$. In particular $?_{0\times X}$ has to be the unique morphism given by the universal property of the cone over the product diagram of 0 and X with legs $?_X$ and 1_0. Again this shows that π_0 is left inverse to $?_{0\times X}$. As before the uniqueness in the universal property of the cone with legs π_0 and π_X ensures that π_0 also is a right inverse.

(iv) By assumption $\langle 1_1, 1_1\rangle$ is a splitting of the restriction idempotent $\emptyset_1$, since we have $\overline{\emptyset_{1X}} = 1_1 = \emptyset_1$. By lemma 1.2.20 this makes 1 into a restriction zero object. But now using (iii) we have $X\cong X\times 1 = X\times 0\cong 0$ via unique isomorphisms. From this the desired equivalence of the categories $\mathcal{R}$ and $\mathbb{1}$ can easily be derived. $\square$

It is common practice to suppress the isomorphism $1\times X\cong X$, so in the following we will often identify X with $1\times X$ without further mentioning.

There is one last thing we have to introduce for restriction categories. The following and more can be found in [CGH12].

Definition 1.2.27

An object X in a cartesian restriction category $\mathcal{R}$ is **discrete**, if the corresponding diagonal morphism $\varDelta_X\colon X\longrightarrow X\times X$ has *partial inverse*, i.e. if there exists a morphism $p_X\colon X\times X\longrightarrow X$ satisfying $p_X\varDelta_X = \overline{\varDelta_X} = 1_X$ and $\varDelta_X p_X = \overline{p_X}$. The restriction idempotent $\overline{p_X}$ is commonly denoted eq_X as it represents the range of the diagonal morphism and thus is a kind of *equality predicate* on the object $X\times X$.

A cartesian restriction category where every object is discrete will be called **discrete cartesian**.

Besides giving this equality morphism, which is interesting for its own, discrete cartesian restriction categories have a very nice characterization.

Proposition 1.2.28

A cartesian restriction category has meets if and only if it is a discrete cartesian restriction category.

Proof (cf. [CGH12] Prop. 2.20)

“$\Longrightarrow$”:

For any object X define $\mathrm{eq}_X \coloneqq \overline{\pi_1\cap\pi_2}\,.$ Then $\pi_1\,\mathrm{eq}_X = \pi_2\,\mathrm{eq}_X$ is a partial inverse to $\varDelta_X$, as

$$(\pi_1\cap\pi_2)\varDelta_X = \pi_1\varDelta_X\cap\pi_2\varDelta_X = 1_X\cap 1_X = 1_X = \overline{\varDelta_X}$$

and by the uniqueness in the universal property of the cone with legs $\pi_1\,\mathrm{eq}_X$ and $\pi_2\,\mathrm{eq}_X$ we have the equation

$$\varDelta(\pi_1\cap\pi_2) = \mathrm{eq}_X\,.$$

“ $\Longleftarrow$ ”:
Let $f_1, f_2 \colon X \longrightarrow Y$ be two arbitrary morphisms and define $f_1 \cap f_2 \coloneqq \pi\,\mathrm{eq}_Y\langle f_1, f_2\rangle$, where π is either of π_1 or π_2. As $\pi_1\,\mathrm{eq}_Y = \pi_2\,\mathrm{eq}_Y$ the definition is indeed independent from this choice and we will continue by using π_1 when possible.

We have for $i = 1, 2$ that

$$f_i\overline{f_1 \cap f_2} = f_i\overline{\pi_i\,\mathrm{eq}_X\langle f_1, f_2\rangle} = f_i\overline{\mathrm{eq}_Y\langle f_1, f_2\rangle} = \pi_i\langle f_1, f_2\rangle\overline{\mathrm{eq}_Y\langle f_1, f_2\rangle} = \pi_i\,\mathrm{eq}_Y\langle f_1, f_2\rangle,$$

hence $f_1 \cap f_2 \subseteq f_i$. Now let $g \colon X \longrightarrow Y$ be an arbitrary morphism satisfying $g \subseteq f_1$ and $g \subseteq f_2$. Then

$$(f_1 \cap f_2)\overline{g} = \pi_1\,\mathrm{eq}_Y\langle f_1, f_2\rangle\overline{g} = \pi_1\,\mathrm{eq}_Y\langle f_1\overline{g}, f_2\overline{g}\rangle = \pi_1\,\mathrm{eq}_Y\langle g, g\rangle = \pi_1\,\mathrm{eq}_Y\,\Delta_Y g = g,$$

so $g \subseteq f_1 \cap f_2$. Further we have for arbitrary morphisms $h \colon W \longrightarrow X$ the equation

$$(f_1 \cap f_2)h = \pi_1\,\mathrm{eq}_X\langle f_1, f_2\rangle h = \pi_1\,\mathrm{eq}_X\langle f_1 h, f_2 h\rangle = f_1 h \cap f_2 h.$$

Hence the restriction category has meets. □

We will need one last result regarding discrete objects.

Lemma 1.2.29
Let X be an discrete object in a cartesian restriction category $\mathcal{R}$. Then every retract Y of X is discrete.

Proof (cf. [Vin12] Prop. 3.2.1)
Let $(m, r) \colon Y \lhd X$ be an embedding-retraction pair and $p \colon X \times X \longrightarrow X$ a partial inverse to $\Delta_X \colon X \longrightarrow X \times X$. Considering the (non-commuting) diagram

$$\begin{tikzcd}
X \arrow[r, shift left, "\Delta_X"] \arrow[d, shift right, "r"'] & X \times X \arrow[l, shift left, "p"] \arrow[d, shift right, "r \times r"'] \\
Y \arrow[r, shift left, "\Delta_Y"] \arrow[u, shift right, "m"'] & Y \times Y, \arrow[l, shift left, "q"] \arrow[u, shift right, "m \times m"']
\end{tikzcd}$$

we define $q \coloneqq rp(m \times m)$. This is a partial inverse to Δ_Y, because

$$q\Delta_Y = rp(m \times m)\Delta_Y = rp\Delta_X m = 1_X$$

and

$$\begin{aligned}\Delta_Y q = \Delta_Y rp(m \times m) &= (r \times r)\Delta_X p(m \times m) = (r \times r)\overline{p}\,(m \times m)\\ &= (r \times r)(m \times m)\overline{p(m \times m)} = \overline{p(m \times m)} = \overline{rp(m \times m)} = \overline{q}\,.\end{aligned}$$

Thus Y is a discrete object. □

2 Categorical Computability and Complexity

2.1 Partial Combinatory Algebras

There have been many different approaches to specify what it means for a function to be computable. A prominent one is Church's λ-*calculus*, which expresses computability using so called λ-*terms*. It is a well known result that all λ-terms can be represented in terms of λ-applications of two *combinators* S and K (cf. [Odi89]) .

This fact leads to an algebraic model of computation, given by a partial application function $\bullet\colon A \times A \rightharpoonup A$, together with elements s and k of A, which mimic this property of the combinators S and K. These so called *partial combinatory algebras* are well studied objects in the realm of realizability theory, see for example [Oos08].

By their algebraic nature, partial combinatory algebras can easily be defined in any cartesian restriction category, as was first done in [CH08]. We characterize arising *categories of computable maps* in terms of morphisms s and k and are able to show basic facts of classical computability theory in this categorical setting. Moreover we then see that the category of partial recursive functions arises as such a category of computable maps. We conclude this section by defining a category of partial combinatory algebras over a cartesian restriction category $\mathcal{C}$ and discussing their morphisms, the so called *simulations*.

§ Definition and Basic Properties

Definition 2.1.1

An **applicative system** $\mathbb{A} = (A, \bullet, \mathfrak{a})$ in a cartesian restriction category $\mathcal{C}$ consists of an object A together with a binary **application morphism** $\bullet\colon A \times A \longrightarrow A$ and a subset $\mathfrak{a} \subseteq \mathrm{Tot}(1, A)$ of **codes**. In the case $\mathfrak{a} = \mathrm{Tot}(1, A)$ we might just write $\mathbb{A} = (A, \bullet)$. Moreover, for the sake of readability, we will sometimes write $\mathtt{a} \bullet \mathtt{b}$ for the composition $\bullet(\mathtt{a} \times \mathtt{b})$.

By left associative application a family of **iterated application morphisms** can be inductively defined as:

$$\bullet^{(0)} := A \xrightarrow{\Delta} A \times A \xrightarrow{\bullet} A$$

$$\bullet^{(1)} := A \xrightarrow{\bullet} A$$

$$\bullet^{(n)} := A \times A^n \xrightarrow{\bullet \times A^{n-1}} A \times A^{n-1} \xrightarrow{\bullet^{(n-1)}} A$$

With this notion at hand we are now in the position to define when a morphism is computable with respect to an application morphism.

Definition 2.1.2

Let $\mathbb{A} = (A, \bullet, \mathfrak{a})$ be an applicative system.

Given a nonnegative integer n, a morphism $f\colon A^n \longrightarrow A$ is said to be $\mathbb{A}$-**computable**, if there exists a (in general not unique) code $\mathtt{c} \in \mathfrak{a}$ such that the diagram to the right commutes and for positive n the morphism

$$\begin{tikzcd} A \times A^n \arrow[r, "\bullet^{(n)}"] & A \\ A^n \arrow[u, dashed, "\mathtt{c} \times A^n"] \arrow[ur, "f"'] & \end{tikzcd}$$

$$A^{n-1} \xrightarrow{\mathtt{c} \times A^{n-1}} A \times A^{n-1} \xrightarrow{\bullet^{(n-1)}} A$$

is total. In this case $\mathtt{c}$ is called a **code for** f.

A morphism $f\colon A^m \longrightarrow A^n$ is $\mathbb{A}$-**computable**, if all its components $\pi_i f\colon A^m \longrightarrow A$ $(i = 1 \dots n)$ are. In particular the product of $\mathbb{A}$-computable functions is $\mathbb{A}$-computable.

The morphism $!_{A^n}\colon A^n \longrightarrow 1$ is $\mathbb{A}$-**computable**, if its domain $1_{A^n}\colon A^n \longrightarrow A^n$ is.

We have to point out that we defined, what [CH08] calls a *relative applicative system.* It will become apparent later, why we decided to immediately use the more general case instead of presenting the theory in the *absolute* case $\mathfrak{a} = \mathrm{Tot}(1, A)$ and switching to the relative case when needed.

The reasons for the iterated application morphisms to be total in the first few arguments are of mainly technical nature. We will find several reasons in the following sections.

Now let us consider the collection of all $\mathbb{A}$-computable morphisms for a given applicative system $\mathbb{A} = (A, \bullet, \mathfrak{a})$. As the definition of $\mathbb{A}$-computability does not imply that identities are $\mathbb{A}$-computable nor that $\mathbb{A}$-computable morphisms compose, the subgraph $\mathrm{Comp}\,\mathbb{A}$ of $\mathbb{A}$-computable morphisms in the ambient category $\mathcal{C}$ in general does not form a subcategory. This motivates the following definition.

Definition 2.1.3

An applicative system $\mathbb{A} = (A, \bullet, \mathfrak{a})$ in a cartesian restriction category $\mathcal{C}$, for which $\mathrm{Comp}\,\mathbb{A}$ is a cartesian restriction category and $\mathfrak{a} = \mathrm{Tot}\,\mathrm{Comp}\,\mathbb{A}(1, A)$, is called **Partial Combinatory Algebra** or PCA for short.

The following theorem states that the computable morphisms of a PCA are all given by some combination of codes, applications and products. Further we will see that the computability of two specific morphisms suffices for an applicative system to be a PCA.

Theorem 2.1.4

Let $\mathcal{C}$ be a cartesian restriction category and let $\mathbb{A} = (A, \bullet, \mathfrak{a})$ be an applicative system, such that $\mathfrak{a} = \mathrm{Tot}\,\mathrm{Comp}\,\mathbb{A}(1, A)$. Denote the smallest cartesian restriction subcategory on the objects $1, A, A^2, \ldots$ containing the application morphism $\bullet$ and all codes $\mathfrak{a}$ by $\mathrm{Poly}\,\mathbb{A}$.

Then the following assertions are equivalent:

(i) $\mathbb{A}$ is a PCA, i.e. $\mathrm{Comp}\,\mathbb{A}$ forms a catesian restriction category.

(ii) Every morphism in $\mathrm{Poly}\,\mathbb{A}$ is $\mathbb{A}$-computable.

(iii) The morphisms

$$k := \quad A \times A \xrightarrow{\pi_1} A$$

and

$$s := \quad A^3 \xrightarrow{\pi_{13} \times \pi_{23}} A^2 \times A^2 \xrightarrow{\bullet \times \bullet} A \times A \xrightarrow{\quad\bullet\quad} A$$

are $\mathbb{A}$-computable with codes $\mathtt{k}, \mathtt{s}$ respectively. Here π_{ij} denotes the projection onto the i-th and j-th component.

Proof (ii) $\Longrightarrow$ (i):

As by definition every computable morphism can be represented in terms of products, codes and application morphisms, surely $\mathrm{Comp}\,\mathbb{A}$ is a subgraph of $\mathrm{Poly}\,\mathbb{A}$. Hence, if every morphism in $\mathrm{Poly}\,\mathbb{A}$ is computable, $\mathrm{Comp}\,\mathbb{A}$ must be equal to $\mathrm{Poly}\,\mathbb{A}$ and in particular has to be a cartesian restriction category.

(i) $\Longrightarrow$ (ii):

On the other hand, if $\mathrm{Comp}\,\mathbb{A}$ is a cartesian restriction category, the identity 1_A is computable by a code $\mathtt{i}$, i.e. the diagram on the right commutes.

$$\begin{tikzcd} A \times A \arrow[r, "\bullet"] & A \\ A \arrow[u, "\mathtt{i} \times A"] \arrow[ur, "1_A"'] & \end{tikzcd}$$

From this we can derive the computability of the application morphism by stating the commutativity of the following diagram.

$$\begin{tikzcd}
A\times A^2 \arrow[r, "\bullet\times A"] & A\times A \arrow[r, "\bullet"] & A \\
& A\times A \arrow[ul, "\mathtt{i}\times A^2"] \arrow[u, "1_A\times A"] \arrow[ur, "\bullet"'] &
\end{tikzcd}$$

Further, for any $n \in \mathbb{N}$, the computability of the identity morphism 1_{A^n} is by definition equivalent to the computability of the projection morphisms $\pi_j \colon A^n \longrightarrow A$ by codes $\mathtt{p}_j$ ($j = 1 \dots n$). Now given any code $\mathtt{a} \colon 1 \longrightarrow A$ it is computable by the code $\mathtt{p}_1 \bullet \mathtt{a}$ as the following decomposition of the desired equality $\bullet^{(0)}(\mathtt{p}_1 \bullet \mathtt{a}) = \mathtt{a}$ into a commutative diagram demonstrates.

$$\begin{tikzcd}
A \arrow[r, "\Delta"] & A\times A \arrow[r, "\bullet"] & A \arrow[dd, equal] \\
& A\times A\times A \arrow[u, "\bullet\times A"'] & \\
& A\times A \arrow[u, "\mathtt{p}_1\times A\times A"'] \arrow[r, "\pi_1"] & A \\
1 \arrow[uuu, "\mathtt{p}_1\bullet\mathtt{a}"] \arrow[r, "\mathtt{p}_1\bullet\mathtt{a}"'] & A \arrow[u, "\mathtt{a}\times A"'] \arrow[r, "!_A"'] & 1 \arrow[u, "\mathtt{a}"']
\end{tikzcd}$$

Here we used the fact that as $\mathtt{p}_1 \bullet \mathtt{a}$ is total by definition of the computability of π_1, the composite $!_A\,(\mathtt{p}_1 \bullet \mathtt{a})$ is the identity 1_1.

Thus, since by assumption computable morphisms are closed under composition, all morphisms in Poly $\mathbb{A}$ are computable.

(ii) $\Longrightarrow$ (iii):
By definition s and k are morphisms in Poly $\mathbb{A}$ and thus by assumption $\mathbb{A}$-computable.

(iii) $\Longrightarrow$ (ii):
Idea: We wish to show that every possible combination of codes and applications is $\mathbb{A}$-computable. However this is very tedious and rather impossible to do manually. Instead we find that such combinations, considered as words, can be represented by a word involving s and k only. As this representation is $\mathbb{A}$-computable, so is the morphism.

By assumption we have a set of total points $\mathfrak{a} = \{\mathtt{a}, \mathtt{b}, \mathtt{c}, \dots\}$. Now fix a set of variables $\Lambda = \{x, y, \dots\}$ to inductively define the set of formal expressions over $\mathfrak{a}$ via

$$\mathfrak{a} \subseteq \Sigma, \qquad \Lambda \subseteq \Sigma, \qquad e, e' \in \Sigma \implies (ee') \in \Sigma.$$

Note that the parentheses are part of the expression. However, we will omit outer parentheses in the following.

These formal expressions exhibit a interpretation $I \colon \Sigma \longrightarrow \operatorname{Mor} \mathcal{C}$ as morphisms in our category $\mathcal{C}$. It is inductively defined as

$$\begin{aligned} I(\mathtt{a}) &= \mathtt{a}\colon 1 \longrightarrow A \ \ \forall \mathtt{a} \in \mathfrak{a}, \\ I(x) &= 1_A\colon A \longrightarrow A \ \ \forall x \in \varLambda, \\ I(ee') &= I(e) \bullet I(e')\colon A^n \longrightarrow A \ \ \forall e, e' \in \varSigma, \end{aligned}$$

where ee' is assumed to contain variables $x_1, \ldots, x_n$. With this interpretation at hand we define expressions e and e' to be equivalent when $I(e) = I(e')$ and denote it by $e \simeq e'$.

Note that the codes $\mathtt{k}$ and $\mathtt{s}$ satisfy $(\mathtt{k}x)y \simeq x$ and $((\mathtt{s}x)y)z \simeq (xz)(yz)$ by definition. Now any expression $e \in \varSigma$ can be represented as expression $[\lambda^* x.e] \in \varSigma$, whose variables are exactly those of e excluding x and which satisfies $[\lambda^* x.e]x \simeq e$. It is, yet again, defined by induction on the structure of $\varSigma$:

$$\begin{aligned} [\lambda^* x.\mathtt{a}] &\coloneqq \mathtt{ka} \\ [\lambda^* x.y] &\coloneqq \mathtt{k}y \\ [\lambda^* x.x] &\coloneqq (\mathtt{sk})\mathtt{k} \\ [\lambda^* x.(ee')] &\coloneqq (\mathtt{s}[\lambda^* x.e])[\lambda^* x.e'] \end{aligned}$$

The following calculations show that this definition satisfies the requirements.

$$\begin{aligned} [\lambda^* x.\mathtt{a}]x &\simeq (\mathtt{ka})x \simeq \mathtt{a} \\ [\lambda^* x.y]x &\simeq (\mathtt{k}y)x \simeq y \\ [\lambda^* x.x]x &\simeq ((\mathtt{sk})\mathtt{k})x \simeq (\mathtt{k}x)(\mathtt{k}x) \simeq x \\ [\lambda^* x.(ee')]x &\simeq ((\mathtt{s}[\lambda^* x.e])[\lambda^* x.e'])x \simeq ([\lambda^* x.e]x)([\lambda^* x.e']x) \simeq ee'. \end{aligned}$$

To reduce the notational burden, we will write $[\lambda^* xy.e]$ for $[\lambda^* x.[\lambda^* y.e]]$.

Finally any morphism in Poly $\mathbb{A}$ is defined in terms of codes, products and the application morphism. These building blocks admit a representation as an expression in $\varSigma$ via

$$\begin{aligned} \mathtt{a}\colon\ 1 \ &\longrightarrow A \mathrel{\hat{=}} \mathtt{a} \\ \pi_k\colon A^n &\longrightarrow A \mathrel{\hat{=}} [\lambda^* x_1 \ldots x_n.x_k]x_1 \ldots x_n \\ 1_A\colon A \ &\longrightarrow A \mathrel{\hat{=}} [\lambda^* x.x]x \end{aligned}$$

and composition can also be expressed in $\varSigma$ as

$$A^m \xrightarrow{f} A^n \xrightarrow{g} A \mathrel{\hat{=}} [\lambda^* x_1 \ldots x_m.\mathtt{g}(\mathtt{f}_1 x_1 \ldots x_m) \ldots (\mathtt{f}_n x_1 \ldots x_m)]x_1 \ldots x_m$$

where we omitted brackets forcing left associative application for the sake of readability and denoted the components of f by $f_1, \ldots, f_n$.

Hence any morphism $f\colon A^n \longrightarrow A$ in Poly $\mathbb{A}$ admits a representation as expression in $\varSigma$ which, by definition of the λ^* notation, can be interpreted as a morphism of the form $\bullet^n([\lambda^* x_1 \ldots x_n.e] \times A^n)$, which makes it computable. $\square$

It is important to note that the λ^*-representation of computable morphisms in terms of s and k heavily depends on the choice of codes $\mathtt{s}$ and $\mathtt{k}$. Hence we may denote a PCA $\mathbb{A}$ by $\mathbb{A} = (A, \bullet, \mathfrak{a}, \mathtt{s}, \mathtt{k})$, if we need to work with their specific representation.

Corollary 2.1.5
Let $\mathcal{F}\colon \mathcal{C} \longrightarrow \mathcal{D}$ be cartesian restriction functor and $\mathbb{A} = (A, \bullet, \mathfrak{a})$ be PCA in $\mathcal{C}$.

Then the image $\mathcal{F}\mathbb{A} = (\mathcal{F}A, \mathcal{F}\bullet, \mathcal{F}\mathfrak{a})$ is a PCA in $\mathcal{D}$.

Proof It is obvious that $\mathcal{F}\mathbb{A}$ is an applicative system in $\mathcal{D}$. A cartesian restriction functor $\mathcal{F}$ preserves restriction and products and in particular total points. Hence the image of any $\mathbb{A}$-computable morphism is $\mathcal{F}\mathbb{A}$-computable. In particular $\mathcal{F}s_{\mathbb{A}} = s_{\mathcal{F}\mathbb{A}}$ and $\mathcal{F}k_{\mathbb{A}} = k_{\mathcal{F}\mathbb{A}}$ are $\mathcal{F}\mathbb{A}$-computable, so by theorem 2.1.4 every morphism in Poly $\mathcal{F}\mathbb{A}$ is $\mathcal{F}\mathbb{A}$-computable. Thus $\mathcal{F}\mathbb{A}$ is a PCA. $\square$

One remarkable thing about PCAs is that basic definitions of computability theory like lists or primitive recursion can be formulated just by using the combinators s and k. Details can be found in [Oos08]. These internal definitions have consequences for the category of computable maps as we will see now.

Corollary 2.1.6
Let $\mathbb{A}$ be a PCA. Then there are computable morphisms $m\colon A^2 \longrightarrow A$, $r_1\colon A \longrightarrow A$ and $r_2\colon A \longrightarrow A$ assembling into a computable embedding-retraction pair $(m, r)\colon A \times A \triangleleft A$ with $r = \langle r_1, r_2\rangle$. Inductively, every object A^n in Comp $\mathbb{A}$ is a retract of A via an computable embedding-retraction pair.

Proof Using the correspondence established in the proof of theorem 2.1.4, define

$$\begin{aligned} m\colon A^2 &\longrightarrow A \mathrel{\hat{=}} [\lambda^* xyu.uxy]xy \\ r_1\colon A &\longrightarrow A \mathrel{\hat{=}} [\lambda^* v.v[\lambda^* xy.x]]v \\ r_2\colon A &\longrightarrow A \mathrel{\hat{=}} [\lambda^* v.v[\lambda^* xy.y]]v \end{aligned}$$

The discerning reader might notice that strictly speaking m should correspond to a morphism $A \times A \times A \longrightarrow A$ as the lambda term contains three variables. However, in this particular instance, we use the code of this three variable morphism to define another morphism on two variables. This then has the property that it behaves like the three variable morphism when applied to three variables.

We can easily verify that (m, r) is an embedding by performing the following calculation.

$$r_1 m \mathrel{\hat{=}} [\lambda^* v.v[\lambda^* xy.x]]([\lambda^* xyu.uxy]xy) = [\lambda^* xyu.uxy]xy[\lambda^* xy.x] = [\lambda^* xy.x]xy \mathrel{\hat{=}} \pi_1$$

From this and the same calculation for $r_2 m$ we can deduce $rm = 1_{A\times A}$. Inductively we can define computable embeddings like

$$m^{(n)} := \quad A \times A^{n-1} \xrightarrow{A \times m^{(n-1)}} A \times A \xrightarrow{\quad m \quad} A$$

with retractions given by

$$r^{(n)} := \quad A \xrightarrow{\quad r \quad} A \times A \xrightarrow{A \times r^{(n-1)}} A \times A^{n-1}.$$

$\square$

Corollary 2.1.7 (Fixed Point Theorem / First Recursion Theorem)
Let $\mathbb{A} = (A, \bullet, \mathfrak{a})$ be a PCA and $f\colon A \longrightarrow A$ be a total $\mathbb{A}$-computable morphism. Then there is a code $\mathtt{a} \in \mathfrak{a}$ such that $\bullet(f\mathtt{a} \times A) = \bullet(\mathtt{a} \times A)$.

Proof (generalizing [Oos08] Prop. 1.3.4)

Speaking of $\mathbb{A}$-computable morphisms we can rephrase the statement in the following way: Let f have a code $\mathtt{c}$, then there is a code $\mathtt{a}$ such that $\mathtt{ca}x \simeq \mathtt{a}x$.

Define the *strict fixed point combinator* $\mathtt{Z} \coloneqq \mathtt{uu}$, where $\mathtt{u} \coloneqq [\lambda^* ucx.c(uuc)x]$. It satisfies

$$\mathtt{Zc} \simeq \mathtt{uuc} \simeq [\lambda^* ucx.c(uuc)x]\mathtt{uc} \simeq [\lambda^* x.\mathtt{c}(\mathtt{uuc})x] \simeq [\lambda^* x.\mathtt{c}(\mathtt{Zc})x].$$

In particular the code $\mathtt{a} \coloneqq \mathtt{Zc}$ exists. Furthermore we have

$$\mathtt{a}x \simeq \mathtt{Zc}x \simeq [\lambda^* x.\mathtt{c}(\mathtt{Zc}x)]x \simeq \mathtt{c}(\mathtt{Zc})x \simeq \mathtt{ca}x,$$

so $\mathtt{a}$ is indeed the desired code. □

Up until now we have only seen a very abstract notion of computability. However this notion neatly generalizes the usual definition of computable functions. To see this we have to recall some facts about partial recursive functions, taken from [Odi89].

The partial recursive functions admit the *standard system of indices* $(\varphi_e^n)_{e,n\in\mathbb{N}}$, i.e. for every partial recursive function $f\colon \mathbb{N}^n \longrightarrow \mathbb{N}$ there exists an index $e \in \mathbb{N}$ such that $f \simeq \varphi_e^n$. This system of indices satisfies the following conditions:

Enumeration:
The partial function given by the assignment $(e, x_1, ..., x_n) \mapsto \varphi_e^n(x_1, ..., x_n)$ is partial recursive, i.e. there exists an index a such that

$$\varphi_e^n(x_1, ..., x_n) \simeq \varphi_a^{n+1}(e, x_1, ..., x_n).$$

Smn or *Parametrization*:
For every $m, n \in \mathbb{N}$ there is a total recursive function S_n^m, which satisfies

$$\varphi_e^{m+n}(x_1, ..., x_n, y_1, ..., y_m) \simeq \varphi^m_{S_n^m(e,x_1,...,x_n)}(y_1, ..., y_m).$$

For the following any *system of indices* (ψ_e^n) satisfying these conditions will be suitable.

Let $\mathcal{C} = \mathsf{Par}$ be the ambient cartesian restriction category and consider $A = \mathbb{N}$. We define the **Kleene application** $e \bullet x \coloneqq \psi_e^1(x)$ on $\mathbb{N}^2$. Since (ψ_e^n) satisfies *enumeration*, $\bullet\colon \mathbb{N}^2 \longrightarrow \mathbb{N}$ is a partial recursive function. Furthermore by *parametrization* there is a total recursive function S_{n-1}^1 with index s for every n. For any partial recursive function $f\colon \mathbb{N}^n \longrightarrow \mathbb{N}$ we get the chain of equalities

$$\begin{aligned} f(x_1, \ldots, x_n) &\simeq \psi_e^n(x_1, \ldots, x_n) \\ &\simeq S_{n-1}^1(e, x_1, \ldots, x_{n-1}) \bullet x_n \\ &\simeq S_{n-1}^1(s, e, x_1, \ldots, x_{n-2}) \bullet x_{n-1} \bullet x_n \\ &\simeq \ldots \\ &\simeq S_{n-1}^1(s, \ldots, s, e) \bullet x_1 \bullet \cdots \bullet x_n. \end{aligned}$$

Hence we can represent every partial recursive function as iterated application and obtain that $\mathbb{K}_1 \coloneqq (\mathbb{N}, \bullet)$ forms a PCA in Par. The category $\mathrm{Comp}\,\mathbb{K}_1 \eqqcolon \mathsf{Rec}$ is the restriction category of partial recursive functions. As a side note, observe that the totality condition in the definition of computable morphism closely resembles the condition on the *Smn*-functions to be total.

With the PCA $\mathbb{K}_1$ in mind we can make a very interesting observation. Recall that within a PCA we have a kind of λ-notation, with which we can derive Kleene's first recursion theorem. Interestingly the usual recursion theoretic proof of the recursion theorem can be adapted to the categorical setting as well. We think of it as yet another hint towards the Church-Turing-thesis.

Proof (generalizing [Odi89] Thm. II.2.10)

Since $\mathbb{A}$-computable functions compose and are stable under products, the morphism

$$A \times A \xrightarrow{f \bullet^{(0)} \times A} A \times A \xrightarrow{\quad\bullet\quad} A$$

is $\mathbb{A}$-computable with some code $\mathtt{b} \in \mathfrak{a}$, i.e. $\bullet^{(2)}(\mathtt{b} \times A^2) = \bullet(f \bullet^{(0)} \times A)$. Moreover, $\bullet(\mathtt{b} \times A)\colon A \longrightarrow A$ is total, so we have the code $\mathtt{a} \coloneqq \mathtt{b} \bullet \mathtt{b}$. By definition this satisfies the desired property. $\square$

In difference to proofs using the λ-notation, adaptations of the proofs coming from recursion theory are more prone to using categorical methods, like we can see in the following proof of Kleene's second recursion theorem. Hence the λ-notation will only occur infrequently from now on.

Proposition 2.1.8 (Second Recursion Theorem)
Let $\mathbb{A} = (A, \bullet, \mathfrak{a})$ be a PCA and $f\colon A \times A \longrightarrow A$ be an $\mathbb{A}$-computable morphism. Then there exists a code $\mathtt{c}$ such that $f(\mathtt{c} \times A) = \bullet(\mathtt{c} \times A)$.

Proof (cf. [Coc10] Thm. 7.9)

Define the $\mathbb{A}$-computable morphism

$$h \coloneqq \quad A \times A \xrightarrow{\bullet^{(0)} \times A} A \times A \xrightarrow{\quad f \quad} A$$

Pick some code $\mathtt{h}$ for h and recall that $\bullet(\mathtt{h} \times A)$ is total. In particular for the code $\mathtt{c} \coloneqq \mathtt{h} \bullet \mathtt{h}$ the following diagram commutes.

$$\begin{tikzcd}
& A \arrow[ddl, bend right, "\mathtt{c} \times A"'] \arrow[d, "\mathtt{h} \times A"] \arrow[ddr, bend left, "\mathtt{c} \times A"] & \\
& A \times A \arrow[dl, "(\mathtt{h} \bullet A) \times A"'] \arrow[dr, "\bullet^{(0)} \times A"] \arrow[dd, "h"] & \\
A \times A \arrow[dr, "\bullet"'] & & A \times A \arrow[dl, "f"] \\
& A &
\end{tikzcd}$$

$\square$

§ Simulations and the 2-Category of PCAs

Definition 2.1.9

Let $\mathbb{A} = (A, \bullet, \mathfrak{a})$ and $\mathbb{B} = (B, \bullet, \mathfrak{b})$ PCAs in a cartesian restriction category $\mathcal{C}$ and let $\phi\colon A \longrightarrow B$ be a morphism in $\mathcal{C}$.

A morphism $f\colon A^n \longrightarrow A$ **admits a simulation** via ϕ, if there exists an (in general not unique) $\mathbb{B}$-computable morphism $g\colon B^n \longrightarrow B$ such that the diagram on the right commutes.

$$\begin{tikzcd}
A^n \arrow[r, "f"] \arrow[d, "\phi^n"'] & A \arrow[d, "\phi"] \\
B^n \arrow[r, dashed, "\exists g"'] & B
\end{tikzcd}$$

$\phi\colon A \longrightarrow B$ is a **simulation**, if every $\mathbb{A}$-computable morphism can be simulated via ϕ. In this case we write $\phi\colon \mathbb{A} \longrightarrow \mathbb{B}$.

If ϕ has a retraction $r\colon B \longrightarrow A$ in $\mathcal{C}$ such that ϕr is $\mathbb{B}$-computable, we say that ϕ is a **faithful simulation**.

Note that a simulation $\phi\colon \mathbb{A} \longrightarrow \mathbb{B}$ always satisfies that for a code $\mathtt{a}$ in $\mathfrak{a}$ the morphism $\phi\mathtt{a}$ is a code in $\mathfrak{b}$. Indeed, having $A^0 = B^0 = 1$ and $\phi^0 = 1_1$, we obtain that the $\mathbb{A}$-computable code $\mathtt{a}$ admits a simulation via ϕ if and only if there is a code $\mathtt{b}$ in $\mathfrak{b}$ such that $\phi\mathtt{a} = \mathtt{b}$.

An interesting example for simulations is given by natural transformations, as is shown in the following lemma.

Lemma 2.1.10

Let $\mathcal{C}, \mathcal{D}$ be cartesian restriction categories and $\mathbb{A} = (A, \bullet, \mathfrak{a})$ be a PCA in $\mathcal{C}$. Suppose further that $\mathcal{F}, \mathcal{G}\colon \mathcal{C} \longrightarrow \mathcal{D}$ are two cartesian restriction functors, which admit a natural transformation $\alpha\colon \mathcal{F} \implies \mathcal{G}$.

Then the component α_A of α is a simulation between the PCAs $\mathcal{F}\mathbb{A}$ and $\mathcal{G}\mathbb{A}$.

Proof (cf. [CH08] Lem. 5.3)

First we note that for a natural transformation between cartesian restriction functors it holds that $\alpha_{A\times A} = \alpha_A \times \alpha_A$, since $\pi_i \alpha_{A\times A} = \mathcal{G}\pi_i \alpha_{A\times A} = \alpha_A \mathcal{F}\pi_i = \alpha_A \pi_i$ for $i = 1, 2$. But by the universal property of the product $\mathcal{G}A \times \mathcal{G}A$ this shows $\alpha_{A\times A} = \alpha_A \times \alpha_A$.

Now from the naturality square

$$\begin{tikzcd}
\mathcal{F}A \times \mathcal{F}A \arrow[r, "\alpha_A \times \alpha_A"] \arrow[d, "\mathcal{F}\bullet"'] & \mathcal{G}A \times \mathcal{G}A \arrow[d, "\mathcal{G}\bullet"] \\
\mathcal{F}A \arrow[r, "\alpha_A"'] & \mathcal{G}A
\end{tikzcd}$$

we can deduce that α_A is a simulation, because by theorem 2.1.4 every computable morphism has a representation consisting of codes and applications. $\square$

As we will see now, simulations give a good notion of morphism of PCAs turning them into a 1-category. In fact this category has a canonical preorder-structure on the homsets, which again turns it into a 2-category.

Lemma 2.1.11

Let $\mathcal{C}$ be a cartesian restriction category. Then PCAs over $\mathcal{C}$ and their simulations form a 2-category, which we will denote by $\mathsf{PCA}(\mathcal{C})$. The subcategory of faithful simulations is then denoted by $\mathsf{PCA}_f(\mathcal{C})$.

Proof (cf. [CH08, pp. 39-40])

The data already stated turns $\mathsf{PCA}(\mathcal{C})$ into a 1-category, as the identity simulation on a PCA $\mathbb{A}$ is given by the identity 1_A in $\mathcal{C}$ and the composition of two simulations by definition results in a simulation.

Moreover $\mathsf{PCA}(\mathcal{C})$ has a well behaved preorder-structure on its homsets. For parallel simulations $\phi, \psi\colon \mathbb{A} \longrightarrow \mathbb{B}$ we define $\phi \preceq \psi$ precisely when there exists a $\mathbb{B}$-computable morphism $g\colon B \longrightarrow B$ such that the diagram to the right commutes, or in other words, if we can transform the simulation ϕ into the simulation ψ via a computable morphism. Clearly this preorder is preserved by precomposition. Meanwhile, by definition of a simulation it is stable under postcomposition. Hence we have an induced 2-category structure.

$$\begin{tikzcd}
A \arrow[d, "\phi"'] \arrow[dr, "\psi"] & \\
B \arrow[r, dashed, "\exists g"'] & B
\end{tikzcd}$$

$\square$

Note that $\mathsf{PCA}(\mathcal{C})$ is explicitly not poset-enriched. In particular $\phi \preceq \psi$ and $\psi \preceq \phi$ does not imply $\phi = \psi$. Hence we denote this case by $\phi \asymp \psi$.

In the current situation we on the one hand have for any PCA a category of computable morphisms and a notion of homomorphism of PCAs on the other hand. Thus the question arises, whether the assignment $\mathbb{A} \mapsto \operatorname{Comp} \mathbb{A}$ is functorial. Embarrassingly in general it is not [CH08, p. 36]. However, if we restrict ourselves to faithful simulations and allow splitting of idempotents, it is. As the latter will be discussed extensively in the next chapter, we will postpone the discussion of functoriality until then.

Now recall that in 2-categories there is a notion of "weak isomorphism" called *equivalence*. A hint, why the notion of equivalence of PCAs is worth to be considered, is its resemblance to the definition of *acceptable system of indices* in (cf. [Odi89] Def. II.5.2). In our terms, a system of indices is acceptable, if and only if the arising PCA is equivalent to Kleene's first PCA. We characterize equivalences of PCAs in the following lemma.

Lemma 2.1.12

Given PCAs $\mathbb{A}$ and $\mathbb{B}$ and simulations $\phi\colon \mathbb{A} \longrightarrow \mathbb{B}$ and $\psi\colon \mathbb{B} \longrightarrow \mathbb{A}$, the following are equivalent.

(i) ϕ and ψ establish an equivalence of PCAs i.e. $1_A \asymp \psi\phi$ and $\phi\psi \asymp 1_B$.

(ii) $\psi\phi\colon A \longrightarrow A$ is $\mathbb{A}$-computable and has an $\mathbb{A}$-computable retraction; $\phi\psi\colon B \longrightarrow B$ is $\mathbb{B}$-computable and has a $\mathbb{B}$-computable retraction.

Proof (cf. [CH08] Lem. 5.6)

By definition $\psi\phi$ is $\mathbb{A}$-computable if and only if $1_A \preceq \psi\phi$. Similarly $\psi\phi$ has an $\mathbb{A}$-computable retraction r precisely when $\psi\phi \preceq 1_A$.

The same applies for $\phi\psi \preceq 1_B$ and $1_B \preceq \phi\psi$. □

2.2 Turing Categories

The category of computable maps of a PCA $\mathbb{A}$ only consists of the objects A^k. In the setting of Kleene's first PCA this translates to having just the partial functions $\mathbb{N}^m \rightharpoonup \mathbb{N}^n$. However, when speaking about computable subsets of $\mathbb{N}$, it might be convenient to have these subsets as existing objects in our category. *Turing categories* fix this deficit.

At first, the definition of a Turing category seems to be more general than that of the category of $\mathbb{A}$-computable maps. Thus a major part of this section is devoted to showing the result from [CH08] that PCAs and Turing categories indeed give the same notion of category of computable sets, in an appropriate sense. In the remaining part of the section we follow [Coc10] and introduce *recursion categories*, i.e. Turing categories with an enhanced restriction structure.

§ Definition and Relation to PCAs

Definition 2.2.1

Let $\mathcal{C}$ be a cartesian restriction category and T be an object of $\mathcal{C}$.

A family of maps $\tau = \{\tau_{X,Y} : T \times X \longrightarrow Y \mid X, Y \in \operatorname{Ob}\mathcal{C}\}$ is an **applicative family** on T.

A morphism $f\colon Z \times X \longrightarrow Y$ admits a $\tau_{X,Y}$**-index**, if there exists an (in general not unique) total index $i\colon Z \longrightarrow T$ such that the diagram to the right commutes. If for arbitrary objects Z each morphism $f\colon Z \times X \longrightarrow Y$ admits a $\tau_{X,Y}$-index, we say $\tau_{X,Y}$ is **universal**.

$$\begin{tikzcd} T \times X \arrow[r, "\tau_{X,Y}"] & Y \\ Z \times X \arrow[u, "i \times X"] \arrow[ur, "f"'] & \end{tikzcd}$$

If every $\tau_{X,Y}$ of an applicative family τ is universal, $\mathbb{T} = (T, \tau)$ defines a **Turing structure** on $\mathcal{C}$. In this case T is called a **Turing object** and $\tau_{T,T} =: \bullet$ is a **Turing morphism**. A cartesian restriction category, which possesses a Turing structure, is a **Turing category**.

It is important to point out that a Turing category does not come with a specific choice of Turing structure. In fact, in a Turing category many different Turing structures may exist on the same or on different Turing objects. However we will later see that different Turing structures on a Turing category are related in a suitable sense.

Now, as finding a Turing structure may become a bit tedious, we strive for a simpler criterion. The following theorem gives us just that.

Theorem 2.2.2 (Recognition Principle)
Let $\mathcal{C}$ be a cartesian restriction category.

Then $\mathcal{C}$ is a Turing category if and only if there is an applicative system $\mathbb{A} = (A, \bullet)$ such that every object is a retract of A and $\bullet$ is universal (in the sense of the preceding definition).

Proof (cf. [CH08] Lem. 3.3 + Thm. 3.4)

If $\mathcal{C}$ is a Turing category, it has a Turing structure $\mathbb{T} = (T, \tau)$. By definition the Turing morphism $\bullet = \tau_{TT}$ is universal. Now let X be some arbitrary object and $p\colon X \longrightarrow T$ be a τ_{1X}-index for $\pi_X\colon X \times 1 \longrightarrow X$. Using the Turing structure and explicitly writing out the isomorphism $X \cong X \times 1$ (which we suppress otherwise), we obtain the commuting diagram

$$\begin{tikzcd} T \arrow[r, shift left, "\langle 1_T, !_T\rangle"] & T \times 1 \arrow[l, shift left, "\pi_T"] \arrow[r, "\tau_{1X}"] & X \\ X \arrow[r, "\langle 1_X, !_X\rangle"] & X \times 1 \arrow[u, "p \times 1"] \arrow[ur, "\pi_X"'] & \end{tikzcd}$$

establishing X as retract of T.

Conversely, given an applicative system $\mathbb{A} = (A, \bullet)$ such that the application morphism is universal and every object is a retract of A, we can construct a Turing structure $\mathbb{A} = (A, \alpha)$. So let X, Y be arbitrary objects of $\mathcal{C}$ and let $(m_X, r_X)\colon X \lhd A$, $(m_Y, r_Y)\colon Y \lhd A$ be the embedding-retraction pairs. We define

$$\alpha_{XY} := \quad A \times X \xrightarrow{A \times m_X} A \times A \xrightarrow{\quad\bullet\quad} A \xrightarrow{\quad r_Y \quad} Y.$$

It remains to check the universality of this morphism. For this consider an arbitrary morphism $f\colon Z \times X \longrightarrow Y$. The universality can then be verified by stating that the diagram

$$\begin{tikzcd} A \times X \arrow[r, "A \times m_X"] & A \times A \arrow[r, "\bullet"] & A \arrow[r, "r_Y"] & Y \\ Z \times X \arrow[u, "i \times X"] \arrow[r, "Z \times m_X"'] & Z \times A \arrow[u, "i \times A"] \arrow[r, "Z \times r_X"'] & Z \times X \arrow[u, "m_Y f"] \arrow[ur, "f"'] & \end{tikzcd}$$

commutes and turns into

$$\begin{tikzcd} A \times X \arrow[r, "\alpha_{XY}"] & Y \\ Z \times X \arrow[u, "i \times X"] \arrow[ur, "f"'] & \end{tikzcd} ,$$

where we used the universality of $\bullet$ to obtain an index i for the composite $m_Y f(Z \times r_X)$.
$\square$

The recognition principle suggests that there is a close connection between certain applicative systems and Turing categories. For example a major part of a Turing structure can be replaced by iterated application morphisms of the Turing morphism.

Corollary 2.2.3

For every Turing category $\mathcal{T}$ with Turing structure $\mathbb{T} = (T, \tau)$, the iterated Turing morphisms $\bullet^{(n)}: T \times T^n \longrightarrow T$ are universal for all $n \in \mathbb{N}_0$.

Proof (cf. [CH08] Lem. 3.8)

For $n > 0$ we show this by induction.

(IB) The assertion holds for $n = 1$ by assumption ($\bullet^{(1)} = \bullet$ is the Turing morphism).

(IH) Suppose for some n the iterated morphisms $\bullet^{(n)}$ is universal.

(IS) Let $f: X \times T^{n+1} \longrightarrow T$. Then by induction hypothesis (IH) there is an $\bullet^{(n)}$-index $g: X \times T \longrightarrow T$ for f and an $\bullet$-index $h: X \longrightarrow T$ for g. The commutativity of the diagram

$$\begin{tikzcd}
T \times T \times T^n \arrow[r, "\bullet \times T"] & T \times T^n \arrow[r, "\bullet^{(n)}"] & T \\
X \times T \times T^n \arrow[u, "h \times T^{n+1}"] \arrow[r, equal] & X \times T \times T^n \arrow[u, "g \times T^n"] \arrow[ur, "f"'] &
\end{tikzcd}$$

shows that h is an $\bullet^{(n+1)}$-index for f and thus that $\bullet^{(n+1)}$ is universal.

For $n = 0$ it suffices to state the commutativity of the following diagram, where $f: X \longrightarrow T$ and $i: X \longrightarrow T$ is an $\bullet$-index for $f\pi_X: X \times T \longrightarrow T$.

$$\begin{tikzcd}
T \arrow[r, "\Delta"] & T \times T \arrow[r, "\bullet"] & T \\
X \arrow[u, "i"] \arrow[r, "1_X \times i"'] & X \times T \arrow[u, "i \times T"] \arrow[r, "\pi_X"'] & X \arrow[u, "f"']
\end{tikzcd}$$

□

Furthermore, the following theorem shows that the category of computable morphisms of PCAs and Turing categories are closely related.

Theorem 2.2.4 (Structure theorem)

Let $\mathcal{C}$ be a cartesian restriction category.

(i) If $\mathbb{A} = (A, \bullet, \mathfrak{a})$ is a PCA in $\mathcal{C}$, then $\operatorname{Comp}\mathbb{A}$ is a Turing category with a Turing structure induced by $\mathbb{A}$.

(ii) Conversely, if $\mathcal{T}$ is a cartesian restriction subcategory of $\mathcal{C}$, which possesses a Turing structure $\mathbb{T} = (T, \tau)$, then $\mathbb{T} = (T, \bullet, \operatorname{Tot}_{\mathcal{T}}(1, T))$ is a PCA in $\mathcal{C}$ and $\operatorname{Comp}\mathbb{T}$ is Morita-equivalent to $\mathcal{T}$.

Proof (i) (cf. [CH08] Thm. 4.6)

In corollary 2.1.6 we have seen that in $\operatorname{Comp}\mathbb{A}$ every object is an $\mathbb{A}$-computable retract of $\mathbb{A}$. Moreover $\bullet$ is universal as for some $\mathbb{A}$-computable morphism $f: A^n \longrightarrow A$ with code c the following diagram commutes

$$\begin{tikzcd}
A \times A \arrow[r, "\bullet"] & A \\
A^{n-1} \times A \arrow[u, "i \times A"] \arrow[ur, "f"'] &
\end{tikzcd}$$

where

$$i := \quad A^{n-1} \xrightarrow{\mathsf{c} \times A^{n-1}} A \times A^{n-1} \xrightarrow{\bullet^{(n-1)}} A$$

is by definition of $\mathbb{A}$-computability total. Thus we can apply the recognition principle theorem 2.2.2.

(ii) (cf. [CH08] Lem. 4.7 + Thm. 4.8 + Thm. 4.11)

We first show that $\operatorname{Comp}\mathbb{T}$ forms a subcategory of $\mathcal{T}$. For this consider any morphism $f\colon T^n \longrightarrow T$ in $\mathcal{T}$. Applying the universality of $\bullet$ iteratively on the indices we get the commuting diagram

$$\begin{tikzcd}
T \times T^n \arrow[r, "\bullet \times T^{n-1}"] & T \times T^{n-1} \arrow[r, "\cdots"] & T \times T \arrow[r, "\bullet"] & T \\
T^n \arrow[u, "\mathsf{c} \times T^n"] \arrow[r, equal] & T \times T^{n-1} \arrow[u, "j \times T^{n-1}"] \arrow[r, equal] & T^{n-1} \times T \arrow[u, "i \times T"] \arrow[ur, "f"'] &
\end{tikzcd}$$

establishing f as $\mathbb{T}$-computable function. In particular $\operatorname{Comp}\mathbb{T}$ is the full subcategory of $\mathcal{T}$ on the objects $1, T, T^2, ...$ and $\mathbb{T}$ forms a PCA in $\mathcal{C}$.

Now, as every object X of $\mathcal{T}$ is a retract of T i.e. there is at least one embedding-retraction pair $(m, r)\colon X \lhd T$ in $\mathcal{T}$ and the composite mr is an $\mathbb{T}$-computable idempotent on T i.e. contained in $\operatorname{Comp}\mathbb{T}$. Splitting these idempotents in $\operatorname{Comp}\mathbb{T}$ we obtain a category $\mathcal{S}$ to which $\mathcal{T}$ is equivalent. □

At this point it should be clear, why we decided to work with the relative notion of PCAs. It gives the opportunity to express this relation between PCAs and Turing categories in a clean way, telling us that Turing categories are just inflations of categories of computable maps.

In the preceding proof we have used that the iterated application morphisms are total in the first few arguments. Besides the reasons for this given right when introducing $\mathbb{A}$-computability one can argue in favor of it as follows: It makes PCAs and Turing categories compatible. One might ask, why we care about Turing categories at all, when we already have a notion of computability at hand. The point is that Turing categories evolve from categories of computable maps by splitting idempotents, hence the notion of Morita-equivalence appears. Recalling the remark that splitting restriction idempotents realizes restriction domains as objects, Turing categories act as a framework, in which one can inspect a notion of computable sets. Before we do just that in the following section, we will briefly investigate how well PCAs and Turing categories work together.

Corollary 2.2.5

If $\mathcal{T}$ is a Turing category with Turing structures $\mathbb{T} = (T, \tau)$ and $\mathbb{S} = (S, \sigma)$, the corresponding PCAs $\mathbb{T}$ and $\mathbb{S}$ are equivalent.

Proof (cf. [CH08] Prop. 5.7)

By corollary 2.1.6 T is a retract of S via some morphism $\phi\colon T \longrightarrow S$ and conversely, S is a retract of T via some morphism $\psi\colon S \longrightarrow T$. Since $\mathbb{T}$ and $\mathbb{S}$ are two Turing structures on the same category $\mathcal{T}$ every $\mathbb{T}$-computable morphism is $\mathbb{S}$-computable and vice versa. Thus $\psi\phi$ is $\mathbb{T}$-computable and so is the composition of the corresponding retractions. Similarly $\phi\psi$ is $\mathbb{S}$-computable and has an $\mathbb{S}$-computable retraction. By lemma 2.1.12 this establishes an equivalence of PCAs.

Another proof is given in [CH08] Lem. 3.10. □

In the spirit of category theory it is a reasonable question, whether simulations, i.e. morphisms of PCAs induce functors between the corresponding Turing categories. Sadly, in general this is not the case [CH08]. However a certain subclass of simulations, that of faithful simulations, does. This is expressed in the following lemma.

Lemma 2.2.6

Let $\mathcal{C}$ be a cartesian restriction category and $\mathbb{A}, \mathbb{B}$ be two PCAs in $\mathcal{C}$.

Then any faithful simulation $\phi\colon \mathbb{A} \longrightarrow \mathbb{B}$ induces a cartesian restriction functor of the form $\Phi\colon \operatorname{Split}\operatorname{Comp}\mathbb{A} \longrightarrow \operatorname{Split}\operatorname{Comp}\mathbb{B}$.

Moreover the assignment $\operatorname{Split}\operatorname{Comp}(-)\colon \mathsf{PCA}_f(\mathcal{C}) \longrightarrow \mathsf{Cat}$ (ignoring the restriction structure) is 2-functorial.

Proof (cf. [CH08] Lem. 5.8)

First note that by functoriality of idempotent splitting, shown in lemma 1.2.13, a restriction functor $\Phi'\colon \operatorname{Comp}\mathbb{A} \longrightarrow \operatorname{Split}\operatorname{Comp}\mathbb{B}$ extends to $\Phi\colon \operatorname{Split}\operatorname{Comp}\mathbb{A} \longrightarrow \operatorname{Split}\operatorname{Comp}\mathbb{B}$. Thus it suffices to define the functor on $\operatorname{Comp}\mathbb{A}$.

Let r be a restriction of the faithful simulation ϕ. By definition $e = \phi r$ is an $\mathbb{B}$-computable idempotent on B and as such an object in $\operatorname{Split}\operatorname{Comp}\mathbb{B}$. Define $\Phi'(A^n) := e^n$, the n-fold product of e in $\operatorname{Split}\operatorname{Comp}\mathbb{A}$, corresponding to the idempotent $e^n\colon B^n \longrightarrow B^n$ in $\operatorname{Comp}\mathbb{B}$. Given a morphism $f\colon A^n \longrightarrow A^m$ we set $\Phi'(f) := \phi^m f r^n\colon e^n \longrightarrow e^m$. In particular this means that Φ' preserves products. By $r\phi = 1_A$ this assignment preserves composition. Moreover for 1_{A^n} we have that $\Phi'(1_{A^n}) = \phi^n r^n = e^n = 1_{e^n}$ and so the assignment is functorial. It is left to show that Φ' preserves the restriction structure. This is done representatively for a morphism $f\colon A \longrightarrow A$ by the following calculation, in which we use that faithful simulations are monic and as such total.

$$\overline{\Phi' f} = \overline{\phi f r}\,\phi r = \phi r \overline{\phi f r \phi r} = \phi r \overline{\phi f r} = \phi r \overline{f r} = \phi \overline{f}\, r = \Phi' \overline{f}$$

By construction the functors induced by composable faithful simulations compose and as such the assignment is 1-functorial. Moreover a 2-cell $\phi \preceq \psi$ between faithful simulations (ϕ, r) and (ψ, s) comes by definition with a $\mathbb{B}$-computable morphism $g\colon B \longrightarrow B$ satisfying $g\phi = \psi$. For this morphism we have $\psi s g \phi r = \psi r$, which implies that ψr is $\mathbb{B}$-computable since the left side is. Moreover this establishes ψr as a morphism from ϕr to ψs in $\operatorname{Split}\operatorname{Comp}\mathbb{B}$. This will be the component of the induced natural transformation at A. Like in the construction of the functor Φ' we can use the same to obtain the components at A^n. Further, as seen in lemma 1.2.13, the natural transformation $\alpha'\colon \Phi' \Longrightarrow \Psi'$ obtained in this way lifts to a natural transformation $\alpha\colon \Phi \Longrightarrow \Psi$. $\square$

We will see as an easy corollary from this that equivalent PCAs induce Morita-equivalent categories of computable maps. In fact (some sort of) the opposite holds true as well.

To put things in context recall that distinct acceptable systems of indices give a self equivalence of the PCA $\mathbb{K}_1$ and represent the same category Rec. The following theorem generalizes this, but requires us to make the following definition.

Definition 2.2.7

Let $\mathcal{C}$ be a split cartesian restriction category. Two subcategories $\mathcal{D}_1, \mathcal{D}_2$ of $\mathcal{C}$ are **Morita equivalent over** $\mathcal{C}$, if there is an equivalence of categories $\mathcal{F}\colon \operatorname{Split}\mathcal{D}_1 \simeq \operatorname{Split}\mathcal{D}_2$ such that the triangle

$$\begin{tikzcd}
\operatorname{Split}\mathcal{D}_1 \arrow[rr, "\mathcal{F}"] \arrow[dr] & {} \arrow[d, phantom, "\cong"] & \operatorname{Split}\mathcal{D}_2 \arrow[dl] \\
& \mathcal{C} &
\end{tikzcd}$$

commutes up to a natural isomorphism, where the arrows pointing downwards are the faithful functors induced by the embeddings $\mathcal{D}_1 \hookrightarrow \mathcal{C}$ and $\mathcal{D}_2 \hookrightarrow \mathcal{C}$.

The natural isomorphism arises, because in $\mathcal{C}$ there could be many objects, which split an idempotent. Now we can state the following correspondence.

Theorem 2.2.8

Let $\mathcal{C}$ be a split cartesian restriction category. PCAs $\mathbb{A}$ and $\mathbb{B}$ in $\mathcal{C}$ are equivalent if and only if the corresponding Turing categories $\operatorname{Comp}\mathbb{A}$ and $\operatorname{Comp}\mathbb{B}$ are Morita-equivalent over $\mathcal{C}$.

Proof " $\Longrightarrow$ " (cf. [CH08] Prop. 5.9):
Lemma 2.1.12 tells us that we can assume that the simulations $\phi\colon \mathbb{A} \longrightarrow \mathbb{B}$ and $\psi\colon \mathbb{B} \longrightarrow \mathbb{A}$ establishing the equivalence of PCAs are faithful. Thanks to the preceding lemma we get functors $\Phi\colon \operatorname{Split}\operatorname{Comp}\mathbb{A} \longrightarrow \operatorname{Split}\operatorname{Comp}\mathbb{B}$ and $\Psi\colon \operatorname{Split}\operatorname{Comp}\mathbb{B} \longrightarrow \operatorname{Split}\operatorname{Comp}\mathbb{B}$. By 2-functoriality of the assignment these establish an equivalence of categories. In particular $\operatorname{Comp}\mathbb{A}$ and $\operatorname{Comp}\mathbb{B}$ are Morita-equivalent.

" $\Longleftarrow$ " (cf. [CH08] Prop. 5.10):
Let $\mathcal{M}_{\mathbb{A}}\colon \operatorname{Split}\operatorname{Comp}\mathbb{A} \longrightarrow \mathcal{C}$ and $\mathcal{M}_{\mathbb{B}}\colon \operatorname{Split}\operatorname{Comp}\mathbb{B} \longrightarrow \mathcal{C}$ denote the faithful restriction functors induced by the inclusions $\operatorname{Comp}\mathbb{A} \hookrightarrow \mathcal{C}$ and $\operatorname{Comp}\mathbb{B} \hookrightarrow \mathcal{C}$. Suppose $\mathcal{F}\colon \operatorname{Split}\operatorname{Comp}\mathbb{A} \longrightarrow \operatorname{Split}\operatorname{Comp}\mathbb{B}$ and $\mathcal{G}\colon \operatorname{Split}\operatorname{Comp}\mathbb{B} \longrightarrow \operatorname{Split}\operatorname{Comp}\mathbb{A}$ together with a natural isomorphism $\alpha\colon \mathcal{M}_{\mathbb{A}} \overset{\sim}{\Longrightarrow} \mathcal{M}_{\mathbb{B}}\mathcal{F}$ state an equivalence over $\mathcal{C}$. The PCA $\mathbb{A}$ in $\operatorname{Split}\operatorname{Comp}\mathbb{A}$ is sent to a PCA $\mathbb{A}'$ in $\operatorname{Split}\operatorname{Comp}\mathbb{B}$ via $\mathcal{F}$ and then to a PCA $\mathbb{A}''$ in $\mathcal{C}$ via $\mathcal{M}_{\mathbb{B}}$. Now, by lemma 2.1.10 α establishes an isomorphism of PCAs. As we wish to show an equivalence of the PCAs $\mathbb{A}$ and $\mathbb{B}$ we might as well assume $\mathbb{A} = \mathbb{A}''$.

In $\operatorname{Split}\operatorname{Comp}\mathbb{B}$ the Object $A' = \mathcal{F}A$ is a retract of the Turing object B via some embedding restriction pair $(m_{A'}, r_{A'})\colon A' \longrightarrow B$. Thus the PCA $\mathbb{A}'$ is simulated by $\mathbb{B}$, as the simulation of a morphism $f\colon A'^n \longrightarrow A'$ is given by mfr^n. Applying $\mathcal{M}_{\mathbb{B}}$ to the simulation $m\colon \mathbb{A}' \longrightarrow \mathbb{B}$ we get a simulation $\phi = m_{A'}\colon \mathbb{A}'' = \mathbb{A} \longrightarrow \mathbb{B}$. Switching the roles of $\mathbb{A}$ and $\mathbb{B}$ we get a simulation $\psi\colon \mathbb{B}'' = \mathbb{B} \longrightarrow \mathbb{A}$.

It is left to show that $\psi\phi$ and $\phi\psi$ are isomorphic to the identity. Citing lemma 2.1.12 we will do this by showing that $\psi\phi$ is $\mathbb{A}$-computable and has a $\mathbb{A}$-computable retraction, the case of $\phi\psi$ being completely analogous. Applying $\mathcal{G}$ to ϕ we get a morphism $\mathcal{G}\mathbb{A}' \longrightarrow \mathcal{G}\mathbb{B}$ in $\operatorname{Split}\operatorname{Comp}\mathbb{A}$, so the composite $\psi(\mathcal{G}\phi)$ is $\mathbb{A}$-computable and its image under $\mathcal{M}_{\mathbb{A}}$ is $\psi\phi$ (modulo computable isomorphism). Similarly we have that $r_B(\mathcal{G}r_A)$ is a retraction of $\psi(\mathcal{G}\phi)$, which is $\mathbb{A}$-computable as morphisms in $\operatorname{Split}\operatorname{Comp}\mathbb{A}$. Again its umage under $\mathcal{M}_{\mathbb{A}}$ is a retraction of $\psi\phi$ (modulo computable isomorphism). □

By now we have seen that a reasonable notion of morphism and equivalence of PCAs behaves well with respect to the general categorical setting, in which it lives. The following proposition tells us that it also behaves well with respect to Turing structures.

Proposition 2.2.9

Let $\mathcal{T}$ be a Turing category and $\mathcal{E}$ be a set of idempotents containing all identities and which is closed under products. The cartesian restriction category $\operatorname{Split}_{\mathcal{E}}\mathcal{T}$ is again a Turing category.

Proof (cf. [CH08] section 3.5)

See theorem 1.2.12 and lemma 1.2.13 for the fact that $\operatorname{Split}_{\mathcal{E}}\mathcal{T}$ is indeed a cartesian restriction category.

Fix some Turing structure $\mathbb{T} = (T, \tau)$ and consider the corresponding applicative system $(T, \bullet)$. Its embedding $(1_T, (1_T, \bullet, 1_T))$ into $\operatorname{Split}_{\mathcal{E}}\mathcal{T}$ is an applicative system in $\operatorname{Split}_{\mathcal{E}}\mathcal{T}$. By construction an idempotent e on X in $\mathcal{T}$ is an object in $\operatorname{Split}_{\mathcal{E}}\mathcal{T}$ which is a retract of 1_X. Now as every object X is a retract of T in $\mathcal{T}$, every object in $\operatorname{Split}_{\mathcal{E}}\mathcal{T}$ is a retract of 1_T.

A morphism $(1_T, f, e_X)\colon e_X \times 1_T \longrightarrow 1_T$ in $\operatorname{Split}_{\mathcal{E}}\mathcal{T}$ in particular corresponds to a morphism $f\colon X \times T \longrightarrow T$ in $\mathcal{T}$, which has an index $i\colon X \longrightarrow T$. But now the morphism $(1_T, ie_X, e_X)\colon e_X \longrightarrow 1_T$ is an index for $(1_T, f, e_X)$. Invoking the recognition principle 2.2.2 we get that $\operatorname{Split}_{\mathcal{E}}\mathcal{T}$ is a Turing category. □

In this light, the preceding theorem generalizes the fact that different *acceptable systems of indices* (cf. [Odi89] Def. II.5.2) by definition represent the same category of partial recursive functions. Moreover, recalling the fact that a splitting of restriction idempotents realizes the restriction domain as a real object in the category, the content of this chapter can be summarized in the following slogan.

Turing categories interpolate between categories of computable maps
and categories of computable sets.

§ Recursion Categories

We have seen that Turing categories offer the opportunity to consider a notion of computable sets. However, to be able to investigate them thoroughly we need more structure to make appropriate statements. For this reason the following definition was given in [Coc10].

Definition 2.2.10

A **recursion category** is a Turing category, in which every object is restriction discrete and which possesses compatible joins. In particular by proposition 1.2.28 it has meets.

It would be a pity, if recursion categories did not have a correspondence to PCAs. Luckily, when given some additional data, they fit right into the scheme.

Definition 2.2.11

Let $\mathcal{C}$ be a discrete cartesian restriction category with joins. An **interleaving PCA** is a PCA $\mathbb{A} = (A, \bullet, \mathfrak{a})$ in $\mathcal{C}$, which satisfies the following conditions.

- The restriction zero morphism $\emptyset_A \colon A \longrightarrow A$ is $\mathbb{A}$-computable.
- A is a restriction discrete object and the partial inverse $p_A \colon A \times A \longrightarrow A$ to $\Delta_A \colon A \longrightarrow A \times A$ is $\mathbb{A}$-computable.
- There is a (not necessarily $\mathbb{A}$-computable) total **interleaving morphism** $\theta \colon A \times A \longrightarrow A$, which satisfies that $\bullet(\theta \times A) \colon A \times A \times A \longrightarrow A$ is an interleaving of $\bullet(\pi_1 \times A)$ and $\bullet(\pi_2 \times A)$ and that $\mathfrak{a}$ is closed under θ.

We have to stress that the interleaving morphism constructs an interleaving of codes. As such it can happen, that given morphisms f, g with codes $\mathtt{f}_1, \mathtt{f}_2$ and $\mathtt{g}_1, \mathtt{g}_2$ respectively, the interleaved codes $\theta(\mathtt{f}_1, \mathtt{g}_1)$ and $\theta(\mathtt{f}_2, \mathtt{g}_2)$ are codes of distinct computable morphisms.

The conditions of the preceding definition suffice to generate recursion categories, as we will see now.

Theorem 2.2.12

Let $\mathcal{C}$ be a cartesian restriction category with joins and meets and let $\mathbb{A}$ be an interleaving PCA. Then $\mathrm{Comp}\,\mathbb{A}$ is restriction discrete and has compatible joins, making it recursion category.

Proof (cf. [Coc10] Thm. 7.3)

We have seen that every object in $\mathrm{Comp}\,\mathbb{A}$ is a retract of A. Since A is restriction discrete by assumption and every object in $\mathrm{Comp}\,\mathbb{A}$ is a retract of A, by lemma 1.2.29 every object in $\mathrm{Comp}\,\mathbb{A}$ is restriction discrete. By the interleaving morphism and lemma 1.2.10(iv), it inherits the joins of $\mathcal{C}$. □

We now turn our focus on the things we can express with this additional structure.

2.3 Computability and Complexity

We have already seen some reformulations of classical results in the language of PCAs, e.g. in the form of Kleene's recursion theorems 2.1.7 and 2.1.8. In the following section we will investigate selected topics of computability in our framework of Turing / Recursion categories.

At first we define the notion of *m-reduction* and state some of its properties, which will be of use when discussing *decidability*. Following [Coc10] we introduce generalizations of *recursively inseparable predicates* and *index sets*.

Extending the definition of an interleaving PCA, we arrive at a very general notion of *complexity class* and discuss a general construction of a hierarchy of restriction idempotents, which closely resembles that of the *arithmetical hierarchy* or *polynomial hierarchy*.

At last we give a definition of *relative computability* and show that a first selection of results generalizes well.

§ Many-One Reductions

In the classical setting of computability theory a subset $A \subseteq \mathbb{N}$ *many-one reduces* to another subset $B \subseteq \mathbb{N}$, if there is a total computable function $f\colon A \longrightarrow B$ satisfying that $x \in A$ if and only if $f(x) \in B$ (cf. [Odi89] Def. III.2.1). In terms of the restriction structure in Par, where we write $a, b\colon \mathbb{N} \longrightarrow \mathbb{N}$ for the restriction idempotents representing the sets A, B respectively, this is equivalent to saying that $a(x)\downarrow$ if and only if $bf(x)\downarrow$. By totality of f this is equivalent to having $fa(x)\downarrow$ if and only if $bf(x)\downarrow$. This leads to the following definition.

Definition 2.3.1

Let $\mathcal{T}$ be a Turing category over a cartesian restriction category $\mathcal{C}$ and e_1, e_2 be two restriction idempotents on objects X, Y respectively. Then e_1 **many-one reduces** (or m**-reduces** for short) to e_2, if there is a total map $f\colon X \longrightarrow Y$ in $\mathcal{T}$ such that the following diagram commutes. In this case we write $e_1 \leq_m e_2$.

$$\begin{tikzcd} X \arrow[r, "f"] \arrow[d, "e_1"'] & Y \arrow[d, "e_2"] \\ X \arrow[r, "f"'] & Y \end{tikzcd}$$

A restriction idempotent in $\mathcal{T}$ is m**-complete**, if every other restriction idempotent in $\mathcal{T}$ m-reduces to it.

If f is a monomorphism, e_1 **one-one reduces** (or **1-reduces** for short) to e_2, denoted by $e_1 \leq_1 e_2$.

We must admit that this may not be the definition found in [CH08], which is the only source to date generalizing m-reductions to the categorical setting. There an m-reduction is defined in terms of a *pullback*, which is a limit. However neither the universal property nor any other hint of which kind of limit (restriction limit, *latent limit* as in [CGH12], limit in the underlying category) can be found in this source. We believe that the definition given here is the right generalization, since its derivation can be easily verified and we will see that many other things generalize well with it. In fact, the proofs in [CH08] seem to use precisely our definition.

Note that the reduction morphism f is assumed to be computable, since m-reductions are defined within a Turing category. We can state some basic properties.

Lemma 2.3.2

Let $\mathcal{T}$ be a Turing category over a cartesian restriction category $\mathcal{C}$ and let e_1, e_2 be two restriction idempotents on objects X, Y respectively. The following holds.

(i) If $e_1 \leq_m e_2$ and e_2 is in $\mathcal{T}$, then e_1 is in $\mathcal{T}$.

(ii) Assume e_1, e_2 to have complements e_1^c and e_2^c. Then $e_1 \leq_m e_2$ if and only if $e_1^c \leq_m e_2^c$.

(iii) If e_1 and e_2 are in $\mathcal{T}$, e_1 is m-complete and $e_1 \leq_m e_2$, then e_2 is m-complete.

Proof (i) The composite $e_2 f = f e_1$ is in $\mathcal{T}$ and so is its restriction domain $\overline{f e_1} = e_1$.

(ii) By the symmetry of the problem it suffices to show that $e_1 \leq_m e_2$ implies $e_1^c \leq_m e_2^c$. For this suppose e_1 m-reduces to e_2 via some total morphism $f\colon X \longrightarrow Y$. The equation

$$e_2^c f = \overline{e_2^c}\, f = f\overline{e_2^c f} = f\overline{e_2 f}^{\,c} = f\overline{f e_1}^{\,c} = f\overline{f e_1^c} = f\overline{e_1^c} = f e_1^c$$

shows the claim. We used the fact, that since f is total we have

$$\overline{f e_1^c} = \overline{f e_1}^{\,c} \quad \text{and} \quad \overline{e_2^c f} = \overline{e_2 f}^{\,c},$$

which can easily verified by checking the following equations.

$$\begin{aligned}
\overline{f e_1}\,\overline{f e_1^c} &= e_1 e_1^c = \emptyset \\
\overline{f e_1} \sqcup \overline{f e_1^c} &= e_1 \sqcup e_2^c = 1 \\
\overline{e_2 f}\,\overline{e_2^c f} &= \overline{e_2 f \overline{e_2 f}} = \overline{e_2 e_2^c f} = \emptyset \\
\overline{e_2 f} \sqcup \overline{e_2^c f} &= \overline{e_2 f \sqcup e_2^c f} = \overline{(e_2 \sqcup e_2^c) f} = \overline{f} = 1
\end{aligned}$$

(iii) Follows immediately from the definition of m-completeness. □

It is a classic result of computability theory that the *halting sets* given by $\{x \in \mathbb{N} \mid \varphi_x(x)\downarrow\}$ and $\{(x, y) \in \mathbb{N}^2 \mid \varphi_y(x)\downarrow\}$ are m-complete (cf. [Odi89] Thm. III.2.4). This holds true in any Turing category.

Proposition 2.3.3

In any Turing category with Turing structure $\mathbb{T} = (T, \tau)$ the halting idempotents $\overline{\bullet^{(0)}}$ and $\overline{\bullet}$ are m-complete.

Proof (cf. [CH08] p. 21)

We first show, that the restriction domain of $\tau_{1,T}$ is m-complete. The rest will then follow by reduction.

Consider an arbitrary restriction idempotent $e\colon X \longrightarrow X$ and let m be an embedding $m\colon X \longrightarrow T$. Then by universality of $\tau_{1,T}$ there is an index $h\colon X \longrightarrow T$ for me such that the following diagram commutes.

$$\begin{tikzcd}
T \arrow[r, "\tau_{1,T}"] & T \\
X \arrow[u, "h"] \arrow[ur, "me"'] &
\end{tikzcd}$$

Using $\overline{me} = e$ by totality of m, we derive $\overline{\tau_{1,T}}\, h = h\overline{\tau_{1,T} h} = h\overline{me} = he$, which proofs $e \leq_m \overline{\tau_{1,T}}$. By corollary 2.2.3, $\bullet^{(0)}$ is part of a Turing structure and thus m-complete.

By $\overline{\bullet}\,\Delta = \Delta\overline{\bullet\Delta} = \Delta\overline{\bullet^{(0)}}$ we see that $\Delta\colon T \longrightarrow T \times T$ m-reduces $\overline{\bullet^{(0)}}$ to $\overline{\bullet}$, making $\overline{\bullet}$ m-complete. □

§ Decidability

In the classical setting of computability theory a set or language $A \subseteq \mathbb{N}$ is said to be *decidable*, if the total characteristic function $\chi_A \colon \mathbb{N} \longrightarrow \{0,1\}$ is computable. While $\mathbb{N}$ can be generalized to be a Turing object, it is not perfectly clear what object $\{0,1\}$ should be in an arbitrary Turing category. However, using Post's theorem (cf. [Odi89] Thm. II.1.19), one can equivalently define a decidable set to be a set, which is a domain of a partial recursive function and whose complement is the domain of another partial recursive function. This is the basis for the following definition.

Definition 2.3.4

Let $\mathcal{R}$ be a recursion category. A restriction idempotent e in $\mathcal{R}$ is said to be **decidable**, if it is complemented, i.e. if $\mathcal{R}$ contains a complement e^c of e. Restriction idempotents, which are not decidable, are commonly called **undecidable**.

Usually undecidability is preserved under m-reductions. This also holds in the categorical setting.

Lemma 2.3.5

Let $\mathcal{R}$ be a recursion category in a discrete cartesian restriction category $\mathcal{C}$ with joins and let e_1 be an undecidable restriction idempotent in $\mathcal{R}$, which possesses a complement in the ambient category $\mathcal{C}$. If e_2 is a restriction idempotent in $\mathcal{R}$ with $e_1 \leq_m e_2$, then e_2 is undecidable.

Proof *by contraposition*

Suppose e_2 to be decidable, i.e. having a complement in $\mathcal{R}$. By lemma 2.3.2 the complement e_1^c of e_1 m-reduces to the complement e_2^c of e_2 by a total morphism f in $\mathcal{R}$. But now by totality of f we have $\overline{fe_1^c} = e_1^c$ and as $fe_1^c = e_2^c f$ is a morphism in $\mathcal{R}$ its domain is in $\mathcal{R}$ as well. Hence e_1 is decidable. □

In computability theory a very important fact is the undecidability of the halting set (cf. [Odi89] Thm.II.2.3). Again, this can be expressed in the setting of Turing categories.

Theorem 2.3.6

In a nontrivial recursion category with Turing structure $\mathbb{T} = (T, \tau)$, the halting idempotent $\overline{\bullet^{(0)}}$ is undecidable.

Proof *by contraposition* (cf. [Coc10] Thm. 7.5)

For the sake of readability we will temporarily denote the halting idempotent $\overline{\bullet^{(0)}}$ by h. Let h^c be the complementing restriction idempotent of h. In particular there is a total code i for h^c making the diagram on the right commute. The equation

$$
\begin{tikzcd}
T \times T \arrow[r, "\bullet"] & T \\
T \arrow[u, "i \times T"] \arrow[ur, "h^c"'] &
\end{tikzcd}
$$

$$hi = \overline{\bullet^{(0)}}\, i = \overline{\bullet\Delta}\, i = i\overline{\bullet\Delta i} = i\overline{\bullet\langle i, i\rangle} = i\overline{h^c i} = h^c i$$

implies $hi = hhi = hh^c i = \emptyset_{1T}$ and similarly $h^c i = \emptyset_{1T}$. But since $h \sqcup h^c = 1_T$ we have $\emptyset = \overline{\emptyset \sqcup \emptyset} = \overline{hi \sqcup h^c i} = \overline{(h \sqcup h^c) i} = \bar{i} = 1_1$. By lemma 1.2.26(iv) this turns the restriction terminal object into a restriction initial object and makes the whole category trivial.

□

Combining the preceding lemma and the preceding theorem we also get that every m-complete set is undecidable.

Corollary 2.3.7

Let $\mathcal{R}$ be a nontrivial recursion category. Then every m-complete restriction idempotent is undecidable. In particular, given a Turing structure $\mathbb{T} = (T, \tau)$, the halting idempotent $\overline{\bullet}$ is undecidable.

Proof Let e be some m-complete restriction idempotent, in particular the halting idempotent $\overline{\bullet^{(0)}}$ m-reduces to e. Since $\overline{\bullet^{(0)}}$ is undecidable by the previous theorem, by lemma 2.3.5 e is undecidable. □

Another definition made in classical computability theory is that of *recursive inseparability* (cf. [Odi89] Def. II.2.4), which in some sense generalizes undecidability. The following is a straightforward reformulation in categorical terms.

Definition 2.3.8

In any recursion category two disjoint restriction idempotents e_1 and e_2 are **recursively inseparable**, if there is no decidable restriction idempotent e with $e_1 \subseteq e$ and $e_2 \subseteq e^c$ or equivalently, if there is no restriction idempotent e satisfying $e_i \subseteq e$ and $e_j e = \emptyset$ for distinct $i, j \in \{1, 2\}$.

In the classical setting such recursively inseparable sets do exist (cf. [Odi89] Thm.II.2.5) and again this generalizes to Turing categories.

Theorem 2.3.9

In a nontrivial recursion category $\mathcal{R}$ with Turing structure $\mathbb{T} = (T, \tau)$ there are recursively inseparable restriction idempotents.

Proof (cf. [Coc10] Thm. 7.8)

First note that in the presence of meets a code $\mathtt{p}\colon 1 \longrightarrow T$ can be identified with the morphism $\widehat{\mathtt{p}} \coloneqq \mathtt{p}!_T \cap 1_T \colon T \longrightarrow T$, which by lemma 1.2.4(ii) is a restriction idempotent.[7]

Since $\mathcal{R}$ is nontrivial, we have two distinct codes $\mathtt{p}_1, \mathtt{p}_2\colon 1 \longrightarrow T$ giving rise to disjoint corresponding morphisms $\widehat{\mathtt{p}_1}$ and $\widehat{\mathtt{p}_2}$. To see this note that we have the equation

$$\widehat{\mathtt{p}_1}\widehat{\mathtt{p}_2} = (\mathtt{p}_1!_T \cap 1_T)(\mathtt{p}_2!_T \cap 1_T) = \mathtt{p}_1!_T \cap \mathtt{p}_2!_T \cap 1_T = (\mathtt{p}_1 \cap \mathtt{p}_2)!_T \cap 1_T.$$

So take codes $\mathtt{z}\colon 1 \longrightarrow T$ for $\emptyset_T$ and $\mathtt{i}\colon 1 \longrightarrow T$ for 1_T and denote their meet by $\mathtt{z} \cap \mathtt{i} = t$. We calculate

$$\bullet(t \times T) = \bullet(\mathtt{z}\bar{t} \times T) = \bullet(\mathtt{z} \times T)(\bar{t} \times T) = \emptyset(\bar{t} \times T) = \emptyset \times T.$$

At the same time we have

$$\bullet(t \times T) = \bullet(\mathtt{i}\bar{t} \times T) = \bullet(\mathtt{i} \times T)(\bar{t} \times T) = 1_T(\bar{t} \times T) = \bar{t} \times T.$$

As $!_{1\times T}$ is split epi with right inverse $\langle 1_1, \mathtt{i}\rangle$ we can derive from $\emptyset = \bar{t} \times T = \overline{t \times T} = \overline{t!_{1\times T}}$ that $\emptyset_1 = \bar{t}$ and hence $t = \mathtt{z} \cap \mathtt{i} = \emptyset_{1T}$.

Now let $\mathtt{p}_1, \mathtt{p}_2$ be two such codes and define $k_i \coloneqq \overline{\widehat{p_i} \bullet \varDelta}$ for $i = 1, 2$, i.e. the restriction idempotent representing those elements, which applied to themselves return p_i. The restriction idempotents k_1 and k_2 are disjoint, since

$$k_1 k_2 = \overline{\widehat{p_1} \bullet \varDelta \overline{\widehat{p_2} \bullet \varDelta}} = \overline{\widehat{p_1}\widehat{p_2} \bullet \varDelta} = \emptyset_T.$$

Assume there is a complemented restriction idempotent $u\colon T \longrightarrow T$ with $k_1 \subseteq u, k_2 \subseteq u^c$.

Define $q = p_2!_T u \cup p_1!_T u^c$, which is total as $\bar{q} = \overline{p_2!_T u} \cup \overline{p_1!_T u^c} = u \cup u^c = 1_T$. Let $\mathtt{q}$ be a code for q, i.e. $q = \bullet(\mathtt{q} \times T)$. We calculate

$$\begin{aligned} k_1\mathtt{q} &= \overline{\widehat{p_1} \bullet \varDelta}\,\mathtt{q} = \mathtt{q}\overline{\widehat{p_1} \bullet \varDelta\mathtt{q}} = \mathtt{q}\overline{\widehat{p_1} \bullet (\mathtt{q} \times T)\mathtt{q}} = \mathtt{q}\overline{\widehat{p_1}(p_2!_T u \cup p_1!_T u^c)\mathtt{q}} \\ &= \overline{\widehat{p_1}p_2!_T u \cup \widehat{p_1}p_1!_T u^c}\,\mathtt{q} = \overline{\emptyset_T \cup p_1 u^c}\,\mathtt{q} = \mathtt{q}\overline{p_1 u^c \mathtt{q}} = u^c\mathtt{q} \end{aligned}$$

and thus $u^c\mathtt{q} = u^c u^c \mathtt{q} = u^c k_1 \mathtt{q} = \emptyset_{1T}$. Similarly one can show $k_2\mathtt{q} = u\mathtt{q}$ and $u\mathtt{q} = uu\mathtt{q} = uk_2\mathtt{q} = \emptyset_{1T}$. But now $1_1 = \overline{\mathtt{q}} = \overline{(u \cup u^c)\mathtt{q}} = \overline{u\mathtt{q}} \cup \overline{u^c\mathtt{q}} = \emptyset_1$. From this we can deduce by lemma 1.2.26(iv) that $\mathcal{R}$ is trivial. A contradiction.

Therefore k_1 and k_2 are recursively inseparable restriction idempotents. □

Another object of consideration in classical computability theory are so called *index sets*. These are sets of the form $X = \{n \in \mathbb{N} \mid \varphi_n \in F\}$, where $F \subseteq \mathsf{Rec}(\mathbb{N}, \mathbb{N})$ is a family of partial recursive functions. An equivalent characterization of such a set X is that it satisfies that if $\varphi_m = \varphi_n$ then it holds that $m \in X$ if and only if $n \in X$. The following definition given in [Coc10] generalizes this.

Definition 2.3.10

Let $\mathcal{R}$ be a recursion category with Turing structure $\mathbb{T} = (T, \tau)$.

A restriction idempotent $e\colon T \longrightarrow T$ is a **index idempotent**, if for all parallel morphisms $f, g\colon T^n \longrightarrow T$ in $\mathcal{R}$ the assertion $ef = f$ and $\bullet(f \times T) = \bullet(g \times T)$ implies $eg\overline{f} = g\overline{f}$.

An index idempotent is **nontrivial**, if there is a point $\mathtt{p}$ with $e\mathtt{p} = \mathtt{p}$ (in particular $e \neq \emptyset_T$) and a point $\mathtt{q}$ with $e^c\mathtt{q} = \mathtt{q}$ (in particular $e \neq 1_T$).

Note that due to problems with the restriction structure it does not suffice to make the definition completely analogous to the classical one by considering codes only. Indeed we will need that it is applicable to more than codes. However it really is a generalization of an index set, as the following remark shows. Confer the codes $\mathtt{p}, \mathtt{q}$ with the indices m, n from above.

Remark 2.3.11

Note that for any restriction idempotent e we have for any code $\mathtt{p}$ with $e\mathtt{p}$ being total that $e\mathtt{p} = \mathtt{p}$. Hence, if $\mathtt{p}, \mathtt{q}$ are points with $e\mathtt{p} = \mathtt{p}$ and $\bullet(\mathtt{p} \times T) = \bullet(\mathtt{q} \times T)$, then $e\mathtt{q} = \mathtt{q}$.

The following lemma is clear in the classical setting but needs some work in the context of Turing categories.

Lemma 2.3.12

Let $\mathcal{R}$ be a recursion category with Turing structure $\mathbb{T} = (T, \tau)$.

If e is a complemented index idempotent in $\mathcal{R}$, so is its complement e^c.

Proof (cf. [Coc10] Lem 7.10)

Let f, g be parallel morphisms satisfying $e^c f = f$ and $\bullet(f \times T) = \bullet(g \times T)$. Since we have the identity $g\overline{f} = eg\overline{f} \sqcup e^c g\overline{f}$, it suffices to show $eg\overline{f} = \emptyset$.

Now $eeg\overline{f} = eg\overline{f}$ and

$$\bullet(eg\overline{f} \times T) = \bullet(g\overline{eg}\,\overline{f} \times T) = \bullet(g \times T)(\overline{eg}\,\overline{f} \times T) = \bullet(f \times T)(\overline{f}\,\overline{eg} \times T) = \bullet(e^c f\overline{eg} \times T),$$

so, because e is an index idempotent, we get $e^c f\overline{eg}\,\overline{eg\overline{f}} = ee^c f\overline{eg}\,\overline{eg\overline{f}} = \emptyset$. Moreover we have $\overline{e^c f\overline{eg}} = \overline{f\overline{eg}} \subseteq \overline{eg\overline{f}}$, so in fact $f\overline{eg} = \emptyset$ holds. From this we can deduce $\overline{f\overline{eg}} = \overline{f}\,\overline{eg} = \emptyset$, hence $eg\overline{f} = eg\overline{eg}\,\overline{f} = \emptyset$. □

A very important result of classical computability theory is Rice's theorem. Informally it tells us that every nontrivial property of computable functions is undecidable. More formally a property can be identified with the index set of all those computable functions, which possess this property. Then Rice's theorem states that every nontrivial index set is undecidable.

[7] In Par the morphism $\widehat{p}$ is the partial identity on the set T, which is only defined on the element p. Indeed, this can be obtained by intersecting the total, constant function sending everything in T to p with the identity function on T.

Theorem 2.3.13 (Rice's Theorem)
Let $\mathcal{R}$ be a nontrivial recursion category with Turing structure $\mathbb{T} = (T, \bullet)$ and be a subcategory of a discrete cartesian restriction category $\mathcal{C}$ with joins. Then every nontrivial index idempotent is undecidable.

Proof (cf. [Coc10] Thm. 7.11)

Suppose there exists a nontrivial and complemented index idempotent in $\mathcal{R}$. We show that $\mathcal{R}$ is trivial.

Let $\mathtt{p}, \mathtt{q}$ be two codes satisfying $e\mathtt{p} = \mathtt{p}$ and $e^c\mathtt{q} = \mathtt{q}$. Define the morphism

$$f := \bullet(\mathtt{p} \times T)\pi_1(e^c \times T) \sqcup \bullet(\mathtt{q} \times T)\pi_1(e \times T)$$

By the second recursion theorem 2.1.8 there is a code $\mathtt{c}$ satisfying

$$\begin{aligned}\bullet(\mathtt{c} \times T) &= f(\mathtt{c} \times T)\\ &= (\bullet(\mathtt{p} \times T)\pi_1(e^c \times T) \sqcup \bullet(\mathtt{q} \times T)\pi_1(e \times T))(\mathtt{c} \times T)\\ &= \bullet(\mathtt{p} \times T)\pi_1(e^c\mathtt{c} \times T) \sqcup \bullet(\mathtt{q} \times T)\pi_1(e\mathtt{c} \times T)\\ &= \bullet(\mathtt{p} \times T)(\overline{e^c\mathtt{c}} \times T) \sqcup \bullet(\mathtt{q} \times T)(\overline{e\mathtt{c}} \times T).\end{aligned}$$

With this we can calculate

$$\begin{aligned}\bullet(e\mathtt{c} \times T) &= \bullet(\mathtt{c} \times T)(\overline{e\mathtt{c}} \times T)\\ &= \bullet(\mathtt{p} \times T)(\overline{e^c\mathtt{c}} \times T)(\overline{e\mathtt{c}} \times T) \sqcup \bullet(\mathtt{q} \times T)(\overline{e\mathtt{c}} \times T)(\overline{e\mathtt{c}} \times T)\\ &= \bullet(\mathtt{q} \times T)(\overline{e\mathtt{c}} \times T)\\ &= \bullet(\mathtt{q}\overline{e\mathtt{c}} \times T).\end{aligned}$$

Since $ee\mathtt{c} = e\mathtt{c}$ and e is a restriction idempotent we have $e\mathtt{q}\overline{e\mathtt{c}} = \overline{q}\,\overline{e\mathtt{c}}$. But $e\mathtt{q}\overline{e\mathtt{c}} = \emptyset\overline{e\mathtt{c}} = \emptyset$, so we get $e\mathtt{c} = \emptyset$. By symmetry we also have $e^c\mathtt{c} = \emptyset$. But since $\mathtt{c} = (e \sqcup e^c)\mathtt{c} = \emptyset$ is total, we can deduce by lemma 1.2.26(iv) that $\mathcal{R}$ is trivial. ☐

§ Relative Complexity

Now let us define some sort of complexity within Turing categories. The intuition behind our approach is that the codes for a computable function f represent all the algorithms, which compute f. If we have some resource, like time or space, we can compare algorithms by inspecting how they use the resource on a given input and return the output of that algorithm, which uses the resource in a more efficient way. The following definition tries to capture this in the setting of PCAs. Because it requires the representation of computable morphisms via λ^*-expressions, we fix codes $\mathtt{k}$ and $\mathtt{s}$ (cf. 2.1.4).

Definition 2.3.14
Let $\mathbb{A} = (A, \bullet, \mathfrak{a}, \mathtt{k}, \mathtt{s})$ be a PCA in a discrete cartesian restriction category $\mathcal{C}$ with joins. A **complexity comparator** on $\mathbb{A}$ is an interleaving morphism $\theta\colon A \times A \longrightarrow A$ (in the sense of 2.2.11), which satisfies the following conditions.

- There is a code $\mathtt{i}$ for the identity 1_A such that for any other code $\mathtt{a}$ the interleaved code $\theta\langle \mathtt{i}, \mathtt{a}\rangle$ is a code for the identity 1_A.
- Let $f\colon A^m \longrightarrow A^n$ and $g\colon A^n \longrightarrow A$ be two $\mathbb{A}$-computable morphisms, let $\mathtt{g}$ be a code for g and let $\mathtt{f}_i$ be a code for the component f_i of f. Recall, that

$$\mathtt{c} := [\lambda^* x_1 ... x_m . \mathtt{g}(\mathtt{f}_1 x_1 ... x_m) ... (\mathtt{f}_n x_1 ... x_m)]$$

  is a code for the composite gf, which depends on the choice of $\mathtt{s}$ and $\mathtt{k}$.

Now if there exists a code $\mathtt{r}$ such that $\theta\langle \mathtt{c}, \mathtt{r}\rangle$ is a code for the composite gf, then for every component f_i the code $\mathtt{f}_i$ satisfies that $\theta\langle \mathtt{f}_i, \mathtt{r}\rangle$ is a code for f_i. We will refer to this as θ being *monotone*.

Having such a complexity comparator at hand we can use a set of codes as reference or bounds for defining a complexity class. For example in the classical setting, if we identify codes with Turing machines $(T_n)_{n\in\mathbb{N}}$ and use time as resource, we can define $\mathsf{TIME}\langle m, n\rangle$ to return the code k of simulating both Turing-machines in parallel and returning the output, which is computed faster. Clearly this construction satisfies the first condition, since we can use the Turing-machine i, which returns the input immediately. Moreover the Turing machine c, which operates the Turing-machine g on the output of the Turing-machine f satisfies that it takes at least so long as running f solely. Hence the monotonicity condition is satisfied as well.

Now if we take a set of Turing-machines, say the Turing-machines (or better, algorithms) computing the *time-constructible* polynomials, then we obtain the class PTIME of all computable functions, which possess a polynomial-time algorithm, just by comparing algorithms. This leads us to the following definition.

Definition 2.3.15

Let $\mathbb{A} = (A, \bullet, \mathfrak{a}, \mathtt{k}, \mathtt{s})$ be a PCA in a discrete cartesian restriction category $\mathcal{C}$ with joins. Let further $\partial\mathscr{C}$ be a subset of $\mathfrak{a}$ and $\theta\colon A \times A \longrightarrow A$ be a complexity comparator. We say a morphism $f\colon A^n \longrightarrow A$ has **θ-complexity** less than $\partial\mathscr{C}$, if there is a code $\mathtt{f}$ for f and a **bounding code** $\mathtt{b} \in \partial\mathscr{C}$ such that $\theta\langle \mathtt{f}, \mathtt{b}\rangle$ is a code for f. A general morphism $f\colon A^m \longrightarrow A^n$ has **θ-complexity** less than $\partial\mathscr{C}$, if there is a **bounding code** $\mathtt{b} \in \partial\mathscr{C}$ such that for every component f_i there is a code $\mathtt{f}_i$ such that $\theta\langle \mathtt{f}_i, \mathtt{b}\rangle$ is a code for f_i. The **θ-complexity class bounded by** $\partial\mathscr{C}$ is the set $\mathscr{C}$ of all those morphisms with θ-complexity less than $\partial\mathscr{C}$.

A **categorical** complexity class is a complexity class $\mathscr{C}$, which is

- *closed under composition* in the sense that for morphisms f, g with codes $\mathtt{f}, \mathtt{g}$ bounded by codes $\mathtt{b}', \mathtt{b}''$ in $\partial\mathscr{C}$ there is a bounding morphism $\mathtt{b}$ for the given codes $\mathtt{c}_i$ of the composite gf.
- *closed under restriction* in the sense that for any morphism f in $\mathscr{C}$ with bounding code $\mathtt{b}$ we also have that $\mathtt{b}$ is a bounding code for $\overline{f}$.

A complexity class is **cartesian**, if it is categorical and closed under products with A.

Note that $\mathscr{C}$ being categorical requires $\partial\mathscr{C}$ to have enough bounding codes. A categorical complexity class is in fact a subcategory of the ambient category of computable maps, as it contains the identities by definition of θ and morphisms in it can be composed by assumption.

In the following we will turn our focus on restriction idempotents within complexity classes. Here it is important to note that restriction idempotents inherently represent partial functions. Thus the example of PTIME cannot be taken as source of intuition, since there the bounding codes represent total functions. However we can adjust the example by allowing all partial functions, which hold in polynomial time when defined.

To avoid notational clutter let us fix a PCA $\mathbb{A} = (A, \bullet, \mathfrak{a}, \mathtt{k}, \mathtt{s})$, a complexity comparator θ and the ambient discrete cartesian restriction category $\mathcal{C}$. From now on we will assume further that $\mathcal{C}$ has joins and complements.

Our first result will be that, under mild assumptions, our notion of m-reduction works well with this definition of complexity class.

Lemma 2.3.16

Let $\mathscr{C}$ be a categorical θ-complexity class and e_2 a restriction idempotent in $\mathscr{C}$. If $e_1 \leq_m e_2$ via a morphism in $\mathscr{C}$, then e_1 has θ-complexity $\mathscr{C}$.

Proof By assumption there is a total morphism f in $\mathscr{C}$ such that the following diagram commutes.

$$\begin{tikzcd} A^m \arrow[r, "f"] \arrow[d, "e_1"'] & A^n \arrow[d, "e_2"] \\ A^m \arrow[r, "f"'] & A^n \end{tikzcd}$$

Further there is a bounding code $\mathtt{b}$ and for every component of $e_2 f$ a code $\mathtt{c}_i$ such that $\theta\langle \mathtt{c}_i, \mathtt{b}\rangle$ is a code for c_i. As $\mathscr{C}$ is closed under restriction we have that $\mathtt{b}$ is a bounding code for $\overline{e_2 f} = \overline{f e_1} = e_1$. This shows $e_1 \in \mathscr{C}$. □

This leads to the following definition.

Definition 2.3.17

Let $\mathscr{C}$ be a categorical θ-complexity class.

An m-reduction with a morphism in $\mathscr{C}$ will be called a **$\mathscr{C}$-reduction** and denoted by $e_1 \leq_{\mathscr{C}} e_2$. A restriction idempotent e is **$\mathscr{C}$-complete**, if every restriction idempotent in $\mathscr{C}$ $\mathscr{C}$-reduces to it.

In classical computability theory one often tries to classify (not necessarily computable) sets by relating them via representations as formulas. More specifically, starting with a given class $\Sigma_0 = \Pi_0$ of sets one inductively defines a set $A \subseteq \mathbb{N}^k$ to be in Π_{n+1} if and only if there is a $u \in \mathbb{N}_0$ and a set $S \subseteq \mathbb{N}^{k+u}$ in Σ_n such that $A = \{a \in \mathbb{N}^k \mid \forall b \in \mathbb{N}^u \colon (a, b) \in S\}$ or equivalently $A \times \mathbb{N}^u \subseteq S$. Meanwhile a set $A \subseteq \mathbb{N}^k$ is defined to be in Σ_{n+1}, when its complement $\mathbb{N}^k \setminus A$ is in Π_{n+1}. A prominent example is that of the *arithmetical hierarchy* [Rob15].

We now seek to lift this construction in the categorical setting.

Definition 2.3.18

Let $\mathscr{C}$ be a cartesian θ-complexity class. The **(alternating) hierarchy over** $\mathscr{C}$ is defined inductively as follows.

Set $\Sigma_0^{\mathscr{C}} = \Pi_0^{\mathscr{C}}$ be the set of all restriction idempotents in $\mathscr{C}$.

A restriction idempotent e on A^k in $\mathcal{C}$ is in $\Pi_{n+1}^{\mathscr{C}}$, when there exists some $u \in \mathbb{N}_0$ and a restriction idempotent s on A^{k+u} in $\Sigma_n^{\mathscr{C}}$, such that $e \times A^u \subseteq s$.

A restriction idempotent e on A^k is in $\Sigma_{n+1}^{\mathscr{C}}$, if its complement e^c is in $\Pi_{n+1}^{\mathscr{C}}$.

The set of restriction idempotents evolving from the union of all $\Sigma_n^{\mathscr{C}}$ and $\Pi_n^{\mathscr{C}}$ will be denoted by $\mathscr{C}H$.

The name hierarchy has a reason, as we will see now.

Lemma 2.3.19

Let $\mathscr{C}$ be a cartesian θ-complexity class. Then for any $n \in \mathbb{N}_0$ we have

(i) $\Sigma_n^{\mathscr{C}} \subseteq \Pi_{n+1}^{\mathscr{C}}$ and $\Pi_n^{\mathscr{C}} \subseteq \Sigma_{n+1}^{\mathscr{C}}$.

(ii) $\Pi_n^{\mathscr{C}} \subseteq \Pi_{n+1}^{\mathscr{C}}$ and $\Sigma_n^{\mathscr{C}} \subseteq \Sigma_{n+1}^{\mathscr{C}}$.

Proof (i) Taking $u = 0$ and $s = e$ we get that if $e \in \Sigma_n^{\mathscr{C}}$ also $e \in \Pi_{n+1}^{\mathscr{C}}$. Further, if $e \in \Pi_n^{\mathscr{C}}$ then $e^c \in \Sigma_n^{\mathscr{C}}$, so $e^c \in \Pi_{n+1}^{\mathscr{C}}$ and thus $e \in \Sigma_{n+1}^{\mathscr{C}}$.

(ii) *by induction*

(IB) For any restriction idempotent $e \colon A^k \longrightarrow A^k$ in $\Pi_0^{\mathscr{C}}$ we can simply take $s = 1_{A^{k+1}}$. Analogously to (i) we have $\Sigma_0^{\mathscr{C}} \subseteq \Sigma_1^{\mathscr{C}}$.

(IH) Let the assumption hold for some $n \in \mathbb{N}$, i.e. for all $m < n$ we have $\Sigma_m^{\mathscr{C}} \subseteq \Sigma_{m+1}^{\mathscr{C}}$ and $\Pi_m^{\mathscr{C}} \subseteq \Pi_{m+1}^{\mathscr{C}}$.

(IS) Let $e\colon A^k \longrightarrow A^k$ be a restriction idempotent in $\Pi_n^{\mathscr{C}}$ with corresponding restriction idempotent $s\colon A^{k+u} \longrightarrow A^{k+u}$ in $\Sigma_{n-1}^{\mathscr{C}}$ for some $u \in \mathbb{N}_0$. By induction hypothesis (IH) s is in $\Sigma_n^{\mathscr{C}}$ and thus e is in $\Pi_{n+1}^{\mathscr{C}}$.

Like seen in (i), if e is in $\Sigma_n^{\mathscr{C}}$, e^c is in $\Pi_n^{\mathscr{C}}$ and also in $\Pi_{n+1}^{\mathscr{C}}$, so e is in $\Sigma_{n+1}^{\mathscr{C}}$. $\square$

The preceding lemma justifies the name alternating hierarchy. Indeed we can depict it like in the following image, using $\Delta_n^{\mathscr{C}} := \Sigma_n^{\mathscr{C}} \cap \Pi_n^{\mathscr{C}}$.

$$\begin{tikzcd}
 & & \Sigma_1^{\mathscr{C}} \arrow[dr, hook] & & \Sigma_2^{\mathscr{C}} \arrow[dr, hook] & & \cdots \\
\Sigma_0^{\mathscr{C}} = \Delta_0^{\mathscr{C}} = \Pi_0^{\mathscr{C}} \arrow[r, hook] & \Delta_1^{\mathscr{C}} \arrow[ur, hook] \arrow[dr, hook] & & \Delta_2^{\mathscr{C}} \arrow[ur, hook] \arrow[dr, hook] & & \Delta_3^{\mathscr{C}} \arrow[ur, hook] \arrow[dr, hook] & \\
 & & \Pi_1^{\mathscr{C}} \arrow[ur, hook] & & \Pi_2^{\mathscr{C}} \arrow[ur, hook] & & \cdots
\end{tikzcd}$$

It is a classic result that this type of hierarchy collapses to a given layer n, if Σ_n and Π_n agree. The *arithmetical hierarchy* is known to be strict, so we have to refer to the *polynomial hierarchy* [AB09], which does not match our definition cause of occurring boundedness conditions. There a set A is in Π_{n+1}^p, if and only if there is some S in Σ_n such that A is given by the identity $A = \{a \mid \forall b, b \text{ polynomially bounded in } a\colon (a,b) \in S\}$. However the collapse theorem holds true for our definition as well.

Theorem 2.3.20

Let $\mathscr{C}$ be a cartesian θ-complexity class and $n \in \mathbb{N}$.

If $\Sigma_n^{\mathscr{C}} = \Pi_n^{\mathscr{C}}$ the hierarchy collapses to n, i.e. we have $\Sigma_m^{\mathscr{C}} = \Sigma_n^{\mathscr{C}} = \Pi_n^{\mathscr{C}} = \Pi_m^{\mathscr{C}}$ for all $m \geq n$.

Proof by induction

(IB) For $m = n$ the statement holds by assumption.

(IH) Suppose $\Sigma_{m-1}^{\mathscr{C}} = \Sigma_n^{\mathscr{C}} = \Pi_n^{\mathscr{C}} = \Pi_{m-1}^{\mathscr{C}}$ for some $m > n$.

(IS) We show $\Pi_m^{\mathscr{C}} \subseteq \Pi_n^{\mathscr{C}}$:

Let e be a restriction idempotent on A^k in $\Pi_m^{\mathscr{C}}$, i.e. there is a $u \in \mathbb{N}_0$ and a restriction idempotent s on A^{k+u} in $\Sigma_{m-1}^{\mathscr{C}}$ such that $e \times A \subseteq s$. By induction hypothesis (IH) we have that s is in $\Pi_n^{\mathscr{C}}$, so there is a $v \in \mathbb{N}_0$ and a restriction idempotent s' on A^{k+u+v} in $\Sigma_{n-1}^{\mathscr{C}}$ with $s \times A^v \subseteq s'$. In particular we have $e \times A^{u+v} \subseteq s'$, hence $e \in \Pi_n^{\mathscr{C}}$.

Again by using complements we immediately derive $\Sigma_m^{\mathscr{C}} \subseteq \Sigma_n^{\mathscr{C}}$. $\square$

Again, the hierarchy is stable under $\mathscr{C}$-reductions.

Proposition 2.3.21

Let $\mathscr{C}$ be a cartesian θ-complexity class. For any $n \in \mathbb{N}_0$ the following holds.

If e_2 is in $\Pi_n^{\mathscr{C}}$ and e_1 $\mathscr{C}$-reduces to e_2, then we have e_1 is in $\Pi_n^{\mathscr{C}}$. The same is true for $\Sigma_n^{\mathscr{C}}$.

Proof by induction

(IB) The case $n = 0$ was already handled in lemma 2.3.16.

(IH) Let the assumption hold for a given n, i.e. $\Sigma_n^{\mathscr{C}}$ and $\Pi_n^{\mathscr{C}}$ are closed with respect to m-reduction with morphisms in $\mathscr{C}$.

(IS) We first show that $\Pi^{\mathscr{C}}_{n+1}$ is stable under $\mathscr{C}$-reductions:

Let e_2 be a restriction idempotent on A^k in $\Pi^{\mathscr{C}}_{n+1}$ with corresponding restriction idempotent s_2 on A^{k+u} in $\Sigma^{\mathscr{C}}_n$ for some $u \in \mathbb{N}_0$. Let e_1 be a restriction idempotent on A^l with $e_1 \leq_m e_2$ via a morphism f in $\mathscr{C}$, i.e. $e_2 f = f e_1$. Using the fact that s_2 is an idempotent, we obtain the commuting diagram

$$\begin{tikzcd}
A^{l+u} \arrow[r, "f \times A^u"] \arrow[d, "\overline{s_2(f\times A^u)}"'] & A^{k+u} \arrow[d, "s_2"] \\
A^{l+u} \arrow[r, "f \times A^u"] & A^{k+u}
\end{tikzcd}$$

establishing an m-reduction of $s_1 := \overline{s_2(f \times A^u)}$ onto s_2 via $f \times A^u$. Moreover, because $f \times A^u$ is a morphism in $\mathscr{C}$ as the latter is cartesian, by induction hypothesis (IH) s_1 is a restriction idempotent in $\Sigma^{\mathscr{C}}_n$.

It is left to show $e_1 \times A^u \leq s_1$, i.e. $s_1(e_1 \times A^u) = e_1 \times A^u$. This can be calculated, making extensive use of lemma 1.2.4(ii).

$$\begin{aligned}
s_1(e_1 \times A^u) &= \overline{s_2(f \times A^u)}\,(e_1 \times A^u) = \overline{f \times A^u\,\overline{s_2(f \times A^u)}\,(e_1 \times A^u)} \\
&= \overline{s_2(e_2 \times A^u)(f \times A^u)} = \overline{(e_2 \times A^u)(f \times A^u)} = \overline{(f \times A^u)(e_1 \times A^u)} \\
&= e_1 \times A^u.
\end{aligned}$$

We can deduce that $\Sigma^{\mathscr{C}}_{n+1}$ is stable under $\mathscr{C}$-reductions by applying lemma 2.3.2 on the statement for $\Pi^{\mathscr{C}}_{n+1}$. □

Corollary 2.3.22
Let $\mathscr{C}$ be a cartesian θ-complexity class.

If there exists an $n \in \mathbb{N}$ with $\mathscr{C}H \subseteq \Sigma^{\mathscr{C}}_n$ or $\mathscr{C}H \subseteq \Pi^{\mathscr{C}}_n$, then for any $m > n$ we have the chain of equalities $\Sigma^{\mathscr{C}}_m = \Pi^{\mathscr{C}}_m = \mathscr{C}H$. In particular, if there is some restriction idempotent $e \in \mathscr{C}H$, which every restriction idempotent in $\mathscr{C}H$ $\mathscr{C}$-reduces onto, these equalities hold.

Proof Suppose $\mathscr{C}H \subseteq \Sigma^{\mathscr{C}}_n$, which implies $\Pi^{\mathscr{C}}_{n+1} \subseteq \Sigma^{\mathscr{C}}_n$. Since by lemma 2.3.19 $\Sigma^{\mathscr{C}}_n \subseteq \Pi^{\mathscr{C}}_{n+1}$ we have $\Sigma^{\mathscr{C}}_{n+1} = \Pi^{\mathscr{C}}_{n+1}$, so by theorem 2.3.20 the hierarchy collapses onto $\Sigma^{\mathscr{C}}_{n+1} = \Pi^{\mathscr{C}}_{n+1} = \mathscr{C}H$.

The remark then immediately follows from proposition 2.3.21. □

To conclude the discussion of this hierarchy we note that its construction does not really depend on having a complexity class, which gives the initial set of restriction idempotents. In fact, to make the definition work out, it suffices to have a set $\Sigma_0 = \Pi_0$ of restriction idempotents containing the identities. For example we might take the set of all complemented computable restriction idempotents. In the context of classical recursion theory this corresponds to taking the set of *recursive sets* as base level of the hierarchy, which leads to the definition of the *arithmetical hierarchy* [Rob15]. Yet we hesitate to say that our hierarchy over the complemented restriction idempotents is a generalized arithmetical hierarchy, since it remains to be shown that basic properties generalize.

§ Relative Computability

The last thing we seek to generalize is that of relative computability. In ordinary computability theory non-computable functions must exist, since there are countably many partial recursive functions but at the same time uncountably many partial functions from $\mathbb{N}$ to $\mathbb{N}$. Hence it is natural to wonder, how non-computable functions are related. Usually the concept of an *oracle* for some non-computable function is introduced and one tries to characterize the functions, which can be computed when having access to this oracle (cf. [Odi89] Def. II.3.1). However, when lifting this concept into categorical realms, the problem with this definition of relative computability is that the notion of oracle computation is intertwined with the underlying model of computability. Indeed, in the categorical setting it is not clear, why for a given PCA $\mathbb{A}$ and a non-computable morphism f there should exist a PCA $\mathbb{A}[f]$ on the same object, which extends the notion of $\mathbb{A}$-computability to that of $\mathbb{A}$-computability with an oracle for f. Hence we have to deviate to using simulations to express relative computability.

Definition 2.3.23

Let $\mathcal{C}$ be a cartesian restriction category and $\mathbb{A} = (A, \bullet, \mathfrak{a})$ be a PCA in $\mathcal{C}$. Given not necessarily $\mathbb{A}$-computable morphisms $f \colon A^{k_0} \longrightarrow A^{l_0}$ and $g_1 \colon A^{k_1} \longrightarrow A^{l_1}, ..., g_n \colon A^{k_n} \longrightarrow A^{l_n}$ we say that f is $\mathbb{A}[g_1, ..., g_n]$**-computable**, if for any PCA $\mathbb{B} = (B, \bullet, \mathfrak{b})$ in $\mathcal{C}$ and faithful simulation $\phi \colon \mathbb{A} \longrightarrow \mathbb{B}$ we have that whenever all morphisms $g_1, \ldots, g_n$ admit a simulation via ϕ, then f admits a simulation via ϕ.

This notion of relative computability is inspired by the result of Oosten (cf. [Oos08] Thm. 1.7.5), which essentially states that for any PCA $\mathbb{A}$ in Par and non-computable morphism f there is a PCA $\mathbb{A}[f]$ on the same object, such that there is a simulation $\iota \colon \mathbb{A} \longrightarrow \mathbb{A}[f]$ and which satisfies the following universal property. For every PCA $\mathbb{B}$, for which f admits a simulation via $\phi \colon \mathbb{A} \longrightarrow \mathbb{B}$ there is a unique simulation $\psi \colon \mathbb{A}[f] \longrightarrow \mathbb{B}$, such that the diagram on the right commutes. Clearly the existence of such a PCA might not be given in every arbitrary restriction category.

$$\begin{tikzcd} \mathbb{A} \arrow[r, "\iota"] \arrow[dr, "\phi"'] & \mathbb{A}[f] \arrow[d, dashed, "\exists! \psi"] \\ & \mathbb{B} \end{tikzcd}$$

Now, clearly every $\mathbb{A}$-computable morphism is $\mathbb{A}[g_1, ..., g_n]$ computable, because it admits a simulation via ϕ independent of having simulations for morphisms $g_1, ..., g_n$. At the same time it is certainly possible to have a completely uncomputable morphism in the sense that no PCA admits a simulation.

One primary focus of relative recursion lies on decidability relative to an oracle. This generalizes well, when speaking of an oracle for a function f. Meanwhile we have to point out the same problem discussed in the context of decidability. Usually one defines computability relative to a set E by using an oracle for the characteristic function χ_E. As a restriction idempotent e only resembles a semidecidable characteristic function for some set E, we have to assume to have an oracle for both e and e^c.

Definition 2.3.24

Let $\mathcal{C}$ be a cartesian restriction category containing a PCA $\mathbb{A} = (A, \bullet, \mathfrak{a})$. Given a morphism $g \colon A^k \longrightarrow A^l$, a complemented restriction idempotent e on A^n in $\mathcal{C}$ is $\mathbb{A}[g]$**-decidable**, if for any PCA $\mathbb{B} = (B, \bullet, \mathfrak{b})$ and faithful simulation $\phi \colon \mathbb{A} \longrightarrow \mathbb{B}$ where g admits a simulation via ϕ, both e and e^c admit a simulation via ϕ.

We say that a complemented restriction idempotent e_1 on A^n **Turing-reduces** to a complemented restriction idempotent e_2 on A^m, if e_1 is $\mathbb{A}[e_2, e_2^c]$-decidable. In this case we write $e_1 \leq_T e_2$.

The following observations are straightforward.

Lemma 2.3.25

Let $\mathbb{A} = (A, \bullet, \mathfrak{a})$ be a PCA in a cartesian restriction category $\mathcal{C}$.

(i) For any complemented restriction idempotent e on A^n we have that $e^c \leq_T e$ and $e \leq_T e^c$.

(ii) Given complemented restriction idempotents e_1 and e_2 on powers of A it holds that if $e_1 \leq_T e_2$ and e_2 is $\mathbb{A}$-decidable then e_1 is $\mathbb{A}$-decidable.

Proof (i) This holds by definition of relative decidability.

(ii) Since e_2 is $\mathbb{A}$-decidable and in particular admits a simulation via $1_A \colon \mathbb{A} \longrightarrow \mathbb{A}$, we have that both e_1 and e_1^c admit a simulation via 1_A and as such are $\mathbb{A}$-computable. □

As a last result we wish to show that the well known fact of m-reducibility implying Turing-reducibility (cf. [Rob15] Thm. 11.3) holds in our setting.

Proposition 2.3.26

Let $\mathbb{A} = (A, \bullet, \mathfrak{a})$ be a PCA in a cartesian restriction category $\mathcal{C}$ and let e_1, e_2 be two complemented restriction idempotents. Then $e_1 \leq_m e_2$ implies $e_1 \leq_T e_2$.

Proof Let $\mathbb{B} = (B, \bullet, \mathfrak{b})$ be another PCA in $\mathcal{C}$ and $\phi\colon \mathbb{A} \longrightarrow \mathbb{B}$ be a faithful simulation. We first show that for restriction idempotents e_1 and e_2 on objects A^n and A^m respectively, we have that if e_2 admits a simulation via ϕ and $e_1 \leq_m e_2$ then e_1 admits a simulation via ϕ.

By assumption we have $e_2 f = f e_1$ for some total $\mathbb{A}$-computable morphism $f\colon A^n \longrightarrow A^m$, which in particular admits a simulation $f'\colon B^n \longrightarrow B^m$ via ϕ. Furthermore e_2 admits a simulation $e_2'\colon B^m \longrightarrow B^m$ via ϕ. Now consider the following diagram.

$$\begin{tikzcd}
B^n \arrow[rrr, "f'"] \arrow[ddd, "\overline{e_2'f'}"'] & & & B^m \arrow[ddd, "e_2'"] \\
& A^n \arrow[ul, "\phi^n"] \arrow[r, "f"] \arrow[d, "e_1"'] & A^m \arrow[ur, "\phi^m"'] \arrow[d, "e_2"] & \\
& A^n \arrow[r, "f"'] \arrow[dl, "\phi^n"'] & A^m \arrow[dr, "\phi^m"] & \\
B^n \arrow[rrr, "f'"'] & & & B^m
\end{tikzcd}$$

Note that, besides the left square, the diagram commutes by assumption and that the restriction idempotent $\overline{e_2'f'}$ is $\mathbb{B}$-computable. It is left to show the commutativity of the left square, so we calculate

$$\overline{e_2'f'}\,\phi^n = \phi^n\overline{e_2'f'\phi^n} = \phi^n\overline{f'\phi^n e_1} = \phi^n e_1,$$

where the last equation holds since $f'\phi^n = \phi^m f$ is total. This exhibits $\overline{e_2'f'}$ as simulation of e_1 via ϕ.

From this observation we can now easily deduce the proposition, since by lemma 2.3.2 e_1^c m-reduces to e_2^c via f as well. $\square$

3 Conclusion

The first part of this thesis was devoted to developing a categorical framework, in which computability can be generalized. After introducing common terminology of categories and 2-categories, we explored restriction categories. Trying to reformulate concepts usually found in the category of sets and partial functions, we developed restriction (co)limits. Their discussion certainly sufficed for the introduction of abstract computability, however their behaviour showed us that the relationship between restriction categories and ordinary categories needs further investigation. Especially the definition of the restriction operator can be unsatisfying from a categorical point of view and may admit a reformulation in purely categorical terms. Besides these rather philosophical problems, the application of the theory of restriction categories to other parts of mathematics remains to be interesting.

In the second part we turned our focus on describing computability within restriction categories. We introduced and characterized applicative systems, partial combinatory algebras and Turing categories. Interestingly enough, their definition encompassed the intuition of Turing machines, properties of the lambda calculus and that of partial recursive functions. This reassured us that this abstract notion of computability incorporates the classical intuition. Filling in necessary details we found that many results of recursion theory can be expressed in this setting. We extended these observations by our own definition of complexity in a PCA, which essentially works by comparing codes. Moreover we gave a reformulation of the construction of alternating hierarchies in suitable restriction categories. Our definition of relative computability was inspired by the definition of relative recursion in [Oos08].

In the process of writing this thesis the following questions arose and remain unanswered.

- The set theoretic notion of a partial combinatory algebra comes with its own associated *categories of assemblies*, as are studied in [Oos08]. The author is unaware of any progress trying to connect Turing categories to them or related constructions.
- Similarly, to the author's knowledge there are no examples of Turing categories, which are not motivated by classical computability theory. In particular it is an interesting question, if Turing categories can be used to obtain other notions of computability, like that of topological or algebraic computations.
- In [CHH14] much work has been put into showing that certain categories of functions arising in complexity theory, like polynomial or linear time functions, can be rediscovered as total maps of Turing categories. Our approach to complexity is fundamentally different and seems to be more in line with other generalizations of classical results, discussed in this thesis. Nevertheless the author would like to know, whether there is a connection between these two accounts of complexity in Turing categories.
- Last but not least, we introduced a notion of relative computability. A direct reformulation of Oosten's construction of a PCA with oracle did not look convincing at a first glance. However its universal property suggests that there might be an underlying categorical construction, which we would have liked to work out.

To wrap up this conclusion we note that Turing categories are only one of many different attempts to understand the categorical side of computability theory. There have been and will certainly be many more interesting approaches to define what it means to be computable in a general sense.

A Appendix

A.1 Ordered Sets

The following definitions and lemmas are taken from [Sch03].

Definition A.1.1
A **preorder** on a set X is a binary relation $\preceq$ on X, which satisfies the following conditions.

- For all x in X it holds that $x \preceq x$. (*reflexivity*)
- For any x, y, z in X the assertions $x \preceq y$ and $y \preceq z$ imply $x \preceq z$. (*transitivity*)

A **partial order** is a preorder $\leq$, which further satisfies the following rule.

- For all x, y in X the assertions $x \leq y$ and $y \leq x$ imply $x = y$. (*antisymmetry*)

The pair $(X, \leq)$ is commonly called **poset**, abbreviating partial ordered set.

Definition A.1.2
Let $\leq$ be a partial order on some set X and A be a subset of X.

An element x of X is an *upper bound* of A, if for all a in A we have $a \leq x$. It is a **join** of A, if it is the *lowest upper bound*, i.e. if for any other upper bound y of A we have $x \leq y$, and a **maximum** of A, if it is a join contained in A. We denote a join of A by $\bigvee A$, if it is exists.

Similarly x is a **meet** of A, if it is the *greatest lower bound*, i.e. if for any other lower bound y of A we have $y \leq x$. A meet of A is denoted by $\bigwedge A$, if it exists.

If $A = \{a_1, ..., a_n\}$ is finite, one also writes $a_1 \vee ... \vee a_n$ and $a_1 \wedge ... \wedge a_n$ respectively.

It is important to note that meets and joins might not always exist. Furthermore there is a subtle difference between a *maximal element* of a set A, i.e. an element of A, for which no larger element in A exists, and a maximum. The latter needs to be comparable with any other element in A, while the former does not. One can make the following observations.

Lemma A.1.3
Let $(X, \leq)$ be a poset. Then the following assertions hold.

(i) Joins and meets are unique (if they exist).

(ii) For every element x in X we have $x \vee x = x$ and $x \wedge x = x$.

(iii) Taking joins / meets is associative in the sense that given subsets A and B of X we have the equalities $\bigvee\{\bigvee A, \bigvee B\} = \bigvee(A \cup B)$ and $\bigwedge\{\bigwedge A, \bigwedge B\} = \bigwedge(A \cup B)$, if the corresponding joins / meets exist. In particular for elements a, b, c in X we have the equalities $a \vee (b \vee c) = (a \vee b) \vee c$ and $a \wedge (b \wedge c) = (a \wedge b) \wedge c$, assuming the joins and meets exist.

Definition A.1.4
A poset, in which every nonempty finite subset possesses a join / meet, is a **join-semilattice** / **meet-semilattice**. If it has both finite joins and finite meets, it is called a **lattice**.

A join-semilattice / meet-semilattice / lattice, which has a largest / smallest / largest and a smallest element, is said to be **complete**.

B Bibliography

Computability Theory and Partial Combinatory Algebras

[AB09] S. Arora and B. Barak. *Computational Complexity: A Modern Approach.* 1st. New York, NY, USA: Cambridge University Press, 2009. ISBN: 0521424267, 9780521424264.

[HMJ06] J.E. Hopcroft, R. Motwani, and J.D.Ullman. *Introduction to Automata Theory, Languages, and Computation.* 3rd. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 2006. ISBN: 0321455363.

[Odi89] P. Odifreddi. *Classical Recursion Theory.* Vol. 125. Studies in Logic and the Foundations of Mathematics. North-Holland Publishing Co., Amsterdam, 1989, p. 668. ISBN: 0-444-87295-7.

[Oos08] J. van Oosten. *Realizability: An Introduction to its Categorical Side.* Vol. 152. Studies in Logic and the Foundations of Mathematics. North-Holland Publishing Co., Amsterdam, 2008, p. 328. ISBN: 9780444515841.

[Rob15] B. Robic. *The Foundations of Computability Theory.* 1st. Springer-Verlag Berlin Heidelberg, 2015, pp. XX, 331. ISBN: 978-3-662-44808-3, 978-3-662-44807-6, 978-3-662-51601-0. DOI: 10.1007/978-3-662-44808-3.

Category Theory

[Bor94] F. Borceux. *Handbook of Categorical Algebra.* Vol. 1. Encyclopedia of Mathematics and its Applications. Cambridge University Press, 1994, pp. XV, 345. DOI: 10.1017/CBO9780511525858.

[KS74] G.M. Kelly and R. Street. "Review of the elements of 2-categories". In: (1974), 75–103. Lecture Notes in Math., Vol. 420.

[Mac78] S. MacLane. *Categories for the Working Mathematician.* 2nd. Graduate Texts in Mathematics. Springer-Verlag New York, 1978, pp. XII, 317. ISBN: 978-0-387-98403-2. DOI: 10.1007/978-1-4757-4721-8.

[Rie14] E. Riehl. *Categorical Homotopy Theory.* New Mathematical Monographs. Cambridge University Press, 2014. DOI: 10.1017/CBO9781107261457.

[Str96] R. Street. "Categorical structures". In: *Handbook of Algebra* 1 (1996), pp. 529–577. ISSN: 1570-7954. DOI: 10.1016/S1570-7954(96)80019-2.

Restriction Categories and Turing Categories

[CGH12] J.R.B. Cockett, X. Guo, and P.J.W. Hofstra. "Range Categories II: Towards Regularity". In: vol. 26. 18. 2012, pp. 453–500.

[CH08] J.R.B. Cockett and P.J.W. Hofstra. "Introduction to Turing Categories". In: *Annals of Pure and Applied Logic* 156.2 (2008), pp. 183 –209. ISSN: 0168-0072.

[CHH14] J.R.B. Cockett, P.J.W. Hofstra, and P. Hrubeš. "Total Maps of Turing Categories". In: *Electronic Notes in Theoretical Computer Science* 308 (2014). Proceedings of the 30th Conference on the Mathematical Foundations of Programming Semantics (MFPS XXX), pp. 129 –146. ISSN: 1571-0661. DOI: https://doi.org/10.1016/j.entcs.2014.10.008.

[CL02] J.R.B. Cockett and S. Lack. "Restriction categories I: categories of partial maps". In: *Theoretical Computer Science* 270.1 (2002), pp. 223 –259. ISSN: 0304-3975.

[CL07] J.R.B. Cocket and S. Lack. "Restriction categories III: colimits, partial limits and extensivity". In: *Mathematical Structures in Computer Science* 17.4 (2007), 775–817. DOI: 10.1017/S0960129507006056.

[Coc10] J.R.B. Cockett. "Categories and Computability". university lecture. 2010.

[Vin12] P. Vinogradova. “Investigating Structure in Turing Categories”. MA thesis. University of Ottawa, 2012.

Others

[Pav12] Dusko Pavlovic. “Monoidal computer I: Basic computability by string diagrams”. In: *CoRR* abs/1208.5205 (2012). arXiv: `1208.5205`.

[Sch03] Bernd Schroeder. *Ordered Sets: An Introduction*. Birkhäuser Basel, 2003, p. 391. ISBN: 9781461265917.