Investigating Structure in Turing Categories

Polina Vinogradova

Thesis submitted to the Faculty of Graduate and Postdoctoral Studies
in partial fulfillment of the requirements for the degree of Master of Science in
Mathematics [1]

Department of Mathematics and Statistics
Faculty of Science
University of Ottawa

[1] The M.Sc. program is a joint program with Carleton University, administered by the Ottawa-Carleton Institute of Mathematics and Statistics

Abstract

The concept of a computable function is quite a well-studied one, however, it is possible to capture certain important properties of computability categorically. A special type of category used for this purpose is called a Turing category. This thesis starts with a brief overview of Turing categories, followed by a study of additional categorical structure they may contain, based on the types of structure found in the world of computable functions, and how this is reflected in the underlying combinatorial structures.

Acknowledgements

First of all, I would like to thank my supervisor, Dr. Pieter Hofstra, without whose structured approach to research and presentation of results I would never have completed this thesis, for his advice, guidance, and financial support. It has been a very positive learning experience working with him.

I also wish to thank the vice-dean of graduate studies in science at the University of Ottawa, Dr. Richard Blute, for approving my admission to the Master of Mathematics program at the University under special circumstances. I would not have had the opportunity to undertake this research project otherwise.

Finally, I would like to express my gratitude to Dr. Robin Cockett, professor of Mathematics at the University of Calgary, for providing me with the exciting opportunity to participate in the Foundational Methods in Computer Science 2011 workshop, and present the results of my research to a group of people largely responsible for the development of the theory on which it builds. Once again I wish to thank Dr. Hofstra for encouraging me to take part in the conference and helping me prepare for my presentation.

Contents

Introduction **2**

1 Overview of Turing Categories **6**

1.1 Restriction Categories 6

1.1.1 Cartesian Restriction Categories. 10

1.2 Turing Categories 11

1.3 Partial Combinatory Algebras 14

2 Ranges in Turing Categories **19**

2.1 Range Categories 19

2.2 Ranges and Retractions 23

2.3 The Beck-Chevalley Condition 27

2.4 Ranges and PCA's 28

3 Equality in Turing Categories **33**

3.1 Types of Categorical Equality 34

3.2 Equality in Turing Categories 38

3.3 Equality and PCA's 41

3.4 Equality and Ranges 43

3.5 Equality and Meets 46

4 Coproducts in Turing Categories 48
4.1 Coproducts in Cartesian Restriction Categories 48
4.2 Coproducts and Turing Structure 50
4.3 Distributivity 53
4.4 Biproducts 55
Conclusion 57
Bibliography 60

Introduction

Traditional computation on the natural numbers, as well as the abstract theory of computability built around it, has historically been the main type of computation studied in mathematics and computer science, mainly due to its widespread applicability. However, there have been several attempts at capturing the essence of computation through various mathematical abstractions. The focus of this thesis will be a categorical approach to describing computation which is quite general, and encompasses some of the existing abstractions.

Traditional Computation

The main focus of traditional computation is on the natural numbers and computable functions from $\mathbb{N}$ to $\mathbb{N}$. The collection of these computable maps can be enumerated (for example by systematically listing all possible Turing machines); it follows that every computable function f will have a code c_f in the enumeration. Then evaluating the function f at an argument $a \in \mathbb{N}$ can be represented by the mapping $(c_f, a) \mapsto f(a)$. The set of all possible computations can then be captured by a map $\bullet : \mathbb{N} \times \mathbb{N} \to \mathbb{N}$. This map applies the function coded by the first argument to the second argument, and is known as the universal computable function.

Furthermore, it is important to note that computations do not necessarily halt on every input, so that a given element of $\mathbb{N}$ may not be in the domain of the corresponding map. The domain of a partial computable function (which is a subset of $\mathbb{N}$)

is called a recursively enumerable set. Equivalently, recursively enumerable sets are precisely the ranges of computable functions.

A third property of the standard model of computation is that there are effective encodings of (enumerable subsets of) powers of the natural numbers into $\mathbb{N}$ itself, in the form of computable bijections $\mathbb{N}^n \cong \mathbb{N}$, for $n > 0$.

These results, as well as some other facts about traditional computabilty theory used in this thesis, can be found in [4].

Categorical Approaches

The collection of all recursively enumerable sets, together with the computable maps, forms a category. This category is a subcategory of sets and partial functions, Par, and inherits some of its structure, along with having the additional structure that comes with enumerability (for common category theory results found in this thesis, see [8]). When looking for a more general category to perform computation in, three properties can be highlighted from the above description of traditional computation:

1. Some kind of categorical structure capturing partiality
2. A special object A and a application $\bullet : A \times A \to A$ that represents all possible maps $A \to A$
3. Embeddings of every other object in the category into A

The categorical framework which incorporates these three features is called a Turing category, and is the main object of study in this thesis. The partiality in such a category is handled by means of idempotents: we may represent the domain of a partial map in the form of a specific type of idempotent. Then, the special object A and the application $\bullet$ generalize the idea of the universal computable function.

Finally, the condition that every object in a category is a retract of A generalizes the fact that all recursively enumerable sets of powers of $\mathbb{N}$ are isomorphic to subsets of $\mathbb{N}$.

The motivating example of a Turing category, as described above, has a lot of additional structure due to the very special nature of the natural numbers. Part of the motivation behind this study is exploring how we can bridge the gap between this specific example and the completely general setting, by adding structure to a Turing category in a controlled way.

Plan of the Thesis

This thesis is divided into chapters by the structure which they cover, starting with an overview of Turing categories, developed by Dr. Hofstra and Dr. Cockett in [2]. In the second chapter, the concept of range of a function is generalized: the partial identity map on $\mathbb{N}$, defined only on the range of some computable function, is abstracted by a special type of idempotent. The main problem we address is finding necessary and sufficient conditions for a Turing category to have ranges, and to understand what they look like.

In the third chapter, different types of equality are considered, again with the usual notion of integer equality as the basis. It is natural to represent the integer equality test as a map $\mathbb{N} \times \mathbb{N} \to \{0, 1\}$, with 0 output when two integers are equal, and 1 when they are not. This is one of the types of equality maps considered in the chapter, but, importantly, there are other types as well. The results presented are largely about the generalization of the different types of equality in a Turing category, when exactly they are present, and in what form.

The fourth chapter talks about a commonly encountered categorical structure - coproducts. Once more, coproducts are found in the standard model, and we want to understand what it means for a general Turing category to have coproducts. We investigate conditions under which Turing categories have coproducts. Other implications of having coproducts in a Turing category will be discussed.

Contributions

The key results in this thesis are concerned with understanding and characterizing Turing categories with additional categorical structure. Each of these describes conditions under which a Turing category will have the desired type of structure. The main results are:

(1.) The necessary and suffient condition for a Turing category to have ranges is formulated in terms of the underlying universal application.

(2.) The necessary and suffient condition for a Turing category to be discrete (have equality), is stated, again, in terms of the universal application.

(3.) An alternative sufficient condition for a Turing category to have ranges is formulated in terms of discreteness.

(4.) In a split Turing category, the necessary and sufficient conditions for existence of distributive coproducts are presented.

Chapter 1

Overview of Turing Categories

In this chapter, Turing categories are introduced, assuming familiarity with basic category theory (see [8]).

1.1 Restriction Categories

In order to explain what Turing structure is, what sort of category it can exist in, and how exactly it abstracts computation, this overview will begin by covering some concepts essential to this discussion. The first concept to be discussed is the notion of restriction categories, as defined in [3]. Consider the following definition:

Definition 1.1.1 A **restriction category** is a category C endowed with a combinator $\overline{(-)}$, sending $f : A \to B$ to $\overline{f} : A \to A$, such that the following axioms hold:

[**R.1**] $f\overline{f} = f$

[**R.2**] $\overline{f}\,\overline{g} = \overline{g}\,\overline{f}$ whenever $\mathrm{dom}(f) = \mathrm{dom}(g)$

[**R.3**] $\overline{g\overline{f}} = \overline{g}\,\overline{f}$ whenever $\mathrm{dom}(f) = \mathrm{dom}(g)$

[**R.4**] $\overline{g}f = f\overline{gf}$ whenever $\mathrm{cod}(f) = \mathrm{dom}(g)$

It follows immediately that when gf is defined, $\overline{\overline{g}f} = \overline{gf}$.

Example 1.1.2 In Par, the category of sets and partial functions, a $\overline{(-)}$ combinator, for a map $f : X \to Y$, may be defined as follows:

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

The map $\overline{f}$ corresponds to the domain $S \subseteq X$ of the map f in the sense that it is defined (and coincides with 1_X) exacly on that subset of X.

In this example, and subsequently, $f(x) \downarrow$ means that f is defined at x, and $\uparrow$ means it is undefined. Note that in this category the map $\overline{f}$ is actually a partial identity on X. Furthermore, with this example it is easy to see that a restriction combinator is not necessarily unique in a category — another possiblity for Par is simply $\overline{f} = 1_X$, which also satisfies all the axioms.

The concept of a restriction combinator prompts the definition of total function: a map $f : X \to Y$ is total whenever $\overline{f} = 1_X$. In particular, any embedding is total. Furthermore, it is possible to define a subcategory of C, Tot(C), consisting of all the objects of C, and all total maps of C.

Another important structure that naturally exists in a restriction category C is a partial ordering $\leq$ on hom-sets, defined as follows:

$$f \leq g \Rightarrow g\overline{f} = f$$

For example, in the case of sets and partial functions, $f \leq g$ means that the domain of f is a subset of the domain of g, and that $f(x) = g(x)$ for all x in the domain of f.

Often, there is a minimal map with respect to this ordering, denoted $0_{A,B} : A \to B, A, B \in$ C, such that for all $f : A \to B, 0_{A,B} \leq f$. In the case of sets and partial functions, this amounts to the map $A \to B$ which is undefined for all $a \in A$.

Splitting Idempotents. Because idempotents are an integral part of the focus, it is necessary to define what an idempotent is, and what it means for one to split, for reference see [9]. The following definition combines traditional idempotent concepts with restriction structure, defined in [3].

Definition 1.1.3 Let C be a category, and $e : X \to X$ — a map in C. Then

(i) e is said to be an **idempotent** when $ee = e$

(ii) e is said to **split** when there exists an object $Y \in \mathsf{C}$, as well as maps $m : Y \to X$ and $r : X \to Y$ such that $e = mr$, and $rm = 1_Y$.

(iii) idempotents e, e' are said to be **equivalent** when they have isomorphic splittings

(iv) e is a **restriction idempotent** when $\overline{e} = e$.

Note that all restriction maps $\overline{f} : A \to A$ are restriction idempotents, and $\overline{f} = f$ implies $ff = f\overline{f} = f$. Furthermore, each restriction idempotent e is the restriction of some map (such as e itself).

From this point on, it will sometimes be assumed that all idempotents in a category C split. Alternatively, Split(C) will denote the category obtained from C by formally splitting all idempotents (see [9] for details). The objects in this category are pairs of the form (A, e) where $A \in \mathsf{C}$ and $e : A \to A$ is an idempotent of C, and whose morphisms are triples of the form

$$(e, f, e') : (A, e) \to (A', e')$$

where $f : A \to A'$ is a morphism of C satisfying $e'f = f = fe$. In the category Split(C), the splitting of an idempotent e may now be given by the object (A, e), with an embedding $(e, e, 1)$ and retraction (e, e, e). As with any idempotent splitting, the object (A, e) is unique up to isomorphism.

Example 1.1.4 Let $\mathbb{N}$ denote the one-object category of natural numbers and partial recursive functions. Here, restriction structure is inherited from Par, and restriction idempotents in this category correspond to subsets of $\mathbb{N}$ (i.e. recursively enumerable sets), which are exactly the the domains of partial recursive functions. For a general idempotent $e : \mathbb{N} \to \mathbb{N}$, the image of e is again a recursively enumerable set. Thus, the images of idempotents can be represented by recursively enumerable subsets, and the objects of $\mathrm{Split}(\mathbb{N})$ may be taken to be the r.e. subsets.

Open Maps. The category of topological spaces and partial continuous functions with open domains is another noteworthy example of a restriction category. There is a special class of maps in this category, namely the *open maps*: a continuous function $f : X \to Y$ is called open when the direct image of an open set in X under f is an open set in Y (see [7]). The concept of open maps has been previously studied in the categorical setting — see [10].

In [5], it has been shown how the concept of openness can be defined in a general restriction category. Let $\mathcal{O}(A)$ denote the poset of restriction idempotents of an object A in a restriction category C. Then, for $f : A \to B$, let $f^* : \mathcal{O}(B) \to \mathcal{O}(A)$ denote the "inverse image" function such that for any $e \in \mathcal{O}(B), f^*(e) = \overline{ef} \leq \overline{f}$.

Furthermore, composing restriction idempotents is denoted by $ee' = e \wedge e'$. This $\wedge$ operation is actually the meet in the poset $\mathcal{O}(A)$; a definition which extends this to more general types of maps will be given in the "Equality" chapter.

Definition 1.1.5 In a restriction category, a map $f : A \to B$ is **open** if and only if there is a poset morphism $\exists_f : \mathcal{O}(A) \to \mathcal{O}(B)$ such that

[O1] $\exists_f(f^*(e')) \leq e'$ for all $e' \in \mathcal{O}(B)$

[O2] $e \wedge f^*(e') \leq f^*(\exists_f(e) \wedge e')$ for all $e \in \mathcal{O}(A), e' \in \mathcal{O}(B)$

[O3] $e' \wedge \exists_f(e) \leq \exists_f(f^*(e') \wedge e)$ for all $e \in \mathcal{O}(A), e' \in \mathcal{O}(B)$

In the case of Par, identifying restriction idempotents with the subsets of $A \in \mathsf{Par}$ they restrict A to,

$$f^*(e) = \{x \in A : f(x) \in e\} \qquad \exists_f(e') = \{f(x) : x \in e'\}$$

Note that generally, in a restriction category, the map f^* actually lets us recover the restriction of f, as $f^*(e) = \overline{ef}$, and taking $e = 1$ gives the result $f^*(1) = \overline{f}$. The map $\exists_f$ is almost left adjoint to the f^* map; however, since f may be partial this is only true when we restrict $\exists_f$ to the domain of f. The O1 axiom gives the counit of the almost-adjunction. The next 2 axioms make sure the 2 maps behave correctly with respect to composition with idempotents.

With this open map definition, it turns out that in any restriction category, all restriction idempotents are open maps, and (see [5])

Lemma 1.1.6 Suppose $f : A \to B$ and $g : B \to C$ are open maps in a cartesian restriction category. Then the composition gf is also an open map.

These facts will prove quite useful in exploring the dual concept of the restriction combinator — the range combinator, in the next chapter. Now, the upcoming definitions provide the setting for an even more specific class of categories that Turing categories are a part of — cartesian restriction categories, described in [2].

1.1.1 Cartesian Restriction Categories.

Definition 1.1.7

(i) Let 1 be an object in a restriction category C. An object 1 is **restriction terminal** when for any object $A \in \mathsf{C}$, there exist a unique total map $!_A : A \to 1$, such that for every $f : A \to B$, $!_B f = !_A \overline{f}$.

(ii) A **partial product** of two objects $A, B \in \mathsf{C}$ is an object $A \times B$, together with total projections $\pi_A : A \times B \to A, \pi_B : A \times B \to B$ such that for each pair of maps

$f : C \to A, g : C \to B$ there exists a unique map $\langle f, g \rangle : C \to A \times B$ such that $\pi_A \langle f, g \rangle \leq f, \pi_B \langle f, g \rangle \leq g$, and $\overline{\langle f, g \rangle} = \overline{f}\overline{g}$.

This definition implies that a restriction terminal object in C is a genuine terminal object in $\mathrm{Tot}(\mathsf{C})$, and a restriction product $A \times B$ is an actual product in $\mathrm{Tot}(\mathsf{C})$.

Definition 1.1.8 A restriction category C is called **cartesian** if it has all binary (partial) products as well as a (restriction) terminal object.

1.2 Turing Categories

Now, turning to the key definition used in this thesis, namely that of a Turing category (as stated in [2]), the following lemmas and definitions will be the basis of results obtained in the subsequent chapters.

Definition 1.2.1 Let C be a cartesian restriction category.

(i) Given a morphism $\tau_{X,Y} : A \times X \to Y$, a morphism $f : Z \times Y \to Y$ is said to **admit a $\tau_{X,Y}$-index** h when there exists a total map $h : Z \to A$ for which the following diagram commutes:

$$\begin{tikzcd}
A \times X \arrow[r, "\tau_{X,Y}"] & Y \\
Z \times X \arrow[u, "h \times 1_X"] \arrow[ur, "f"'] &
\end{tikzcd}$$

(ii) The morphism $\tau_{X,Y}$ is called a **universal application** when every $f : Z \times Y \to Y$ admits a $\tau_{X,Y}$-index.

(iii) A **Turing object** in C is an object A such that for each $X, Y \in \mathsf{C}$, there is a universal application $\tau_{X,Y} : A \times X \to Y$.

(iv) The category C is called a **Turing category** if it possesses a Turing object.

It is important to note that Turing categories are different from cartesian closed (CC) categories. CC categories have a terminal object, all finite products and all exponentials X^Y. This amounts to the natural bijection $Hom(X \times Y, Z) \cong Hom(X, Z^Y)$. The diagram depicting this is similar to that used in the definition of Turing categories:

$$\begin{tikzcd}
Z^Y \times Y \arrow[r, "ev"] & Z \\
X \times Y \arrow[u, "\tilde{f}\times 1_Y"] \arrow[ur, "f"'] &
\end{tikzcd}$$

The difference is that in a Turing category, $\tilde{f}$ need not be unique.

The preceeding definition pertains to the defining property of a Turing category — the presence of a Turing object, now the following one will outline the key structure found in a Turing category.

Definition 1.2.2 Let C be a Turing category and A — an object of C.

(i) A family of maps $\tau = \{\tau_{X,Y} : A \times X \to Y | X, Y \in \mathsf{C}\}$ is an **applicative family** for A.

(ii) An applicative family τ for A is called **universal** when each $\tau_{X,Y}$ is universal. In this case, τ is also called a **Turing structure** on A.

(iii) The pair (A, τ), where τ is universal for A is called a **Turing structure** on C.

Suppose $\tau_{X,Y} : A \times X \to Y$ is a universal application in C, and $f : Z \times X \to Y$ is a map. Since C has all products, X may itself be an arbitrary product, so it makes sense to focus on the case when $Z = 1$. In this case, the index for the map f, denoted $c_f : 1 \to A$, is called a code for f. The isomorphism $1 \times X \to X$ will often be supressed in this and the following chapters.

The following definition will play an important role in describing maps in a Turing category.

Definition 1.2.3 Suppose C is a category with objects X and A. X is said to be a retract of A if there exists a pair of maps, $m_X : X \to A$ and $r_X : A \to X$, such that $r_X m_X = 1_X$. The pair (m_X, r_X) is called an embedding-retraction pair.

Note that in the above definition, the map m_X must be total, while r_X is an epimorphism. To further describe all maps in a Turing category, the following lemma will be useful.

Lemma 1.2.4 Suppose C is a Turing category with Turing object A. Then every $X \in \mathsf{C}$ is a retract of A.

The embedding-retraction pair is given by $(\widetilde{\pi_X}, \tau_{1,X}\pi_A^{-1})$, as in the following diagram:

$$\begin{tikzcd}
A \times 1 \arrow[r, "\tau_{1,X}"] & X \\
X \times 1 \arrow[u, "\widetilde{\pi_X} \times 1"] \arrow[ur, "\pi_X"'] &
\end{tikzcd}$$

This implies that $\tau_{A,A}$ is a generic universal application in a Turing category. Observe that an arbitrary map $f : X \to Y$ factors through $\tau_{A,A}$, an embedding $m_X : X \to A$ and a retraction $r_Y : A \to Y$:

$$\begin{tikzcd}
A \times A \arrow[r, "\tau_{A,A}"] & A \arrow[d, "r_Y"] \\
1 \times X \arrow[u, "c_f \times m_X"] \arrow[r, "f"] & Y
\end{tikzcd}$$

Example 1.2.5 The standard model for computation is the Turing category of finite powers of the natural numbers — $\{1, \mathbb{N}, \mathbb{N}^2, ...\}$, together with the (enumerable) set of all computable functions $\mathbb{N}^n \to \mathbb{N}^m$. A Turing object in this category is $\mathbb{N}$, and there exist embedding - retraction pairs for $\mathbb{N}^n, n \geq 1$ which are in fact isomorphisms — this is not generally true in all Turing categories. The $\mathbb{N}^2 \to \mathbb{N}$ isomorphism may be defined by

$$(a,b) \mapsto (a+b+1)(a+b+2)/2 - a - 1$$

The application $\tau_{\mathbb{N},\mathbb{N}} : \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ is defined as follows: for any $n, m \in \mathbb{N}$,

$$n \cdot m = \phi_n(m),$$

where $\phi_n(m)$ is the computable function with index n. The index comes from a Gödel numbering framework of all computable functions $f : \mathbb{N} \to \mathbb{N}$.

Note that the category of powers of $\mathbb{N}$ and (partial) computable functions is a subcategory of Par, with the same terminal object and same restriction products.

1.3 Partial Combinatory Algebras

This section will introduce an abstraction of the universal application in the standard model which represents all the possible computations on powers of $\mathbb{N}$.

Let C be a cartesian restriction category. An applicative system $\mathbb{A} = (A, \bullet)$ in C consists of an object A and a morphism $\bullet : A \times A \to A$, called an application. Now, define $\bullet^n : A \times A^n \to A$ inductively for $n = 2, 3, \ldots,$ by $\bullet^n = \bullet(1 \times \bullet^{n-1})$ and $\bullet^1 : A \times A \to A$. Finally, it makes sense to define $\bullet^0 = \bullet\Delta$, where $\Delta : A \to A \times A$ is the diagonal functor. The following definition, found in [2], describes partial combinatory algebras in terms of its maps and objects.

Definition 1.3.1 Suppose $A \in \mathsf{C}$, a cartesian restriction category, and $\bullet : A \times A \to A$ is a map in C, referred to as the application.

(i) Consider the smallest cartesian restriction subcategory of C containing every total map $1 \to A$ as well as the application. Any morphism $f : A^n \to A^m$ in this subcategory is called a **polynomial** orphism.

(ii) A morphism $f : A^n \to A$ is said to be $\mathbb{A}$ - **computable** when it factors through $\bullet^n$ via some code $c_f : 1 \to A$. A morphism $A^n \to A^m$ is $\mathbb{A}$ - computable when each of its m components is.

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

(iii) When every polynomial morphism $A^n \to A^m$ is $\mathbb{A}$ - computable, the applicative system $\mathbb{A}$ is said to be **combinatory complete**, and is referred to as a **partial combinatory algebra**.

Suppose $\bullet : A \times A \to A$ is in C. Write $\mathbb{A} = (A, \bullet)$, and let $\mathrm{Comp}(\mathbb{A})$ denote the graph where

$\mathrm{Obj}(\mathrm{Comp}(\mathbb{A})) = \{1, A, A^2, \ldots\}$
$\mathrm{Arr}(\mathrm{Comp}(\mathbb{A})) = \{\text{polynomial maps}\}$.

The following result, also found in [2], describes the connection between Turing structure and combinatory completeness.

Lemma 1.3.2 When the system $\mathbb{A}$ is combinatory complete, $\mathrm{Comp}(\mathbb{A})$ is a Turing category, with Turing object A. Furthermore, when every object in C is a retract of A, A is also a Turing object in C, with $\tau_{A,A} = \bullet$.

It also makes sense to consider the full one-object subcategory A of $\mathrm{Comp}(\mathbb{A})$, $A \in \mathsf{A}$, with all $\mathbb{A}$ - computable maps $f : A \to A$. This is not, however, a Turing category, since it does not contain a terminal object. To summarize, a Turing category C with Turing object A gives rise to 3 related categories based on PCA $\mathbb{A}$, which all have similar properties:

(i) A,

and the following are Turing categories

(ii) Comp($\mathbb{A}$), and

(iii) Split(Comp($\mathbb{A}$))

These categories embed as follows:

$$\mathsf{A} \hookrightarrow \text{Comp}(\mathbb{A}) \hookrightarrow \mathsf{C} \hookrightarrow \text{Split}(\text{Comp}(\mathbb{A}))$$

Alternatively, a PCA can be identified by the presense of specific elements in A (see [6] for more details). Here, the notation ab, $a, b, \in A$ is shorthand for $a \cdot b \in A$, and $abc = (ab)c, c \in A$, whereas $ab \downarrow$ denotes that ab is defined. The $\cong$ symbol denotes Kleene equality.

Lemma 1.3.3 A partial applicative structure $(A, \bullet)$ is combinatory complete whenever it has elements k and s such that for all $a, b \in A$:

(i) $kab \cong a$

(ii) $sa\downarrow, sab\downarrow$, and $sabc \cong ac(bc)$

In a combinatory complete structure, elements k and s, as well as all elements that can be built up from these and other elements of A by using the application $\bullet$, are called combinators.

The above lemma is in set-theoretic notation, but it may also be interpreted diagramatically. Combinators are usually considered as elements of the underlying PCA set. However, in a Turing category, this may be too specific, as the object A in a PCA is not necessarily a set (in Par).

To completely integrate the concept of PCA combinators into an arbitrary Turing category, they must be regarded as the total point maps into a Turing object, $a : 1 \to A$. That is, they are exactly the codes for the maps Comp($\mathbb{A}$). This is also the reason behind using Kleene equality instead of equality in the combinator PCA definition.

In the general categorical case, the application $a \cdot b$ is shorthand for $\bullet\langle a, b\rangle : 1 \to A$, where $a, b : 1 \to A$ are codes. A map $f : A \to A$ with a code c_f may be donoted by $c_f \cdot - = \bullet\langle c_f !_A, 1_A\rangle$. Observe that the combinators k, s indeed represent maps in a Turing category with Turing object A:
k is a code for $\pi_0 : A \times A \to A$, whereas s is a code for the composition

$$A \times A \times A \overset{\langle \pi_{13}, \pi_{23}\rangle}{\longrightarrow} (A \times A) \times (A \times A) \overset{\bullet\times\bullet}{\longrightarrow} A \times A \overset{\bullet}{\longrightarrow} A$$

Many different computations can be built up with just those combinators. Itself, k is a code for a projection onto the first coordinate, and the following are a few other combinator computations:

$$\begin{aligned} ((sk)k)x &\cong (kx)(kx) \\ &\cong x \end{aligned}$$

$$\begin{aligned} k^* xy &= ((sk)x)y \\ &\cong ky(xy) \\ &\cong y, \end{aligned}$$

so that k^* is the combinator for the projection onto the second coordinate. Certain properties of Turing categories, such as existence of range maps or coproducts, will actually enforce the existence of other particular combinators in the underlying PCA.

A PCA structure may be regarded as a model of a certain theory called combinatory logic. There is a close connection between combiatory logic and lambda calculus. For example, the combinator k is present in lambda calculus in the form $\lambda ab.a$. All the lambda calculus computations are, in turn, expressible through the application within a PCA. A proof of this statement may be found in [6].

This combinator approach will be quite useful in pinpointing the conditions for the presence of categorical structure in the upcoming chapters. The following are some further examples of PCA's:

(i) $(\mathbb{N}(A), \bullet)$, computation with an oracle $A \subset \mathbb{N}$ that answers the question "is x in A?", denoted $n \cdot_A m$. The set of all computations with oracle A is still enumerable, since a code can be assigned to the oracle query.

Then, every A-computable $f : \mathbb{N} \to \mathbb{N}$ has a code $c_f \in \mathbb{N}$, and that there is a universal application $\bullet : \mathbb{N} \times \mathbb{N} \to \mathbb{N}$.

This application is different from the standard model application because A is now computable in this model.

(ii) Reflexive Structures: in a cartesian closed category, an object A is said to be *reflexive* whenever A^A is a retract of A, so that there is a retraction $r : A \to A^A$. Then, the application can be defined as the following composition:

$$A \times A \overset{r \times 1}{\to} A^A \times A \overset{ev}{\to} A$$

where ev is the evaluation map.

Chapter 2

Ranges in Turing Categories

There are two equivalent definitions for a recursively enumerable set — the domain of a computable map, or the range of a computable map. The very definition of a restriction cateogry allows us to talk about domains of maps, however, the notion of range is not part of the structure. Ranges in restriction categories take the form of restriction idempotents, and are related to a topological concept of open maps. The axioms for a category with ranges are described in [5]. In this chapter, the overlap of Turing and range categories is investigated.

2.1 Range Categories

Consider the range of a mapping $f : A \to B$ in Par. It is a subset S of B such that for every $y \in S$ there exists an $x \in A$ such that $f(x) = y$. Now, in line with the concept of domains expressed as restriction idempotents, the range of f can also be viewed as an idempotent

$$e(y) = \begin{cases} y & \text{if } \exists x \in B \text{ such that } f(x) = y \\ \uparrow & \text{otherwise.} \end{cases}$$

To explore ranges in categories other than Par, maps must again have corresponding range idempotents, described in the following definition.

Definition 2.1.1 A restriction category C is called a **range category** if it has an operator $\widehat{(-)}$, which sends $f : A \to B$ to a map $\hat{f} : B \to B$, and satisfies the following axioms:

[**RR.1**] $\overline{\hat{f}} = \hat{f}$

[**RR.2**] $\hat{f} f = f$

[**RR.3**] $\widehat{\overline{g} f} = \overline{g}\hat{f}$ with codom(f) = dom(g)

[**RR.4**] $\widehat{g\hat{f}} = \widehat{gf}$ with codom(f) = dom(g)

This range theory captures many properties of the conventional concept of range in Par. However, often a fifth axiom is added:

[**RR.5**] $f\hat{g} = h\hat{g}$ whenever $fg = hg$

With this axiom, certain non-intuitive examples of ranges are eliminated, namely when $\overline{f} = 1$ for all $f \in \mathsf{C}$, so that $\hat{f} = 1$.

In a category with ranges, any morphism $r : X \to Y$ such that $\hat{r} = 1_Y$, in particular, any retraction, is called a *surjection*. Thus, intuitively, RR.5 says that every morphism is surjective onto its range.

Parts (iii) and (iv) involve all the maps in a range category. However, it may be the case, in a cartesian restriction category C, that it would be desirable to define some semblance of range for a single map in the category. Recall from the previous chapter that restriction idempotents are open, including range maps, and that they compose openly. The (always non-empty, containing the identity maps) subcategory of C of open maps is in fact a range category, with $\exists_f(e) = \widehat{fe}$, where e is a restriction idempotent.

So ranges of maps are defined by $\hat{f} = \exists_f(1)$. From now on, for an open map f the notation $\hat{f}, f : A \to B$, will refer to the range of f in the open map subcategory of C, where $\widehat{f}$ makes sense. Sometimes, we will say that a category satisfies RR5 even if it is not a range category. By that, we mean that the subcategory of open maps satisfies RR5.

Such notation helps aggregate the original definition of a range category and the openness of certain maps to show a category has a range combinator (as in the upcoming proof). Since the range combinator is in fact unique once a restriction combinator in the category is fixed, this means that there is no ambiguity in writing $\hat{f}$ when f is open.

Before exploring ranges in Turing categories, another important point to note about range maps is their connection to partial inverses. We first recall the definition of a partial isomorphism:

Definition 2.1.2 A **partial inverse** of a map $f : X \to Y$ in a cartesian restriction category C is a map $f^{-1} : Y \to X$ such that:

(i) $f^{-1}f = \overline{f}$

(ii) $ff^{-1} = \overline{f^{-1}}$

Then, if f has a partial inverse, then it is called a **partial isomorphism**.

As with ordinary isomorphisms, a partial inverse to a map, if it exists, is necessarily unique. Not all maps have a partial inverse, for eg., a non-injective map in Par does not.

As the range combinator is, in a way, the dual concept to the restriction combinator, the domain of a (partial) inverse of a map f is in fact the range of f. The following proposition ties together the concepts of partial inverses and ranges, saying just that. This result about ranges will also be useful in the next chapter on equality.

Proposition 2.1.3 Suppose C is a cartesian restriction category. Then, if a map $f : X \to Y$ in C has a partial inverse, it is open.

Proof: Let f^{-1} be a partial inverse of $f : X \to Y$. For $e \in \mathcal{O}(X), e' \in \mathcal{O}(Y)$, define

$\exists_f(e) = (f^{-1})^*(e)$, and verify the open map axioms:

[**O1**].

$$\begin{aligned} e' &\geq e'\overline{f^{-1}} \\ &= \overline{e'\overline{f^{-1}}} \\ &= (ff^{-1})^*(e'), \text{as} ff^{-1} = \overline{f^{-1}}, \text{so that} \end{aligned}$$

$$\begin{aligned} \exists_f(f^*(e')) &= (f^{-1})^*(f^*(e')) \\ &= (ff^{-1})^*(e') \\ &\leq e' \end{aligned}$$

[**O2**]

$$\begin{aligned} ef^*(e') &= e\overline{e'f} \\ &= e\overline{e'f}\,\overline{f} \\ &= e\overline{f}\,\overline{e'f} \\ &= \overline{f}^*(e)\overline{e'f} \\ &= (f^{-1}f)^*(e)f^*(e') \\ &= f^*(f^{-1})^*(e)f^*(e') \\ &= f^*((f^{-1})^*(e)e') \\ &= f^*(\exists_f(e)e') \end{aligned}$$

[**O3**].

$$e'\exists_f(e) = e'(f^{-1})^*(e)$$

$$
\begin{aligned}
&= e'\overline{ef^{-1}f^{-1}} \\
&= e'\overline{f^{-1}}(f^{-1})^*(e) \\
&= (\overline{f^{-1}})^*(e')(f^{-1})^*(e) \\
&= (ff^{-1})^*(e')(f^{-1})^*(e) \\
&= (f^{-1})^*(f^*(e')e) \\
&= \exists_f(f^*(e')e)
\end{aligned}
$$

▮

2.2 Ranges and Retractions

Now, to prepare for describing Turing categories with ranges, the following lemma about ranges and idempotents will help verify the necessary technical details.

Lemma 2.2.1 Suppose (m, r) is an embedding-retraction pair of X into $A \in \mathsf{C}$, and the idempotent mr is open. Then

(i) m is open, and $\hat{m} = \widehat{mr}$

(ii) r is open, and for any $e \in \mathcal{O}(A), \exists_r(e) = m^*\exists_{mr}(e)$

(iii) for any $e \in \mathcal{O}(A)$, when $r = r\hat{m}$, $\widehat{re} = rem$

(iv) the map $r' = r\widehat{m}$ is also a retraction for X, and, assuming RR5, mr' is a restriction idempotent equivalent to mr

Proof:

(i) First, define $\widehat{me} = \exists_m(e) = \exists_{mr}(r^*(e))$, and check the axioms for $e \in \mathcal{O}(A), e' \in \mathcal{O}(X)$:

[**O1**].

$$\begin{aligned}\exists_m(m^*(e')) &= \exists_{mr}(r^*(m^*(e'))) \\ &= \exists_{mr}((mr)^*(e')) \\ &\leq e'\end{aligned}$$

by O1 for the open map mr.

[**O2**]. As mr is open, by O2, it follows that

$u \wedge (mr)^*(u') \leq (mr)^*(\exists_{mr}(u) \wedge u')$

Then, taking $u = r^*(e), u' = e'$,

$$\begin{aligned}u \wedge (mr)^*(u') &= r^*(e) \wedge (mr)^*(e') \\ &= r^*(e) \wedge r^*m^*(e') \\ &= r^*(e \wedge r^*m^*(e')), \text{and} \\ (mr)^*(\exists_{mr}(u) \wedge u') &= r^*m^*(\exists_{mr}(r^*(e)) \wedge e') \\ &= r^*m^*(\exists_m(e) \wedge e'), \text{so that} \\ r^*(e \wedge r^*m^*(e')) &\leq r^*m^*(\exists_{mr}(r^*(e)) \wedge e')\end{aligned}$$

Now, r^* preserves order, as for any restriction idempotents e_1, e_2, the following sequence of inequalities holds

$$\begin{aligned}r^*(e_1) &\leq r^*(e_2) \\ \overline{e_1 r} &\leq \overline{e_2 r} \\ r\overline{e_1 r} &\leq r\overline{e_2 r} \\ e_1 r &\leq e_2 r, \text{which is equivalent to saying} \\ e_2 r\overline{e_1 r} &= e_1 r\end{aligned}$$

$$\begin{aligned}
e_2e_1r &= e_1r, \text{and since } r \text{ is an epimorphism,}\\
e_2e_1 &= e_1, \text{so that exactly}\\
e_1 &\leq e_2
\end{aligned}$$

Concluding, then, that $e \wedge r^*m^*(e') \leq m^*(\exists_{mr}(r^*(e)) \wedge e')$, which implies $e \wedge (mr)^*(e') \leq m^*(\exists_m(e) \wedge e')$.

[**O3**]. By axiom O3 for mr,

$$u' \wedge \exists_{mr}(u) \;\; \leq \;\; \exists_{mr}((mr)^*(u') \wedge u)$$

Now, let $u = r^*(e'), u' = e$, so that

$$\begin{aligned}
e' \wedge \exists_{mr}(r^*(e)) &\leq \exists_{mr}(r^*m^*(e') \wedge e) \Rightarrow\\
e' \wedge \exists_m(e) &\leq \exists_m(m^*(e') \wedge e)
\end{aligned}$$

as needed.

Finally, $\hat{m} = \exists_{mr}(r^*(1)) = \widehat{mr\overline{r}} = \widehat{mr}$

(ii) For r, define: $\exists_r(e) = m^*\exists_{mr}(e)$, and check the axioms for $e \in \mathcal{O}(A), e' \in \mathcal{O}(X)$:

[**O1**]. By axiom O1 for mr, with idempotents $e, r^*(e') \in \mathcal{O}(A)$,

$$\begin{aligned}
\exists_{mr}((mr)^*(r^*(e'))) &\leq r^*(e'), \text{ applying } m^*,\\
m^*\exists_{mr}((mr)^*(r^*(e'))) &\leq m^*r^*(e'), \text{ so that}\\
\exists_r((rmr)^*(e')) &\leq (rm)^*(e') \text{ as } rm = 1,\\
\exists_r(r^*(e')) &\leq e'
\end{aligned}$$

[**O2**]. By O2 for mr, with idempotents $e, r^*(e') \in \mathcal{O}(A)$, we get

$$\begin{aligned}
e(mr)^*(r^*(e')) &\leq (mr)^*(\exists_{mr}(e)r^*(e')) \Rightarrow\\
er^*(e') &\leq r^*(m^*\exists_{mr}(e)m^*r^*(e'))
\end{aligned}$$

$$e \wedge r^*(e') \quad \leq \quad r^*(\exists_r(e) \wedge e')$$

[**O3**]. Once more, by O3 for mr, with idempotents $e, r^*(e') \in \mathcal{O}(A)$, conclude

$$\begin{aligned}
r^*(e')\exists_{mr}(e) &\leq \exists_{mr}((mr)^* r^*(e')e), \text{ applying } m^* \\
m^*(r^*(e')\exists_{mr}(e)) &\leq m^*\exists_{mr}(r^*(e')e) \\
m^*r^*(e')m^*\exists_{mr}(e) &\leq m^*\exists_{mr}(r^*(e')e) \text{ so that exactly} \\
e' \wedge \exists_r(e) &\leq \exists_r(r^*(e') \wedge e)
\end{aligned}$$

(iii) For any $e \in \mathcal{O}(A)$,

$$\overline{rem} = rm\overline{rem} = r\overline{re}m = r\overline{r}em = rem$$

then, the following sequence of equalities holds, proving the result:

$$\widehat{re} = \widehat{re\widehat{m}} = \widehat{rem} = \widehat{\overline{rem}} = \overline{rem} = rem$$

(iv) (m, r') is indeed an embedding-retraction pair of X into A:

$$r\widehat{m}m = rm = 1_X$$

Now, to show that $mr\widehat{m} = mr\widehat{mr}$ is indeed a restriction idempotent,

$$mr = mrmr = 1_A mr$$

and apply RR5:

$$mr\widehat{mr} = \widehat{mr}$$

∎

For an object X with an open embedding-retraction pair (m, r), it is safe to consider the retraction $r = r\widehat{m}$, as such a morphism always exists. When RR5 holds, one may assume $mr = \overline{mr}$, as such an idempotent exists.

2.3 The Beck-Chevalley Condition

The Bech-Chevalley condition has long been studied in a variety of contexts, and the following definition, found in [1], introduces it in the context of ranges and restriction products. This condition will be useful in describing Turing categories with ranges.

Definition 2.3.1 A cartesian restriction category C is a **cartesian range category** when it is a range category and the following condition, known as the **Beck-Chevalley condition** (see [1]):

$$\exists_{f\times 1_X}(\pi_C^*(e)) = \pi_A^*(\exists_f(e))$$

holds true for all (pullback) squares of the form

$$\begin{tikzcd}
C \times B \arrow[loop above, "\pi_C^*(e)"] \arrow[r, "f\times 1_X"] \arrow[d, "\pi_C"'] & A \times B \arrow[d, "\pi_A"] \\
C \arrow[loop below, "e"] \arrow[r, "f"'] & A
\end{tikzcd}$$

An arbitrary cartesian restriction category C is said to satisfy the BCC whenever Open(C) is a cartesian range category.

To say C satisfies BCC is equivalent to saying that for all open maps $f, g \in \mathsf{C}, \widehat{f \times g} = \hat{f} \times \hat{g}$. Indeed, suppose $\widehat{f \times g} = \hat{f} \times \hat{g}$. Then

$$\begin{aligned}
\exists_{f\times 1_X}(\pi_C^*(e)) &= \exists_{f\times 1_X}(e \times 1_X) \\
&= (\widehat{f \times 1_X})(\widehat{e \times 1_X})
\end{aligned}$$

$$\begin{aligned} &= \widehat{fe} \times 1_X (\text{by assumption}) \\ &= \pi_A^*(\exists_f(e)) \end{aligned}$$

And conversely, when $\exists_{f\times 1_X}(\pi_C^*(e)) = \pi_A^*(\exists_f(e))$,

$$\begin{aligned} \widehat{(f \times 1)} &= \hat{f} \times 1, \text{and} \\ \widehat{(1 \times g)} &= 1 \times \hat{g}, \text{so} \\ \widehat{f \times g} &= (f \times \widehat{1)(1 \times} g) \\ &= \widehat{(f \times 1)}\widehat{(1 \times g)} \\ &= (\hat{f} \times \widehat{1)(1 \times} \hat{g}) \\ &= \widehat{\hat{f} \times \hat{g}} \\ &= \hat{f} \times \hat{g} \end{aligned}$$

Generalizing to arbitrary finite products, it follows that for any $f =< f_i, ..., f_n >$: $C \to \Pi_{i=1}^n A_i$ in $\mathsf{C}, \hat{f} =< \hat{f}_i, ..., \hat{f}_n >$. The following statement is true about range categories and their subcategories:

Lemma 2.3.2 Consider a range category C with a subcategory D. Then D satisfies the Beck-Chevalley condition if C satisfies the Beck-Chevalley condition.

The proof is immediate, since the inclusion $\mathsf{D} \hookrightarrow \mathsf{C}$ preserves the ranges and the products. This lemma is applicable to the case of a Turing cartesian range category with PCA $\mathbb{A}$, and its subcategory Comp($\mathbb{A}$).

2.4 Ranges and PCA's

We now investigate conditions on a PCA under which the corresponding Turing category will have ranges.

Proposition 2.4.1 Suppose C is a Turing category with a universal application $\bullet : A \times A \to A$ which is open. Suppose also that every code (total map) $c_f : 1 \to A$ is open, and assume BCC. Then the categories Comp($\mathbb{A}$),C, and Split(Comp($\mathbb{A}$)) are range categories, and

$$\text{Comp}(\mathbb{A}) \hookrightarrow \mathsf{C} \hookrightarrow \text{Split}(\text{Comp}(\mathbb{A}))$$

are range preserving inclusions.

Proof: First, note that the assumptions in the proposition imply that every computable map $f : A \to A$ is open, as it can be expressed as $\bullet\langle c_f !_A, 1_A\rangle$, which is a composition of an open map and a product of open maps (open by BCC). That means in particular that every computable idempotent on A is open. By the previous lemma it follows that all embedding-retraction pairs are open as well.

An arbitrary map $f : X \to Y$ is again a composition of open maps, with $f = r_Y \bullet \langle c_f !_A, 1_A\rangle m_X$.

∎

To avoid ambiguity in notation, the range of a composition of several maps will be denoted by $(-)_{ran}$ in the upcoming sequence of equalities. Now, for $f : X \to Y$ in C, the range can be computed as follows:

$$\hat{f} = (r_Y \bullet \langle c_f !_A, 1_A\rangle m_X)_{ran}$$

Note that $f = r_Y \widehat{m_Y} m_Y r_Y \bullet \langle c_f !_A, 1_A\rangle m_X$, so that $c'_f \cdot - = m_Y r_Y (c_f \cdot -)$ is another code for f. Then, without loss of generality, we can assume $r = r\hat{m}$.

By the previous lemma, with $e = \widehat{(c_f \cdot -)\widehat{m_X}} = \widehat{(c_f \cdot -)m_X}$, which is the range of a composition of two open maps, the result follows:

$$\hat{f} \quad = \quad (r_Y \widehat{(c_f \cdot -)m_X})_{ran}$$

$$= r_Y((c_f\cdot -)m_X)_{ran}m_Y$$
$$= r_Y((c_f\cdot -)\widehat{m_X r_X})_{ran}m_Y$$

Next, we consider what happens in the underlying partial combinatory algebra side of things in terms of combinators.

Definition 2.4.2 Suppose $\mathbb{A} = (A, \bullet)$ is a PCA in a cartesian restriction category C. Then

(i) $\mathbb{A}$ has (**weak**) **range combinators** whenever for every code $a : 1 \to A$ there exists a combinator r_a such that $(r_a\cdot a)\cdot - = \widehat{a\cdot -}$.

(ii) $\mathbb{A}$ has a **strong range combinator** if there exists a combinator r such that $(r\cdot a)\cdot - = \widehat{a\cdot -}$ for every code $a : 1 \to A$.

Using the above results, the next theorem gives a characterization of when a Turing category is a range category.

Theorem 2.4.3 Consider a cartesian restriction category C with a PCA $(A, \bullet)$. Then A has weak range combinators if and only if Comp($\mathbb{A}$) is a range category. Furthermore, Split(Comp($\mathbb{A}$)) is then also a range category.

Proof: Suppose Comp($\mathbb{A}$) is a range category, with a PCA $(A, \bullet)$, which contains a code $a : 1 \to A$. Then the map $a\cdot - : A \to A$ is open, so let b be a code for $\widehat{a\cdot -}$. Let r_a be the combinator for the lambda calculus expression $\lambda x.b$, so that $r_a\cdot a = b$. It follows that $\widehat{a\cdot -} = b\cdot - = (r_a\cdot a)\cdot -$, so that Comp($\mathbb{A}$) has a weak range combinators.

It follows from the definition that having weak range combinators in $\mathbb{A}$ means that for any map $f : A \to A$ in Comp($\mathbb{A}$), with code a, there exists a range combinator r_a, with $\widehat{f} = (r_a\cdot a)\cdot -$, so that it is open.

Then, as shown in the previous proposition, every map $X \to Y$, where X, Y are retracts of A, must also be open. It follows that Split(Comp($\mathbb{A}$)), as well as Comp($\mathbb{A}$) are range categories.

∎

Having a strong range combinator in $\mathbb{A}$ allows us to compute the range of a map in a Turing category in a uniform manner. So, for any $f : X \to Y$, the range of f may be expressed as

$$\hat{f} \;=\; r_Y \widehat{f m_X r_X} m_Y, \text{ so to illustrate,}$$

$$\begin{tikzcd}
A \arrow[loop above, "\widehat{m_X r_X}"] \arrow[r, "c_f\cdot -"] \arrow[d, shift left, "r_X"] & A \arrow[loop above, "\widehat{(c_f\cdot -)m_X r_X}"] \arrow[d, shift left, "r_Y"] \\
X \arrow[u, shift left, "m_X"] \arrow[r, "f"'] & Y \arrow[u, shift left, "m_Y"] \arrow[loop below, "\hat{f}"]
\end{tikzcd}$$

Let c_{mr} be a code for $m_X r_X$, and c_f be a code for f. At a point (map) x,

$$\begin{aligned}
c_f(c_{mr}x) &= ((kc_f)x)(c_{mr}x) \\
&= s(kc_f)c_{mr}x, \text{ now applying the range combinator,} \\
\widehat{f m_X r_X} &= r(s(kc_f)c_{mr})\cdot -, \text{ so that} \\
\hat{f} &= r_Y(rs(kc_f)c_{mr}\cdot -)m_Y
\end{aligned}$$

Example 2.4.4 The standard model Comp($\mathbb{N}$) is a range category, with range combinator inherited from Par. The underlying PCA in fact has a strong range combinator r, which may be expressed using pseudo-code as follows:

```
IsInRange $(n, x)$ {
    For(int $i = 0$ to $\infty$) {
        For(int $j = 0$ to $i$) {
```

```
      Do 1 step of each computation $\phi_n(j)$
      If $\phi_n(j)$ halts at this step {
        Test $\phi_n(j) = x$
        If TRUE, return $x$
      }
    }
  }
}
```

This algorithm computes the value of the map $n\cdot - : \mathbb{N} \to \mathbb{N}$ at each integer i sequentially, using interleaving, and tests the equality $n\cdot i = x$. The algorithm halts if the equality holds for some i, and outputs x. It continues testing infinitely if x is not in the range of n.

For each n, this computation corresponds to some (computable) function of x, with code $\phi_r(n)$ by the S^m_n theorem (see [4]). This amounts to saying exactly that $(r\cdot n)\cdot x = \widehat{\phi}_n(x)$.

Note that in the computation, a test for equality, $\phi_n(i) = x$, takes place. This test is indeed possible in $\mathbb{N}$, but such equality tests may not exist in other PCA's, which would make a similar computation strategy infeasible. The following chapter will elaborate on equality in Turing categories.

Chapter 3

Equality in Turing Categories

In an arbitrary cartesian restriction category C, it may be possibile to establish a comparison map that captures some key properties of equality in the category Par. In this category, equality is usually formulated as, for each object X, a morphism $EQ_X : X \times X \to \{0, 1\}$ which is then typically defined as

$$EQ_X(a, b) = \begin{cases} 0 & \text{if } a = b \\ 1 & \text{otherwise.} \end{cases}$$

The standard computability model Comp($\mathbb{N}$) inherits this equality from Par. One way to compute equality of two integers (x, y) is by taking their binary representations (the bits in front of the most significant bit are assumed to be 0's) and performing the calculation

```
Equality(x,y) {
        If (XOR($x, y$) = 0)
          Return 0
        Else
          Return 1
```

}

Only if $x = y$ will the output be 0, but this computation always halts, meaning that $EQ_{\mathbb{N}}$ is total. This type of equality is known as *strong equality*, or *decidability*, and will be discussed later on in the chapter.

Having equality that is a total map may be a lot to ask for in a category. The more common type of equality studied in restriction categories is weak equality. A weak equality's domain, in the world of sets, only covers the subset of pairs of the form $(a, a), a \in X$. That is, in Par, weak equality Eq_X on a set X would be

$$Eq_X(a,b) = \begin{cases} a & \text{if } a = b \\ \uparrow & \text{otherwise.} \end{cases}$$

This is defined on the subset $\{(a, a) : a \in X\}$, which is exactly the range of the diagonal map Δ_X. This situation can be interpreted in a more general categorical setting.

In this chapter, we will first look at categorical definitions of equality, then examine equality in Turing categories and underlying PCA's. Finally, we look at the interaction between equality and ranges and meets.

3.1 Types of Categorical Equality

The following is the usual definition of (weak) equality in a cartesian restriction category C (see [1]).

Definition 3.1.1 An object X in a cartesian restriction category is **discrete** when there exists a map $Eq_X : X \times X \to X$, called **equality**, that is a partial inverse of the diagonal map Δ_X. That is,

(i) $Eq_X \Delta_X = 1_X$, and

(ii) $\Delta_X Eq_X = \overline{Eq_X}$

A category is said to be discrete, or with equality, when every object is discrete.

Recall from the previous chapter that whenever a map has a partial inverse, it is open. So, $\widehat{\Delta_X}$ exists for a discrete object X, and furthermore,

$$\begin{aligned}
\pi_0\widehat{\Delta_X} &= \pi_0\overline{Eq_X} \\
&= \pi_0\Delta_X Eq_X \\
&= Eq_X
\end{aligned}$$

Similarly, this holds true for π_1 as well. Using these facts, it is possible to craft necessary and sufficient conditions for Eq_X to exist in a category the way it is defined in [1]:

Proposition 3.1.2 In a cartesian restriction category C, an object X is discrete, and $Eq_X = \pi_0\widehat{\Delta_X}$, whenever the diagonal map Δ_X is open and $\pi_0\widehat{\Delta_X} = \pi_1\widehat{\Delta_X}$.

Proof: Suppose C is a cartesian restriction category, containing a discrete object X. As already shown, when Eq_X exists, the diagonal map Δ_X is open, and $\pi_0\widehat{\Delta_X} = \pi_1\widehat{\Delta_X}$.

Conversely, suppose the diagonal map Δ_X is open, and $\pi_0\widehat{\Delta_X} = \pi_1\widehat{\Delta_X}$. Let $Eq_X = \pi_0\widehat{\Delta_X}$. Observe that

$$\begin{aligned}
Eq_X\Delta_X &= \pi_0\widehat{\Delta_X}\Delta_X \\
&= \pi_0\Delta_X \\
&= 1_X, \text{ and}
\end{aligned}$$

$$\begin{aligned}
\pi_0\widehat{\Delta_X} &= \pi_1\widehat{\Delta_X} \Rightarrow \\
\Delta_X\pi_0\widehat{\Delta_X} &= \Delta_X\pi_1\widehat{\Delta_X}
\end{aligned}$$

testing for equality in both components of the product,

$$\begin{aligned}
\pi_0\Delta_X\pi_0\widehat{\Delta_X} &= \pi_0\widehat{\Delta_X}, \\
\pi_1\Delta_X\pi_0\widehat{\Delta_X} &= \pi_0\widehat{\Delta_X} \\
&= \pi_1\widehat{\Delta_X}, \text{so that exactly} \\
\Delta_X\pi_0\widehat{\Delta_X} &= \widehat{\Delta_X},\ \text{which is then equal to} \\
&= \overline{\widehat{\Delta_X}} \\
&= \overline{\overline{\pi_0}\widehat{\Delta_X}} \\
&= \overline{\pi_0\widehat{\Delta_X}} \\
&= \overline{Eq_X}
\end{aligned}$$

∎

Discreteness is not found in all categories, but it follows from the above proposition that cartesian range categories satisfying the RR5 axiom are automatically discrete. In such a category, the diagonal is open, and by RR.5 $\pi_0\Delta = \pi_1\Delta$ implies $\pi_0\widehat{\Delta} = \pi_1\widehat{\Delta}$.

Consider again the motivation behind Turing categories, the standard computation model. The domain of definition of the strong equality $EQ_{\mathbb{N}}$ is all of $\mathbb{N} \times \mathbb{N}$. For $EQ_{\mathbb{N}}$, as well as for every EQ_X, $X \in \mathsf{Par}$, strong equality is conveniently presented as a map into $\{0, 1\}$.

Immediately, the weak version can be derived from EQ_X by composing it with a restriction idempotent e on $\{0, 1\}$

$$e(x) = \begin{cases} 0 & \text{if } x = 0 \\ \uparrow & \text{otherwise.} \end{cases}$$

Since $EQ_X(x, y)$ returns 0 when $x = y$ and is undefined otherwise, the projection of the restriction, $\pi_0\overline{eEQ_X}(x, y)$, gives the output $Eq_X(x, y)$. Symmetrically, one may define the idempotent e^c, restricting $\{0, 1\}$ to the subset $\{1\}$. The expression $\pi_0\overline{e^cEQ_X}$ gives another map, which is only defined on pairs of distinct elements. This type of map is another map related to equality, denoted Eq^c_X. In this case the two projections will be distinct, but the choice of π_0 is somewhat arbitrary.

In Comp($\mathbb{N}$), while $Eq_{\mathbb{N}}$ exists, comparing two algorithms Alg$_1$ and Alg$_2$ to determine if they amount to the same $\mathbb{N} \to \mathbb{N}$ map is a different story. The only way to answer this question may be by testing Alg$_1$ and Alg$_2$ on all inputs, using interleaving, and comparing the outputs. This approach will only output an answer if at some input x, Alg$_1(x)$ $\neq$Alg$_2(x)$, that is, the algorithms are different. If they are the same, this computation will not halt. This situation is an example of a distinctness comparison, which can be defined in a general setting.

Definition 3.1.3 Suppose C is a cartesian restriction category with object A and zero map $0_A : A \to A$. A **distinctness map** $Eq^c_A : A \times A \to A$ is a morphism such that $\overline{Eq}_A\overline{Eq^c}_A = 0_{A\times A}$, and is maximal under the $\leq$ ordering.

In the type of situation as with Eq_A and Eq^c_A, Eq^c_A is said to be a complement of Eq_A. In general, two maps $f, g : X \to Y$ in a cartesian restriction category are said to complement each other when $\overline{f}\overline{g} = 0_X$, and f (symmetrically, g) is the greatest such map under the $\leq$ ordering. With these two comparison maps established, it is now possible to define strong equality in a more general setting too.

Definition 3.1.4 Let C be a cartesian restriction category containing an object A. Suppose also that both maps Eq_A and Eq^c_A are defined. Then **strong equality** on

A is a map $EQ_A : A \times A \to X$ such that

(i) $\pi_0 \overline{eEQ_A} = Eq_A$,

(ii) $\pi_0 \overline{e_c EQ_A} = Eq^c_A$,

where $e, e_c \in \mathcal{O}(X)$ complement each other.

In Par, strong equality is a map into $1 + 1 \cong \{0, 1\}$, with e defined on 0, and e_c defined on 1.

3.2 Equality in Turing Categories

The equality that is the focus of this chapter is the weak type, and the following proposition about retracts and discreteness sets the stage for finally describing equality in Turing categories.

Proposition 3.2.1 Let C be a cartesian restriction category with a discrete object A. Then, if $X \in \mathsf{C}$ is a retract of A, it is also discrete.

Proof:

Consider the diagram

$$\begin{tikzcd}
A \arrow[r, shift left, "\Delta_A"] \arrow[d, shift left, "r"] & A \times A \arrow[l, shift left, "Eq_A"] \arrow[d, shift left, "r\times r"] \\
X \arrow[r, shift left, "\Delta_X"] \arrow[u, shift left, "m"] & X \times X \arrow[l, shift left, "Eq_X"] \arrow[u, shift left, "m\times m"]
\end{tikzcd} \tag{3.2.1}$$

The map $Eq_X = rEq_A(m \times m)$ is indeed the partial inverse of Δ_X:

$$\begin{aligned}
Eq_X\Delta_X &= rEq_A(m \times m)\Delta_X \\
&= rEq_A\Delta_A m \\
&= r1_A m \\
&= 1_X, \text{ while}
\end{aligned}$$

$$\begin{aligned}
\Delta_X Eq_X &= \Delta_X rEq_A(m \times m) \\
&= (r \times r)\Delta_A Eq_A(m \times m) \\
&= (r \times r)\overline{Eq_A}(m \times m) \\
&= (r \times r)\overline{r \times r}\,\overline{Eq_A}(m \times m) \\
&= (r \times r)\overline{(r \times r)\overline{Eq_A}}(m \times m) \\
&= (r \times r)(m \times m)\overline{(r \times r)\overline{Eq_A}(m \times m)} \\
&= \overline{(r \times r)\overline{Eq_A}(m \times m)} \\
&= \overline{(r \times r)\Delta_A Eq_A(m \times m)} \\
&= \overline{\Delta_X rEq_A(m \times m)} \\
&= \overline{\overline{\Delta_X}Eq_X} \\
&= \overline{Eq_X}
\end{aligned}$$

∎

An immediate corollary of the above lemma is its application to Turing objects.

Corollary 3.2.2 Suppose C is a Turing category with a discrete Turing object. Then C is a discrete category.

Recall that the diagonal map is open when an object is discrete. Together with the preceeding lemma, these imply that the diagonal map on every retract of a discrete object is also open. However, it is often useful to directly compute the factorization of a map through a Turing object. The upcoming lemma will give the expression for the direct image map, and the proof will be a direct verification of the axioms of openness.

Lemma 3.2.3 Suppose C is a Turing category with discrete Turing object A. Then for every $X \in \mathsf{C}$, the diagonal map is open, with the direct image map defined as follows (refer to (3.1.1)):

$$\exists_{\Delta_X}(e) = (m \times m)^*(\exists_\Delta(r^*(e)))$$

Proof: To verify the 3 openness axioms, let $u \in \mathcal{O}(A \times A), u' \in \mathcal{O}(A)$, and apply the open axioms for Δ:

Let $u = (r \times r)^*(e), u' = r^*(e'), e \in \mathcal{O}(X \times X), e' \in \mathcal{O}(X)$, then
[**O1**].

$$\begin{array}{rcl}
\exists_\Delta(\Delta^*(r \times r)^*(e')) & \leq & (r \times r)^*(e), \text{ then} \\
\exists_\Delta(\Delta^*(r \times r)^*(e'))(m \times m) & \leq & (r \times r)^*(e)(m \times m) \Rightarrow \\
\overline{\exists_\Delta(r^*\Delta_X^*(e'))(m \times m)} & \leq & \overline{(r \times r)^*(e)(m \times m)} \Rightarrow \\
(m \times m)^*(\exists_\Delta(r^*\Delta_X^*(e'))) & \leq & (m \times m)^*((r \times r)^*(e)), \text{ and the axiom follows :} \\
\exists_{\Delta_X}(\Delta_X^*(e')) & \leq & e
\end{array}$$

[**O2**].

$$\begin{array}{rcl}
r^*(e')\Delta^*((r \times r)^*(e)) & \leq & \Delta^*(\exists_\Delta(r^*(e'))(r \times r)^*(e)) \Rightarrow \\
r^*(e')r^*\Delta_X^*(e) & \leq & \Delta^*(\exists_\Delta(r^*(e'))(r \times r)^*(e)) \Rightarrow \\
r^*(e'\Delta_X^*(e)) & \leq & \Delta^*((r \times r)^*(\exists_\Delta(r^*(e'))e)) \Rightarrow
\end{array}$$

$$r^*(e'\Delta_X^*(e)) \;\leq\; r^*\Delta_X^*((m\times m)^*(\exists_\Delta(r^*(e'))e))$$

Since r^* preserves order, O3 for Δ_X follows:
$e' \wedge \Delta_X^*(e) \leq \Delta_X^*((m\times m)^*(\exists_\Delta(r^*(e')) \wedge e))$

[**O3**].

$$\begin{aligned}
(r\times r)^*(e)\exists_\Delta(r^*(e')) &\leq \exists_\Delta(\Delta^*(r\times r)^*(e)r^*(e')),\ \text{apply } m\times m^* \\
(m\times m)^*((r\times r)^*(e)\exists_\Delta(r^*(e'))) &\leq (m\times m)^*(\exists_\Delta(\Delta^*(r\times r)^*(e)r^*(e'))) \Rightarrow \\
e(m\times m)^*\exists_\Delta(r^*(e')) &\leq (m\times m)^*(\exists_\Delta(\Delta^*(r\times r)^*(e)r^*(e'))) \Rightarrow \\
e\exists_{\Delta_X}(e') &\leq (m\times m)^*(\exists_\Delta(r^*\Delta_X^*(e)r^*(e'))) \Rightarrow \\
e\exists_{\Delta_X}(e') &\leq (m\times m)^*(\exists_\Delta(r^*(\Delta_X^*(e)e'))) \Rightarrow \\
e\wedge\exists_{\Delta_X}(e') &\leq \exists_{\Delta_X}(\Delta_X^*(e)\wedge e')
\end{aligned}$$

▮

3.3 Equality and PCA's

For a category C with a PCA $(A,\bullet)$ such that A is discrete, it makes sense to study what happens in the partial combinatory algebra in terms of combinators. The necessary combinator would be c_{Eq}, making the following diagram commute:

$$\begin{tikzcd}
(A\times A)\times A \arrow[r, "\bullet\times 1_A"] & A\times A \arrow[r, "\bullet"] & A \\
A\times A \arrow[u, "\langle c_{Eq}!,1_A\rangle\times 1_A"] \arrow[urr, "Eq_A"'] & &
\end{tikzcd}$$

Using set-theoretic notation, this would amount to defining c_{Eq} as follows: for all $a, b \in A$,

$$c_{Eq} \cdot a \cdot b = \begin{cases} a & \text{if } a = b \\ \uparrow & \text{otherwise.} \end{cases}$$

It is also worth pointing out that similar diagrams must commute when combinators Eq^c and EQ, corresponding to distinctness and strong equality are found in the category. Even though EQ_A may be a map to some object $X \neq A$, it may be treated as a map into A as well (as every X is a retract of the Turing object), and has a corresponding combinator. Now, the following condition can be formulated regarding the weak equality combinator in a PCA and the category built around it.

Theorem 3.3.1 Suppose C is a cartesian restriction category with a PCA $(A, \bullet)$. The PCA $(A, \bullet)$ has an equality combinator c_{Eq} if and only if the category Comp($\mathbb{A}$) is discrete.

Proof: When the category Comp($\mathbb{A}$) is discrete, it immediately implies that the map Eq_A, and therefore the combinator c_{Eq} as defined in the diagram above, are present in the underlying PCA $(A, \bullet)$.

Now, suppose the combinator c_{Eq} exists in the PCA $(A, \bullet)$. The equality $Eq_A : A \times A \to A$ is then also a map in Comp($\mathbb{A}$). So, A is discrete and it follows by the retract argument that all the other objects are discrete also. ▮

For $n \geq 2$ the equality on A^n, $Eq_{A^n} : A^n \times A^n \to A^n$ can actually be constructed using Eq_A and the product structure by first applying the following isomorphim

$$\phi : (A_1 \times ... \times A_n) \times (A_{n+1} \times ... \times A_{n+n}) \to (A_1 \times A_{n+1}) \times (A_2 \times A_{n+2}) \times ... \times (A_n \times A_{n+n})$$

with $A_i = A$ for $1 \leq i \leq n$, which permutes the objects, then applying the product of n maps Eq_A to $(A \times A)^n$, mapping into A^n. Moving on to the larger Turing categories containing Comp($\mathbb{A}$), the following is a corollary of the previous 2 propositions that gives a sufficient condition for a Turing category to be discrete.

Corollary 3.3.2 Suppose C is a cartesian restriction category, and $(A, \bullet)$ is a PCA in C. Suppose also that $(A, \bullet)$ has an equality combinator. Then, Split(Comp($\mathbb{A}$)) is a discrete category, and furthermore, when A is a Turing object in C, C is also discrete.

Proof:

Having an equality combinator in a PCA $(A, \bullet)$ means that A is a discrete object, and an earlier proposition has already established that every object that is a retract of a discrete object A is discrete. It follows immediately that Split(Comp($\mathbb{A}$)) is discrete, and so is C when A is its Turing object.

∎

3.4 Equality and Ranges

It is interesting to note that when Eq_A exists for an object A in a cartesian restriction category C, this also ensures the presence of certain ranges (openness of certain maps) in the category. Using Eq_A, it is possible to restrict the domain of any idempotent such that the resulting map is an (open) restriction idempotent which is equivalent to the original. This result is analogous to a result stated in lemma (2.2.1) in the Ranges chapter.

Lemma 3.4.1 Let C be a cartesian restriction category with a discrete object A. Then, for every object $X \in \mathsf{C}$ that is a retract of A, the embedding and retraction maps are open, and it is possible to define a restriction idempotent on A, with a splitting that is isomorphic to X.

Proof:

Observe that it follows from the definition that for any idempotent u on $A \in \mathsf{C}$, the map $Eq_A(u \times 1)\Delta_A$ is a restriction idempotent:

$$\begin{aligned}
\Delta_A Eq_A &\leq 1, \text{ since } Eq_A \text{ is a partial inverse to } \Delta_A \\
\Delta_A Eq_A(u \times 1)\Delta_A &\leq (u \times 1)\Delta_A, \text{ since composing preserves } \leq \\
\pi_0 \Delta_A Eq_A(u \times 1)\Delta_A &\leq \pi_0(u \times 1)\Delta_A, \text{ by properties of partial products, this is} \leq \pi_0\Delta_A, \text{ so} \\
\pi_0 \Delta_A Eq_A(u \times 1)\Delta_A &\leq \pi_0\Delta_A \\
\pi_0 \Delta_A Eq_A(u \times 1)\Delta_A &\leq 1, \text{ and} \\
Eq_A(u \times 1)\Delta_A &\leq 1,
\end{aligned}$$

meaning exactly that $Eq_A(u \times 1)$ is a restriction idempotent. At the same time, as the projections are equal, so

$$\begin{aligned}
\pi_1 \Delta_A Eq_A(u \times 1)\Delta_A &\leq \pi_1(u \times 1)\Delta_A \\
&\leq u, \text{ which implies} \\
Eq_A(u \times 1)\Delta_A &\leq u, \text{ so that} \\
uEq_A(u \times 1)\Delta_A &= Eq_A(u \times 1)\Delta_A \\
&= \overline{uEq_A(u \times 1)\Delta_A}
\end{aligned}$$

Suppose $X \in \mathsf{C}$ with an embedding retraction pair (m, r). Then the pair $(m, r') = (m, rEq_A(mr \times 1_A)\Delta)$ is also an embedding-retraction pair for X, and it is such that $mr' = \overline{mr'}$:

$$r'm = rEq_A(mr \times 1_A)\Delta m$$

$$\begin{aligned}
&= rEq_A(mr \times 1_A)(m \times m)\Delta_X \\
&= rEq_A(m \times m)\Delta_X \\
&= rEq_A\Delta m \\
&= rm = 1, \text{ and}
\end{aligned}$$

$$\begin{aligned}
\overline{mr'} &= \overline{mrEq_A(mr \times 1_A)} \\
&= mrEq_A(mr \times 1_A) \text{ as demonstrated above.}
\end{aligned}$$

That is, m and r' are partial isomorphisms, so that the $\widehat{m} = Eq_A(mr \times 1_A)\Delta$ is the range of m. It follows that r and so, mr, are open.

∎

Indeed, saying that for an idempotent e on X, $e = Eq_X(e \times 1)\Delta_X$ is equivalent to describing e as a restriction idempotent. This lemma sets the stage for the theorem characterizing when a Turing category with a discrete Turing object is a range category.

Theorem 3.4.2 Let C be a Turing category with a discrete Turing object A. Suppose the application map $\bullet : A \times A \to A$ is an open map and assume BCC. Then the entire category is a range category.

Proof: By previous lemma, for any object $X \in \mathsf{C}$, there exists an embedding - retraction pair (m, r') such that $mr' \in \mathcal{O}(A)$, and m and r must then be open. Now, by proposition (2.4.1) from the range chapter, when the codes (embeddings of 1 into A) and the $\bullet$ map are open, and by BCC, every $A \to A$ map is open. It follows that the entire category C is a range category. ∎

In the standard model, making an arbitrary idempotent into a restriction idempotent using this corollary would work like this: suppose $X = \{0,1,2,3,4\}$, and (m,r) is an embedding retraction pair for X such that

$$\begin{aligned} m(x) &= x \\ r(a) &= a \bmod 5, \text{ then} \end{aligned}$$

$$\begin{aligned} r'(x) &= rEq_{\mathbb{N}}(mr \times 1_A)\Delta_{\mathbb{N}}(x) \\ &= rEq_A(mr \times 1_A)(x,x) \\ &= r\begin{cases} x & \text{if } x = x \bmod 5 \\ \uparrow & \text{otherwise.} \end{cases} \\ &= \begin{cases} x & \text{if } x \in \{0,1,2,3,4\} \\ \uparrow & \text{otherwise.} \end{cases} \end{aligned}$$

3.5 Equality and Meets

Recall that the meet combinator $\wedge$ appeared when we considered meets of idempotents in the poset $\mathcal{O}(X)$. In some cases, we can extend the meet combinator to all hom-sets $\mathsf{C}(X,Y)$.

Definition 3.5.1 In a restriction category C, a **meet** combinator $- \wedge -$ sends a pair of maps $f, g : A \to B$ to a map $f \wedge g : A \to B$ such that

[M.1] $f \wedge f = f$

[M.2] $f \wedge g \leq f$ and $f \wedge g \leq g$

[M.3] $(f \wedge g)h = fh \wedge gh$ for any $h : A' \to A$

In Par, $\wedge$ would be defined in the following way: for $f, g : A \to B$,

$$(f \wedge g)(a) = \begin{cases} f(a) & \text{if } f(a) = g(a) \\ \uparrow & \text{otherwise.} \end{cases}$$

The same definition is inherited by $\mathbb{N}$. As the equality comparison is computable, the above definition is computable for any two computable maps.

There are various relations between the concepts of meets, open maps and discreteness. The most important relation is the following, found in [1].

Lemma 3.5.2 A cartesian restriction category C has meets whenever it is discrete.

The idea of the proof is that for any $X \in \mathsf{C}$, when Eq_X is defined, $\wedge$ can be defined by

$$f \wedge g = Eq_X(f \times g)\Delta$$

Conversely, Eq_X can be defined in terms of $\wedge$:

$$\begin{aligned} Eq_X &= \pi_0\overline{\pi_0 \wedge \pi_1} \\ &= \pi_1\overline{\pi_0 \wedge \pi_1} \end{aligned}$$

Thus, in a range category with $\pi_0\widehat{\Delta_X} = \pi_1\widehat{\Delta_X}$, equality maps, and therefore meets, are immediately present. In particular, in Turing categories, having a $\wedge$ combinator is equivalent to having discreteness. The underlying PCA would then have a code $c_{f\wedge g}$ for the map $f \wedge g$. In fact, the PCA will automatically have a stronger version of the code for $\wedge$, namely $c_\wedge$, with $c_\wedge \cdot c_f \cdot c_g = c_{f\wedge g}$ for arbitrary maps $f, g : A \to A$. This is because $c_\wedge \cdot c_f \cdot c_g$ is exactly the code for the existing map $Eq_A(f \times g)\Delta_A$.

Chapter 4

Coproducts in Turing Categories

The dual concepts of a restriction terminal object and partial products may also appear in a cartesian restriction category C — these are the initial object and coproducts. In a cartesian restriction category, the term product usually refers to a partial product, which is equivalent to the usual categorical product in the subcategory Tot(C). In the case of coproducts as well as initial objects, however, partiality incorporates seamlessly into the definitions, and they stay true to the usual categorical ones.

In this chapter, coproducts in cartesian restriction categories will be discussed, followed by a detailed investigation of the precise conditions under which a Turing category has coproducts. We also include a discussion of distributivity in Turing categories and explain why a Turing category cannot have biproducts.

4.1 Coproducts in Cartesian Restriction Categories

Recall that, as defined in [8],

Definition 4.1.1 A **coproduct** of objects A and B in a category C is an object

$A + B$ for which there exist two injections, $i_A : A \to A + B$ and $i_B : B \to A + B$, such that for each pair of maps $f : A \to C, g : B \to C, C \in \mathsf{C}$ there exists a unique map $[f, g] : A + B \to C$, with

$$[f, g]i_A = f \qquad [f, g]i_B = g$$

Note that in a restriction category C, with the minor assumption that it has a terminal object, the injections for a coproduct $A + B$ must be total maps.

Lemma 4.1.2 Suppose C is a cartesian restriction category containing the coproduct $A + B$. Then the coproduct injections i_A, i_B are total maps.

Proof: Consider the unique total maps $!_A, !_B$ into the terminal object 1 :

$$\begin{aligned} [!_A, !_B]i_A &= !_A \Rightarrow \\ \overline{[!_A, !_B i_A]} &= \overline{!_A} \\ \overline{\overline{[!_A, !_B]}i_A} &= 1_A \end{aligned}$$

on the other hand,

$$\begin{aligned} \overline{\overline{[!_A, !_B]}i_A} &\leq \overline{i_A}, \text{ so} \\ \overline{i_A} &= 1_A \end{aligned}$$

∎

Since idempotent splittings will again play an important role in the subsequent sections, we briefly address the question of how idempotents and coproducts interact. Suppose we have a coproduct $A + A$ and want to understand the poset $\mathcal{O}(A + A)$. Right away, the idempotents that can be defined on $A + A$ are $e_1 + e_2$, where e_1, e_2 are idempotents on A, and all idempotents must be of this form. Indeed, for an arbitrary idempotent e on $A+A$, consider the following commuting diagram:

$$\begin{tikzcd}
A \arrow[r, "i_{A_1}"] \arrow[d, "\overline{ei_{A_1}}"'] & A+A \arrow[d, "e"] & A \arrow[l, "i_{A_2}"'] \arrow[d, "\overline{ei_{A_2}}"] \\
A \arrow[r, "i_{A_1}"] & A+A & A \arrow[l, "i_{A_2}"']
\end{tikzcd}$$

Which shows exactly that $e = \overline{ei_{A_1}} + \overline{ei_{A_2}}$.

Another useful observation is concerned with the existence of coproducts in split categories:

Lemma 4.1.3 Suppose $A+A$ exists in a cartesian restriction category C. Suppose furthermore that X and Y are retracts of A. Then the coproduct $X+Y$ exists in Split(C).

Proof: Let e_1 and e_2 be idempotents with splittings X and Y respectively, and the embedding-retraction pairs $(m_1, r_1, (m_2, r_2)$. The sum $e_1 + e_2 = r_1 m_1 + r_2 m_2 = (r_1 + r_2)(m_1 + m_2)$ also splits in Split(C), so that the corresponding embedding-retraction pair for the splitting would be $(m_1 + m_2, r_1 + r_2)$:

$$(m_1 + m_2)(r_1 + r_2) = m_1 r_1 + m_2 r_2 = 1_X + 1_Y = 1_{X+Y}$$

Thus, $(m_1 + m_2, r_1 + r_2)$ is the embedding-retraction pair for $X+Y$. ∎

4.2 Coproducts and Turing Structure

In order to study the question when arbitrary binary coproducts exist in a Turing category C, it is beneficial to first consider the coproduct $1+1$, as the existence of that particular coproduct already has significant consequences in Turing categories.

Our first observation is that the object $1+1$ can be embedded into the Turing object in a particular way:

Lemma 4.2.1 Suppose C is a Turing category with Turing object A, and containing the coproduct $1+1$. Then $[k, k^*]$ is an embedding of $1+1$ into A.

Proof: An embedding of $1+1$ into A is of the form $[x, y]$, where $x, y : 1 \to A$ are codes. Let e be a combinator for the lambda calculus computation $\lambda a.a \cdot x \cdot y$, so that

$$ek = kxy = x, \qquad ek^* = k^*xy = y$$

Letting $E = \bullet\langle e!_A, 1_A\rangle : A \to A$, we have $E[k, k^*]i_1 = e \cdot k = x, E[k, k^*]i_2 = e \cdot k^* = y$. It follows that $E[k, k^*] = [x, y]$. If r is the retraction of the embedding $[x, y]$, then $rE : A \to 1+1$ is a retraction for the embedding $[k, k^*]$:

$$rE[k, k^*] = r[x, y] = 1_{1+1}$$

∎

With the $[k, k^*]$ embedding, any map of the form $[f, g] : 1+1 \to X, X \in \mathsf{C}$, factors in the following way:

$$\begin{tikzcd}
1 \arrow[r, "i_1"] \arrow[dddr, "f"'] & 1+1 \arrow[d, "\phi"] & 1 \arrow[l, "i_2"'] \arrow[dddl, "g"] \\
 & A \times A \times A \arrow[d, "\bullet^2"] & \\
 & A \arrow[d, "rX"] & \\
 & X &
\end{tikzcd}$$

where $\phi = \langle [k, k^*], c_f!, c_g! \rangle$, and $c_f = m_X f, c_g = m_X g$ are codes for f and g respectively. Then, $[f, g] = r_X \bullet^2 \phi$.

The pair of injections $i_1, i_2 : 1 \rightarrow 1 + 1$ possesses a property which will be important in what follows:

Definition 4.2.2 In a category C, two maps $f : A \rightarrow C, g : B \rightarrow C$ are said to be **jointly epimorphic** whenever for any maps $h, h' : C \rightarrow D$,

$$hf = h'f \text{ and } hg = h'g \qquad \Rightarrow \qquad h = h'$$

Similarly, we can talk about arbitrary families of jointly epimorphic maps.

Note that any pair of coproduct injections is jointly epimorphic. The following condition regarding jointly epimorphic families will be useful in describing when a Turing category will have certain coproducts.

Definition 4.2.3 Suppose $E = \{f_i : A_i \rightarrow B | i \in I\}$ is a set of jointly epimorphic maps. Then, the familiy E is called **pullback stable along projections** (henceforth, stable) whenever $\{f_i \times C : A_i \times C \rightarrow B \times C | i \in I\}$ is also a set of jointly epimorphic maps.

Now, it is possible to say when a coproduct of a Turing object with itself is present in a Turing category.

Proposition 4.2.4 Suppose C is a Turing category with PCA $(A, \bullet)$. Suppose the coproduct $1 + 1$ exists in C, and assume the epimorphic family of injections $\{i_1, i_2 : 1 \rightarrow 1 + 1\}$ is pullback stable along projections. Then the coproduct $A + A$ exists in C.

Proof: With $1+1$, we are going to show that $A+A$ may be taken to be $(1+1) \times A$ in C. If $i_1, i_2 : 1 \rightarrow 1 + 1$ are injections, taking the product of $(1 + 1)$ with A, and the injections with 1_A gives the coproduct injections $i_{A_1} = 1_A \times i_1, i_{A_2} = 1_A \times i_2$ of A into $(1+1) \times A$.

For the universal property, first consider maps $f, g : A \to A$. The map $[f, g]$ is illustrated in the following coproduct diagram:

$$\begin{tikzcd}
A \cong 1 \times A \arrow[r, "i_{A_1}"] \arrow[ddr, "f"'] & (1+1) \times A \arrow[d, "\psi \times 1_A"] & 1 \times A \cong A \arrow[l, "i_{A_2}"'] \arrow[ddl, "g"] \\
 & A \times A \arrow[d, "\bullet"] & \\
 & A &
\end{tikzcd}$$

where $\psi = \bullet^2 \langle [k, k^*], c_f!, c_g! \rangle : 1 + 1 \to A$ is the map illustrated in the $1+1$ coproduct diagram above. That is, $[f, g] = \bullet(\psi \times 1_A)$. Then

$$\begin{aligned}
\bullet(\psi \times 1_A) i_{A_1} &= \bullet(\psi \times 1_A)(i_1 \times 1_A) \\
&= \bullet(c_f \times 1_A) \\
&= f,
\end{aligned}$$

and the corresponding derivation holds true for the g as well.

The uniqueness of the map $[f, g]$ is a result of the injections $i_{A_1} = i_1 \times 1_A, i_{A_2} = i_2 \times 1_A : 1 \times A \to (1+1) \times A$ being jointly epimorphic, as the epimorphic family of injections $\{i_1, i_2 : 1 \to 1 + 1\}$ is pullback stable. ∎

4.3 Distributivity

The way we define the coproduct $A + A$ in a Turing category is a specific instance of distributivity of products over coproducts. Distributivity is defined as follows:

Definition 4.3.1 Let C be a category with products and coproducts. Then C is said to be **distributive** when for every $X, Y, Z \in \mathsf{C}$, the canonical map

$$[i_X \times 1_Z, i_Y \times 1_Z] : X \times Z + Y \times Z \to (X + Y) \times Z$$

is an isomorphism.

And in general, it is one of the features of Turing categories with coproducts. The proof for this claim follows a similar logic as proposition (4.2.4).

Lemma 4.3.2 Assuming pullback stability (along projections) for every family of coproduct injections, a Turing category with coproducts is distributive.

Proof: Suppose C is a Turing category with Turing object A. If $f : X \to B, g : Y \to B$ are maps in C with indices h_f and h_g respectively. It follows from the definition of $X + Y$ that there exists a unique h making the diagram commute:

$$\begin{tikzcd}
X \times Z \arrow[r, "i_X \times 1_Z"] \arrow[dr, "h_f \times 1_Z"] \arrow[ddr, "f"'] & (X + Y) \times Z \arrow[d, "h \times 1_Z"] & Y \times Z \arrow[l, "i_Y \times 1_Z"'] \arrow[dl, "h_g \times 1_Z"'] \arrow[ddl, "g"] \\
 & A \times Z \arrow[d, "\tau_{Z,B}"] & \\
 & B &
\end{tikzcd}$$

This shows exactly that $(X + Y) \times Z$ is a coproduct of $X \times Z$ and $Y \times Z$, with injections $i_{X \times Z} = i_X \times 1_Z, i_{Y \times Z} = i_Y \times 1_Z$. Here,

$$[f, g] = \tau_{Z,B}(h \times 1_Z)$$

is the necessary map. Again, pullback stability along projections of the $\{i_X, i_Y\}$ family of injections guarantees uniqueness of this map. ∎

By the same logic, the category Split(C) is also distributive under the condition.

When a coproduct of Turing objects is found in a Turing category C, it does not always guarantee that arbitrary coproducts exist. However, all coproducts are present if idempotents are required to split.

Theorem 4.3.3 Let C be a split Turing category with Turing object A. Suppose that $1+1$ exists in C and that the coproduct injections are stable. Then all binary coproducts exist in C and are distributive.

Proof: By proposition (4.2.4), the coproduct $A+A$ exists. Now, take an arbitrary pair of objects X, Y; then by lemma (4.1.2), the coproduct $X+Y$ exists, because C is split. Now, by proposition (4.3.2), distributivity follows.

∎

Corollary 4.3.4 Let C be a cartesian restriction category with PCA $(A, \bullet)$. If there exists a combinator c in A such that $c \cdot -$ is an idempotent with splitting $1+1$, then Split(Comp($\mathbb{A}$)) has all coproducts.

Note that if A is a Turing object, with $A+A$ in C, $A+A$ is another Turing object, assuming that the zero map $0_A : A \to A$ exists in the category. An embedding-retraction pair of A into $A+A$ is $(i_{A_1}, [1,0])$, and it follows that the coproduct of an any number of Turing objects is also Turing.

4.4 Biproducts

There are limitations on the coproduct structure in C, such as, for one, products and coproducts in C cannot coincide. This actually follows from the distributivity of A, but will be shown here explicitly.

Lemma 4.4.1 In a non-trivial Turing category C products and coproducts (if they exist) must be distinct — that is, it does not have biproducts.

Proof: Suppose $A \oplus A$ is a biproduct, so that $\bullet = [f, g]$ for some maps $f, g : A \to A$. Let $d : A \to A$ be some map in C with code c. Then for the map $[1, d] : 1 \times A \to A$,

$$\begin{tikzcd}
A \times A \arrow[r, "\bullet", "{[f,g]}"'] & A \\
1 \times A \arrow[u, "c\times 1"] \arrow[ur, "{[1,d]}"'] &
\end{tikzcd}$$

So that $[fc, g] = [1, d]$. Then

$$\begin{aligned}
[fc, g]i_A &= [1, d]i_A, \\
g &= d
\end{aligned}$$

so that C has only one $A \to A$ map, and must be trivial. ∎

Example 4.4.2 As a primary example, consider again Comp($\mathbb{N}$). Although $\mathbb{N} + \mathbb{N} \cong \mathbb{N} \times \mathbb{N}$, with $i_1 a + i_2 b \to (a, b)$, for a map such as $Eq_{\mathbb{N}}$ there are no maps $Eq_1, Eq_2 : \mathbb{N} \to \mathbb{N}$ making this diagram commute:

$$\begin{tikzcd}
\mathbb{N} \arrow[r, "i_1"] \arrow[dr, "Eq_1"'] & \mathbb{N} \oplus \mathbb{N} \arrow[d, "Eq_{\mathbb{N}}"] & \mathbb{N} \arrow[l, "i_2"'] \arrow[dl, "Eq_2"] \\
 & \mathbb{N} &
\end{tikzcd}$$

Conclusion

This thesis establishes several connections between categorical structure and computational aspects of Turing categories. The exploration of equality, range and coproduct structures in a Turing category showcases examples of how to determine what a general computational setting might behave like based on what is known about traditional computation.

There is indeed a large amount of structure to be studied in the category of natural numbers and computable functions, and much of it is likely to have some corresponding implications in Turing categories. It may also be of interest to study particular examples of Turing categories containing (or lacking) certain structure, e.g. looking for Turing categories with PCA's that have weak range combinators or weak equality, but not the strong versions. Examining free PCA's, where the only relations between combinators are those imposed by the rules of combinatory completeness, may be a good place to start when searching for examples.

This thesis is a study of three examples of categorical structure found in the motivating example based on traditional computation. One can also consider specific computational features of the standard model and try to identify categorical structure capturing those features. For example, in the standard model, a key feature is the fact that every r.e set can be approximated by its finite subsets. Thus, it would be interesting to pursue notions of finiteness in Turing categories. Similarly, in recursion theory one often uses the fact that the natural numbers are well-ordered. Thus, it makes sense to look for Turing objects that have an analogous order structure on them.

These are just a few of the possible different directions one can pursue in the study of Turing categories. Each of these is motivated by the desire to better understand the meaning of the various aspects of traditional computation in the categorical language.

Bibliography

[1] J.R.B. Cockett and Pieter Hofstra. Range categories II: Towards regularity. Submitted for publication in 2011, available at "http://mysite.science.uottawa.ca/phofstra/".

[2] J.R.B. Cockett and P.J.W. Hofstra. Introduction to Turing categories. *Annals of Pure and Applied Logic*, 156(2-3):183–209, 2008.

[3] J.R.B Cockett and S. Lack. Restriction categories I. *Theoretical Computer Science*, 270:223–259, 2002.

[4] N.J. Cutland. *Computability: An introduction to recursive function theory*. Cambridge University Press, 1980.

[5] J.R.B. Cockett, Xiuzhan Guo and Pieter Hofstra. Range categories I. Submitted for publication, available at "http://mysite.science.uottawa.ca/phofstra/".

[6] Pieter J.W. Hofstra. Partial combinatory algebras and realizability toposes. 2004.

[7] John L. Kelley. *General Topology*. Graduate Texts In Mathematics. Springer, Birkhauser, 1975.

[8] Saunders Mac Lane. *Categories for the Working Mathematician*. Graduate Texts In Mathematics. Springer, New York, second edition edition, 1998.

[9] Peter J. Freyd, Andrej Scedrov. *Categories, Allegories.* North-Holland Mathematical Library. Elsevier, Amsterdam, 1990.

[10] A. Joyal, M. Nielson, G. Winskel. Bisimulation and open maps. *Proceedings of Eighth Annual IEEE Symposium on Logic in Computer Science*, LICS '93:418 – 427, 1993.