Realizability as the Connection between Computable and Constructive Mathematics

Andrej Bauer
Department of Mathematics and Physics
University of Ljubljana*

October 16, 2005

Abstract

These are lecture notes for a tutorial seminar which I gave at a satellite seminar of "Computability and Complexity in Analysis 2004" in Kyoto. The main message of the notes is that *computable mathematics is the realizability interpretation of constructive mathematics.* The presentation is targeted at an audience which is familiar with computable mathematics but less so with constructive mathematics, category theory or realizability theory.

Contents

*Address: Jadranska 19, 1000 Ljubljana, Slovenia. E-mail: Andrej.Bauer@andrej.com

1 Introduction

I have attended a number of talks on computable analysis in which I got a distinct feeling that the subject could be presented more cleanly and results arrived at more effortlessly if the speaker used a bit of *realizability theory*. This would allow him or her to import known results from constructive mathematics into the computable setting and avoid having to discover things that can be computed mechanically. The purpose of these notes is to help those who are interested in the basics of realizability theory, especially in relation to computable and constructive analysis. These notes ignore complexity issues and focus entirely on computability.

Realizability theory is a subject which originated with Kleene's [Kle45] interpretation of intuitionistic number theory and has since developed into a large body of work in logic and theoretical computer science. We shall focus only on two basic flavors of realizability, number realizability and function realizability, which were both invented by Kleene [Kle45, KV65]. They correspond to two best known "schools" of computable mathematics, namely Recursive Mathematics and Type Two Effectivity.

We assume familiarity with computable analysis at the level of Prof. Weihrauch's gentle textbook [Wei00]. We presuppose only very rudimentary knowledge of logic, intuitionistic logic, and category theory.

By *classical mathematics* we mean mathematics developed in the realm of classical logic. By *constructive mathematics* we mean mathematics developed in the realm of intuitionistic logic, as was done by Bishop [Bis67]. By *computable mathematics* we mean the study of computability in classical mathematics.

Warning. This is a draft version which I wrote hurriedly to have it ready for the audience at the tutorial seminar at "Computability and Complexity in Analysis 2005" in Kyoto. I guarantee that the notes contain errors and that bibliography omits important references. I gratefully collect error reports and comments about the notes that will help me improve them.

2 Models of Computability

There is a universally accepted notion of computation, namely that of Turing machines, or any of its equivalent formulations. One would think that this is the end of the story as far as computability is concerned. Surely, if everyone agrees on what it means to compute, then everyone should agree on, for example, what it means to compute the solution of a differential equation. But this is not so.

Any specific description of computation involves manipulation of concrete mathematical objects, such as tapes, numbers, or syntactic expressions. When we study computability on abstract mathematical entities, we are thus naturally lead to their *representations* by such concrete objects. It turns out that the choice of representations, and more importantly of the *type* of representing objects, is a source of differences between various schools of computability.

Let us give an example of how different representations lead to differences in computability. There are at least two well known representations of real numbers:

Type I reals: a real number x is *represented by a number* $n \in \mathbb{N}$, such that the n-th Turing machine (in an effective enumeration of all Turing machines) on input $k \in \mathbb{N}$ terminates and outputs a pair of integers $a, b \in \mathbb{Z}$ such that $|x - a/b| < 2^{-k}$. A real number which has such a representation is *computable*.

Type II reals: a real number x is *represented by an infinite tape* containing a sequence of rational approximations a_1/b_1, a_2/b_2, ..., such that $|x - a_k/b_k| < 2^{-k}$ for all $k \in \mathbb{N}$. A real number is *computable* if there is a computable sequence[1] representing it.

Whereas in Type II we can represent *all* real numbers, in Type I we cannot do so because there are only countably many Turing machines but uncountably many reals. Nevertheless, it is easy to show that Type I computable reals are the same as Type II computable reals, so we do not have a difference in computability yet. We denote the set of computable reals by $\mathbb{R}_c$, and similarly let $[0,1]_c = \mathbb{R}_c \cap [0,1]$.

The next step is to define what it means for a real function to be computable:

Type I functions: A function $f : \mathbb{R}_c \to \mathbb{R}_c$ is *computable* if there exists a Turing machine T such that whenever $n \in \mathbb{N}$ represents $x \in \mathbb{R}_c$, then T on input n terminates and outputs a representation of $f(x)$.

Type II functions: A function $f : \mathbb{R} \to \mathbb{R}$ is *computable* if there exits a Turing machine T such that if $a_1/b_1, a_2/b_2, \ldots$ represents $x \in \mathbb{R}$, the machine T outputs a representation of $f(x)$ when given as input the infinite tape containing the sequence $a_1/b_1, a_2/b_2, \ldots$[2]

We may analogously define Type I computable functions $[0,1]_c \to \mathbb{R}_c$ and Type II computable functions $[0,1] \to \mathbb{R}$. A first difference between Type I and Type II computability is that Type I computable functions accept as inputs and give as outputs only the computable reals. In contrast, Type II computable functions may in principle accept any real as input. It is easy to see that a Type II computable function outputs a computable real when given a computable real as input. So perhaps Type I computable functions are just the restrictions of Type II computable functions to computable reals. That this is not the case is witnessed by the following theorems.

Theorem 2.1 (Type I) *There is a computably* unbounded *computable function* $f : [0,1]_c \to \mathbb{R}_c$.

Theorem 2.2 (Type II) *Every computable function* $f : [0,1] \to \mathbb{R}$ *is computably* bounded.

[1] A sequence is computable if there exists a Turing machine which on input n outputs the n-th term of the sequence.

[2] The machine T has an input tape, a *write-only* output tape, and a working tape. The computation itself is infinite, but we can imagine stopping it in practice when the desired approximation has been reached. Note that T writes out any finite part of its output in finite time.

The function $f : [0,1]_c \to \mathbb{R}_c$ from the first theorem is not the restriction of any Type II computable function $g : [0,1] \to \mathbb{R}$. If it were, we could compute a bound M for g by the second theorem. But then M would be a bound for f too, which is impossible by the first theorem. Therefore, Type I computable functions are different from Type II computable functions.

The preceding example shows that questions about computability in analysis are intricate, and that seemingly unimportant decisions may have unexpected consequences. It is thus useful to have a tool such as realizability theory, which helps organize various choices one can make into a coherent framework. As an added benefit, the realizability interpretation of logic connects the Bishop-style constructive mathematics with the computable world.

2.1 Type I Computability

In Type I computability objects of computation are represented by finite strings of symbols from a given finite alphabet. Computations are performed by Turing machines which operate on such strings. As we are not interested in complexity issues, we may instead use natural numbers instead of finite strings and partial recursive functions instead of Turing machines.

Let $(\boldsymbol{\varphi}_n)_{n\in\mathbb{N}}$ be an effective enumeration of partial recursive functions and $\langle -,- \rangle$ an effective pairing function on natural numbers with projection functions $\mathsf{fst}\,\langle m,n\rangle = m$ and $\mathsf{snd}\,\langle m,n\rangle = n$. The pairing function allows us to view multivariate functions as univariate, i.e., instead of having an $f : \mathbb{N}\times\mathbb{N} \rightharpoonup \mathbb{N}$ which takes two arguments m and n, we may have instead $f : \mathbb{N} \rightharpoonup \mathbb{N}$ which takes a single argument $\langle m,n\rangle$.

We recall the important utm and smn theorems from the theory of partial recursive functions.

Theorem 2.3 (utm) *There is a universal partial recursive function, i.e., a partial recursive function $\boldsymbol{u} : \mathbb{N} \rightharpoonup \mathbb{N}$ such that $\boldsymbol{u}\langle m,n\rangle \simeq \boldsymbol{\varphi}_m(n)$.*

Theorem 2.4 (smn) *For every partial recursive function $p : \mathbb{N} \rightharpoonup \mathbb{N}$ there is a recursive function $s : \mathbb{N} \to \mathbb{N}$ such that $p\langle m,n\rangle \simeq \boldsymbol{\varphi}_{s(m)}(n)$. Moreover, the function s may be obtained effectively from a description of p.*

We denote the model of computability based on natural numbers and partial recursive functions by K_1, because it is also known as *Kleene's first partial combinatory algebra*.

2.2 Type II Computability

In Type II computability objects of computation are represented by *infinite* strings of symbols from a given finite alphabet. Computations are performed by Turing machines which write infinite outputs on a write-only tape, i.e., once a machine writes a symbol on the output tape it is not allowed to alter it. This is necessary to prevent the machine from changing the output indefinitely. Type II computations are in principle infinite, but we may well imagine that in practice a computation is aborted when a sufficient amount of output has been computed.

Once again, since we are not interested in computational complexity, we may replace infinite strings by infinite sequences of natural numbers. This is a convenient but inessential change—the diligent reader can verify that realizability may be formulated just as well in terms of actual infinite strings over a finite alphabet. We denote sequences of natural numbers by Greek letters α, β, γ, ..., and denote the set of all such sequences by $\mathbb{B} = \mathbb{N}^{\mathbb{N}}$. The set $\mathbb{B}$ carries the product topology, which makes it into a zero-dimensional Hausdorff space, known as the *Baire space*.

A partial map $f : \mathbb{B} \rightharpoonup \mathbb{B}$ is *continuous* if it is continuous as a map $f : \mathsf{dom}(f) \to \mathbb{B}$, where the domain $\mathsf{dom}(f)$ is equipped with the subspace topology. It can be shown that every partial continuous map $f : \mathbb{B} \rightharpoonup \mathbb{B}$ can be extended to a partial continuous one whose domain is a G_δ-set (a countable intersection of open sets) [Bau00, 1.1.6].

A partial function $f : \mathbb{B} \rightharpoonup \mathbb{B}$ is *computable*, if there exists a Turing machine, which takes as input $\alpha \in \mathbb{B}$, written on an infinite input tape, and outputs $f(\alpha)$, or diverges when $f(\alpha)$ is undefined. It can be shown that such an f is continuous and that its domain of definition is

a G_δ-set. Furthermore, there is an encoding $\boldsymbol{\eta}$ of such functions by sequences. The idea is to encode f by a sequence of tuples $\langle [a_0, \ldots, a_n], m, b\rangle$, where such a tuple means "if $[a_0, \ldots, a_n]$ is a prefix of α then $f(\alpha)_m = b$". Conversely, every $\gamma \in \mathbb{B}$ determines a partial continuous $\boldsymbol{\eta}_\gamma : \mathbb{B} \rightharpoonup \mathbb{B}$ whose domain is a G_δ-set, see [Wei00, 2.3] for details.

As in Type I computability, there is a computable pairing operation $\langle -, - \rangle ; \mathbb{B} \times \mathbb{B} \to \mathbb{B}$, which simply interleaves sequences, together with projections fst and snd. Also in Type II computability we get the utm and smn theorems [Wei00, 2.3] in which we use the encoding $\boldsymbol{\eta}$ in place of $\boldsymbol{\varphi}$.

Theorem 2.5 (utm) *There is a computable partial function $\boldsymbol{u} : \mathbb{B} \rightharpoonup \mathbb{B}$ such that $\boldsymbol{u}\langle \alpha, \beta\rangle \simeq \boldsymbol{\eta}_\alpha(\beta)$.*

Theorem 2.6 (smn) *For every computable partial map $p : \mathbb{B} \rightharpoonup \mathbb{B}$ there is a computable map $s : \mathbb{B} \to \mathbb{B}$ such that $p\langle \alpha, \beta\rangle \simeq \boldsymbol{\eta}_{s(\alpha)}(\beta)$. Furthermore, the function s may be computed from a description of p.*

There are in fact three variations of Type II computability:

Effective: Input and output tapes are only allowed to contain *computable* sequences. Only computable partial maps $\mathbb{B} \rightharpoonup \mathbb{B}$ are considered.

Continuous: Input and output tapes may contain arbitrary sequences. All partial *continuous* maps $\mathbb{B} \rightharpoonup \mathbb{B}$ whose domains are G_δ-sets are allowed.

Mixed: Input and output tapes may contain arbitrary sequences. Only computable partial maps $\mathbb{B} \rightharpoonup \mathbb{B}$ are allowed.

We have so far described the mixed version, which is most commonly used in computable analysis. We shall not use the effective version, while occasionally we will refer to the continuous one. There are also continuous versions of the utm and smn theorems, in which "computable" is replaced by "continuous with G_δ domain".

We denote the mixed version of Type II computability by K_2 and the continuous one by K_2^{cont}. They are related to *Kleene's second partial combinatory algebra*.

2.3 Other Models

The Type I and Type II models of computability are the ones that are used almost exclusively in computable mathematics. General realizability theory, however, considers a wider class of models, called *(typed) partial combinatory algebras (pca)*. There are many examples of pca's from topology, domain theory and the theory of programming languages. Constructions of realizability categories over these models give many more mathematical universes that support development of computable mathematics, e.g., equilogical space [BBS04] and domain representations [Bla97] to name just two.

A comprehensive account of computable mathematics should encompass a rich spectrum of models of computability. We know that they are there, and we know that they do *not* all give the same computable mathematics, but we know less on how they relate to each other. By focusing on just a couple of models in the name of simplicity and historical importance of Turing machines, an opportunity for the view of a larger picture is missed.

3 Representations

The basic idea of realizability is implicitly known to every programmer: entities with which we want to compute must first be suitably represented, or *realized*, in the computer. In computable mathematics, this idea is formalized by the notions of numberings (Type I) and Baire space representations (Type II). In this section we first recall what these are and formulate them in a way that is best suited for realizability theory.

3.1 Basic Definitions

Definition 3.1 A *numbered set* (A, ν_A) is a set A with a partial surjection $\nu_A : \mathbb{N} \rightharpoonup A$, called the *numbering*. We define $|A| = \{n \in \mathbb{N} \mid \nu_A(n)\downarrow\}$ to be the domain of ν_A. When $x = \nu_A(n)$ we say that n is a *name* of x, or that n *represents* x.

A *realized function* $f : (A, \nu_A) \to (B, \nu_B)$ between numbered sets is a function $f : A \to B$ for which there exists a partial recursive function $p : \mathbb{N} \rightharpoonup \mathbb{N}$ such that $f(\nu_A(m)) = \nu_B(p(n))$ for all $n \in |A|$. We say that p *tracks* or *realizes* f.

Definition 3.2 A *Baire space representation*, or a *representation* for short, $\mathbf{A} = (A, \delta_A)$, is a set A with a partial surjection $\delta_A : \mathbb{B} \rightharpoonup A$. We define $|A| = \{\alpha \in \mathbb{B} \mid \delta_A(\alpha)\downarrow\}$ to be the domain of δ_A, and say that α is a *name* for x, or that α *represents* x, when $x = \delta_A(\alpha)$.

A *computably (resp. continuously) realized function* $f : \mathbf{A} \to \mathbf{B}$ between representations is a function $f : A \to B$ for which there exists a computable (resp. continuous) partial function $p : \mathbb{B} \rightharpoonup \mathbb{B}$ such that $f(\delta_A(\alpha)) = \delta_B(p(\alpha))$ for all $\alpha \in |A|$.

In realizability theory it is customary to think of a representation $\delta_A : \mathbb{B} \rightharpoonup A$ in terms of its graph, i.e., as a relation between $\mathbb{B}$ and A.

Definition 3.3 The *realizability relation* $\Vdash_A$ corresponding to a representation $\mathbf{A}$ is the graph of δ_A:

$$\alpha \Vdash_A x \iff \delta_A(\alpha) = x\ .$$

Because δ_A and $\Vdash_A$ determine each other, we may define representations in terms of their realizability relations.

Definition 3.4 (equivalent to Definition 3.2) A *representation* $\mathbf{A} = (A, \Vdash_A)$ is a set A with a relation $\Vdash_A \subseteq \mathbb{B} \times A$ which is surjective, meaning that

$$\forall\, x \in A\,.\,\exists\, \alpha \in \mathbb{B}\,.\,\alpha \Vdash_A x\ , \tag{1}$$

and single-valued, meaning that, for all $\alpha \in \mathbb{B}$, $x, y \in A$,

$$\alpha \Vdash_A x \wedge \alpha \Vdash_A y \implies x = y\ . \tag{2}$$

A *computably (resp. continuously) realized map* $f : \mathbf{A} \to \mathbf{B}$ is a function $f : A \to B$ for which there exists a computable (resp. continuous) partial map $p : \mathbb{B} \rightharpoonup \mathbb{B}$ such that

$$\alpha \Vdash_A x \implies p(\alpha) \Vdash_B f(x)\ .$$

Condition (1) for $\Vdash_A$ corresponds to the condition that δ_A be surjective, while (2) corresponds to the condition that δ_A be single-valued. If we drop (2) we get a more general notion of *multi-valued* representations, which however we shall not consider here. In realizability theory representations are also known as *modest sets*, and the multi-valued ones as *assemblies*.

To keep things organized, we define two categories of representations, corresponding to the mixed and continuous case:

$$\mathsf{Rep}(K_2) : \text{representations and computably realized functions,}$$
$$\mathsf{Rep}(K_2^{\mathrm{cont}}) : \text{representations and continuously realized functions,}$$

We shall work with $\mathsf{Rep}(K_2)$, and only occasionally in $\mathsf{Rep}(K_2^{\mathrm{cont}})$. Thus, unless explicitly stated otherwise, when we speak of "realized maps" or "morphisms" we mean computably realized maps.

Exercise 3.5 Express the definition of numbered sets with realizability relations instead of numberings and define the category $\mathsf{Rep}(K_1)$ of numbered sets and realized maps.

Representations $\mathbf{A}$ and $\mathbf{B}$ are *isomorphic*, written $\mathbf{A} \cong \mathbf{B}$, if there exists a bijection $f : A \to B$ such that both f and f^{-1} are realized. Note that the realizers for f and f^{-1} need not be inverses of each other.

3.2 Constructions of Representations

One of the basic tasks in Type II computability theory is search for good representations of important spaces $\mathbb{N}$, $\mathbb{R}$, $\mathbb{C}$, ℓ_2, $L^p[0,1]$, etc. There are known criteria for what makes a good representation, such as admissibility, but on the whole one often gets the feeling that good representations are discovered rather than computed. Typically, one faces several possible representations for a given set. It is desirable to have precise criteria for picking the right one. There are various results, see e.g. the "robustness" theorems in [Wei00], which help choose the right representations, but on the whole a bit of experience is needed to always get things right. Realizability theory, with help of category theory and constructive logic, replaces much of the discovery and guesswork process by straightforward computation of representations. Basic constructions of representations, such as products, exponentials (function spaces), disjoint sums, and quotients have standard category-theoretic definitions. Therefore, we may simply *compute* them in the category of representation $\mathsf{Rep}(K_2)$.

To demonstrate the technique, we shall compute products and exponentials of representations starting from their category-theoretic definitions. Many more advanced examples from analysis will follow in later on.

3.2.1 Products

Let us first recall the definition of a product in category, applied to $\mathsf{Rep}(K_2)$. Given representations $\mathbf{A}$ and $\mathbf{B}$, their product is a representation $\mathbf{P}$ together with realized maps $p_1 : P \to A$ and $p_2 : P \to B$, such that for all realized maps $f : C \to A$, $g : C \to B$ there exists a unique realized map $\langle f, g\rangle : C \to P$ satisfying for all $z \in C$

$$p_1\langle f, g\rangle(z) = f(z) \quad \text{and} \quad p_2\langle f, g\rangle(z) = g(z) \; .$$

We first apply this definition to the singleton representation $\mathbf{C} = \mathbf{one}$, which is the singleton set $1 = \{\star\}$ with representation $\delta_1(\alpha) = \star$. Functions $\{\star\} \to A$ are the same thing as elements of A. Thus, for every $x \in A$ and $y \in B$ there must exists a unique $z \in P$, write it as $\langle x, y\rangle$, such that $p_1\langle x, y\rangle = x$ and $p_2\langle x, y\rangle = y$. Therefore, the set P is just the usual cartesian product of sets A and B, with p_1 and p_2 the canonical projections. It remains to compute the representation δ_P. Whatever it is, the canonical projections $p_1 : P \to A$ and $p_2 : P \to B$ are realized by computable partial maps $q_1 : \mathbb{B} \rightharpoonup \mathbb{B}$ and $q_2 : \mathbb{B} \rightharpoonup \mathbb{B}$, which means that

$$\delta_P(\alpha) = \langle x, y\rangle \implies \delta_A(q_1(\alpha)) = x \wedge \delta_B(q_2(\alpha)) = y \; .$$

Thus from a name α of a pair $\langle x, y\rangle$, we should be able to compute names $q_1(\alpha)$ and $q_2(\alpha)$ of its components. This condition tells us that we must put in α enough information to be able to recover the names of x and y. On the other hand, because products are limits (in the sense of category theory), they must be constructed as "economically" as possible, which suggests that a name of a pair should contain just enough information to recover names of its components. In other words, what we need is a pairing operation $\langle -, -\rangle$ on $\mathbb{B}$, and we already know that there is one. To summarize, the product of $\mathbf{A}$ and $\mathbf{B}$ is $\mathbf{A} \times \mathbf{B}$ where

$$\delta_{A\times B}\langle \alpha, \beta\rangle = \langle \delta_A(\alpha), \delta_B(\beta))\rangle \; .$$

The projections $p_1 : A \times B \to A$ and $p_2 : A \times B \to B$ are realized by the projections fst and snd, respectively.

3.2.2 Exponentials

Given representations $\mathbf{A}$ and $\mathbf{B}$, we may ask for a good representation of the set of realized maps $\mathbf{A} \to \mathbf{B}$. In categorical language, we would like to compute the *exponential* of $\mathbf{A}$ and $\mathbf{B}$. By definition, this is a representation $\mathbf{W}$ together with a morphism $e : \mathbf{W} \times \mathbf{A} \to \mathbf{B}$, called the *evaluation*, such that for every morphism $f : \mathbf{C} \times \mathbf{A} \to \mathbf{B}$ there exists a unique morphism $\widetilde{f} : \mathbf{C} \to \mathbf{W}$, called the *transpose* of f, such that, for all $x \in A$, $z \in C$,

$$e\langle \widetilde{f}(z), x\rangle = x \ .$$

Let us first compute the exponential in $\mathsf{Rep}(K_2^{\mathrm{cont}})$, i.e., when morphisms are continuously realized. Applying the definition of exponentials to $\mathbf{A}$ and $\mathbf{B}$ with $\mathbf{C} = \mathbf{one}$ and taking into account the isomorphism $\mathbf{one} \times \mathbf{A} \cong \mathbf{A}$, we see that for every realized map $f : A \to B$ there is a unique element $t \in W$ such that $f(x) = e\langle t, x\rangle$, and conversely, for every element $t \in W$, there is a realized map $A \to B$ defined by $x \mapsto e\langle t, x\rangle$. Therefore, W is the set of all continuously realized functions from A to B.

Since $f(x) = e\langle f, x\rangle$ for every $f \in W$ and $x \in A$, e is the usual evaluation map which applies a function to an argument. We must also check that it is realized. We do not know yet what δ_W might be, but suppose anyway that $\delta_W(\alpha) = f$ for some $f \in W$, that $p : \mathbb{B} \rightharpoonup \mathbb{B}$ tracks f, and $\delta_A(\beta) = x$. A realizer $u : \mathbb{B} \rightharpoonup \mathbb{B}$ for e must satisfy $\delta_B(u\langle\alpha, \beta\rangle) = f(x) = \delta_B(p(\beta))$, which suggests that we should require $u\langle\alpha, \beta\rangle = p(\beta)$. Clearly, there ought to be some relation between a name α of f and a realizer p of f, so we may reasonably expect to have a map h such that $h(\alpha) = p$. Now we get

$$u\langle\alpha, \beta\rangle = p(\beta) = h(\alpha)(\beta) \ . \tag{3}$$

which is a familiar equation: it is the utm theorem for Baire space representations. Hence we can take $h = \boldsymbol{\eta}$, the canonical encoding of partial continuous maps $\mathbb{B} \rightharpoonup \mathbb{B}$ with G_δ domain, and $u = \boldsymbol{u}$ to be the universal map from the utm Theorem 2.5. The representation δ_W is then given by

$$\delta_W(\alpha) = f \iff \boldsymbol{\eta}_\alpha \text{ tracks } f \ .$$

Lastly, we need to check that with this (W, δ_W) every $f : C \times A \to B$ has a unique transpose. Suppose $p : \mathbb{B} \rightharpoonup \mathbb{B}$ tracks $f : C \times A \to B$. The transpose $\widetilde{f} : C \to W$, if it exists, must satisfy, for all $x \in A$, $z \in C$,

$$\widetilde{f}(z)(x) = f(z, x)$$

which determines $\widetilde{f}$ uniquely as a function between sets. To verify that the codomain of $\widetilde{f}$ is W, we must show that the map $x \mapsto f(z, x)$ is continuously realized for every $z \in C$, which it is by $\beta \mapsto \boldsymbol{\eta}_\alpha\langle\gamma, \beta\rangle$, where $\gamma \in \mathbb{B}$ is any name of z. This establishes $\widetilde{f} : C \to W$ as a function. For it to be a morphism, we seek a partial continuous $s : \mathbb{B} \rightharpoonup \mathbb{B}$ such that

$$\boldsymbol{\eta}_{s(\gamma)}(\beta) = p\langle\gamma, \beta\rangle \ .$$

This is exactly what the smn theorem 2.6 gives us. The transpose $\widetilde{f}$ is exists and is unique.

Let us summarize the construction we have just computed: in $\mathsf{Rep}(K_2^{\mathrm{cont}})$, the exponential of $\mathbf{A}$ and $\mathbf{B}$ is $\mathbf{W} = \mathbf{B}^{\mathbf{A}}$ where

$$W = \{f : A \to B \;\big|\; f \text{ is continuously realized}\}$$

and

$$\delta_W(\alpha) = f \iff \boldsymbol{\eta}_\alpha \text{ tracks } f \ .$$

How about the mixed case $\mathsf{Rep}(K_2)$? Is W the set of computably realized or the set of continuously realized functions? As it turns out, we need to take the continuously realized ones. To see the

reason for this, consider $\theta \in \mathbb{B}$ which is *not* computable. Let $\mathbf{C}$ be the representation $C = \{\dagger\}$ with realizability relation

$$\delta_C(\alpha) = \dagger \iff \alpha = \theta \,.$$

A realizer p for a function $f : C \times A \to B$ may use the realizer θ as an oracle. Thus a realizer for $\widetilde{f}(\dagger) \in W$ is a function with access to an oracle θ, so we cannot expect it to be computable in general.

Exercise 3.6 Calculate products and exponentials in $\mathsf{Rep}(K_1)$.

The representations of products and exponentials which we computed are isomorphic to the standard representations used in Type II computability, cf. [Wei00, 3.3.3 and 3.3.13]. The point we want to make here is that even if we never heard of pairing functions, the utm and the smn theorems, the category-theoretic definitions of products and exponentials would lead us to them.

Without going through further calculations in the same detail, we next state other basic constructions of representations. They are obtained by applying general category-theoretic definitions to $\mathsf{Rep}(K_2)$. To get used to the realizability notation, we shall sometimes use $\Vdash_A$ instead of δ_A.

3.2.3 Disjoint sums

We first fix notation for disjoint sums. Given sets S and T, let $S+T$ be their disjoint sum (disjoint union) and $i_0 : S \to S + T$ and $i_1 : S \to S + T$ the canonical inclusions. We may denote the elements of $S + T$ as $i_0(x)$ or $i_1(y)$, depending on whether they belong to the first or the second component. Thus

$$S + T = \{i_0(x) \;\big|\; x \in S\} \cup \{i_1(y) \;\big|\; y \in T\} \,.$$

It is assumed that the maps i_0 and i_1 are chosen so that their images never overlap. One common definition is $i_0(x) = \langle 0, x\rangle$ and $i_1(y) = \langle 1, y\rangle$. When no confusion may arise, we drop the inclusions i_0 and i_1, and pretend that $A \subseteq A + B$ and $B \subseteq A + B$. A common case when this is done is definition by cases of a function $f : A + B \to C$, which we write as

$$f(z) = \begin{cases} f_0(z) & \text{if } z \in A \,, \\ f_1(z) & \text{if } z \in B \,. \end{cases}$$

It would be more correct to write down two separate cases, one for $x \in A$ and one for $y \in B$, as

$$f(i_0(x)) = f_0(x) \quad \text{and} \quad f(i_0(y)) = f_1(y) \,.$$

The disjoint sum of representations $\mathbf{A}$ and $\mathbf{B}$ is the representation $\mathbf{A} + \mathbf{B} = (A+B, \Vdash +A + B)$ where $\Vdash_{A+B}$ is characterized by

$$\begin{aligned} \langle \alpha, \beta\rangle \Vdash_{A+B} i_0(x) &\iff \alpha(0) = 0 \wedge \beta \Vdash_A x \\ \langle \alpha, \beta\rangle \Vdash_{A+B} i_1(y) &\iff \alpha(0) = 1 \wedge \beta \Vdash_B y \,. \end{aligned}$$

The canonical inclusions $i_0 : A \to A + B$ and $i_1 : B \to A + B$ are realized by $\beta \mapsto \langle 0^\omega, \beta\rangle$ and $\beta \mapsto \langle 1^\omega, \beta\rangle$, respectively. If $f : A \to C$ and $g : B \to C$ are realized by p and q, respectively, there is a realized map $[f, g] : A + B \to C$, defined by

$$[f, g](z) = \begin{cases} f(z) & \text{if } z \in A, \\ g(z) & \text{if } z \in B \,. \end{cases}$$

It is realized by the map $\mathbb{B} \rightharpoonup \mathbb{B}$ defined by

$$\alpha \mapsto \texttt{if } (\mathsf{fst}\,\alpha)(0) = 0 \texttt{ then } p(\mathsf{fst}\,\alpha) \texttt{ else } q(\mathsf{snd}\,\alpha) \,.$$

A common disjoint sum is **one**+**one**. It is isomorphic to the representation **2** of *Booleans* $2 = \{0, 1\}$ with realizability relation

$$\alpha \Vdash_2 0 \iff \alpha(0) = 0 \quad \text{and} \quad \alpha \Vdash_2 1 \iff \alpha(1) = 1 \,.$$

3.2.4 Quotients

Let $\mathbf{A}$ be a representation and $\sim$ and equivalence relation on its underlying set A. The equivalence class of $x \in A$ is the set $[x]_\sim = [x] = \{y \in A \mid x \sim y\}$. The underlying set of the quotient representation $\mathbf{A}/\sim$ is the set of $\sim$-equivalence classes $A/\sim$ with the representation

$$\delta_{A/\sim}(\alpha) = [\delta_A(\alpha)]_\sim \,.$$

The canonical quotient map $[-]_\sim : A \to A/\sim$, defined by $x \mapsto [x]_\sim$, is realized by the identity map. For any realized map $f : A \to B$ which respects $\sim$, i.e., $x \sim y$ implies $f(x) = f(y)$, there is a unique factorization $\overline{f} : A/\sim \to B$ such that $\overline{f}([x]) = f(x)$. The factorization is tracked by at least those maps $\mathbb{B} \rightharpoonup \mathbb{B}$ that track f.

3.2.5 Subrepresentations

Recall that a *monomorphism*, or *mono* for short, is a morphism $m : A \rightarrowtail B$ that can be cancelled on the left: if $m \circ f = m \circ g$ then $f = g$. Monos $m : A \rightarrowtail B$ and $m' : A' \rightarrowtail B$ are isomorphic if there exists an isomorphism $h : A \to A'$ such that $m = m' \circ h$.

In $\mathsf{Rep}(K_2)$ a mono is an *injective* realized map $m : \mathbf{A} \rightarrowtail \mathbf{B}$, i.e., one that satisfies $m(x) = m(y) \implies x = y$. Note that a realizer of such a mono need not be injective.

Exercise 3.7 Prove what we just stated, namely, that $m : \mathbf{A} \to \mathbf{B}$ is mono in $\mathsf{Rep}(K_2)$ if, and only if, $m : A \to B$ is injective.

We may ask whether every mono is isomorphic to one that has a particularly nice form. The next definition and proposition give one possible answer.

Definition 3.8 A mono $i : \mathbf{A} \rightarrowtail \mathbf{B}$ is *canonical* if $A \subseteq B$, i is the subset inclusion, and i is tracked by fst.

Proposition 3.9 *Every mono is isomorphic to a canonical one.*

Proof. Suppose a mono $m : \mathbf{A} \rightarrowtail \mathbf{B}$ is tracked by p. To get a canonical mono $i : \mathbf{A}' \rightarrowtail \mathbf{B}$, let $A' = \mathsf{im}(m) = \{y \in B \mid \exists\, x \in A \,.\, y = m(x)\}$ be the image of m, and define $\Vdash_{A'}$ by

$$\langle \alpha, \beta \rangle \Vdash_{A'} y \iff \alpha \Vdash_B y \wedge \beta \Vdash_A m^{-1}(y) \,.$$

Let $i : A' \to B$ be the subset inclusion. It is tracked by fst so that $i : \mathbf{A}' \to \mathbf{B}$ is a canonical mono. The isomorphism $h : \mathbf{A} \to \mathbf{A}'$ between m and i is defined as $h(x) = m(x)$. It is tracked by the map $\alpha \mapsto \langle p(\alpha), \alpha \rangle$, and its inverse h^{-1} is tracked by $\beta \mapsto \mathsf{snd}\,\beta$. To see that all this really works, a few easy details need to be checked, which we leave as exercise. ∎

Proposition 3.9 allows us to replace a mono with a canonical one that is isomorphic to it. Moreover, by inspecting the above proof, we may improve the description of monos even further. Observe that $\Vdash_{A'}$ is defined so that whenever $\langle \alpha, \beta \rangle \Vdash_{A'} y$ we may replace α by any other $\alpha' \Vdash_B y$ to obtain a realizer $\langle \alpha', \beta \rangle \Vdash_{A'} y$. But this means that it does not matter which α we have in $\langle \alpha, \beta \rangle \Vdash_{A'} y$, as long as it realizes y. Thus we are lead to a simple description of monos, given in the following definition, which only involves realizers β and elements y. The α's are not needed, as they can recovered from δ_B.

Definition 3.10 A *realizability predicate*, or *predicate* for short, on a representation $\mathbf{A}$ is a relation $P \subseteq \mathbb{B} \times A$. The canonical mono $i_P : \{\mathbf{A}|P\} \rightarrowtail \mathbf{A}$ determined by P is given by

$$\{A|P\} = \{x \in A \mid \exists\, \alpha \in \mathbb{B} \,.\, \langle \alpha, x \rangle \in P\} \quad \text{and} \quad \langle \alpha, \beta \rangle \Vdash_{\{A|P\}} x \iff \alpha \Vdash_A x \wedge \langle \beta, x \rangle \in P \,. \quad (4)$$

A *realizability relation* R between representation $\mathbf{A}_1, \ldots, \mathbf{A}_n$ is a realizability predicate on the product $\mathbf{A}_1 \times \cdots \times \mathbf{A}_n$.

Proposition 3.11 *Every mono is isomorphic to one determined by a realizability predicate.*

Proof. A mono $m : \mathbf{A} \rightarrowtail \mathbf{B}$, tracked by p, determines a predicate P by

$$P = \{\langle \alpha, y\rangle \in \mathbb{B} \times B \;\big|\; \exists\, x \in A\,.\,(\alpha \Vdash_A x \wedge m(x) = y)\}\ .$$

Let $i_P : \{\mathbf{A}|P\} \rightarrowtail \mathbf{B}$ be the canonical mono determined by P, as in (4). Observe that $\{A|P\}$ is the image of m and that

$$\langle \alpha, \beta\rangle \Vdash_{\{A|P\}} y \iff \alpha \Vdash_B y \wedge \exists\, x \in A\,.\,(\beta \Vdash_A x \wedge m(x) = y)\ .$$

The isomorphism $h : A \to \{A|P\}$ between m and i_P is defined by $h(x) = m(x)$. It is tracked by the map $\alpha \mapsto \langle p(\alpha), \alpha\rangle$, and its inverse is tracked by snd, which is easily checked. ∎

A predicate $P \subseteq \mathbb{B} \times A$ on $\mathbf{A}$ tells us two things. Firstly, it tells us which elements $x \in A$ satisfy the predicate P, namely those belonging to $\{A|P\}$, defined in (4). Secondly, if $\langle \alpha, x\rangle \in P$, then α is the *reason* as to why x satisfies P. It may seem strange that a sequence of numbers α can be seen as a "reason" for anything, but do not forget that useful information may be encoded by it. For example, a sequence of prime numbers $\alpha = [p_0, q_0, p_1, q_1, p_2, q_2, \ldots]$ such that $p_k + q_k = 2k + 4$ for all $k \in \mathbb{N}$ would be a very good reason to believe in the truth of the famous Goldbach's conjecture, which states that every even number larger than two is the sum of two primes. It is not important how we could get such a sequence, or even how we could check that it had the desired property. The point is that it is *possible* to encode a reason for the truth of Goldbach's conjecture with a sequence of numbers.

There is a customary notation for realizability predicates, which we shall use extensively. We read $\langle \alpha, x\rangle \in P$ as "x has property P because of α", or "α realizes the fact that x has property P", so we write $\langle \alpha, x\rangle \in P$ more intuitively as

$$\alpha \Vdash P(x)\ ,$$

and read it as "α realizes $P(x)$". More generally, for a realizability relation R between representations $\mathbf{A}_1, \ldots, \mathbf{A}_n$ we write

$$\alpha \Vdash R(x_1, \ldots, x_n)$$

instead of $\langle \alpha, x_1, \ldots, x_n\rangle \in R$. Because it is not clear from this notation what the domain of R is, we may also write more precisely

$$x_1 : \mathbf{A}_1, \ldots, x_n : \mathbf{A}_n \;\big|\; \alpha \Vdash R(x_1, \ldots, x_n)\ .$$

The sequence $x_1 : \mathbf{A}_1, \ldots, x_n : \mathbf{A}_n$ is called the *context*. It makes it clear that R is a relation between representations $\mathbf{A}_1, \ldots, \mathbf{A}_n$ and that $x_1, \ldots, x_n$ are elements of $A_1, \ldots, A_n$, respectively.

We say that realizability predicates P and Q are *equivalent*, written $P \equiv Q$, when they represent isomorphic monos.

Proposition 3.12 *Realizability predicates P and Q on $\mathbf{A}$ are equivalent if, and only if, there exist computable partial maps $p : \mathbb{B} \rightharpoonup \mathbb{B}$ and $q : \mathbb{B} \rightharpoonup \mathbb{B}$ such that, for all $\alpha, \beta \in \mathbb{B}$, $x \in A$,*

$$\alpha \Vdash_A x \wedge \beta \Vdash P(x) \implies p\langle \alpha, \beta\rangle \Vdash Q(x)$$

and

$$\alpha \Vdash_A x \wedge \beta \Vdash Q(x) \implies q\langle \alpha, \beta\rangle \Vdash P(x)\ .$$

Proof. Suppose P and Q are equivalent. Then there exists an isomorphism $h : \{\mathbf{A}|P\} \to \{\mathbf{A}|Q\}$ between $i_P : \{\mathbf{A}|P\} \rightarrowtail A$ and $i_Q : \{\mathbf{A}|Q\} \rightarrowtail \mathbf{A}$, and because i_P and i_Q are subset inclusions, this isomorphism is the identity map, $h(x) = x$. Let s be a realizer for h and t a realizer for h^{-1}. Define $p(\alpha) = \mathsf{snd}\, s(\alpha)$ and $q(\alpha) = \mathsf{snd}\, t(\alpha)$. If $\alpha \Vdash_A x$ and $\beta \Vdash P(x)$ then $\langle \alpha, \beta\rangle \Vdash_{\{A|P\}} x$, hence $s\langle \alpha, \beta\rangle \Vdash_{\{A|Q\}} x$ and $p\langle \alpha, \beta\rangle \Vdash Q(x)$, as required. By a symmetric argument, $\alpha \Vdash_A x$ and $\beta \Vdash Q(x)$ implies $q\langle \alpha, \beta\rangle \Vdash P(x)$.

Conversely, suppose there are p and q as in the proposition. For any $x \in A$, with $\alpha \Vdash_A x$, if $x \in \{A|P\}$ then there is $\beta \in \mathbb{B}$ such that $\beta \Vdash P(x)$, hence $p\langle \alpha, \beta\rangle \Vdash Q(x)$ and $x \in \{A|Q\}$, too. This

shows $\{A|P\} \subseteq \{A|Q\}$, and $\{A|Q\} \subseteq \{A|P\}$ is proved similarly. To see that $i_P : \{\mathbf{A}|P\} \rightarrowtail \mathbf{A}$ and $i_Q : \{\mathbf{A}|Q\} \rightarrowtail \mathbf{A}$ are isomorphic, we only need to show that the identity maps $\{A \mid P\} \to \{A|Q\}$ and $\{A|Q\} \to \{A|P\}$ are realized, which they are by $\alpha \mapsto \langle \alpha, p(\alpha)\rangle$ and $\beta \mapsto \langle \beta, q(\beta)\rangle$, respectively. ∎

We say that a predicate P on $\mathbf{A}$ has *irrelevant realizers* if $\alpha \Vdash P(x)$ implies $\beta \Vdash P(x)$ for all $\beta \in \mathbb{B}$. Such realizers are indeed of no help as we cannot compute anything from them. We say that a predicate is *without computational content* when it is equivalent to one with irrelevant realizers.

Exercise 3.13 Let $\mathbf{A}$ be a representation. Every subset $S \subseteq A$ determines a realizability predicate P_S with irrelevant realizers, namely the one defined by

$$\alpha \Vdash P_S(x) \iff x \in S\ .$$

Prove that every predicate on $\mathbf{A}$ without computational content is isomorphic to one of this form. Thus, the realizability predicates on $\mathbf{A}$ without computational content are, up to isomorphism, subsets of A.

4 Realizability Logic

In this section we study further properties of subobjects and realizability predicates. Then we give rules for interpreting intuitionistic logic by realizability predicates.

4.1 The Poset of Subobjects

Consider monos $m : A \rightarrowtail C$ and $n : B \rightarrowtail C$. We say that m is below n, and write $m \leq n$, if m factors through m, i.e., if there exists a morphism $k : A \to B$ such that $m = n \circ k$. It is easily shown (exercise) that k is mono and unique, if it exists. The relation $\leq$ is reflexive and transitive, but it is not in general antisymmetric. In fact, $m \leq n$ and $n \leq m$ holds precisely when m and n are isomorphic. Thus, the preorder $\leq$ on monos induces a partial order $\leq$ on the set of equivalence classes of isomorphic monos, which are called *subobjects*.

Since realizability predicates represent monos, we may define a preorder on them, too. If P and Q are realizability predicates on $\mathbf{A}$, we let $P \leq Q$ when $i_P \leq i_Q$.

Proposition 4.1 *Let P and Q be realizability predicates on $\mathbf{A}$. Then $P \leq Q$ if, and only if, there exists a computable $p : \mathbb{B} \rightharpoonup \mathbb{B}$ such that, for all $\alpha, \beta \in \mathbb{B}$, $x \in A$,*

$$\alpha \Vdash_A x \wedge \beta \Vdash P(x) \implies p\langle\alpha, \beta\rangle \Vdash Q(x) \ .$$

Proof. This is half of Proposition 3.12. Half of its proof is the proof of this proposition. ∎

Our next task is to give an interpretation of (intuitionistic) first-order logic in $\mathsf{Rep}(K_2)$. We follow the standard techniques of categorical logic and interpret logical predicates as subobjects, or rather as realizability predicates that represent subobjects. The basic logical operations are interpreted as operations on the poset of subobject, or rather on the preorder of realizability predicates. Specifically, for realizability predicates P and Q on $\mathbf{A}$ we have:

- truth $\top$ is the greatest subobject of $\mathbf{A}$,
- falsehood $\bot$ is the least subobject of $\mathbf{A}$,
- conjunction $P \wedge Q$ is the greatest lower bound of P and Q,
- disjunction $P \vee Q$ is the least upper bound of P and Q,
- implication $P \Rightarrow Q$ is the greatest predicate S such that $P \wedge S \leq Q$,
- negation $\neg P$ is defined as $P \Rightarrow \bot$,

For a realizability relation R on $\mathbf{A} \times \mathbf{B}$, we interpret quantifiers as follows:

- existential quantification $\exists_A R$ is the least predicate on $\mathbf{B}$ through which the canonical projection $\mathbf{A} \times \mathbf{B} \to \mathbf{B}$ factors,
- universal quantification $\forall_A R$ is the largest predicate on $\mathbf{B}$ that factors through the canonical projection $\mathbf{A} \times \mathbf{B} \to \mathbf{B}$.

It would take too much time and effort to explain the precise meaning and motivation for the above interpretation. The interested reader may consult a standard text in categorical logic [MM92, McL95, Joh02]. We shall content ourselves with the resulting rules for computing logical operations in $\mathsf{Rep}(K_2)$.

4.2 The Realizability Interpretation

Our language is a multi-sorted first-order logic. It consists of *sorts*, which in our case are representations, *predicates* and *relations* on sorts, which to us are realizability predicates and relations. It is simpler to work with predicates than with relations, so below we give rules for predicates only. To obtain rules for general relations, just remember that relations are predicates on products and apply the rules accordingly.

4.2.1 Constructions of Sorts

Since sorts are interpreted as representations, we may use the constructions from §3.2 to define new sorts from old ones, for example products $\mathbf{A} \times \mathbf{B}$, disjoint sums $\mathbf{A} + \mathbf{B}$ and exponentials $\mathbf{B}^{\mathbf{A}}$.

If P is a realizability predicate on $\mathbf{A}$, the subrepresentation corresponding to P is $\{\mathbf{A}|P\}$, defined in (4), also written as $\{x \in A \mid P(x)\}$. It is interesting to look at the realizers for the elements of a subrepresentation. A realizer for $x \in \{A|P\}$ is a pair $\langle \alpha, \beta \rangle$ where $\alpha \Vdash_A x$ and $\beta \Vdash P(x)$. Thus, to know that an element belongs to a subrepresentation is to know not only a realizer for the element, but also a reason for its being in the subrepresentation. In particular, it is possible to have a predicate P on $\mathbf{A}$ such that $\{A|P\} = A$, that is for every $x \in A$ there is $\beta \in \mathbb{B}$ such that $\beta \Vdash P(x)$, yet $\mathbf{A}$ and $\{\mathbf{A}|P\}$ are not isomorphic, because the realizers of $\{\mathbf{A}|P\}$ carry more information that those of $\mathbf{A}$.

We need to be careful with quotients, since we do not want to mix up ordinary equivalence relations and realizability equivalence relations. Suppose R is a realizability relation on $\mathbf{A} \times \mathbf{A}$. It determines an ordinary relation $R' \subseteq A \times A$ by $R' = \{\langle x, y \rangle \in A \times A \mid \exists\, \alpha \in \mathbb{B}\,.\, \langle \alpha, x, y \rangle \in R\}$. We define $\mathbf{A}/R$ to be the quotient of $\mathbf{A}$ by the least equivalence relation containing R'.

4.2.2 Truth and Falsehood

The constant $\top$ signifies truth, and is interpreted as the predicate which is always realized, i.e., $x : A \mid \alpha \Vdash \top$ holds for all $x \in A$ and all $\alpha \in \mathbb{B}$.

The constant $\bot$ signifies falsehood, and it is interpreted as the predicate which is never realized, i.e., $x : A \mid \alpha \Vdash \bot$ holds for no $x \in A$ and $\alpha \in \mathbb{B}$.

Exercise 4.2 Show that, for any realizability predicate P on $\mathbf{A}$, $\bot \leq P \leq \top$.

4.2.3 Conjunction

Let P and Q be predicates on $\mathbf{A}$. Their conjunction $P \wedge Q$ is the predicate on $\mathbf{A}$, given by

$$\langle \alpha, \beta \rangle \Vdash P(x) \wedge Q(x) \text{ iff } \alpha \Vdash P(x) \text{ and } \beta \Vdash Q(x).$$

Exercise 4.3 Prove that $P \wedge Q$ is the greatest lower bound of P and Q, i.e., that $P \wedge Q \leq P$, $P \wedge Q \leq Q$, and that whenever $R \leq P$ and $R \leq Q$ then $R \leq P \wedge Q$.

4.2.4 Disjunction

Let P and Q be predicates on $\mathbf{A}$. Their disjunction $P \vee Q$ is the predicate on $\mathbf{A}$, given by

$$\langle \alpha, \beta \rangle \Vdash P(x) \vee Q(x) \text{ iff } (\alpha(0) = 0 \text{ and } \beta \Vdash P(x)) \text{ or } (\alpha(0) = 1 \text{ and } \beta \Vdash Q(x)).$$

Notice that a realizer for a disjunction not only provides a realizer for one of the disjuncts, but also explicit information as to which disjunct the realizer is for.

Exercise 4.4 Prove that $P \vee Q$ is the least upper bound of P and Q, i.e., that $P \leq P \vee Q$, $Q \leq P \vee Q$, and that whenever $P \leq R$ and $Q \leq R$ then $P \vee Q \leq R$.

4.2.5 Implication

Let P and Q be predicates on $\mathbf{A}$. The implication $P \Rightarrow Q$ is the predicate on $\mathbf{A}$, given by

$$\alpha \Vdash P(x) \Rightarrow Q(x) \text{ iff whenever } \beta \Vdash_A x \text{ and } \gamma \Vdash P(x) \text{ then } \boldsymbol{\eta}_\alpha \langle \beta, \gamma \rangle \Vdash Q(x).$$

Thus, a realizer for $P(x) \Rightarrow Q(x)$ is (the name of) a program which transforms realizers for x and $P(x)$ to realizers for $Q(x)$.

Exercise 4.5 Prove that $P \Rightarrow Q$ is the largest predicate R such that $P \wedge R \leq Q$, i.e., that $P \wedge (P \Rightarrow Q) \leq Q$ and that whenever $P \wedge R \leq Q$ then $R \leq (P \Rightarrow Q)$.

4.2.6 Negation

Let P be a predicate on $\mathbf{A}$. Negation $\neg P(x)$ is defined as $P(x) \Rightarrow \bot$, so we may simply compute its interpretation. Suppose $\beta \Vdash_A x$. If $\alpha \Vdash P(x) \Rightarrow \bot$ then, for all $\gamma \Vdash P(x)$, $\boldsymbol{\eta}_\alpha\langle\beta,\gamma\rangle \Vdash \bot$. Since $\bot$ is never realized, this means that $\gamma \Vdash P(x)$ cannot happen. So $\neg P(x)$ is realized only if $P(x)$ is not. And if $P(x)$ is not realized, then any α is a realizer for $\neg P(x)$. So we have

$$\alpha \Vdash \neg P(x) \text{ iff for all } \beta \in \mathbb{B}, \text{ not } \beta \Vdash P(x).$$

Later on we shall need to know the realizers of a doubly negated predicate. From the above we may compute

$$\alpha \Vdash \neg\neg P(x) \text{ iff for some } \beta \in \mathbb{B},\ \beta \Vdash P(x).$$

Thus, *double negation destroys the computational content* of a predicate, since it replaces computationally meaningful realizers with irrelevant ones.

4.2.7 Existential quantification

Let R be a relation on $\mathbf{A} \times \mathbf{B}$. By quantifying existentially over A we may form the predicate $\exists_A R$ on $\mathbf{B}$. If we think of R as a two-place relation $R(x,y)$, we may write $\exists\, x \in A\,.\, R(x,y)$ instead of $\exists_A R$. The existential quantification is characterized by

$$\langle\alpha,\beta\rangle \Vdash \exists\, x \in A\,.\, R(x,y) \text{ iff there exists } x_0 \in A \text{ such that } \alpha \Vdash_A x_0 \text{ and } \beta \Vdash R(x_0,y).$$

The realizer $\langle\alpha,\beta\rangle$ consists of a realizer α for some $x_0 \in A$ and a realizer β which realizes $R(x_0,y)$.

4.2.8 Universal quantification

Let R be a relation on $\mathbf{A} \times \mathbf{B}$. By quantifying universally over A we may form the predicate $\forall_A R$ on $\mathbf{B}$. If we think of R as a two-place relation $R(x,y)$, we may write $\forall\, x \in A\,.\, R(x,y)$ instead of $\forall_A R$. The universal quantification is characterized by

$$\alpha \Vdash \forall\, x \in A\,.\, R(x,y) \text{ iff whenever } \beta \Vdash_A x \text{ then } \boldsymbol{\eta}_\alpha(\beta) \Vdash R(x,y).$$

4.2.9 Equality

Equality on a representation $\mathbf{A}$ is the diagonal map $\mathbf{A} \rightarrowtail \mathbf{A} \times \mathbf{A}$, $x \mapsto \langle x,x\rangle$. Up to isomorphism, it is represented by the realizability relation $=_A$ on $\mathbf{A} \times \mathbf{A}$, defined by

$$\alpha \Vdash x =_A y \text{ iff } x = y.$$

Exercise 4.6 Prove that the diagonal map $\mathbf{A} \rightarrowtail \mathbf{A} \times \mathbf{A}$ is indeed isomorphic to the mono determined by $=_A$.

It is evident that the realizer α is irrelevant, so *equality has no computational content.* (This is a property of the category of representations, which does not hold in realizability toposes, which are generalizations of categories of representations.)

We usually omit the subscript in $=_A$, especially when the representation $\mathbf{A}$ is clear from the context. But it should be remembered that every representation has its own equality relation.

4.3 Basic Properties of the Realizability Interpretation

We say that a realizability predicate P on $\mathbf{A}$ is *true* or *valid*, if it is equivalent to truth $\top$. Since $P \leq \top$ anyhow, validity amounts to $\top \leq P$, which is equivalent to existence of a certain computable map:

Definition 4.7 A predicate P on $\mathbf{A}$ is *true* or *valid*, written $\models P$, when there exists a computable partial map $p : \mathbb{B} \rightharpoonup \mathbb{B}$ such that if $\alpha \Vdash_A x$ then $p(\alpha) \Vdash P(x)$. We say that p *validates*, or *realizes* P.

A predicate P on the singleton representation $\mathbf{1}$ is called a *sentence*. It is valid if, and only if, it is realized.

Exercise 4.8 Show that up to equivalence there are two sentences in $\mathsf{Rep}(K_2^{\mathrm{cont}})$. Then prove that in $\mathsf{Rep}(K_2)$ sentences, quotiented by equivalence, are the *Medvedev degrees* from computability theory.

Proposition 4.9 *For realizability predicates P and Q on $\mathbf{A}$:*

1. *$\models P$ if, and only if, there exists a realizer for $\forall\, x \in A\,.\, P(x)$,*
2. *$P \leq Q$ if, and only if, $\models P \Rightarrow Q$.*
3. *P and Q are equivalent if, and only if $\models (P \Rightarrow Q) \wedge (Q \Rightarrow P)$.*

Proof. If p validates P, then any α such that $\boldsymbol{\eta}_\alpha = p$ realizes $\forall\, x \in A\,.\, P(x)$. Conversely, if $\alpha \Vdash \forall\, x \in A\,.\, P(x)$ then $\boldsymbol{\eta}_\alpha$ validates P. The second claim is equally easy, while the third one follows directly from the second one. ∎

We may build up realizability predicates using the logical connectives and quantifiers of first-order logic. It is natural to ask about the relationship between provability of such predicates in first-order logic and their validity in the realizability interpretation.

Proposition 4.10 *Intuitionistic first-order logic is sound for the realizability interpretation, which means: if intuitionistic logic proves that Q follows from P, then $P \leq Q$. In particular, if intuitionistic logic proves P, then P is valid.*

Proof. We omit the proof. One checks that intuitionistic rules of inference preserve validity, from which it follows that that provable predicates are valid. ∎

Normally, validity of a predicate is important but the map validating is of little interest. The practical value of Proposition 4.10 is that it allows us to establish validity of a predicate by proving it in intuitionistic logic, thus avoiding completely the construction of its validating map.

4.3.1 Realizability Logic is not Classical

The interpretation of logic given above validates the *intuitionistic* rules of reasoning. One may wonder why we insist on intuitionistic logic. It would certainly be more convenient to use classical logic, which most mathematicians are already familiar with. But which logic we use is not a matter of choice, convenience, or philosophical conviction. The logic of $\mathsf{Rep}(K_2)$ is a property of $\mathsf{Rep}(K_2)$, and not an extra thing that we add to it. In fact, following the standard techniques of categorical logic, we have *computed* its rules. So far the rules have turned out to support intuitionistic logic. We may also compute whether the logic of $\mathsf{Rep}(K_2)$ happens to be classical.

The difference between classical and intuitionistic logic is in the rule known as Excluded Middle, which states that, for every predicate P,

$$P \vee \neg P\ .$$

Does this rule hold in the realizability relation? Let P be a realizability predicate on $\mathbf{A}$. Suppose $\models P \vee \neg P$. Then there exists a computable $p : \mathbb{B} \rightharpoonup \mathbb{B}$ such that, for all $\alpha \Vdash x$, $p(\alpha) \Vdash P(x) \vee \neg P(x)$. By definition of disjunction, this means that

$$\mathsf{fst}\,(p(\alpha))(0) = 0 \text{ and } \mathsf{snd}\,(p(\alpha)) \Vdash P(x),$$

or

$$\mathsf{snd}\,(p(\alpha))(0) = 1 \text{ and } \mathsf{snd}\,(p(\alpha)) \Vdash \neg P(x).$$

Define $q : \mathbb{B} \rightharpoonup \{0,1\}$ by $q(\alpha) = \mathsf{fst}\,(p(\alpha))(0)$, which is obviously a computable map. For all $\alpha \Vdash_A x$,

$$q(\alpha) = \begin{cases} 1 & \text{if } \models P(x), \\ 0 & \text{if } \models \neg P(x). \end{cases}$$

We have the following conclusion: if a realizability predicate P on $\mathbf{A}$ satisfies the Excluded Middle, then there is a *computable* decision procedure q, which tells us for any given $\alpha \in |A|$ whether $P(\delta_A(\alpha))$ holds.

It is not hard to think of an example which does not have such a computable decision procedure. To make things simple, let us take $\mathbf{A} = (\mathbb{B}, \mathrm{id}_{\mathbb{B}})$ and let $S \subseteq \mathbb{B}$ be a subset which is neither closed nor open. Define the realizability predicate P by

$$\alpha \Vdash P(\beta) \iff \beta \in S\,.$$

We claim that $\not\models P \vee \neg P$. Suppose on the contrary that $\models P \vee \neg P$. By the argument above, there exists a computable map $q : \mathbb{B} \to \{0,1\}$ such that

$$q(\alpha) = \begin{cases} 1 & \text{if } \alpha \in S, \\ 0 & \text{if } \alpha \notin S. \end{cases} \tag{5}$$

Since q is computable it is also continuous, hence $q^{-1}(1) = S$ should be an open and closed subset of $\mathbb{B}$, which it is not. Therefore $\not\models P \vee \neg P$. We have *proved* that Excluded Middle fails in $\mathsf{Rep}(K_2)$. The same argument shows that Excluded Middle fails in $\mathsf{Rep}(K_2^{\mathrm{cont}})$ as well, as there is not even a continuous q satisfying (5).

Since Excluded Middle does not hold for all predicates, it makes sense to give a name to those for which it does.

Definition 4.11 A realizability predicate P on $\mathbf{A}$ is *decidable* if $\models P \vee \neg P$.

4.4 First Uses of Realizability Logic

4.4.1 Computability for Free

It is often said that in intuitionistic logic an existential statement $\exists\, x \in A\,.\, P(x)$ must be proved by an explicit construction of an x_0 satisfying P, whereas in classical logic one may prove $\exists\, x \in A\,.\, P(x)$ indirectly by proving $\neg \forall\, x \in A\,.\, \neg P(x)$. Realizability offers one way of understanding what this means precisely.

Proposition 4.12 *Let P be a predicate on $\mathbf{A}$ and suppose $\models \exists\, x \in A\,.\, P(x)$. Then there is $x_0 \in A$ and a computable $\alpha \in \mathbb{B}$ such that $\alpha \Vdash_A x_0$ and $\models P(x_0)$.*

Proof. Since $\models \exists\, x \in A\,.\, P(x)$ there exists a computable β such that $\beta \Vdash \exists\, x \in A\,.\, P(x)$. Then $\alpha = \mathsf{fst}\,\beta$ is also computable and there is $x_0 \in A$ such that $\alpha \Vdash_A x_0$ and $\mathsf{snd}\,\beta \Vdash P(x_0)$, hence $\models P(x_0)$. ∎

Corollary 4.13 *If intuitionistic logic proves $\exists\, x \in A\,.\, P(x)$ then there exists a computable $x_0 \in A$ such that $P(x_0)$ is valid.*

Proof. This follows directly from Propositions 4.10 and 4.12. ∎

It should be noted that intuitionistic logic does *not* assume that everything is computable. In fact, there are various models for it which have nothing to do with computability, such as sheaves on a topological space. So by Corollary 4.13 we really do get computability for free from an intuitionistic proof.

A common application of Corollary 4.13 is as follows. Suppose we want to show the existence of a computably realized map $f : \mathbf{A} \to \mathbf{B}$ satisfying a certain condition. If we can express this condition as a realizability predicate P, then it suffices to prove intuitionistically the statement $\exists\, f \in B^A \,.\, P(f)$. In §4.4.4 we will see how a similar trick allows us to infer the existence of computably realized multi-valued maps from intuitionistic proofs.

4.4.2 Classical Predicates

In §4.2.6 we saw that double negation destroys computational content of a realizability predicate. Let us look at the relationship between P and its double negation $\neg\neg P$.

Proposition 4.14 *For any predicate P on $\mathbf{A}$, $\models P(x) \Rightarrow \neg\neg P(x)$.*

Proof. First proof: we look for a computable $p : \mathbb{B} \rightharpoonup \mathbb{B}$, such that if $\alpha \Vdash x$ then $p(\alpha) \Vdash P(x) \Rightarrow \neg\neg P(x)$. There is a computable $\beta \in \mathbb{B}$ such that $\boldsymbol{\eta}_\beta(\gamma) = 0^\omega$ for all $\gamma \in \mathbb{B}$. Define $p(\alpha) = \beta$. Now if $\alpha \Vdash_A x$ and $\gamma \Vdash P(x)$ then $\boldsymbol{\eta}_{p(\alpha)}\langle\alpha, \gamma\rangle = \boldsymbol{\eta}_\beta\langle\alpha, \gamma\rangle = 0^\omega \Vdash \neg\neg P(x)$, as required.

Second proof: we prove $P(x) \Rightarrow \neg\neg P(x)$ intuitionistically. Recalling the definition of negation, we need to show $P(x) \Rightarrow (P(x) \implies \bot) \implies \bot$. But this is trivial: if $P(x)$ and $P(x) \Rightarrow \bot$ then $\bot$. ∎

It cannot be the case that $\models \neg\neg P \Rightarrow P$ for all predicates P because that would imply Excluded Middle.

Definition 4.15 A realizability predicate P is *classical* if $\neg\neg P \equiv P$.

Exercise 4.16 Show that a decidable predicate is classical. This is best done by proving $(P \vee \neg P) \Rightarrow \neg\neg P \Rightarrow P$ in intuitionistic logic.

Proposition 4.17 *A realizability predicate is classical if, and only if, it is without computational content.*

Proof. If P and $\neg\neg P$ are equivalent then P is without computational content since $\neg\neg P$ has irrelevant realizers.

Conversely, suppose P is without computational content. Then it is equivalent to a predicate Q with irrelevant realizers. But then $\neg\neg Q$ is the same predicate as Q, so we have $P \equiv Q = \neg\neg Q \equiv \neg\neg P$. ∎

Classical realizability predicates are particularly easy to understand because realizers do not play a role in their meaning. Thus we may read a classical realizability predicate as if it were and ordinary predicate. There is a family of classical predicates which may easily be recognized from their form.

Definition 4.18 A predicate is *negative* if it is built from basic classical predicates, equality, $\bot$, $\top$, $\wedge$, $\Rightarrow$, $\neg$, and $\forall$.

Proposition 4.19 *A negative predicate is classical.*

Proof. We prove the proposition by induction on the structure of a negative predicate P. For each case we prove intuitionistically that $\neg\neg P \Rightarrow P$.

We already know that equality is classical, as it is without computational content, and clearly $\bot$ and $\top$ are classical, too.

Suppose P is $P_1 \wedge P_2$ where P_1 and P_2 are classical. Assume $\neg\neg(P_1 \wedge P_2)$. If $\neg P_1$ then also $\neg(P_1 \wedge P_2)$, which contradicts our assumption, hence $\neg\neg P_1$, and because P_1 is classical, P_1 holds. An analogous argument establishes P_2. Therefore $P_1 \wedge P_2$.

Suppose P is $P_1 \Rightarrow P_2$ where P_1 and P_2 are classical. Assume $\neg\neg(P_1 \Rightarrow P_2)$. Assume P_1. If $\neg P_2$ then $\neg(P_1 \Rightarrow P_2)$, which contradicts our assumption. Therefore $\neg\neg P_2$ and since P_2 is classical, P_2 holds. Therefore $P_1 \Rightarrow P_2$.

Suppose P is $\neg P_1$ where P_1 is classical. Assume $\neg\neg\neg P_1$. Since $\neg\neg P_1$ is equivalent to P_1, we obtain $\neg P_1$.

Suppose P is $\forall\, x \in A\,.\, P_1(x)$, where P_1 is classical. Assume $\neg\neg\forall\, x \in A\,.\, P_1(x)$. Consider an arbitrary $x \in A$. If we had $\neg P_1(x)$ then it would follow that $\neg\forall\, x \in A\,.\, P_1(x)$, which contradicts our assumption. Therefore $\neg\neg P_1(x)$ holds and since P_1 is classical, $P_1(x)$ holds. We may conclude $\forall\, x \in A\,.\, P_1(x)$. ∎

Exercise 4.20 If you think that the above proof is ugly, try writing down the realizers instead.

When P is a classical predicate on $\mathbf{A}$, the subrepresentation $\{\mathbf{A}|P\}$ has a particularly simple form. Recall that a realizer for $x \in \{A|P\}$ is a pair $\langle \alpha, \beta \rangle$ such that $\alpha \Vdash_A x$ and $\beta \Vdash P(x)$. Since P is classical, we might as well drop the irrelevant realizer β.

Proposition 4.21 *If P is a classical predicate on $\mathbf{A}$, then up to isomorphism $\{\mathbf{A}|P\}$ is the representation*

$$\{A|P\} = \{x \in A \;\big|\; \models P(x)\} \qquad \alpha \Vdash_{\{A|P\}} x \iff \alpha \Vdash_A x\,. \tag{6}$$

Proof. Without loss of generality, we may assume that P has irrelevant realizers. Representations defined by (4) and (6) have the same underlying set $\{A|P\}$. To see that they are isomorphic, we only need to show that the identity map $\{A|P\} \to \{A|P\}$ is realizes as a morphism between the two representations. From (4) to (6) it is realized by fst, and in the other direction by $\alpha \mapsto \langle \alpha, 0^\omega \rangle$. ∎

In $\mathsf{Rep}(K_2)$ there are predicates which are classical even though they are not negative. One such important example is Markov Principle.

Proposition 4.22 (Markov Principle) *Suppose P is a decidable predicate on $\mathbf{N}$. Then the predicate $\exists\, n \in N\,.\, P(n)$ is a classical.*

Proof. We give an informal proof. If P is decidable there is a computable $q : \mathbb{N} \rightharpoonup \{0,1\}$ such that $q(n) = 1 \iff \models P(n)$. If $\models \neg\neg\exists\, n \in N\,.\, P(n)$ then we may compute $m \in \mathbb{N}$ such that $\models P(m)$ as $m = \min_{k \in \mathbb{N}}(q(k) = 1)$. This m is well defined because q is not constantly zero because $\models \neg\neg\exists\, n \in N\,.\, P(n)$. ∎

Exercise 4.23 Show that a *parameterized* Markov Principle holds in $\mathsf{Rep}(K_2)$: if R is a relation on $\mathbf{A} \times \mathbf{N}$ and $\models \forall\, x \in A\,.\, (R(x,n) \vee \neg R(x,n))$ then $\exists\, n \in N\,.\, R(x,n)$ is a classical predicate on $\mathbf{A}$.

4.4.3 Monos and Epis

Realizability logic is only useful if we can express interesting properties of representations and morphisms in it. This is indeed the case, and we start with simple examples.

Proposition 4.24 *A morphism $f : \mathbf{A} \to \mathbf{B}$ is mono if, and only if,*

$$\models \forall\, x, y \in A\,.\, (f(x) =_B f(y) \Rightarrow x =_A y)\;. \tag{7}$$

Proof. Observe that (7) is a negative predicate. Therefore its meaning in the realizability interpretation is the same as its ordinary meaning. But the ordinary meaning is that $f : A \to B$ is injective, i.e., that $f : \mathbf{A} \to \mathbf{B}$ is mono.

Let us also write down a more detailed proof. First of all, $\forall\, x, y \in A\,.\,\cdots$ is an abbreviation for $\forall\, x \in A\,.\,\forall\, y \in A\,.\,\cdots$. A realizer for (7) is a computable $\theta \in \mathbb{B}$ such that, whenever $\alpha \Vdash_A x$ and $\beta \Vdash_A y$, then $\boldsymbol{\eta}_{\boldsymbol{\eta}_\theta(\alpha)}(\beta) \Vdash f(x) =_B f(y) \Rightarrow x =_A y$. Since $=_B$ has trivial realizers, this means that for all $\gamma \in \mathbb{B}$, if $f(x) = f(y)$ then $\boldsymbol{\eta}_{\boldsymbol{\eta}_{\boldsymbol{\eta}_\theta(\alpha)}(\beta)}\langle\langle \alpha, \beta\rangle, \gamma\rangle \Vdash x =_A y$. Now if such a θ exists, then $f(x) = f(y)$ implies that $x =_A y$ is realized by $\boldsymbol{\eta}_{\boldsymbol{\eta}_{\boldsymbol{\eta}_\theta(\alpha)}(\beta)}\langle\langle \alpha, \beta\rangle, \gamma\rangle$, therefore $x = y$. So f is indeed injective. Conversely, if f is injective, then there is a computable θ such that $\boldsymbol{\eta}_{\boldsymbol{\eta}_{\boldsymbol{\eta}_\theta(\alpha)}(\beta)}\langle\langle \alpha, \beta\rangle, \gamma\rangle = 0^\omega$ for all α, β, γ. This θ realizes (7). ∎

We have not said anything yet about epimorphisms. Recall that $e : A \to B$ is an *epimorphism*, or *epi*, if it can be cancelled on the right: $f \circ e = g \circ e \implies f = g$. In $\mathsf{Rep}(K_2)$, a realized map $e : \mathbf{A} \to \mathbf{B}$ is epi if, and only if, $e : A \to B$ is surjective.

Exercise 4.25 Prove that in $\mathsf{Rep}(K_2)$ a morphism is epi if, and only if, its underlying map is surjective.

How can we express the fact that a realized map $f : \mathbf{A} \to \mathbf{B}$ is surjective with a realizability predicate? Classically, surjectivity of f is the statement

$$\forall\, y \in B\,.\,\exists\, x \in A\,.\, f(x) = y\,. \tag{8}$$

Let us see what the same statement means in realizability logic. We should suspect that it means something else because (8) is not a negative formula as it contains an existential quantifier. A realizer for (8) is a computable $\theta \in \mathbb{B}$ such that if $\beta \Vdash_B y$ then $\boldsymbol{\eta}_\theta(\beta) \Vdash \exists\, x \in A\,.\, f(x) =_B y$, which means that there is $x_0 \in A$ such that

$$\mathsf{fst}\,(\boldsymbol{\eta}_\theta(\beta)) \Vdash_A x_0 \quad \text{and} \quad \mathsf{snd}\,(\boldsymbol{\eta}_\theta(\beta)) \Vdash f(x_0) =_B y\,.$$

Since $=_B$ has irrelevant realizers, we may safely ignore $\mathsf{snd}\,(\boldsymbol{\eta}_\theta(\beta))$, except for the fact that it tells us that $f(x_0) = y$. However, $\mathsf{fst}\,(\boldsymbol{\eta}_\theta(\beta))$ is very interesting. The map $p(\beta) = \mathsf{fst}\,(\boldsymbol{\eta}_\theta(\beta))$ is computable and, given $\beta \Vdash_B y$, it computes a realizer $p(\beta)$ for some $x_0 \in A$ such that $f(x_0) = y$. Does this mean that p realizes a morphism $g : \mathbf{B} \to \mathbf{A}$ such that $f(g(y)) = y$? If that were the case then $\beta \Vdash_B y$ and $\beta' \Vdash_B y$ would imply $p(\beta) \Vdash_A g(y)$ and $p(\beta') \Vdash_A g(y)$, but there is nothing to force p to map two realizers β, β' for y to two realizers $p(\beta)$, $p(\beta')$ for the same element in A. However, p does realizes a *multi-valued map* [Wei00, 3.1.3] $g : A \rightrightarrows B$ such that $f(g(y)) = y$ for all $y \in B$. We have proved:

Proposition 4.26 *For a morphism $f : \mathbf{A} \to \mathbf{B}$,*

$$\models \forall\, y \in B\,.\,\exists\, x \in A\,.\, f(x) =_B y \tag{9}$$

is valid if, and only if, f has a computable multi-valued right inverse.

Exercise 4.27 If you are familiar with category theory, prove that (9) is valid if, and only if, f is a regular epi.

We have found a way to express in realizability logic the notion of an epi which has a computable multi-valued right inverse, but we still do not know how to write down a predicate which says that $f : \mathbf{A} \to \mathbf{B}$ is epi. A common trick is to try a predicate which is equivalent to (8) in classical logic, but not in intuitionistic. The problem with (8) is that its realizers compute something useful because of the existential quantifier. So we just need to "destroy" the computational content of $\exists$, and we know that double negation does that.

Proposition 4.28 *For a morphism $f : \mathbf{A} \to \mathbf{B}$,*

$$\models \forall\, y \in B\,.\,\neg\neg\exists\, x \in A\,.\, f(x) =_B y$$

is valid if, and only if, f is epi.

Proof. $\forall\, y \in B\,.\,\neg\neg\exists\, x \in A\,.\,f(x) =_B y$ is intuitionistically equivalent to $\forall\, y \in B\,.\,\neg\forall\, x \in A\,.\,\neg f(x) =_B y$, which is a negative formula. Its ordinary meaning is that $f : A \to B$ is surjective, i.e., that $f : \mathbf{A} \to \mathbf{B}$ is epi. ∎

Exercise 4.29 Prove in intuitionistic logic that $\neg\neg\exists\, x \in A\,.\,P(x)$ is equivalent to $\neg\forall\, x \in A\,.\,\neg P(x)$.

4.4.4 Multi-valued maps and $\forall\exists$ predicates

In §4.4.3 we saw that a realizer for (9) also realized a multi-valued map. Let us look at the general situation. Consider a realizability relation R on $\mathbf{A} \times \mathbf{B}$ and a $\forall\exists$ predicate

$$\forall\, x \in A\,.\,\exists\, y \in B\,.\,R(x, y)\ .$$

A realizer for it is a computable $\theta \in \mathbb{B}$ such that if $\alpha \Vdash_A x$ then $\boldsymbol{\eta}_\theta(\alpha) \Vdash \exists\, y \in B\,.\,R(x, y)$, hence there is $y_0 \in B$ such that

$$\mathsf{fst}\,(\boldsymbol{\eta}_\theta(\alpha)) \Vdash_A y_0 \quad \text{and} \quad \mathsf{snd}\,(\boldsymbol{\eta}_\theta(\alpha)) \Vdash R(x, y_0)\ .$$

As before, the map $p(\alpha) = \mathsf{fst}\,(\boldsymbol{\eta}_\theta(\alpha))$ is computable and it realizes a multi-valued map $g : A \rightrightarrows B$ such that, whenever $\alpha \Vdash_A x$ then $p(\alpha) \Vdash_B y$ where $y \in g(x)$ and $\models R(x, y_0)$. This gives us an idea how to prove the existence of a computable multi-valued map using intuitionistic logic.

Proposition 4.30 *If intuitionistic logic proves $\forall\, x \in A\,.\,\exists\, y \in B\,.\,R(x, y)$ then there exists a computably realized multi-valued choice map $g : A \rightrightarrows B$ such that $\models R(x, g(x))$ for all $x \in A$.*

Remark: the notation $R(x, g(x))$ is not entirely correct, as $g(x)$ is a set of values. What we mean is that *any* value of $g(x)$ would work: for all $x \in A$, $y \in B$, if $y \in g(x)$ then $\models R(x, y)$.

Proof. Suppose intuitionistic logic proves $\forall\, x \in A\,.\,\exists\, y \in B\,.\,R(x, y)$. Then there is a computable $p : \mathbb{B} \rightharpoonup \mathbb{B}$ which validates the predicate. Define a multi-valued map $g : A \rightrightarrows B$ by

$$g(x) = \{y \in B \;\big|\; \exists\, \alpha \in \mathbb{B}\,.\,(\alpha \Vdash_A x \wedge \mathsf{fst}\,(p(\alpha)) \Vdash y)\}\ .$$

A computable realizer for g is the map $\alpha \mapsto \mathsf{fst}\,(p(\alpha))$. To see that g is a choice map for R, consider any $x \in A$ and $y \in g(x)$. There exists $\alpha \in \mathbb{B}$ such that $\alpha \Vdash_A x$ and $\mathsf{fst}\,(p(\alpha)) \Vdash_B y$, hence $p(\alpha) \Vdash R(x, y)$. ∎

In practice we most often see $\forall\exists$ predicates $\forall x A \exists\, y \in B\,.\,R(x, y)$ where R is a classical relation. These are particularly simples as $\models R(x, y)$ may be read as an ordinary relation.

4.4.5 Axiom of Choice

As we saw in §4.4.4, valid $\forall\exists$ predicates give us choice functions which are multi-valued in general. Existence of single-valued choice functions is expressed by (one form of) the axiom of choice. We examine those representations for which the axiom of choice is valid in the realizability logic.

Definition 4.31 A representation $\mathbf{A}$ is said to *satisfy choice* when

$$\models (\forall\, x \in A\,.\,\exists\, y \in B\,.\,R(x, y)) \Rightarrow \exists\, f \in B^A\,.\,\forall\, x \in A\,.\,R(x, f(x))\ . \tag{10}$$

for every representation $\mathbf{B}$ and relation R on $\mathbf{A} \times \mathbf{B}$,

Statement (10) says that every total relation R with domain A has a (single-valued) choice function. Of course, Definition 4.31 is of value only if we can characterize in a simple way representations satisfying it.

Definition 4.32 A representation $\mathbf{A} = (A, \delta_A)$ is *canonically projective* if its representation function $\delta_A : \mathbb{B} \rightharpoonup A$ is injective, or equivalently, if for all $\alpha, \beta \in \mathbb{B}$, $x \in A$,

$$\alpha \Vdash_A x \wedge \beta \Vdash_A x \implies \alpha = \beta \,.$$

A representation is *projective* if it is isomorphic to a canonically projective one.

It seems odd that we call a representation "projective" when its representation is *injective*. The name comes from a general definition in category theory (projective for regular epis), which in $\mathsf{Rep}(K_2)$ happens to mean precisely injectivity of δ_A.

Proposition 4.33 *A representation satisfies choice if, and only if, it is projective.*

Proof. Suppose $\mathbf{A}$ satisfies choice. Let $\mathbf{B}$ be the representation

$$B = |A| = \{\alpha \in \mathbb{B} \;\big|\; \exists\, x \in A \,.\, \alpha \Vdash_A x\} \quad \text{and} \quad \beta \Vdash_B \alpha \iff \beta = \alpha \,.$$

Define a realizability relation R on $\mathbf{A} \times \mathbf{B}$ by

$$\beta \Vdash R(x, \alpha) \iff \alpha \Vdash_A x \,.$$

Observe that R has irrelevant realizers. Because $\mathbf{A}$ satisfies choice there is a computable $p : \mathbb{B} \rightharpoonup \mathbb{B}$ which validates (10) for our $\mathbf{B}$ and R.

A computable θ such that $\boldsymbol{\eta}_\theta(\alpha) = \langle \alpha, 0^\omega \rangle$ realizes $\forall\, x \in A \,.\, \exists\, \alpha \in B \,.\, R(x, \alpha)$, thus $p(\theta) \Vdash \exists\, f \in B^A \,.\, \forall\, x \in A \,.\, R(x, f(x))$. By Proposition 4.12, there exists a morphism $f : \mathbf{A} \to \mathbf{B}$, realized by a computable $q : \mathbb{B} \rightharpoonup \mathbb{B}$, such that $\models \forall\, x \in A \,.\, R(x, f(x))$. Because $\forall\, x \in A \,.\, R(x, f(x))$ is a valid classical predicate we may read it as an ordinary predicate. It tells us that $f(x) \Vdash_A x$ for every $x \in A$. Now let $\mathbf{C}$ be the representation

$$C = \mathsf{im}(f) = \{\alpha \in \mathbb{B} \;\big|\; \exists\, x \in A \,.\, f(x) = \alpha\} \quad \text{and} \quad \beta \Vdash_C \alpha \iff \beta = \alpha \,.$$

Clearly, $\mathbf{C}$ is canonically projective. We claim that $f : A \to C$ is an isomorphism. The map $f : A \to C$ is surjective by definition and injective because $f(x) = f(y)$ together with $f(x) \Vdash_A x$ and $f(y) \Vdash_A y$ implies $x = y$. As a morphism $\mathbf{A} \to \mathbf{C}$, f is realized by q, and f^{-1} is realized by the identity map. This proves that f is an isomorphism. Therefore $\mathbf{A}$ is projective.

It suffices to prove the converse for a canonically projective $\mathbf{A}$. We argue informally that (10) is realized by showing that there is a computable procedure transforming realizers of the antecedent to realizers of the consequent. Suppose $\theta \Vdash \forall\, x \in A \,.\, \exists\, y \in B \,.\, R(x, y)$. For $\alpha \Vdash_A x$, $\boldsymbol{\eta}_\theta(\alpha) \Vdash \exists\, f \in B^A \,.\, \forall\, x \in A \,.\, R(x, f(x))$. The computable map $p(\alpha) = \mathsf{fst}\,(\boldsymbol{\eta}_\theta(\alpha))$ tracks a morphism $f_0 : \mathbf{A} \to \mathbf{B}$, defined by $f_0(\delta_A(\alpha)) = \delta_B(p(\alpha))$, which is a good definition because $\mathbf{A}$ is canonically projective. Now $\alpha \mapsto \mathsf{snd}\,(\boldsymbol{\eta}_\theta(\alpha))$ maps $\alpha \in \mathbb{B}$ such that $\alpha \Vdash_A x$ to a realizer for $\forall\, x \in A \,.\, R(x, f_0(x))$, therefore the consequent is realized. ∎

A representation $\mathbf{A}$ induces a topology on the underlying set A, namely the quotient topology induced by the representation map $\delta_A : \mathbb{B} \rightharpoonup A$. We say that $\mathbf{A}$ represents a topological space X if $A = X$ and the induced topology on A coincides with the topology of X. In $\mathsf{Rep}(K_2^{\mathrm{cont}})$ there is an interesting topological characterization of those representations that satisfy choice.

Exercise 4.34

1. Prove that zero-dimensional countably-based Hausdorff spaces (Hausdorff spaces with countably based clopen base) are precisely those spaces that embed into Baire space $\mathbb{B}$.
2. Show that a quotient map $f : X \to Y$ between zero-dimensional countably-based Hausdorff spaces is a retraction, i.e, that there exists a continuous $g : Y \to X$ such that $f \circ g = \mathrm{id}_Y$.
3. Prove that in $\mathsf{Rep}(K_2^{\mathrm{cont}})$ a representation is projective if, and only if, it represents a zero-dimensional countably-based Hausdorff space.

By combining Exercise 4.34 and Proposition 4.33, we get an interesting connection between realizability logic and topology.

Corollary 4.35 *A representation satisfies choice in* $\mathsf{Rep}(K_2^{cont})$ *if, and only if, it represents a zero-dimensional countably-based Hausdorff space.*

Exercise 4.36 Figure out why Corollary 4.35 does not work in $\mathsf{Rep}(K_2)$ and recover it by formulating a suitable notion of *computable* zero-dimensional countably-based Hausdorff spaces.

4.4.6 Decidable Predicates

In classical logic we are used to defining a function $f : A \to B$ by cases as

$$f(x) = \begin{cases} f_0(x) & \text{if } P(x), \\ f_1(x) & \text{otherwise,} \end{cases}$$

where P is any (ordinary) predicate on the set A. This defines a function because, for every $x \in A$, exactly one of the possibilities $P(x)$ and $\neg P(x)$ holds. Therefore, definition by cases relies on Excluded Middle, which means that in intuitionistic logic a definition by cases is allowed only when P is decidable.

The *characteristic map* $\chi_P : A \to \{0, 1\}$ is an example of a map defined by cases,

$$\chi_P(x) = \begin{cases} 1 & \text{if } P(x), \\ 0 & \text{if } \neg P(x). \end{cases}$$

Thus in intuitionistic logic *only decidable predicates have characteristic maps into* $\{0, 1\}$. (Even intuitionistically every predicate has a characteristic map into Ω, the set of (intuitionistic) truth values. But we do not have Ω because we are not working with a topos.)

We can also explain the connection between decidable predicates and characteristic maps with realizability logic. Suppose P is a predicate on $\mathbf{A}$. Recall that by the argument in §4.3.1 if P is decidable then there exists a computable map $q : \mathbb{B} \rightharpoonup \{0, 1\}$ such that, for all $\alpha \Vdash_A x$,

$$q(\alpha) = \begin{cases} 1 & \text{if } \models P(x), \\ 0 & \text{if } \models \neg P(x). \end{cases}$$

With the help of q we may then realize the characteristic map

$$\chi_P(x) = \begin{cases} 1 & \text{if } \models P(x), \\ 0 & \text{if } \models \neg P(x). \end{cases}$$

Conversely, if the characteristic map χ_P is realized, then P is decidable because

$$\models P(x) \iff \chi_P(x) = 1 \tag{11}$$

and

$$\models \chi_P(x) = 1 \vee \neg(\chi_P(x) = 1)\ . \tag{12}$$

Exercise 4.37 Write down computable maps which validate (11) and (12).

An important special case of a decidable predicate is decidable equality.

Definition 4.38 A representation $\mathbf{A}$ is *decidable* if its equality relation is decidable, $\models x =_A y \vee \neg(x =_A y)$.

By what we said above, $\mathbf{A}$ is decidable if, and only if, the characteristic map of equality

$$\mathrm{eq}_A(x,y) = \begin{cases} 1 & \text{if } x = y, \\ 0 & \text{if } x \neq y, \end{cases}$$

is realized as a morphism $\mathrm{eq}_A : \mathbf{A} \times \mathbf{A} \to \mathbf{2}$, where $\mathbf{2}$ is the representation of Booleans $\{0,1\}$.

Exercise 4.39 Prove that $\mathbf{2}$ is decidable and give as simple as possible an example of a representation that is not decidable.

Exercise 4.40 In $\mathsf{Rep}(K_2^{\mathrm{cont}})$ there is a topological characterization of decidable representations. Prove that $\mathbf{A}$ is decidable if, and only if, the topology on A induced by δ_A is discrete. What happens in $\mathsf{Rep}(K_2)$?

The *Limited Principle of Omniscience (LPO)* is the statement

$$\forall\, f \in 2^{\mathbb{N}} \,.\, ((\forall\, n \in \mathbb{N} \,.\, f(n) = 0) \vee (\exists\, n \in \mathbb{N} \,.\, f(n) = 1)) \ .$$

Proposition 4.41 *In* $\mathsf{Rep}(K_2)$*, LPO is* not *valid.*

Proof. LPO is intuitionistically equivalent to decidability of $\mathbf{2^N}$. But a decidable representation induces the discrete topology on the underlying set by Exercise 4.40. Since the topology 2^N induced by the representation on $\mathbf{2^N}$ is the usual product topology, $\mathbf{2^N}$ is not decidable. ∎

Exercise 4.42 Fill in the holes in the above proof: (1) prove intuitionistically that LPO is equivalent to "$2^{\mathbb{N}}$ is decidable" (hint: you will need Markov Principle 4.22), and (2) prove that the topology on $2^{\mathbb{N}}$ induced by $\mathbf{2^N}$ is the product topology (hint: see Proposition 5.12).

5 Examples from Computable Mathematics

In §4 we explained realizability logic and computed examples which showed how we can use it to express interesting properties of representations and realized maps. The calculations involved a fair bit of manipulation of realizers. This might give one the impression that in order to work with realizability logic one has to fiddle with realizers all the time. Quite on the contrary, realizability logic allows us to *avoid* realizers and use intuitionistic logic instead, by applying Proposition 4.10, Corollary 4.13, and Proposition 4.30. However, all this is not very useful if we do not know how to speak computable mathematics in the language of intuitionistic logic. Fortunately, the right way of speaking has already been developed, and it is called *constructive mathematics*:

Computable mathematics is
the realizability interpretation of
constructive mathematics.

We do not have to guess how constructions and theorems of computable mathematics translate into realizability logic. According to the slogan, they are just the familiar constructive constructions and theorems. In this section we look at several examples which demonstrate this point.

5.1 Natural Numbers

The best way to arrive at the representation of natural numbers is to apply the category-theoretic definition of natural numbers object to $\mathsf{Rep}(K_2)$. According to the definition, we need a representation $\mathbf{N}$ with an element $0 \in N$ and a morphism $s : \mathbf{N} \to \mathbf{N}$ such that, for every $\mathbf{A}$, $x_0 \in A$, and $f : \mathbf{A} \to \mathbf{A}$ there exists a unique $h : \mathbf{N} \to \mathbf{A}$ such that, for all $n \in N$,

$$h(0) = x_0 \quad \text{and} \quad h(s(n)) = f(h(n)) \; .$$

Since we already have a pretty good idea of how to represent natural numbers, it is easiest to check that our guess satisfies the above condition.

Let $\mathbf{N}$ be the representation $N = \mathbb{N} = \{0, 1, 2, \ldots\}$ with realizability relation

$$\alpha \Vdash_N n \iff \forall\, k \in \mathbb{N} \,.\, \alpha(k) = n \; .$$

Let $s : \mathbf{N} \to \mathbf{N}$ be the successor map, $s(n) = n + 1$, which is clearly realized. Given and $x_0 \in \mathbf{A}$ and $f : \mathbf{A} \to \mathbf{A}$, define $h : \mathbf{N} \to \mathbf{A}$ by simple recursion:

$$h(0) = x_0 \quad \text{and} \quad h(n+1) = f(h(n)) \; .$$

With a bit of effort it can be checked that h is realized. Suppose $g : \mathbf{N} \to \mathbf{A}$ also satisfies $g(0) = x_0$ and $g(s(n)) = f(g(n))$. Then $g = h$ by induction: $g(0) = x_0 = h(0)$ and if $g(n) = h(n)$ then $g(n+1) = f(g(n)) = f(h(n)) = h(n+1)$.

The representation $\mathbf{N}$ behaves in realizability logic much like ordinary natural numbers to in classical logic. For example, we have the usual induction principle.

Proposition 5.1 *For every predicate P on $\mathbf{N}$, the induction principle is valid:*

$$\models (P(0) \wedge \forall\, n \in N \,.\, (P(n) \Rightarrow P(n+1))) \Rightarrow \forall\, n \in n \,.\, P(n) \; .$$

Proof. The induction principle is a negative formula. Since it holds for N in classical logic, it is valid in realizability logic. ∎

Proposition 5.2 $\mathbf{N}$ *is decidable.*

Proof. It would be easy to show directly that the characteristic map of equality $=_N$ is computably realized. Alternatively, we may construct the characteristic map intuitionistically.: define the map $e : \mathbf{N} \times \mathbf{N} \to \mathbf{2}$ by double recursion:

$$\begin{aligned} e\langle 0, 0\rangle &= 1 \\ e\langle 0, n+1\rangle &= 0 \\ e\langle m+1, 0\rangle &= 0 \\ e\langle m+1, n+1\rangle &= e\langle m, n\rangle \ . \end{aligned}$$

Then a double induction on n and m shows that $e\langle n, m\rangle = 1 \iff n = m$. It follows that e is the characteristic map of $=_N$. ∎

Proposition 5.3 (Number Choice) $\mathbf{N}$ *satisfies choice.*

Proof. Clearly, $\mathbf{N}$ is canonically projective, therefore it satisfies choice by Proposition 4.33. ∎

The sets of integers $\mathbb{Z}$ and rational numbers $\mathbb{Q}$ have their standard representations $\mathbf{Z}$ and $\mathbf{Q}$, too. We do not dwell on them, as their treatment is very similar to that of natural numbers. We move on to real numbers, instead.

5.2 Real Numbers

In Type II Computability [Wei00, 4.1] various representations of real numbers are considered. Three of them, $\rho_<$, $\rho_>$ and ρ turn out to be most useful, and it is shown that ρ is equivalent to a representation by rapidly converging Cauchy sequences. On the other hand, the representation by *all* Cauchy sequences is rejected as inappropriate [Wei00, 4.1.14.1].

According to our slogan, the representations $\mathbf{R}_< = (\mathbb{R}, \rho_<)$, $\mathbf{R}_> = (\mathbb{R}, \rho_>)$, and $\mathbf{R} = (\mathbb{R}, \rho)$ ought to be realizability interpretations of well known definitions in constructive mathematics. This is indeed the case.

The standard construction of Cauchy reals $\mathbb{R}_c$ in constructive mathematics consists of two steps. First, the set of rapidly converging rational sequences is defined:

$$\mathbf{C} = \{a \in \mathbf{Q}^{\mathbf{N}} \mid \forall\, n \in N\,.\, |a(n+1) - a(n)| < 2^{-n}\}\ , \tag{13}$$

Then the set of Cauchy reals is obtained as a quotient $\mathbf{R}_c = \mathbf{C}/{\sim}$, where $\sim$ is the coincidence relation on $\mathbf{C}$, defined by

$$a \sim b \iff \forall\, n \in N\,.\, |a(n) - b(n)| < 2^{-n+2}\ .$$

Let us calculate the realizability interpretation of this construction. Because the defining predicate in (13) is a negative ($<$ on $\mathbf{Q}$ is classical, and even decidable) we may use Proposition 4.21 to calculate the representation $\mathbf{C}$ of rapidly converging Cauchy sequences. Not surprisingly, the underlying set C consists of the rapidly converging Cauchy sequences. A realizer for such a sequence simply encodes its terms. The coincidence relation $\sim$ is defined by a negative formula, too, so its realizability interpretation coincides with the ordinary meaning. Thus R_c consists of equivalence classes of coincident Cauchy sequences, i.e., it is the set of real numbers. A real number is represented by a realizer which represents a Cauchy sequence from its equivalence class. We have indeed obtained the Cauchy representation of reals [Wei00, 4.1.5]:

Proposition 5.4 *The realizability interpretation of constructive real numbers is the Cauchy representation of reals, which is isomorphic to the standard representation* $\mathbf{R}$.

There is another well known construction of reals by Dedekind cuts. In constructive mathematics, a Dedekind cut is a pair (L, U) of subsets of $\mathbb{Q}$ such that:

1. L and U are inhabited: $\exists\, q \in \mathbb{Q}\,.\, x \in L$ and $\exists\, r \in \mathbb{Q}\,.\, x \in U$,

2. L is a lower set and U is an upper set: $r \in L \wedge q < r \Rightarrow q \in L$ and $q \in U \wedge q < r \Rightarrow r \in U$,
3. L and U are open: $q \in L \Rightarrow \exists\, r > q\,.\, r \in L$ and $r \in U \Rightarrow \exists\, q > r\,.\, q \in U$,
4. the cut is located: $q < r \Rightarrow (q \in L \vee r \in U)$,
5. L and U are disjoint: $\neg(q \in L \wedge q \in U)$.

The Dedekind reals $\mathbb{R}_d$ are defined as the set of all Dedekind cuts. It is well known that the Cauchy and Dedekind construction give isomorphic sets, if Number Choice is valid, which it is in the realizability interpretation. As stated, the construction of $\mathbb{R}_d$ requires powersets, since $\mathbb{R}_d$ is defined as a subset of $\mathcal{P}\mathbf{Q} \times \mathcal{P}\mathbf{Q}$. However, it turns out that we may consider only the *topologically open* cuts so that $\mathcal{P}\mathbf{Q}$ may be replaced by $\mathbf{\Sigma}^{\mathbf{Q}}$, where $\mathbf{\Sigma}$ is the standard representation of the Sierpinski space. When this is done, the construction makes sense in $\mathsf{Rep}(K_2)$, and it gives a representation that is isomorphic to $\mathbf{R}$.

A *lower* Dedekind cut is a subset $L \subseteq \mathbb{Q}$ which is an inhabited, bounded open lower set. The realizability interpretation of the set of lower Dedekind cuts (again the cuts are interpreted as elements of $\mathbf{\Sigma}^{\mathbf{Q}}$) gives the representation $\mathbf{R}_<$. Similarly, the upper Dedekind cuts give the representation $\mathbf{R}_>$.

Exercise 5.5 Consider representations

$$\{a \in \mathbf{Q}^{\mathbf{N}} \;\big|\; (\forall\, n \in N\,.\, a(n) < a(n+1)) \wedge \exists\, q \in Q\,.\, \forall\, n \in N\,.\, a(n) < q\}/{\approx}\,.$$

and

$$\{a \in \mathbf{Q}^{\mathbf{N}} \;\big|\; (\forall\, n \in N\,.\, a(n) < a(n+1)) \wedge \neg\forall\, q \in Q\,.\, \neg\forall\, n \in N\,.\, a(n) < q\}/{\approx}\,.$$

where $\approx$ is the coincidence relation on $\mathbf{Q}^{\mathbf{N}}$ defined as

$$a \approx b \iff \forall\, k \in N\,.\, \exists\, m \in N\,.\, \forall\, n, n' \in N\,.\, \big(n > k \wedge n' > k \Rightarrow |a(n) - b(n')| < 2^{-k}\big)\,. \qquad (14)$$

Which of the two representations, if any, is isomorphic to $\mathbf{R}_<$? (Hint: trust your constructivist feelings.)

As we mentioned before, Type II Computability rejects the so-called "naive Cauchy" representation of reals $\mathbf{R}_{nc} = (\mathbb{R}, \rho_{nc})$ where a ρ_{nc}-name of $x \in \mathbb{R}$ encodes *any* rational Cauchy sequence converging to x. Let us see what this representation corresponds to in the constructive world. Define the predicate $\mathsf{cauchy}(a)$ on $\mathbf{Q}^{\mathbf{N}}$ by

$$\mathsf{cauchy}(a) \iff \forall\, k \in N\,.\, \exists\, m \in N\,.\, \forall\, n, n' \in N\,.\, \big(n > k \wedge n' > k \Rightarrow |a(n) - a(n')| < 2^{-k}\big)\,.$$

The intuitive meaning of $\theta \Vdash \mathsf{cauchy}(\alpha)$ is that $\boldsymbol{\eta}_\theta$ computes from $k \in \mathbb{N}$ a number $m \in \mathbb{N}$ such that all terms of a after the m-th one are less than 2^{-k} apart from one another.

Proposition 5.6

1. $\{a \in \mathbf{Q}^{\mathbf{N}} \;\big|\; \mathsf{cauchy}(a)\}/{\approx}$ *is isomorphic to the standard representation of reals* $\mathbf{R}$,
2. $\{a \in \mathbf{Q}^{\mathbf{N}} \;\big|\; \neg\neg\mathsf{cauchy}(a)\}/{\approx}$ *is isomorphic to the naive representation* $\mathbf{R}_{nc}$.

Proof.

1. Here $\approx$ is defined as in (14). This is most easily proved in intuitionistic logic. We just need to show that for every Cauchy sequence there is a rapidly converging Cauchy sequence that coincides with it. The intuitionistic proof relies on Number Choice 5.3, and is left as an exercise, or can be looked up in [Bau00, 5.5.2].
2. The defining predicate $\neg\neg\mathsf{cauchy}(a)$ is classical. Thus $\{a \in \mathbf{Q}^{\mathbf{N}} \;\big|\; \neg\neg\mathsf{cauchy}(a)\}$ is the set of all rational Cauchy sequences, where the realizers encode the terms of the sequences. When we quotient this set by $\approx$ we get the set of real numbers, represented by realizers that encode Cauchy sequences converging to them. This is the naive representation $\mathbf{R}_{nc}$.

∎

Even an unexperienced constructive mathematician would immediately reject the second definition in Proposition 5.6 as unnatural. It is also clear from the point of view of computable mathematics that the first definition is better, because it does not throw away the valuable realizers for $\mathsf{cauchy}(a)$ which tell how quickly a converges.

Exercise 5.7 (Reals in Type I Computability) Show that when the construction of rapidly converging Cauchy reals is interpreted in $\mathsf{Rep}(K_1)$ we get the standard numbering of computable real numbers. What do we get when we interpret the construction of lower and upper Dedekind cuts? (Hint: look up the definition of left- and right-computable reals.)

5.3 Metric Spaces

The constructive definition of a metric space is the same as the classical one. A (constructive) metric space (M, d) is a set M with a metric $d : M \times M \to \mathbb{R}$ satisfying the usual axioms:

- $d(x, y) \geq 0$,
- $d(x, y) = 0 \iff x = y$,
- $d(x, y) = d(y, x)$,
- $d(x, z) \leq d(x, y) + d(y, z)$.

Observe that these conditions are negative predicates because $=$ and $\leq$ on real numbers are classical (a fact we did not verify). Because of this, the realizability interpretation of a constructive metric space is not surprising at all: it is a representation $\mathbf{M}$ equipped with a computably realized map $d : \mathbf{M} \times \mathbf{M} \to \mathbf{R}$ which satisfies the above conditions.

It is more interesting to look at constructive *complete* metric spaces and their realizability interpretation. Recall that a metric space is complete if every Cauchy sequence has a limit. We have seen in §5.2 that instead of general Cauchy sequences we may use the rapidly converging ones, which is convenient because the rapidly converging sequences have simpler realizers. So we define a (constructive) metric space (M, d) to be complete if every rapidly converging sequence has a limit. To see what the realizability interpretation of this condition is, we compute its realizers. Let C_M be the representation of rapidly converging Cauchy sequences in M,

$$C_M = \{a \in M^{\mathbb{N}} \mid \forall\, k \in \mathbb{N} \,.\, d(a(k), a(k+1)) < 2^{-k}\} \,.$$

Completeness of M is expressed by the formula

$$\forall\, a \in C_M \,.\, \exists\, x \in M \,.\, \forall\, k \in \mathbb{N} \,.\, d(a(k), x) < 2^{-k+2} \,. \tag{15}$$

The metric space (M, d) is interpreted by a representation with computable metric $(\mathbf{M}, d)$, as above. The interpretation of C_M is a representation $\mathbf{C}_M$ whose underlying set is the set of rapidly converging Cauchy sequences in M. A δ_{C_M}-name of $a \in C_M$ encodes the terms of the sequence, where each term is encoded by its δ_M-name. By Proposition 4.30, validity of (15) is equivalent to the existence of a multi-valued computable map $\ell : C_M \rightrightarrows M$ such that, for all $a \in C_M$,

$$\models \forall\, k \in N \,.\, d(a(k), \ell(a)) < 2^{-k+2} \,.$$

The above realizability predicate is negative so we may read it as an ordinary one. It tells us that $\ell(a)$ is the limit of a. Since the limit of a Cauchy sequence is unique, if it exists, the map ℓ is actually single-valued. To summarize: the realizability interpretation of completeness of a metric space is a computable limit operator which maps (rapidly converging) Cauchy sequences to their limits. This immediately suggest a definition of a computable complete metric space:

Definition 5.8 A *computable complete metric space* is a triple $(\mathbf{M}, d, \ell)$ where $\mathbf{M}$ is a representation, $d : \mathbf{M} \times \mathbf{M} \to \mathbf{R}$ is a computable metric, and $\ell : \mathbf{C}_M \to \mathbf{M}$ is a computable limit operator which maps a (rapidly converging) Cauchy sequence to its limit.

Next, let us find a suitable definition of *separability*. A metric space (M, d) is separable if there exists a sequence $s : \mathbb{N} \to M$ such that

$$\forall\, k \in \mathbb{N}\,.\,\forall\, x \in M\,.\,\exists\, n \in \mathbb{N}\,.\, d(x, s(n)) < 2^{-k}\ .$$

Once again this is a $\forall\exists$ statement (we may rewrite the double $\forall\forall$ as a single one ranging over $\mathbb{N}\times M$). By Proposition 4.30 its validity is equivalent to the existence of a computable multi-valued map $h : \mathbf{N} \times \mathbf{M} \to \mathbf{N}$ such that $d(x, s(h(k, x))) < 2^{-k}$ for all $k \in N$, $x \in M$.

Definition 5.9 A *computable separable metric space* is a quadruple $(\mathbf{M}, d, s, h)$ where $d : \mathbf{M} \times \mathbf{M} \to \mathbf{R}$ is a computable metric on $\mathbf{R}$, $s : \mathbf{N} \to \mathbf{M}$ is a computable sequence, and $h : \mathbf{N} \times \mathbf{M} \rightrightarrows \mathbf{N}$ is a computable multi-valued map such that $d(x, s(h(k, x))) < 2^{-k}$ for all $k \in N$, $x \in M$.

Above definitions are rather rudimentary realizability interpretations of constructive definitions. For a non-trivial application of constructive mathematics we cite a result by Lietz.

Proposition 5.10 *Suppose intuitionistic logic proves that (M, d) is a complete separable metric space. Then the realizability interpretation of M is an admissible representation.*

Proof. Combine Theorems 3.2.7 and 3.2.9 of [Lie04]. ∎

Exercise 5.11 Compute representations of compact metric spaces via the realizability interpretation of constructive *complete totally bounded metric spaces*. Recall that a metric space (M, d) is *totally bounded* when for every $k \in \mathbb{N}$ there exists a finite sequence $y_0, \ldots, y_n \in M$ such that for every $x \in M$ there exists $m \leq n$ such that $d(x, y_m) < 2^{-k}$. Warning: total boundedness is a $\forall\exists\forall\exists$ statement, which yields rather unwieldy realizers if interpreted literally. You should first use intuitionistic logic and Number Choice to simplify it.

5.3.1 The Baire Space

The natural representation for the Baire space it is the exponential $\mathbf{N}^{\mathbf{N}}$. Constructively, Baire space is a complete separable metric space, therefore $\mathbf{N}^{\mathbf{N}}$ has a computable metric, a computable limit operator, and is computably separable.

In $\mathsf{Rep}(K_2)$ Baire space enjoys a few special properties which have important consequences for computable analysis. These properties are not generally valid in other realizability models.

Proposition 5.12 $\mathbf{N}^{\mathbf{N}}$ *satisfies choice.*

Proof. By Proposition 4.33 it suffices to show that $\mathbf{N}^{\mathbf{N}}$ is isomorphic to a canonically projective representation. Let $\mathbf{B}$ be the representation

$$B = \mathbb{N}^{\mathbb{N}} \qquad \alpha \Vdash \beta \iff \alpha = \beta\ .$$

Obviously, $\mathbf{B}$ is canonically projective. We claim that $\mathbf{B} \cong \mathbf{N}^{\mathbf{N}}$. Both representations have the same underlying set so we only have to check that the identity map between them is realized in each direction. The identity map on $\mathbb{N}^{\mathbb{N}}$ is realized as a morphism $\mathbf{N}^{\mathbf{N}} \to \mathbf{B}$ by $\alpha \mapsto (n \mapsto (\boldsymbol{\eta}_\alpha(n^\omega))(0))$. The other direction is realized by an equally ugly realizer. ∎

Exercise 5.13 Show that in $\mathsf{Rep}(K_1)$ Baire space does *not* satisfy choice. Hint: total recursive functions do not have canonical Gödel codes.

If $\alpha \in \mathbb{B}$ and $n \in \mathbb{N}$, let $\overline{\alpha}(n)$ denote the finite prefix $[\alpha(0), \ldots, \alpha(n)]$. The sets $\mathbb{B}$ and $\mathbb{N}$ are metric spaces with the usual computable metrics (where the usual metric on $\mathbb{B}$ is $d(\alpha, \beta) = 2^{-\min_k(\alpha_k \neq \beta_k)}$). A function $f : \mathbb{B} \to \mathbb{N}$ is pointwise continuous in the metric if, and only if,

$$\forall\, \alpha \in \mathbb{B}\,.\,\exists\, n \in \mathbb{N}\,.\,\forall\, \beta \in \mathbb{B}\,.\,\big(\overline{\alpha}(n) = \overline{\beta}(n) \Rightarrow f(\alpha) = f(\beta)\big)$$

This condition says that the value of $f(\alpha)$ depends only on a finite prefix of α.

Proposition 5.14 *In* $\mathsf{Rep}(K_2)$ *the statement "every function* $f : \mathbf{N}^{\mathbf{N}} \to \mathbf{N}$ *is pointwise continuous" is realized:*

$$\models \forall\, f \in N^{N^N}\,.\,\forall\, \alpha \in N^N\,.\,\exists\, n \in N\,.\,\forall\, \beta \in N^N\,.\,\big(\overline{\alpha}(n) = \overline{\beta}(n) \Rightarrow f(\alpha) = f(\beta)\big) \tag{16}$$

Proof. We do not provide a detailed proof. Observe that (16) is a $\forall\exists$ statement, thus its validity may be proved by showing that a suitable computable multi-valued map exists. The argument uses ideas similar to those in the proofs of [Wei00, 6.2.7,6.2.8]. ∎

5.3.2 The square-summable sequences

We borrow an instructive example from Lietz [Lie04, §3.3.3, p. 86]. Recall that a sequence $a : \mathbb{N} \to \mathbb{R}$ is *square-summable* if the sum

$$\|a\|_2 = \sum_{k=0}^{\infty} |a_k|^2$$

converges. The set of square-summable sequences

$$\ell_2 = \{a \in \mathbb{R}^{\mathbb{N}} \mid \textstyle\sum_{k=0}^{\infty} |a_k|^2 \text{ converges}\} \tag{17}$$

is a Hilbert space that can be equipped with at least two important topologies on ℓ_2: the one induced by the norm $\|-\|_2$, and the weak topology (the weakest topology on ℓ_2 for which all continuous linear functionals on ℓ_2 are still continuous). For each of these topologies there are admissible representations of ℓ_2 in Type II Computability. So it looks like our slogan is in trouble—how can both representations be the realizability interpretation of the constructive definition of ℓ_2? Can there be two constructive definitions of ℓ_2? Yes!

The first constructive definition of ℓ_2 is (17). From it one may show constructively that ℓ_2 is a complete separable metric space. By Proposition 5.10 it follows that the realizability interpretation of ℓ_2 is an admissible representation of ℓ_2, equipped with the topology induced by the norm.

To get a second constructive definition, consider

$$\ell_2^{(w)} = \{a \in \mathbb{R}^{\mathbb{N}} \mid \textstyle\sum_{k=0}^{\infty} |a_k|^2 \text{ bounded}\}\,.$$

This is classically equivalent to (17) but not intuitionistically, because in intuitionistic logic we cannot prove that a bounded monotone sequence has a limit. According to Lietz, the realizability interpretation of $\ell_2^{(w)}$ is a representation of the set ℓ_2 equipped with the weak topology.

Exercise 5.15 Peter Lietz did not bother to include a proof that $\ell_2^{(w)}$ really does induce the weak topology on the set ℓ_2. I suspect he meant it to be an exercise for you.

5.4 Continuous Maps

By Proposition (16), the realizability relation validates the statement "all $f : \mathbf{N}^{\mathbf{N}} \to \mathbf{N}$ are pointwise continuous", and by Proposition 5.12 $\mathbf{N}^{\mathbf{N}}$ satisfies choice. It is known [TvD88b, Theorem 7.2.7] that these two together imply constructively the statement "all $f : M \to M'$ are pointwise continuous" for complete separable metric spaces M and M'. Therefore, in the realizability interpretation we have

$$\models \text{all } f \in {M'}^{M} \text{ are pointwise continuous}$$

for any computably complete separable metric spaces $\mathbf{M}$ and $\mathbf{M}'$. We apply this to $\mathbf{M} = \mathbf{M}' = \mathbf{R}$. The statement "all $f \in R^R$ are pointwise continuous" is

$$\forall\, f \in R^R \,.\, \forall\, x \in R \,.\, \forall\, k \in N \,.\, \exists\, m \in N \,.\, \forall\, y \in R \,.\, (|x - y) < 2^{-m} \Rightarrow |f(x) - f(y)| < 2^{-k} \,.$$

This is a $\forall\exists$ statement. Because it is valid it follows by Proposition 4.30 that there is a computable multi-valued map $\mathbf{R}^{\mathbf{R}} \times \mathbf{R} \times \mathbf{N} \rightrightarrows \mathbf{N}$ which computes a modulus of continuity. This is a standard result in computable analysis [Wei00, Corolary 6.2.8], which we proved here by applying standard results in constructive analysis.

The constructive mathematicians (if any are still reading) will be interested to hear that $\mathsf{Rep}(K_2)$ also validates the Fan Theorem. Among the various realizability models, $\mathsf{Rep}(K_2)$ is the one that Brouwer would have liked best.

6 Conclusion

At "Computability and Complexity in Analysis 2001" in Dagstuhl, I was asked to represent my view of computable mathematics. I draw a a pretty slide.

I have since grown older and wiser, and I know the slide I drew was all wrong. Computable and constructive mathematics is like the Matrix. Do you remember the film *Matrix*? Computable mathematics is the Matrix, a world built by intelligent computers to keep people from seeing the world as it really is. The realizers are the little green characters that keep falling down the screens. They seem infinitely more boring and less comprehensible than the Matrix. However, when Neo, the hero who saves humankind, reaches a higher level of awareness he sees the Matrix as it really is—made of little green characters. Intuitionistic logic and category theory are the Architect and the Oracle, but I am not telling which is which.

6.1 Further Reading

I hope these notes make it clear that anyone interested in computable mathematics should be familiar with constructive mathematics. Thus, if you have not yet read, or at least looked at Bishop's constructive mathematics [Bis67, BB85], you should. Another rich source on constructive mathematics is [TvD88a, TvD88b].

If you think Type II computability and realizers are fun you should read [KV65], with which Type II computability originated.

For material on computable mathematics via realizability theory, I recommend Peter Lietz's Ph.D. dissertation [Lie04], which contains advanced topics, such as constructive treatment of admissible representations. I also wrote a Ph.D. dissertation [Bau00] about the realizability approach to computable analysis and topology, which you may find useful to look at.

6.2 Acknowledgment

I gratefully thank the organizers of "Computability and Complexity in Analysis 2005", where I lectured on this topic at a satellite seminar. In particular, I thank Hideki Tsuiki and all the local organizers from Kyoto for their hospitality and generous support. I also thank Alex Simpson for his advice on how to make the notes better.

References

[Bau00] A. Bauer. *The Realizability Approach to Computable Analysis and Topology*. PhD thesis, Carnegie Mellon University, 2000. Available as CMU technical report `CMU-CS-00-164` and at `http://andrej.com/thesis`.

[BB85] E. Bishop and D. Bridges. *Constructive Analysis*, volume 279 of *Grundlehren der math. Wissenschaften*. Springer-Verlag, 1985.

[BBS04] A. Bauer, L. Birkedal, and D.S. Scott. Equilogical spaces. *Theoretical Computer Science*, 1(315):35–59, 2004.

[Bis67] E. Bishop. *Foundations of Constructive Analysis*. McGraw-Hill, New York, 1967.

[Bla97] J. Blanck. *Computability on Topological Spaces by Effective Domain Representations*. PhD thesis, Department of Mathematics, Uppsala University, 1997.

[Joh02] P.T. Johnstone. *Sketches of an Elephant*. Number 43 in Oxford Logic Guides. Oxford University Press, 2002.

[Kle45] S.C. Kleene. On the interpretation of intuitionistic number theory. *Journal of Symbolic Logic*, 10:109–124, 1945.

[KV65] S.C. Kleene and R.E. Vesley. *The Foundations of Intuitionistic Mathematics, especially in relation to recursive functions*. North-Holland Publishing Company, 1965.

[Lie04] P. Lietz. *From Constructive Mathematics to Computable Analysis via the Realizability Interpretation*. PhD thesis, Technische Universität Darmstadt, 2004.

[McL95] C. McLarty. *Elementary Categories, Elementary Toposes*. Oxford Logic Guides. Oxford University Press, 1995.

[MM92] S. Mac Lane and I. Moerdijk. *Sheaves in Geometry and Logic. A First Introduction to Topos Theory*. Springer-Verlag, New York, 1992.

[TvD88a] A.S. Troelstra and D. van Dalen. *Constructivism in Mathematics, An Introduction, Vol. 1*. Number 121 in Studies in Logic and the Foundations of Mathematics. North-Holland, 1988.

[TvD88b] A.S. Troelstra and D. van Dalen. *Constructivism in Mathematics, An Introduction, Vol. 2*. Number 123 in Studies in Logic and the Foundations of Mathematics. North-Holland, 1988.

[Wei00] Klaus Weihrauch. *Computable Analysis*. Springer-Verlag, 2000.