© Copyright, Princeton University Press. No part of this book may be distributed, posted, or reproduced in any form by digital or mechanical means without prior written permission of the publisher.

# Contents



For general queries, contact webmaster@press.princeton.edu

© Copyright, Princeton University Press. No part of this book may be distributed, posted, or reproduced in any form by digital or mechanical means without prior written permission of the publisher.





For general queries, contact webmaster@press.princeton.edu

© Copyright, Princeton University Press. No part of this book may be distributed, posted, or reproduced in any form by digital or mechanical means without prior written permission of the publisher.

# Chapter One

# Introduction

WHAT DOES IT take to perform a certain construction? In computability theory, this question is the basis of a long-term programme which seeks to understand the relationship between dynamic properties of sets and their algorithmic complexity. Our main thesis in this monograph is that where the computably enumerable (c.e.) Turing degrees are concerned, *a degree can compute complicated objects if and only if some functions in the degree are difficult to approximate.* Computability-theoretic tools allow us to quantify precisely what we mean by "difficult to approximate." More specifically, we use a classification of $\Delta^0_2$ functions defined by Ershov in [39–41]. While Ershov's hierarchy of complexity is orthogonal to complexity as measured by Turing reducibility, we show that combining these two notions of complexity yields a new, transfinite hierarchy inside the $\text{low}_2$ c.e. degrees, and that two levels of this hierarchy capture the dynamics of a number of seemingly unrelated constructions in different areas of computability. Further, some of these constructions show that these two levels are naturally definable in the c.e. degrees.

## 1.1 HISTORICAL CONTEXT

The roots of computability theory go back to the work of Borel [8], Dedekind [18], Hermann [50], Dehn [19], and others in the late nineteenth and early twentieth centuries. From a modern point of view, these authors were highly interested in algorithmic procedures in algebra. Around the same time, Hilbert famously posed the *Entscheidungsproblem*, which asked whether there was an algorithmic procedure to decide the validity of statements in first-order logic. To show that the answer is yes, we would need to give such an algorithm, as we do with truth tables in propositional logic. However, to demonstrate that there is *no such algorithm*, we would first need to mathematically specify what an algorithm is. Culminating in the work of Turing [97], several authors gave proofs that first-order logic is undecidable; there is no such algorithm. Turing's work built on Gödel's First Incompleteness Theorem, and gave a beautiful conceptual analysis which convincingly laid the foundations of computability theory. Turing machines gave a universal model of computation.

Following these early results, many problems, such as Hilbert's $10^{\text{th}}$ problem, the word problem for groups, or DNA self-assembly, have been shown to be

For general queries, contact webmaster@press.princeton.edu

© Copyright, Princeton University Press. No part of this book may be distributed, posted, or reproduced in any form by digital or mechanical means without prior written permission of the publisher.



undecidable. These proofs mostly followed a familiar pattern. They used Turing's notion of a *reduction* [98], and typically showed that the halting problem is reducible to the algorithmic decision problem at hand by some effective coding process.

A major impetus for the development of computability theory was Post's [78] which gave an analysis of the fine structure of reductions, and set a research agenda in the "structure theory" of computation. This paper was also famous as it "stripped away the formalism associated with the development of recursive functions in the 1930's and revealed in a clear informal style the essential properties of recursively enumerable sets and their role in Gödel's incompleteness theorem" (Soare [91]). Following Post's paper, there were three major developments:

- The Kleene-Post development of the finite extension method. This and related techniques demonstrated the richness of the structure of the Turing degrees, and were arguably a precursor to Cohen's method of forcing.
- The Friedberg-Muchnik theorem showing that there were intermediate computably enumerable Turing degrees. This result introduced the priority method to computability theory and is a hallmark of the area to this day.
- Sacks's work [81, 82] which culminated in his book [80] which proved a number of penetrating results on the structure of degrees, and developed the infinite-injury priority method, first introduced by Shoenfield [85]. Sacks's book famously proposed a research agenda with a number of difficult questions still open.

There were subsequent books by, for example, Rogers [79], Lerman [66], and Soare [91] exploring the universe of the degrees of unsolvability. But conceptual clarification provided by this early work has seen a flowering of applications of computability theory to many areas of mathematics. These include computable analysis [102] (a subject going back to Turing's [97]), computable algebra and model theory (see, for example, [38]), algorithmic randomness [27, 69, 74], algorithmic learning theory ([45]), and reverse mathematics [89], to name but a few. (See [22] for a general historical discussion of this development, mainly focussing on randomness.) Each of these areas has its own subareas, and hence the area of computability has become remarkably diverse.

This monograph has several goals. Some are in the spirit of Sacks's book. That is, we wish to introduce new techniques and classification tools for understanding the complexity of computation. These include some new nonuniform methods and certain symmetric games in the sense of Lachlan [60], in which obstacles in constructions turn out to reflect the boundary between what is and what is not possible. These games allow us to prove new definability results in the computably enumerable degrees. Another goal is in the spirit of Soare's book; we carefully guide the graduate student through complex techniques involving modern arguments. Our final goal is to formalize the persistent intuition that many of the constructions in the diverse areas of computability theory seem to

For general queries, contact webmaster@press.princeton.edu

© Copyright, Princeton University Press. No part of this book may be distributed, posted, or reproduced in any form by digital or mechanical means without prior written permission of the publisher.



have common combinatorics. How should we explain that? We will draw several areas back together by showing that the hierarchy we introduce can be used to explain, classify, and *unify* combinatorics in these areas.

## 1.2 BACKGROUND: UNIFYING CONSTRUCTIONS AND NATURAL DEFINABILITY

### 1.2.1 Unifying constructions and levels of permitting

Computability theory has a small number of classes of degrees which capture the underlying dynamics of a number of apparently similar constructions. A good example is the class of high degrees, the degrees $\mathbf{d}$ satisfying $\mathbf{d}' \geqslant \mathbf{0}''$. Martin [70] showed that a c.e. degree is high if and only if

(1) it contains a function dominating all computable functions;
(2) it contains a maximal set;
(3) it contains a hyperhypersimple set.

Another example would be the class of the promptly simple degrees (Ambos-Spies, Jockusch, Shore, and Soare [2]), which coincide with the low-cuppable degrees and the non-cappable degrees. A more recent example of current interest is the class of $K$-trivial degrees (see, for example, [28, 72, 73]), which have several characterisations arising from lowness constructions.

The example most relevant to this monograph is the class of array computable degrees, defined by Downey, Jockusch, and Stob [30, 31]. Recall that by Shoenfield's Limit Lemma [84], a function $g\colon \omega \to \omega$ is $\Delta^0_2$ if and only if it has a *computable approximation*: a uniformly computable sequence $\langle g_s \rangle$ of functions which converge to $g$ in the discrete topology, that is, for which for all $n$, $g_s(n) = g(n)$ for all but finitely many $s$. We think of each $g_s$ as a stage $s$ approximation for $g$. Associated to every computable approximation $\langle g_s \rangle$ is its *mind-change function*, which maps each $n$ to the number of stages $s$ such that $g_{s+1}(n) \neq g_s(n)$.

A c.e. Turing degree $\mathbf{a}$ is array computable if every function $g \in \mathbf{a}$ has a computable approximation $\langle g_s \rangle$ such that for all $n$ there are at most $n$ many stages $s$ such that $g_{s+1}(n) \neq g_s(n)$, that is, whose mind-change function is bounded by the identity function. The array computable degrees capture the combinatorics of a wide class of constructions. To wit, we observe that a c.e. degree is array noncomputable if and only if

(1) it is the degree of a perfect thin $\Pi^0_1$ class (Cholak, Coles, Downey, and Herrmann [12]);
(2) it bounds a disjoint pair of c.e. sets which have no separator computing $\varnothing'$ (Downey, Jockusch, and Stob [30]);
(3) it contains a c.e. set with maximal Kolmogorov complexity (Kummer [57]);

For general queries, contact webmaster@press.princeton.edu

© Copyright, Princeton University Press. No part of this book may be distributed, posted, or reproduced in any form by digital or mechanical means without prior written permission of the publisher.



(4) it does not have a strong minimal cover in the Turing degrees (Ishmukhametov [51]);
(5) it has effective packing dimension 1 (Downey and Greenberg [24]);
(6) it contains two left-c.e. reals with no common upper bound in the cl-degrees of left-c.e. reals (Barmpalias, Downey, and Greenberg [7]);
(7) it contains a set which is not reducible to the halting problem with tiny use (Franklin, Greenberg, Stephan, and Wu [43]).

The dynamics captured by classes of degrees are often phrased in terms of *permitting*. We perform some computable construction, often using the priority method. To make the construction succeed, we need to satisfy infinitely many requirements, and to meet each requirement, we need to enumerate some numbers into a c.e. set $A$ that we are building. The question is whether we can perform the construction "below" a given c.e. degree $\mathbf{d}$, which means, can we make $A \leqslant_{\mathrm{T}} \mathbf{d}$? In the standard framework, we choose a c.e. set $D \in \mathbf{d}$, and along with the construction we define a Turing reduction $\Phi$ of $A$ to $D$. Then, when we want to enumerate a number $n$ into $A$, we seek *permission* from $D$ to do so, which means that we want to see some number enter $D$ below the use $\varphi(n)$ that we declared for computing $A(n)$ from $D$ using $\Phi$. Naturally, we will not always receive such permission, and so we need to make several attempts at meeting the requirement, using different potential numbers $n$ to enumerate into $A$.

The "amount" of permitting that is required to carry out the construction (that is, to meet every requirement) corresponds to the class of degrees $\mathbf{d}$ below which we can perform the construction. The most common notion is *simple* permitting, which is given by any nonzero c.e. degree $\mathbf{d}$. Here it suffices for at least one of the attempts made by a given requirement to receive permission. This argument then shows, for example, that every c.e. degree bounds two incomparable c.e. degrees (the Friedberg-Muchnik construction can be performed using simple permitting), or that every c.e. degree bounds a 1-generic sequence.

*Prompt* permission, given by any promptly simple degree, also needs just one attempt to receive permission, but this permission must be given quickly: the required change in $D$ needs to happen within some computable bound given the stage number. At the other extreme from simple permitting is high permitting, in which every requirement makes infinitely many attempts, and to meet the requirement, all but finitely many of these attempts need to be permitted.

Array noncomputable permitting, originally called "multiple permitting," is an intermediate version, in which for each attempt at meeting a requirement, a number of required permissions is stated in advance. The connection with the complexity of approximations of functions in the degree is direct: mind-changes essentially correspond to instances of permission; the computable bound on the number of mind-changes is the same bound on the number of permissions required to meet a requirement. The remarkable fact is that in many cases it is shown that the level of permitting is not only sufficient but also necessary for the construction to succeed.

For general queries, contact webmaster@press.princeton.edu

© Copyright, Princeton University Press. No part of this book may be distributed, posted, or reproduced in any form by digital or mechanical means without prior written permission of the publisher.



As we shall see, in this monograph we introduce a transfinite hierarchy of classes, each of which has its own level of permitting; these classes generalise the array noncomputable degrees.

### 1.2.2 Natural definability and lattice embeddings

Ever since Lachlan and Yates's [59, 105] construction of a minimal pair refuted Shoenfield's conjecture [86] that the c.e. degrees are homogeneous, research in the c.e. degrees tended toward showing that they are as complicated as can be. For example, their theory (as a partial ordering) is computationally equivalent to full first-order arithmetic (see [49, 75]). This paradigm leads us to study definability in the partial ordering of the c.e. degrees, with the expectation that full bi-interpretability with arithmetic would hold. That would entail that a relation in the c.e. degrees is definable if and only if it is induced by a degree-invariant, arithmetic relation on indices of c.e. sets. Currently, this has almost been achieved, up to double jump classes:

**Theorem 1.1** (Nies, Shore, Slaman [75])**.** *Any relation on the c.e. degrees which is invariant under the double jump is definable in the c.e. degrees if and only if it is definable in first-order arithmetic.*

The proof of Theorem 1.1 involves interpreting the standard model of arithmetic in the structure of the c.e. degrees without parameters, and obtaining a definable map from degrees to indices (in the model) which preserves the double jump. The result gives a definition of a large collection of classes of degrees (for example, all jump classes $\mathrm{high}_n$ and $\mathrm{low}_n$, the latter for $n \geqslant 2$).

Theorem 1.1 has two shortcomings. One is the reliance on the invariance of the relation under the double jump. It follows that no collection of c.e. degrees that contains some, but not all, $\mathrm{low}_2$ degrees can be defined using the theorem; these are the kinds of collections that we investigate in this monograph.

Another issue is that the definitions provided by the theorem are not *natural*, as discussed by Shore [88]. The definitions given by Theorem 1.1 are not structural; they do not give insights into the role of the relations being defined in the structure of the c.e. degrees. To date, there are not many examples of natural definitions in the c.e. degrees. Among them are:

- the promptly simple degrees are defined as the non-cappable ones (Ambos-Spies, Jockusch, Shore, and Soare [2]);
- the contiguous degrees are defined as the locally distributive ones (Downey and Lempp [33]) and also as the ones which are not the top of a copy of the pentagon lattice (the nonmodular, 5-element lattice $N_5$) in the c.e. degrees (Ambos-Spies and Fejer [1]);
- a third example takes place in the truth-table c.e. degrees rather than the Turing c.e. degrees: a c.e. truth-table degree is $\mathrm{low}_2$ if and only if it has no minimal cover in the c.e. truth-table degrees (Downey and Shore [34]).

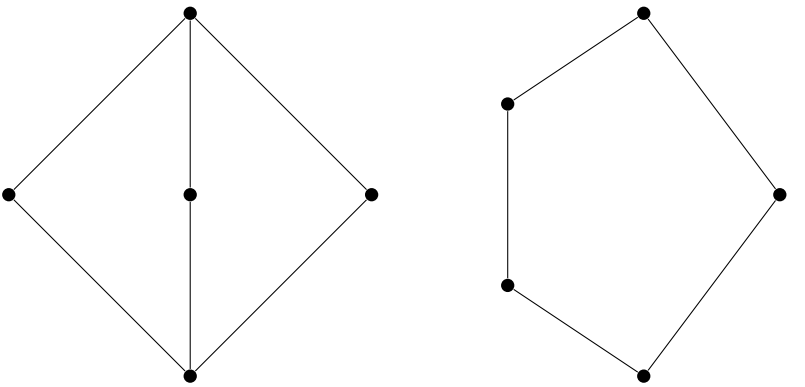
For general queries, contact webmaster@press.princeton.edu

© Copyright, Princeton University Press. No part of this book may be distributed, posted, or reproduced in any form by digital or mechanical means without prior written permission of the publisher.



The example of the contiguous degrees (Turing c.e. degrees all of whose c.e. elements are weak truth-table equivalent) shows that natural definability results can be found when considering lattice embeddings into the c.e. degrees (see, for example, [63, 64, 67]). The question of which finite lattices can be embedded into the c.e. degrees (preserving join and meet) is also closely related to the problem of determining how much of the theory of the c.e. degrees is decidable. For example, Kleene and Post [55] showed that every finite partial ordering is embeddable into the c.e. degrees, and so that the 1-quantifier theory of the c.e. degrees is decidable. Deciding 2-quantifier questions involves lattice embeddings and extensions of embeddings.

All distributive finite lattices are embeddable into the c.e. degrees (Thomason [96], and independently Lerman, unpublished). All nondistributive lattices contain copies of one of the two following lattices:



$M_5$ $N_5$

Figure 1.1. The two basic nondistributive lattices.

As mentioned, the lattice $N_5$ is nonmodular (the relation $a \vee (x \wedge b) = (a \vee x) \wedge b$ fails for some $a \leqslant b$), and every nonmodular lattice contains a copy of $N_5$. The lattice $M_5$, also known as the *1-3-1 lattice*, is modular, and every nondistributive, modular lattice contains a copy of the 1-3-1 lattice. Both lattices are embeddable into the c.e. degrees (Lachlan [61]).

The general question of which finite lattices are embeddable into the c.e. degrees remains open. The 1-3-1 is a significant obstacle, in that a slightly more complicated formation, known as the lattice $S_8$ (fig. 1.2), is not embeddable into the c.e. degrees (Lachlan and Soare [62]).

Thus, the 1-3-1 lattice is "just barely embeddable" in the c.e. degrees. Recalling our discussion above about permitting, the next natural question is how much computational power is required to embed this lattice. The point is that the embedding of the 1-3-1 lattice is quite complicated. Such an embedding, which is often done preserving the bottom element, involves the enumeration of

For general queries, contact webmaster@press.princeton.edu

© Copyright, Princeton University Press. No part of this book may be distributed, posted, or reproduced in any form by digital or mechanical means without prior written permission of the publisher.



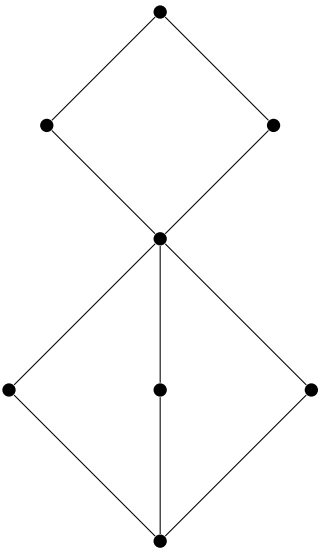


Figure 1.2. The lattice $S_8$.

three c.e. sets, $A_0$, $A_1$ and $A_2$, which pairwise form a minimal pair, and pairwise join above the third. The join and meet requirements interact very badly, and to overcome the difficulties, Lachlan used what became known as "continuous tracing." These difficulties were exploited by Downey [21], who showed that not every c.e. degree bounds a copy of the 1-3-1 lattice. In that paper, Downey noted that the embedding of the 1-3-1 lattice seemed to be tied up with multiple permitting in a way that was similar to non-low$_2$-ness. This intuition was verified by Downey and Shore [35], who showed that every non-low$_2$ c.e. degree bounds a copy of the 1-3-1 lattice in the c.e. degrees.

In attempting to synthesize the exact lattice structure which creates the embedding problems, Downey [21] and Weinstein [103] isolated the notion of a *critical triple*. A critical triple in a lattice consists of elements $a_0, a_1$ and $b$ such that $a_0 \vee b = a_1 \vee b$ but $a_0 \wedge a_1 \leqslant b$ (fig. 1.3)

More generally, in an upper semilattice (which may fail to be a lattice), the meet requirement is replaced by $c \leqslant a_0, a_1 \to c \leqslant b$. Weinstein also introduced the notion of a *weak critical triple* (which we will not use in this manuscript); there the meet requirement is replaced by $c \leqslant a_0, a_1 \to a_0 \not\leqslant b \vee c$. In the 1-3-1 lattice, the middle three elements, in any order, form a critical triple. Downey actually constructed an initial segment of the c.e. degrees in which there are no critical triples, and Weinstein did the same for weak critical triples.

The notion of non-low$_2$-ness seemed too strong to capture the class of degrees which bound a copy of the 1-3-1 lattice, but it was felt that something like that should suffice. On the other hand, Walk [101] constructed an array noncomputable c.e. degree bounding no weak critical triples, and hence it was already known that array noncomputability was not enough for such embeddings. In any

For general queries, contact webmaster@press.princeton.edu

© Copyright, Princeton University Press. No part of this book may be distributed, posted, or reproduced in any form by digital or mechanical means without prior written permission of the publisher.



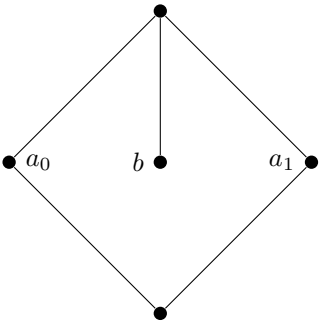



Figure 1.3. A critical triple.

case it was presumed that bounding the 1-3-1 lattice is equivalent to bounding a critical triple (or a weak critical triple). Our main result in this monograph implies that this presumption is false, and completely characterises the amount of permitting required to embed the 1-3-1 lattice.

## 1.3 TOWARD THE HIERARCHY OF TOTALLY $\alpha$-C.A. DEGREES

We now turn to discussing two levels of the new hierarchy that we introduce. Some preliminary ideas and results appear in the companion papers [23, 25], and some related results appeared after our work was discussed with colleagues. Now we will discuss these ideas and results together in a mathematically, rather than historically, coherent way. Later we will discuss in detail the content of this monograph.

### 1.3.1 Totally $\omega$-c.a. degrees

In 2005, J. Miller (unpublished) defined a non-uniform version of the class of array computable degrees. We call a function $\omega$-*computably approximable* ($\omega$-c.a.) if it has a computable approximation whose mind-change function is bounded by some computable function. This is equivalent to the function being weak truth-table reducible to $\varnothing'$. The notion is widely used in computability, with applications in algorithmic randomness as well (for example, in [42, 44, 47, 48]).[1]

This first step toward our new hierarchy is inspired by the above characterisation of array computable c.e. degrees as those which only contain functions with computable approximations with mind-change functions bounded by

[1]The terminology "$\omega$-c.a." is new. In the literature one usually finds "$\omega$-c.e.," although "$\omega$-computable" is also used. In Chapter 2 below we justify the new terminology.

For general queries, contact webmaster@press.princeton.edu

© Copyright, Princeton University Press. No part of this book may be distributed, posted, or reproduced in any form by digital or mechanical means without prior written permission of the publisher.



the identity. This is in some sense a "forced marriage" between two notions of complexity: complexity in terms of Turing degree; and complexity in terms of simplicity of approximations. $\omega$-c.a. functions are in some sense relatively simple, in that we can guess them with few mistakes; on the other hand, they can be Turing equivalent to $\mathbf{0}'$, making them as complicated as possible among all $\Delta^0_2$ functions when we consider Turing reducibility. When we consider approximations of *all* functions in a Turing degree, we get a new, useful concept. Thus Miller defined:

**Definition 1.2.** A c.e. degree is *totally $\omega$-c.a.* if every function in it is $\omega$-c.a.

Array computability is a uniform version of this notion: it requires the same bound on the mind-change function for all functions in the degree.

As discussed, this notion naturally aligns itself with a level of permitting. Recall that in array noncomputable permitting (previously named "multiple permitting"), each requirement plans infinitely many attempts at meeting it. Roughly, for the $n^{\text{th}}$ attempt to succeed, it needs $n$ many permissions on numbers associated with this attempt. This corresponds to the identity bound on the number of mind changes. In *non-totally $\omega$-c.a. permitting*, we again set up infinitely many attempts, but we are allowed to wait to declare how many permissions each attempt requires. Thus, for example, if the $n^{\text{th}}$ attempt is set up at stage $s$, then we could require $s$ many permissions; and $s$ could be much larger than $n$. For each requirement, the function mapping $n$ to the number of permissions required to meet the $n^{\text{th}}$ attempt is computable, but different requirements will define different computable functions, likely with no uniform computable bound on these functions when all requirements are considered.

Using this notion of permitting, the class of totally $\omega$-c.a. degrees captures the dynamics of a number of constructions. The first result appeared in [25], in which the authors, together with R. Weber, proved:

**Theorem 1.3.** *The following are equivalent for a c.e. degree* $\mathbf{d}$*:*

*(a)* $\mathbf{d}$ *bounds a critical triple in the c.e. degrees;*
*(b)* $\mathbf{d}$ *bounds a weak critical triple in the c.e. degrees;*
*(c)* $\mathbf{d}$ *is not totally $\omega$-c.a.*

Note that this theorem shows that the totally $\omega$-c.a. degrees are naturally definable in the c.e. degrees.

In this book we show another equivalence, characterising the dynamics of an existing construction. It considers presentations of left-c.e. reals in the unit interval $[0, 1]$. A real is left-c.e. if the left cut it defines in the rationals is c.e. These reals are the measures of effectively open subsets of Cantor space; equivalently, each such real equals the sum $\sum_{\sigma \in A} 2^{-|\sigma|}$ for some prefix-free c.e. set

For general queries, contact webmaster@press.princeton.edu

© Copyright, Princeton University Press. No part of this book may be distributed, posted, or reproduced in any form by digital or mechanical means without prior written permission of the publisher.



$A \subset 2^{<\omega}$. Such a set $A$ is called a *presentation* of the sum, and is always computable from the sum. However, presentations can be simpler than the sum; in fact, every left-c.e. real has a computable presentation, even though the left-c.e. real itself may be noncomputable. The question is whether we can always code the complexity of a left-c.e. real into one of its presentations. In [32], Downey and LaForte answered this question negatively in a strong way: they constructed a noncomputable left-c.e. real, all of whose presentations are computable. The dynamics of coding complexity into presentations are captured by the totally $\omega$-c.a. degrees:

**Theorem 1.4.**

*(1) If a c.e. degree* $\mathbf{d}$ *is not totally* $\omega$*-c.a. then there is a left-c.e. real* $\varrho \leqslant_{\mathrm{T}} \mathbf{d}$ *and a c.e. set* $B <_{\mathrm{T}} \varrho$ *such that every presentation of* $\varrho$ *is* $B$*-computable.*
*(2) If a left-c.e. real* $\varrho$ *has totally* $\omega$*-c.a. degree then there is a presentation of* $\varrho$ *which is Turing equivalent to* $\varrho$*.*

For more background and details see Chapter 5, where we prove Theorem 1.4.

After our results were announced, Barmpalias and the authors [7] obtained yet another construction whose dynamics are captured by this class. Their results concern the interaction of Turing and weak truth-table reducibility. They showed that a c.e. degree is totally $\omega$-c.a. if and only if every set in that degree is weak truth-table reducible to a ranked set (equivalently, to a hyperimmune set, or to a proper initial segment of a computable, scattered linear ordering). In further work, Brodhead, Downey, and Ng [9] showed that the totally $\omega$-c.a. degrees capture a finite form of randomness.

Also, Adam Day [17] proved that a c.e. degree bounding a generic set which could compute an indifferent subset for itself cannot be totally $\omega$-c.a. In his Ph.D. thesis, McInerney [71] has established similar results relating "multiple genericity" and "integer valued martingales" to being totally $\omega$-c.a.

In the same way that array computability has become a central area of computability theory and its applications, we are confident that once researchers become sensitized to the combinatorics involving the notion of total $\omega$-c.a.-ness, many further applications will be found.

### 1.3.2 Totally $< \omega^\omega$-c.a. degrees

As mentioned, contrary to expectation, we show in this monograph that in the c.e. degrees, bounding critical triples is not equivalent to bounding the 1-3-1 lattice. Very roughly speaking, the "continuous tracing" used in the embedding of the 1-3-1 lattice requires layers over layers of permitting. We now describe the dynamics of the construction, without connecting them to the requirements; more details will be given in Chapter 7.

For general queries, contact webmaster@press.princeton.edu

© Copyright, Princeton University Press. No part of this book may be distributed, posted, or reproduced in any form by digital or mechanical means without prior written permission of the publisher.



The basic cycle in the construction of a critical triple goes as follows. A requirement starts defining a sequence $x_0, x_1, x_2, \ldots$ of numbers which it may want to enumerate into a c.e. set that we are building. At each stage $s$, we choose another number $x_s$ and add it to the list. Then, possibly, at some stage $t$, a primary $\Sigma_1$ event happens (the realisation of a follower), and we want to enumerate these numbers into the sets, starting with $x_t$ and working backwards. For each such number we need to wait for a secondary $\Sigma_1$ event (a new length of agreement of a minimal pair requirement). The requirement is met when the first number $x_0$ is enumerated. In a permitting argument, each such enumeration needs permission, so to meet the requirement we need $t$ many permissions. This kind of permitting is precisely the kind given by non-totally $\omega$-c.a. degrees.

In the 1-3-1 embedding, though, the number of minimal pair requirements stronger than the one we are looking at makes the process more complicated. If there is just one such requirement to contend with, the behaviour is just like the critical triple embedding. If there are two, though, the process is as follows:

(a) Define a sequence $x_0, x_1, x_2, \ldots$, adding a new number at each stage.
(b) When the primary $\Sigma_1$ event happens, start with $x_t$, and repeat the following $t$ times:
  (i) If we are currently dealing with $x_j$ (for $j \leqslant t$), start appointing a sequence $y_0^j, y_1^j, y_2^j, \ldots$, adding a new number at each stage.
  (ii) When a secondary $\Sigma_1$ event happens at some stage $s = s^j$, say, we start enumerating the numbers $y_s^j, y_{s-1}^j, \ldots$, each time waiting for some tertiary $\Sigma_1$ event.
  (iii) When all numbers $y_i^j$ for $i < s^j$ have been enumerated, we also enumerate $x_j$, and repeat the cycle with $x_{j-1}$. If $j = 0$, the requirement is met.

When dealing with three minimal pair requirements, we add a layer:

(a) Define a sequence $x_1, x_2, \ldots$, adding a new number at each stage.
(b) When the primary $\Sigma_1$ event happens, start with $x_t$, and repeat the following $t$ times:
  (i) If we are currently dealing with $x_j$, start appointing a sequence $y_0^j, y_1^j, y_2^j, \ldots$, adding a new number at each stage.
  (ii) When a secondary $\Sigma_1$ event happens at some stage $s = s^j$, start with $y_s^j$, and repeat the following $s$ times:
    (1) If we are currently dealing with $y_i^j$ (for $i < s^j$), we appoint a sequence $z_0^{j,i}, z_1^{j,i}, z_2^{j,i}, \ldots$, adding a new number at each stage.
    (2) When a tertiary $\Sigma_1$ event happens, at some stage $r = r^{j,i}$, we start enumerating the numbers $z_r^{j,i}, z_{r-1}^{j,i}, \ldots$, each time waiting for a quaternary $\Sigma_1$ event.

For general queries, contact webmaster@press.princeton.edu

© Copyright, Princeton University Press. No part of this book may be distributed, posted, or reproduced in any form by digital or mechanical means without prior written permission of the publisher.



(3) When all numbers $z_k^{j,i}$ have been enumerated, we also enumerate $y_i^j$, and repeat the cycle with $y_{i-1}^j$. If $i=0$ then we exit this cycle.

(iii) We enumerate $x_j$; we repeat the outer cycle with $x_{j-1}$. If $j=0$, the requirement is met.

How many permissions are needed to meet the requirement? With two minimal pair requirements constraining us, we need $t+s^0+s^1+\cdots+s^t$ many permissions; with three, we need

$$\begin{aligned} &t+\\ &s^0+r^{0,0}+r^{0,1}+r^{0,2}+\cdots+r^{0,s^0}+\\ &s^1+r^{1,0}+r^{1,1}+r^{1,2}+\cdots+r^{0,s^1}+\\ &\vdots\\ &s^t+r^{t,0}+r^{t,1}+r^{t,2}+\cdots+r^{t,s^t}. \end{aligned}$$

We come now to the key insight. The real question is not how many permissions are required, but what is the reason that the process of meeting a requirement requires only finitely many steps. And the answer to the latter question is that we can attach a transfinite ordinal number to the process, and count down the ordinal along with the steps. With two minimal pair requirements, we start with the ordinal $\omega^2$. When stage $t$ is discovered, we go down to $\omega(t+1)$. When $s^t$ is discovered, we descend to $\omega t+s^t$, and then decrease by 1 each time we enumerate another $y_i^t$. When $y_0^t$ is enumerated, we are at $\omega t$; when $s^{t-1}$ is discovered, we go down to $\omega(t-1)+s^{t-1}$, and repeat. When three minimal pair requirements are present, we need to start at $\omega^3$; then we go down to $\omega^2(t+1)$, then $\omega^2 t+\omega(s^t+1)$, then $\omega^2 t+\omega s^t+r^{t,s^t}$, then decrease by 1 each time some $z_k^{t,s^t}$ is enumerated, and so on. Each time an inner cycle is finished (we enumerate some $y_i^j$) we go past some multiple of $\omega$; each time an outer cycle is finished (we enumerate some $x_j$) we go past some multiple of $\omega^2$.

In terms of permitting, the corresponding notion comes from Ershov's hierarchy of $\Delta_2^0$ functions. We give exact details in Chapter 2, but informally, for a computable ordinal $\alpha$, an *$\alpha$-computable approximation* is a computable approximation $\langle g_s\rangle$ of a $\Delta_2^0$ function $g$ equipped with a counting down $\alpha$ which witnesses the fact that $g_s(n)$ changes only finitely many times: it is a uniformly computable sequence $\langle o_s\rangle$ of functions from $\mathbb{N}$ to $\alpha$ such that for all $n$, $o_s(n)<\alpha$, $o_{s+1}(n)\leqslant o_s(n)$, and if $g_{s+1}(n)\neq g_s(n)$ then $o_{s+1}(n)<o_s(n)$. The function $g$ is called *$\alpha$-computably approximable*, or $\alpha$-c.a. Note that for $\alpha=\omega$ the notion coincides with the definition above. We thus see that in the 1-3-1 embedding, very roughly, to meet a requirement which has to contend with $n$ stronger minimal pair requirements, we need permission from a function which is not $\omega^n$-c.a. Thus we define:

For general queries, contact webmaster@press.princeton.edu

© Copyright, Princeton University Press. No part of this book may be distributed, posted, or reproduced in any form by digital or mechanical means without prior written permission of the publisher.



**Definition 1.5.** A c.e. degree is *totally* $< \omega^\omega$*-c.a.* if every function in it is $\omega^n$-c.a. for some $n$.

And the main theorem in this monograph, which realises the intuitive description above, is:

**Theorem 1.6.** *A c.e. degree bounds a copy of the 1-3-1 lattice if and only if it is not totally* $< \omega^\omega$*-c.a.*

Note that, as above, Theorem 1.6 shows that the class of totally $< \omega^\omega$-c.a. degrees is naturally definable in the c.e. degrees.

*Nonuniform anti-permitting arguments*

When we show that a class of degrees captures the dynamics of a construction (such as we do in Theorems 1.3 and 1.6) the argument has two parts: a permitting argument, which shows that the construction can be performed below a degree which permits accordingly; and an *anti-permitting* argument, which shows the converse. The latter is not a priority argument; we usually have different attempts at constructing objects which give that direction of the theorem, but these have very little interaction with each other. On the other hand, there is a certain nonuniformity to the construction, in that one of the attempts will succeed, but we cannot computably tell which. In the case of totally $< \omega^\omega$-c.a. degrees, we have $\omega$ levels of nonuniformity, which means that even though no injury occurs, only the oracle $\varnothing^{(\omega)}$ can tell which of the constructions we performed actually succeeds. This kind of argument, which we hinted at in [23], is presented in this monograph (in Chapters 6 and 7) in full for the first time. We believe that it will have wider applications.

### 1.3.3 The hierarchy of totally $\alpha$-c.a. degrees

We have characterised the degrees which bound critical triples and degrees which bound a copy of the 1-3-1 lattice; but we have not yet argued that these classes are distinct, that is, that there is a degree which bounds a critical triple but not a copy of the 1-3-1. This will come out of a general investigation into a hierarchy of classes of degrees. The two classes under discussion are two levels of this hierarchy.

Armed with the definition of $\alpha$-c.a. functions (which, as discussed, will require clarification, which we give in Chapter 2), we can extend the definitions above and define a degree to be *totally* $\alpha$*-c.a.* if every function in it is $\alpha$-c.a.; and more generally, to be *totally* $< \alpha$*-c.a.* if every function in it is $\beta$-c.a. for some $\beta < \alpha$. All such degrees are $\mathrm{low}_2$. In the first part of this monograph, we give a detailed investigation of these classes, and in particular we find which are the proper levels of the hierarchy. For example, we show:

For general queries, contact webmaster@press.princeton.edu

© Copyright, Princeton University Press. No part of this book may be distributed, posted, or reproduced in any form by digital or mechanical means without prior written permission of the publisher.



- there is a totally $\alpha$-c.a. degree which is not totally $\beta$-c.a. for any $\beta < \alpha$ if and only if $\alpha$ is a power of $\omega$;
- there is a totally $< \alpha$-c.a. degree which is not totally $< \beta$-c.a. for any $\beta < \alpha$ if and only if $\alpha$ is a limit of powers of $\omega$.

This, in particular, shows that there are $\omega$ many distinct levels between the totally $\omega$-c.a. degrees and the totally $< \omega^\omega$-c.a. degrees.

## 1.4 THE CONTENTS OF THIS MONOGRAPH

In the first part of the monograph, we introduce and investigate our new hierarchy.

In Chapter 2, we give a rigorous treatment of the notion of $\alpha$-c.a. functions. The main issue is to properly define what we mean by a computable function $o$ from $\mathbb{N}$ to $\alpha$, which is required for the definition of $\alpha$-computable approximations. Naturally, to deal with an ordinal $\alpha$ computably, we need a notation for this ordinal, or more generally, a computable well-ordering of order-type $\alpha$. To form the basis of a solid hierarchy, the notion of $\alpha$-c.a. should not depend on which well-ordering we take, rather it should only depend on its order-type. Thus we cannot consider all computable copies of $\alpha$. Rather, we restrict ourselves to a class of particularly well-behaved well-orderings, in a way that ensures that they are all computably isomorphic. For example, when considering copies of $\omega^2$, we must compute not only the collection of limit points and the successor function, but we also need to know which copy of $\omega$ inside $\omega^2$ is which. In general, we need the Cantor normal form to be computable. This turns out to be sufficient for small enough ordinals; we develop the theory for ordinals $\alpha \leqslant \varepsilon_0$. The theory can be pushed further, but not all the way up to $\omega_1^{CK}$; we do not pursue such extensions here.

Having defined $\alpha$-c.a. functions, we also (in Section 2.3) relate these functions to iterations of the bounded jump (the jump inside the weak truth-table degrees). This extends and solidifies work by Coles, Downey, and LaForte [15], and independently Anderson and Csima [3]. Extending the familiar result for $\omega$, we show (Theorem 2.40) that a function is $\omega^\alpha$-c.a. if and only if it is weak truth-table reducible to the $\alpha^{\text{th}}$ iteration of the bounded *function* jump; an analogous result holds for sets.

In Chapter 3 we investigate the hierarchy of totally $\alpha$-c.a. degrees. As mentioned above, we show precisely when this hierarchy collapses (Theorem 3.6), and refine this hierarchy when we consider totally $< \alpha$-c.a. degrees (Theorem 3.25). We further consider *uniform* versions of our classes. Recall that the array computable degrees were a uniform version of the totally $\omega$-c.a. degrees, in that we took a single computable bound on the mind-change function of approximations of functions in the degree. We find the right formulation that generalises this to define *uniformly totally $\alpha$-c.a. degrees*, and show (Theorem 3.20) how they fit in our hierarchy. For a general picture, see Figure 3.3.

For general queries, contact webmaster@press.princeton.edu

© Copyright, Princeton University Press. No part of this book may be distributed, posted, or reproduced in any form by digital or mechanical means without prior written permission of the publisher.



### 1.4.1 Maximality

It is not common to find maximal elements of classes in the c.e. degrees; usually, density prevails. However, in Chapter 4 we show that at every level of our main hierarchy there are maximal degrees (Theorem 4.1). Thus, for example, there are maximal degrees with respect to not bounding a critical triple, namely, maximal totally $\omega$-c.a. degrees. Since the totally $\omega$-c.a. degrees are naturally definable, we obtain a naturally definable antichain in the c.e. degrees; the only previously known such antichain consisted of the maximal contiguous degrees (Cholak, Downey, and Walk [14]).

On the other hand, we show (Theorem 4.12) that maximality cannot go too far, that is, to the next level. For example, no totally $\omega$-c.a. degree can be maximal totally $\omega^2$-c.a. A corollary of the argument shows that there are no maximal totally $< \omega^\omega$-c.a. degrees, that is, no degrees maximal with respect to not bounding a 1-3-1.

We remark that in further work with Katherine Arthur [5] we investigate bounding by maximal degrees. For example, there are totally $\omega$-c.a. degrees bounded by no such maximal degrees. The general picture is interesting. We suspect that, in general, the following holds:

- Let $\alpha \leqslant \beta \leqslant \varepsilon_0$ be powers of $\omega$. Then every totally $\alpha$-c.a. degree is bounded by a maximal totally $\beta$-c.a. degree if and only if $\beta \geqslant \alpha^\omega$.

Further questions consider collapse of our hierarchy in upper cones. Theorem 4.12 implies that every totally $\omega$-c.a. degree is bounded by a strictly greater degree which is totally $\omega^2$-c.a. However we do not know if we can always make that degree not totally $\omega$-c.a. The best result so far, which appears in [5], implies that every totally $\omega$-c.a. degree is bounded by a totally $\omega^4$-c.a. degree which is not totally $\omega$-c.a. Is it $\omega^2$, or $\omega^3$? We cannot yet tell.

### 1.4.2 Calibrating the dynamics of constructions

The second part of the monograph consists of Chapters 5, 6, and 7, in which we discuss and calibrate the dynamics of three different constructions. In Chapter 5 we prove Theorem 1.4 about presentations of left-c.e. reals. In Chapter 7 we prove our main Theorem 1.6. In Chapter 6 we consider *m-topped degrees*, continuing [23]. The notion of $m$-topped degrees comes from a general study of the interaction between Turing reducibility and stronger reducibilities among c.e. sets. For example, this study includes the contiguous degrees. A c.e. Turing degree $\mathbf{d}$ is $m$-topped if it contains a greatest degree among the many-one degrees of c.e. sets in $\mathbf{d}$. Such degrees (other than $\mathbf{0}'$) were constructed in Downey and Jockusch [29]. They are all $\mathrm{low}_2$. In [23] we showed that there are totally $\omega^\omega$-c.a. $m$-topped degrees. Here we show that this is the best possible: no $m$-topped degree is totally $< \omega^\omega$-c.a. (Theorem 6.1). We remark though that in this case we cannot hope to get full equivalence: we cannot prove that every degree which is not totally $< \omega^\omega$-c.a. bounds an $m$-topped degree. This is because $m$-topped

For general queries, contact webmaster@press.princeton.edu

© Copyright, Princeton University Press. No part of this book may be distributed, posted, or reproduced in any form by digital or mechanical means without prior written permission of the publisher.



degrees cannot be low, whereas every level of our hierarchy contains both low degrees and degrees which are $\text{low}_2$ but not low.

### 1.4.3 Promptness

One can ask, regarding the embedding of the 1-3-1 lattice, what it would take to get an embedding preserving the bottom, that is, an embedding whose bottom degree is $\mathbf{0}$ (as is obtained in Lachlan's original construction). We discuss this in Chapter 8, where we introduce prompt versions of all levels in our hierarchy. This generalises the already familiar notion of prompt permitting, which is the prompt version of simple permitting. Prompt array noncomputable permission, for example, allows us to construct a pair of separating classes whose elements form minimal pairs (Theorem 8.22); whereas traditional (non-prompt) array noncomputable permission only gives Turing incomparability [30]. Similarly, a degree which is promptly not totally $<\omega^\omega$-c.a. bounds a copy of the 1-3-1 lattice with bottom $\mathbf{0}$.

This however cannot be reversed: every high degree bounds a copy of the 1-3-1 lattice with bottom $\mathbf{0}$, and there are high degrees which are not promptly simple (let alone promptly non-totally $<\omega^\omega$-c.a.). Informally what this says is that there are at least two ways to get such an embedding: either by quickly getting the precise number of permissions required; or by getting many permissions (cofinitely many), in which case we can wait for the permissions and don't need them promptly.

It would be interesting to find a common generalisation.

## 1.5 AN APPLICATION TO ADMISSIBLE COMPUTABILITY

Combined with results of the second author, our work has an application to admissible computability. This is a generalisation of traditional computability to ordinals beyond $\omega$. In [46] it is shown that for any admissible ordinal $\alpha$, the $\alpha$-c.e. degrees are not elementarily equivalent to the c.e. degrees. This was done in cases, depending on the proximity of $\alpha$ to $\omega$. In one case the separation between the theories is not natural but relies on coding models of arithmetic. However one result is:

**Theorem 1.7** ([46])**.** *Let $\alpha > \omega$ be an admissible ordinal, and let* $\mathbf{a}$ *be an incomplete $\alpha$-c.e. degree. The following are equivalent:*

*(1)* $\mathbf{a}$ *computes a cofinal $\omega$-sequence in $\alpha$.*
*(2)* $\mathbf{a}$ *bounds a copy of the 1-3-1 lattice.*
*(3)* $\mathbf{a}$ *bounds a critical triple.*

Again, it is the analysis of continuous tracing that underlies this result. The basic idea is the following. Consider again the dynamic aspect of the embedding

For general queries, contact webmaster@press.princeton.edu

© Copyright, Princeton University Press. No part of this book may be distributed, posted, or reproduced in any form by digital or mechanical means without prior written permission of the publisher.



of a critical triple which we discussed above. We start by appointing elements $x_0, x_1, x_2, \ldots$, adding one at each stage. When the primary $\Sigma_1$ event happens (the follower is realised), it is important (because of use considerations) that we attempt to enumerate the elements $x_j$ *starting with the last number* $x_t$ and working backwards.

Trying to do this when time goes beyond $\omega$ presents a completely new problem: after $\omega$ many stages, we will have elements $x_j$ for all $j < \omega$, that is, we will not have a last element. We cannot then peel it back, each step removing only the last element. It turns out that this blockage is fundamental. The only case it might be possible for a degree $\mathbf{a}$ to bound a critical triple is if it itself can see that $\alpha$ is far from being a regular cardinal—if it can essentially re-order time and space to order-type $\omega$, so that the construction can be (at least after the fact) seen to have taken $\omega$ steps, avoiding infinite sequences of numbers. In one direction, effectively closed and unbounded sets are used to show that this is necessary. In the other direction, a fine-structural result of Shore's [87] says that an incomplete degree of computable cofinality $\omega$ must be high, and can compute a bijection between $\alpha$ and $\omega$. Working below such a degree, we can translate back to $\omega$-computability, and use non-low$_2$ permitting to embed the 1-3-1 lattice (for a technical reason, we cannot quite use high permitting).

To sum, what this says is that once we go beyond $\omega$, the fine distinctions between totally $\omega$-c.a. degrees and totally $< \omega^\omega$-c.a. degrees completely disappear. Combined with the current work, this gives us a single, natural sentence which separates the elementary theory of the c.e. degrees from the theory of the $\alpha$-c.a. degrees for any admissible $\alpha > \omega$.

**Theorem 1.8.** *Let $\alpha \geqslant \omega$ be admissible. The following are equivalent:*

*(1) There is an incomplete $\alpha$-c.e. degree which bounds a critical triple but not the 1-3-1 lattice.*
*(2) $\alpha = \omega$.*

## 1.6 NOTATION AND GENERAL DEFINITIONS

We recap some notions that we discussed above, and introduce terminology and conventions that will be used throughout the monograph. First, though, we comment on the expected mathematical background a reader will need. We assume that the reader has mastered the basics of computability theory, up to and including basic finite-injury priority arguments, in particular the Friedberg-Muchnik theorem, and basic infinite-injury priority constructions, mainly the construction of a minimal pair of c.e. degrees, as performed on a priority tree. For years, the standard reference in this area has been Soare's [91]. Other possible sources are the second chapter of [27], the first chapter of [74], Cooper's [16], Odifreddi's [77], or Steffen Lempp's unpublished notes on priority arguments in computability theory, available on his website. We also assume some basic

For general queries, contact webmaster@press.princeton.edu

© Copyright, Princeton University Press. No part of this book may be distributed, posted, or reproduced in any form by digital or mechanical means without prior written permission of the publisher.



information on ordinals and ordinal arithmetic; any standard set theory text would be more than sufficient.

### 1.6.1 Computable approximations and enumerations

A *computable approximation* for a function $f\colon \omega \to \omega$ is a uniformly computable sequence $\langle f_s \rangle_{s<\omega}$ of functions such that for all $x$, for almost all $s$, $f_s(x) = f(x)$. In other words, $f = \lim_s f_s$ when we equip $\omega$ with the discrete topology. Shoenfield's limit lemma [84] states that a function $f$ is $\Delta^0_2$-definable if and only if it is computable from the halting set $\varnothing'$ if and only if it has a computable approximation. If $A$ is a *set* (a subset of $\omega$, identified with an element of Cantor space) then a computable approximation of $A$ is a sequence of sets.

A *computable enumeration* of a c.e. set $A$ is a computable, $\subseteq$-increasing sequence of finite sets $\langle A_s \rangle$ such that $A = \bigcup_s A_s$. We can also think of a computable enumeration as a computable approximation of $A$, again by taking characteristic functions. We say that a number $x$ is *enumerated into* $A_s$ if $x \in A_s \setminus A_{s-1}$.

### 1.6.2 Turing functionals

A (Turing) *functional* is a c.e. set of triples $\langle \sigma, x, y \rangle$ consisting of a finite sequence $\sigma$ of natural numbers and a pair of natural numbers $x$ and $y$. We consider such triples as *axioms*, and sometimes write them as $\sigma \mapsto (x, y)$. If $f\colon \omega \to \omega$ and $\Phi$ is a functional, then we define the multi-valued function (i.e., relation) $\Phi(f) \subseteq \omega \times \omega$ by letting $\Phi(f, x) = y$ if there is some finite $\sigma \prec f$ such that the axiom $\sigma \mapsto (x, y)$ is in $\Phi$. We write $\Phi(f, x)\!\downarrow$ for $x \in \operatorname{dom} \Phi(f)$ and $\Phi(f, x)\!\uparrow$ for $x \notin \operatorname{dom} \Phi(f)$.

In general we allow functionals, especially the ones that we build, to be *inconsistent*. That is, we allow them to contain contradictory axioms: a pair of axioms $\sigma \mapsto (x, y)$ and $\tau \mapsto (z, w)$ such that $\sigma$ and $\tau$ are comparable (that means that $\sigma \preccurlyeq \tau$ or $\tau \preccurlyeq \sigma$), $x = z$ but $y \neq w$. A functional $\Phi$ is called *consistent relative to an oracle* $f$ if $\Phi(f)$ is a partial function, i.e., is not multi-valued. A functional is consistent if and only if it is consistent relative to every oracle.

The following are equivalent for $f, g\colon \omega \to \omega$:

(1) there is a consistent functional $\Phi$ such that $\Phi(f) = g$;
(2) there is a functional $\Phi$, consistent relative to $f$, such that $\Phi(f) = g$;
(3) $g \leqslant_{\mathrm{T}} f$.

If $\langle \Phi_s \rangle$ is a computable enumeration of a functional $\Phi$, then each $\Phi_s$ is also a functional. If $\langle f_s \rangle$ is a computable approximation of a function $f\colon \omega \to \omega$, then the finite multi-valued function $\Phi_s(f_s)$ can be effectively obtained from $s$. If for all $s$, $\Phi_s$ is consistent relative to $f_s$, then $\Phi$ is consistent relative to $f$. Note that if, further, $\Phi(f)$ is a total function, then we can extend $\langle \Phi_s(f_s) \rangle$ to a computable approximation of $\Phi(f)$, since $\langle \operatorname{dom} \Phi_s(f_s) \rangle$ is uniformly computable. When the notation $\Phi_s(f_s)$ becomes unwieldy, we sometimes write $\Phi(f)[s]$, and in general may use Lachlan's square bracket notation.

For general queries, contact webmaster@press.princeton.edu

© Copyright, Princeton University Press. No part of this book may be distributed, posted, or reproduced in any form by digital or mechanical means without prior written permission of the publisher.



Suppose that $\Phi$ is a functional which is consistent relative to an oracle $f$. If $x \in \operatorname{dom} \Phi(f)$, we also refer to $\Phi(f, x) = y$ as a "computation." Let $\sigma$ be the shortest initial segment of $f$ for which $\sigma \mapsto (x, y)$ is an axiom in $\Phi$. Often in fact there will be a unique such initial segment. The string $\sigma$ determines the *use* of the computation, denoted by $\varphi(f, x)$ (and when $f$ is clear from the context, by $\varphi(x)$). We will use two conflicting notions:

- If either $f$ or $\Phi$ are given, then the use is the length of $\sigma$.
- If both $f$ and $\Phi$ are built by us then we let the use be $|\sigma| - 1$, the "greatest number queried during the computation." In this case $f$ is usually a c.e. set $A$. The idea is that we may want to void the computation by enumerating the use $\varphi(x)$ into $A$.

If $\langle \Phi_s \rangle$ is a computable enumeration of a Turing functional $\Phi$, and $\langle f_s \rangle$ is a computable approximation of a function $f$ (and again we assume that for all $s$, $\Phi_s$ is consistent relative to $f_s$), $s < \omega$ and $x \in \operatorname{dom} \Phi_s(f_s)$, then we say that the computation $\Phi_s(f_s, x)$ is *destroyed* (or *injured*) at stage $s + 1$ if $\sigma \not\prec f_{s+1}$, where $\sigma$ as above is the shortest axiom applying to $f$ giving the computation at stage $s$. That is, if $f_{s+1}{\restriction_u} \neq f_s{\restriction_u}$ where $u = \varphi_s(f_s, x)$ is the use of the computation, in the case in which either $f$ or $\Phi$ are given; if both are built by us, then the computation is destroyed if $f_s{\restriction_{u+1}} \neq f_{s+1}{\restriction_{u+1}}$, and as described above, this will often happen because we enumerate $u$ into $f_{s+1}$.

In contrast, we say that a computation $\Phi_s(f_s, x) = y$ is *$f$-correct* if $\sigma \prec f$. The fundamental fact about Turing computations, used without mention throughout computability theory, is that $x \in \operatorname{dom} \Phi(f)$ if and only if there is a stage $s$ (equivalently, for almost all stages $s$) such that $x \in \operatorname{dom} \Phi_s(f_s)$ by an $f$-correct computation. When working with c.e. sets we often use the fact that correct computations never go away: if $\langle A_s \rangle$ is a computable enumeration of a c.e. set $A$, and $\Phi_s(A_s, x)$ is an $A$-correct computation, then for all $t \geqslant s$, $x \in \operatorname{dom} \Phi_t(A_t)$ by the same computation.

The following lemma is used when we build functionals which apply to c.e. sets that we enumerate.

**Lemma 1.9.** *Let $\langle \Phi_s \rangle$ be a computable enumeration of a functional $\Phi$, and let $\langle A_s \rangle$ be a computable enumeration of a c.e. set $A$. Suppose that for all $s$,*

*(1) if an axiom $\sigma \mapsto (x, y)$ is enumerated into $\Phi_s$, then $\sigma \prec A_s$;*
*(2) for each $x$, at most one axiom $\sigma \mapsto (x, y)$ is enumerated into $\Phi_s$.*

*Let $s < \omega$, and suppose that $\Phi_s$ is consistent for $A_s$. Suppose that for all $x < \omega$,*

*(3) if an axiom $\sigma \mapsto (x, y)$ is enumerated into $\Phi_{s+1}$, and $x \in \operatorname{dom} \Phi_s(A_s)$, then some number $u \leqslant \varphi_s(A_s, x)$ is enumerated into $A_{s+1}$.*

*Then $\Phi_{s+1}$ is consistent for $A_{s+1}$.*

Hence if conditions (1)–(3) hold at every stage $s$, then $\Phi$ is consistent for $A$. Note that usually $\Phi$ will not be consistent for all oracles: we could void a

For general queries, contact webmaster@press.princeton.edu

© Copyright, Princeton University Press. No part of this book may be distributed, posted, or reproduced in any form by digital or mechanical means without prior written permission of the publisher.



computation $\Phi_s(A_s, x)$ by enumerating $u = \varphi_s(A_s, x)$ into $A_{s+1}$, and then define a new computation $\Phi_{s+1}(A_{s+1}, x)$ with smaller use, so $\Phi_{s+1}$ may be inconsistent for $A_s$.

**Convention 1.10.** We often assume that for a given consistent functional $\Phi$, for any oracle $f$, $\operatorname{dom}\Phi(f)$ is an initial segment of $\omega$. That is, we require that if $\sigma \mapsto (x, y)$ is in $\Phi$, then for all $x' < x$ there is some $\sigma' \preccurlyeq \sigma$ and some $y'$ such that $\sigma' \mapsto (x', y')$ is also in $\Phi$. We simply prevent $\sigma \mapsto (x, y)$ from entering $\Phi$ until we see the other necessary axioms.

In this situation we also assume that if $\langle \Phi_s \rangle$ is a computable enumeration of a Turing functional $\Phi$, then for all $s$ and $f$, $\operatorname{dom}\Phi_s(f)$ is an initial segment of $\omega$.

The point is that if we are only interested in *total* functions computable from an oracle $f$, then we can restrict ourselves to functionals of the type described.

We let $\langle \Phi_e \rangle$ be some enumeration of all *consistent* functionals; associated with which we are given uniformly computable enumerations $\langle \Phi_{e,s} \rangle$ of $\Phi_e$.

**Convention 1.11.** We sometimes identify natural numbers with the von Neumann ordinals isomorphic to them; that is, we identify the natural number $n$ with the set $\{0, 1, 2, \ldots, n-1\}$. In particular, if for some functional $\Phi$ and oracle $f$, $\operatorname{dom}\Phi(f)$ is an initial segment of $\omega$ (per Convention 1.10), then we write $x < \operatorname{dom}\Phi(f)$ for $x \in \operatorname{dom}\Phi(f)$, and $x \leqslant \operatorname{dom}\Phi(f)$ for $\{0, 1, \ldots, x-1\} \subseteq \operatorname{dom}\Phi(f)$.

Functionals which take more than one oracle are treated in a similar fashion. For example, when taking two oracles, axioms will be of the form $(\sigma, \tau) \mapsto (x, y)$. Usually, for a pair of oracles $f, g$ in which we are interested, for each $x$ there will be at most one pair of strings $\sigma \prec f$ and $\tau \prec g$ such that $(\sigma, \tau) \mapsto (x, y)$ is in the functional $\Phi$ we are building or examining. These determine the *$f$-use* and the *$g$-use* of the computation $\Phi(f, g, x)$, according to the notational convention discussed above. When $\Phi$ is not built by us we often assume that the $f$-use and the $g$-use are the same, and that common value is referred to simply as the use $\varphi(f, g, x)$ of the computation.

### 1.6.3 Priority arguments and tree constructions

In our constructions we keep the *convention of small numbers*.

**Convention 1.12.** At stage $s$ of a construction, all numbers played by the "opponent" are bounded by $s$. These are the values of functions that are not defined by us during the construction.

On the other hand, the constructions would often call on us to define new values for functions that are *large*. This means that the new values are picked to be numbers that are larger than any other number previously used or observed in the construction, including the stage number.

For general queries, contact webmaster@press.princeton.edu

© Copyright, Princeton University Press. No part of this book may be distributed, posted, or reproduced in any form by digital or mechanical means without prior written permission of the publisher.



Most terminology we will use in priority constructions is common. We will attempt to meet *requirements*. *Positive requirements* are those which can be met by enumerating numbers into c.e. sets we are enumerating. *Negative requirements* are met by imposing *restraint* on other actors. The numbers enumerated into the c.e. sets are sometimes called *followers*. In the standard Friedberg-Muchnik construction, for example, a requirement attempting to ensure that $\Phi(A) \neq B$ will *appoint* a follower $x$, which means choose some number $x$ (that will not be used by any other requirement), wait until we see that $\Phi(A, x)\downarrow = 0$, and then enumerate it into $B$. The prototypical negative requirements, on the other hand, are met in the Lachlan-Yates minimal pair construction. In most of our constructions, restraint will be imposed by *initialising* other requirements. Typically, initialising a positive requirement means that any follower $x$ it appointed is *cancelled*: this means that the number $x$ will not be involved in the construction any longer. Any new follower will be chosen to be large.

*Tree constructions*, namely priority constructions done with the aid of a *tree of strategies*, are now standard; a reference is Chapter XIV of [91]. Elements of the tree are called *strategies*, or *nodes*; these are finite sequences of symbols. To describe the tree of strategies, we give two pieces of information:

(a) An association of requirements for nodes; we say that a node *works for* the requirement associated with it. Often, but not always, all nodes of a given level of the tree work for the same requirement.
(b) For nodes working for some requirement, the list of *outcomes* of these nodes.

The tree is then defined recursively. The empty node is always on the tree of strategies; if a node $\sigma$ has already been determined to lie on the tree of strategies, and a requirement $R$ has been associated with it, then the immediate successors of $\sigma$ on the tree are the nodes of the form $\sigma^\frown o$, where $o$ is a possible outcome for nodes working for $R$.

The collection of possible outcomes of any node will be linearly ordered; we say that an outcome $o$ is *stronger* than an outcome $o'$ if $o < o'$. This ordering induces a linear ordering of the tree of strategies, by taking a lexicographic amalgamation of the orderings of outcomes: $\sigma < \tau$ if $\sigma \prec \tau$, or if there are $\eta$, $o$ and $o'$ such that $\sigma \succcurlyeq \eta^\frown o$, $\tau \succcurlyeq \eta^\frown o'$, and $o < o'$. We say that a node $\sigma$ is *stronger* than a node $\tau$ if $\sigma < \tau$, and that a node $\sigma$ *lies to the left* of a node $\tau$ if $\sigma < \tau$ but $\sigma \not\succ \tau$. We sometimes write $\sigma <_L \tau$; this has nothing to do with the constructible universe.

At any stage $s$, the construction describes the (finite) collection $\delta_s$ of nodes that are *accessible* at stage $s$. In our constructions this will always be an initial segment of the tree of strategies, linearly ordered by extension of nodes. We will not use constructions with links. Usually, the empty node $\langle\rangle$ is accessible at every stage.

We then say that a node $\sigma$ lies on the *true path* $\delta_\omega$ if there are infinitely many stages $s$ of the construction such that $\sigma \in \delta_s$ (that is, such that $\sigma$ is accessible at stage $s$), but the same is not true for any node $\tau$ that lies to the left of $\sigma$. The

For general queries, contact webmaster@press.princeton.edu

© Copyright, Princeton University Press. No part of this book may be distributed, posted, or reproduced in any form by digital or mechanical means without prior written permission of the publisher.



true path $\delta_\omega$ will be a linearly ordered initial segment of the tree of strategies. We will need to prove that the true path is infinite.

As with simpler constructions, tree constructions will involve initialisations, this time of nodes rather than of requirements. Again, when a node is initialised, all parameters associated with the node (such as followers) are removed (or *cancelled*), and new ones will have to be defined, either immediately, or more often, at the next stage at which the node is accessible. When a stage ends, every node which lies to the right of an accessible node (a node in $\delta_s$) is initialised. Often, but not always, nodes extending the longest node in $\delta_s$ are also initialised at the end of stage $s$. We ensure that whenever a node $\sigma$ is initialised, and $\tau$ is a node weaker than $\sigma$, then $\tau$ is also initialised at the same time.

We say that the construction is *fair* to a node $\sigma$ if $\sigma$ is initialised only finitely many times (i.e., at only finitely many stages of the construction). The main *fairness lemma* for each construction will state that the construction is fair to every node on the true path $\delta_\omega$. If $\sigma$ is a node on the true path and the construction is not fair to $\sigma$ then there will be some node $\tau \prec \sigma$ on the true path which initialises $\sigma$ at infinitely many stages. This is because initialisation has to respect the priority ordering; no node weaker than $\sigma$ can initialise $\sigma$.

Other standard conventions of priority constructions are employed without mention. For example, we use "stickiness" or "persistence" of parameters: if, for example, a requirement $R$ or strategy $\sigma$ has a follower at some stage $s$, and the requirement or node is not tampered with (e.g., initialised) at stage $s+1$, say, then that follower is still considered to be a follower for the requirement or strategy at stage $s+1$.

For general queries, contact webmaster@press.princeton.edu