Title
Weak Lowness Notions for Kolmogorov Complexity

Permalink
https://escholarship.org/uc/item/0q79m72q

Author
Herbert, Ian-Cadoc Robertson

Publication Date
2013

Peer reviewed|Thesis/dissertation
Weak Lowness Notions for Kolmogorov Complexity

by

Ian-Cadoc Robertson Herbert

A dissertation submitted in partial satisfaction of the requirements for the degree of Doctor of Philosophy in Logic and the Methodology of Science in the Graduate Division of the University of California, Berkeley

Committee in charge:

Professor Theodore Slaman, Chair
Professor Leo Harrington
Professor John Steel
Professor W. Hugh Woodin
Professor George Bergman

Spring 2013
Weak Lowness Notions for Kolmogorov Complexity

Copyright 2013
by
Ian-Cadoc Robertson Herbert
Abstract

Weak Lowness Notions for Kolmogorov Complexity

by

Ian-Cadoc Robertson Herbert

Doctor of Philosophy in Logic and the Methodology of Science

University of California, Berkeley

Professor Theodore Slaman, Chair

The prefix-free Kolmogorov complexity, $K(\sigma)$, of a finite binary string $\sigma$ is the length of the shortest self-delimiting program that outputs $\sigma$ and is a measure of the information content of $\sigma$. There are two very natural ways to extend this notion to infinite binary strings (identified as the binary expansions of real numbers in the interval $[0,1]$): one can examine the initial segment complexity of a real, i.e. $K(A \upharpoonright n)$ as a function of $n$, or one can examine the compressive power of $A$ as an oracle, i.e. $K^A(\sigma)$ as a function of $\sigma$. Each of these approaches has a notion of minimality for reals. A real is $K$-trivial if the complexities of its initial segments are up to a constant just the complexities of their lengths. A real is low for $K$ if it provides no more than an additive constant advantage to compressing any string. This dissertation examines weakenings of these notions that arise from replacing the constant bounds with slow-growing functions. The main results are that these weaker lowness notions behave very differently from the standard ones in terms of the sets of reals defined by these notions. We also include applications of these new notions to effective dimension and mutual information.
For my father, Robert Kevin Herbert.
# Contents

1 Introduction 1
   1.1 Introduction ................................................. 1
   1.2 Definitions and Notation ..................................... 2
   1.3 Building Machines ............................................ 4
   1.4 Lowness Notions .............................................. 5
   1.5 Weak Lowness Notions ........................................ 6
   1.6 Finite-to-one Approximations ................................. 7

2 Initial-Segment Complexity 11
   2.1 A failure of Nies’s Theorem in the weaker case ........... 11
   2.2 Finding a universal witness .................................. 15
   2.3 Downwards closure .......................................... 21
   2.4 Strengthenings .............................................. 23

3 Compressive Power 36
   3.1 A Perfect Set .............................................. 36
   3.2 Examples ................................................ 48
   3.3 A Universal Tree ........................................... 49

4 Mutual Information 62
   4.1 Information Content ........................................ 62
   4.2 Further Directions ........................................ 70

Appendix A: A finite-to-one $\Delta^0_2$ function with no finite-to-one approximation 72

Appendix B: The index set of total, finite-to-one approximations is $\Pi^0_3$-complete 74

Bibliography 75
Acknowledgments

I would like to thank my advisor, Ted Slaman, for his guidance and support. He devoted a lot of his time to meeting with me, even when I didn’t have much to say. I am also grateful to George Bergman for several insightful comments on an earlier draft of this dissertation. Thanks also to Adam Day, Denis Hirschfeldt, Joe Miller, and André Nies for some productive discussions on this material.

A large portion of this dissertation was written while I was participating in the Buenos Aires Semester in Computability, Complexity and Randomness. I would like to thank the organizers of the program for providing an incredible environment for developing and writing down these ideas.

I would also like to thank the many friends who made my graduate experience so enjoyable. Thanks to Kelty Allen, Morgan Brown, Quinn Culver, Taryn Flock, Jeff Galkowski, Amit Gupta, Michael Hartglass, Katrina Honigs, Theo Johnson-Freyd, Mushfeq Khan, Aaron Kleinman, Benoît Monin, Ralph Morrison, Ben Orlin, Luca San Mauro, James Tener, Ivan Ventura, Cynthia Vinzant, and Harold Williams.

Finally, I wish to thank my family for their constant love and support. Ne quis nos impediat.
Chapter 1

Introduction

1.1 Introduction

There is a subtle philosophical challenge in giving a precise mathematical definition of randomness, say in distinguishing random data from non-random data. For example, a sequence of 100 zeroes seems non-random; one would not expect to get such a result when flipping 100 coins (where ‘heads’ is 1 and ‘tails’ is 0). On the other hand, any ‘more random’ sequence with zeroes and ones evenly spread out has the same probability of being generated by 100 coin flips. Looking at infinite binary sequences (identified with the binary expansions of real numbers in the interval [0,1]), the probability of getting any given real is 0, but surely we can pick reals at random and some reals are ‘not random’ and we would never expect to pick them this way. Algorithmic Randomness is a subfield of Recursion Theory that tries to formalize and examine notions of randomness in an effective way. There are different approaches to this problem. Following probabilistic intuition, one can define random reals to be those that are not members of any set of measure 0 that can be effectively presented. Various requirements on the effectivity of the presentation yield different notions of randomness. Alternatively, one can look not at probabilities, but at complexities of reals. A random binary sequence should be one that is ‘disordered’ and thus difficult to describe efficiently. These approaches are not mutually exclusive, and several notions have definitions in both the measure- and information-theoretic senses. Here we will deal mainly with the information-theoretic approach, and so we give a short overview.

The essential idea in this approach is to use Kolmogorov complexity to examine the information content of reals. Intuitively, the prefix-free Kolmogorov complexity of a finite binary string is the length of the shortest self-delimiting program that generates that string, i.e. the length of the most efficient description of that string. There are two ways to extend this notion of complexity for finite strings to a notion on infinite sequences: we can examine the initial segment complexity of the infinite sequence, or we can examine the complexities of finite strings relative to the infinite sequence (used as an oracle in the computations). The random reals are those with the ‘highest possible’ initial segment complexity: the easiest
way to describe them is in fact to read off the bits of their expansion. On the other hand, if one is given oracle access to such a real, describing its initial segments should be easy: one just needs to specify the length of the desired initial segment and then query that many bits of the oracle. In the sense of compressive power, then, any random real should have arbitrarily better compression than the oracle-free compression for a certain set of strings.

We focus not on the random reals defined this way, but on those reals on the other side of the spectrum. Defining randomness using Kolmogorov complexity gives us a notion of being ‘far from random:’ those reals that have the low initial segment complexity or that provide very little compressive power. Each sense has a notion of minimality. The lowest possible initial segment complexity is to be able to describe the initial segment of length \( n \) as easily as one describes the number \( n \). Any string contains at least the information of its length, so it is not possible to do better than this. The lowest possible compressive power is to have no better than a constant advantage over the oracle-free compression over all finite strings. These notions have been well studied for several decades now and this study has produced some very striking and beautiful results. In this thesis I examine weakenings of these minimal notions and how they relate both to each other and to the stronger traditional notions. This chapter contains the basic definitions, notation, and background. Chapters 2 and 3 examine different aspects of the traditional notions and to what extent they apply in the weaker case. Chapter 4 contains a discussion of Mutual Information and an application of some earlier results to this setting.

### 1.2 Definitions and Notation

We use \( \omega \) to denote the least countably infinite ordinal (identified with the natural numbers \( \mathbb{N} \)), \( 2^{<\omega} \) to denote the set of finite binary strings, and \( 2^\omega \) to denote Cantor space, the set of infinite binary strings (identified alternately with binary expansions of real numbers between 0 and 1 and with subsets of \( \omega \)). We use \( \prec \) to denote the initial segment relation on \( 2^{<\omega} \times 2^{<\omega} \) and \( 2^{<\omega} \times 2^\omega \). For a string \( \sigma \in 2^{<\omega} \), we let \( |\sigma| \) be the length of \( \sigma \) and \( [\sigma] = \{ X \in 2^\omega : \sigma \prec X \} \). The sets \([\sigma] \) are the basic open (actually clopen) sets of our topology on Cantor space. We use \( \lambda \) for the Lebesgue measure on \( 2^\omega \), given by \( \lambda([\sigma]) = 2^{-|\sigma|} \). This measure on \( 2^\omega \) induces a measure, \( \mu \), on \( 2^{<\omega} \) given by \( \mu(A) = \sum_{\sigma \in A} \lambda([\sigma]) \). We use angle brackets to distinguish binary strings from natural numbers, i.e. ‘10’ signifies the number ten while ‘\( \langle 10 \rangle \)’ signifies the binary string whose first bit is a one and whose second bit is a zero.

A **partial recursive function** is a partial function \( f : \omega \to \omega \) that can be computed by some deterministic algorithm (for concreteness, say by a deterministic Turing machine), whose domain may not be total (on some inputs the algorithm may fail to give an output). If a partial recursive \( f \) converges (gives an output) on some input \( n \) we write \( f(n) \downarrow \), if we wish to specify both that it converges and the value to which it converges, we use \( f(n) \downarrow = x \). If it fails to converge, we write \( f(n) \uparrow \). A partial recursive function that converges on every input is **recursive**. It is possible to effectively list all such algorithms, and we let \( \phi_e \) denote the \( e \)th partial recursive function according to this effective listing.
By the Schoenfield Limit Lemma and Post’s Theorem a function has a recursive approximation if and only if it is $\Delta^0_2$. That is, a partial function $f$ is $\Delta^0_2$ if and only if there is a recursive function $\hat{f}(x, s)$ such that $f(x) = \lim_s \hat{f}(x, s)$ for every $x$ (if this limit does not exist, obviously $f(x)$ is undefined). We view this recursive function as an approximation to $f$, with $\hat{f}(x, s)$ giving the value of $f(x)$ at stage $s$ of this approximation. In practice, we often move the second input into a subscript and write $f_s$ for the function given at stage $s$ of this approximation to $f$.

For two reals (or subsets of $\omega^n$) $A$ and $B$, we write $A \leq_T B$ if there is an algorithm for answering questions about the bits (or about membership) of $A$ by querying the bits of $B$. In this case we say that $A$ is Turing-reducible to $B$ or that $A$ is recursive in $B$. If we wish to specify that the algorithm $\Phi$ is performing the reduction, we may write $\Phi^B = A$.

By a machine we mean a partial recursive function $M : 2^{<\omega} \to 2^{<\omega}$. A machine $M$ is prefix-free if for any $\sigma < \tau$ in $2^{<\omega}$, if $M(\sigma) \downarrow$ then $M(\tau) \uparrow$. It follows that $\mu(\text{dom}(M))$ (where $\mu$ is our extension of Lebesgue measure to $2^{<\omega}$) for such a machine is no more than 1. In practice, all of our machines will be prefix-free and we will tend not to explicitly mention their prefix-freeness. We think of these machines as being decoding algorithms: inputs are codes for their outputs, and the work of the machine is to decode the codes.

A prefix-free machine induces a notion of complexity on $2^{<\omega}$ given by $K_M(\sigma) = \min\{|\tau| : M(\tau) = \sigma\}$, that is, the complexity of $\sigma$ is the length of the shortest $\tau$ that can be used as a code for $\sigma$. $K_M$ is called the (prefix-free) Kolmogorov complexity relative to $M$. Intuitively, this represents the information content of $\sigma$ relative to $M$, because it is a measure of the number of bits of information $M$ needs in order to generate $\sigma$. It is a famous result, independently discovered by Solomonoff, Kolmogorov, and Chaitin, that there is a universal prefix-free machine, i.e. an $M_U$ such that for any other prefix-free machine $M$ there is a constant $c_M$ such that for all $\sigma \in 2^{<\omega}$, $K_{M_U}(\sigma) \leq K_M(\sigma) + c_M$. The idea behind the proof is the observation that any machine is given by an algorithm, so they may be effectively listed, and then an input to the universal machine needs to specify only the position in the list of the machine $M$ and the string $\tau$ to run on it to generate the output $M(\tau)$. Algorithms for non-prefix-free machines can be interpreted as prefix-free ones by ignoring any action on a prefix or end-extension of a string after the machine has converged on that string. The position of $M$ in this list is obviously fixed, so this information can add no more than a constant to the length of description of strings. We fix some universal machine, $U$, which by definition takes into consideration all possible machines that have codes for a given $\sigma$, and make the following definition.

**Definition 1.2.1.** For a finite binary string $\sigma$, the (prefix-free) Kolmogorov complexity of $\sigma$, is

$$K(\sigma) = \min\{|\tau| : U(\tau) = \sigma\}.$$
in its domain (finitely many) and not others. We use $U_s$ to denote the action of the machine in the first $s$ steps of its run. This also gives us an approximation to $K(\sigma)$ for any $\sigma$, since at any stage $s$ we may have seen some descriptions of $\sigma$, but not necessarily the shortest. We use $K_s$ to denote this approximation, and note that it converges to its final value from above.

We will often be interested in the Kolmogorov complexities of a collection of strings only up to an additive constant (for instance, some amount of constant information needed to identify a particular machine that gives short descriptions for all the given strings) and we use $\leq^+$ to denote that an inequality holds up to an additive constant.

### 1.3 Building Machines

We now have a measure of complexity that depends on the action of the universal coding machine, $U$. Unfortunately, after we have fixed a universal machine $U$ it is not possible to control this machine to determine the complexities of strings we are interested in. The best we can usually do is to build a machine that assigns to strings the complexities we want, and then use the universality of $U$ to ensure the actual complexities of these strings differ by no more than a constant from those we chose. There are two common ways to build such machines.

The first and most straightforward way is to give an explicit description of the action of the partial recursive function $M$. In practice, however, this is often not feasible. The most common way to control the complexity of a set of strings is to make use of the following theorem, often called the Kraft-Chaitin Theorem although it appeared independently in work of Levin.

**Theorem 1.3.1.** Let $W \subset 2^{<\omega} \times \omega$ be a recursively enumerable set such that $\sum_{(\sigma,n)\in W} 2^{-n} < 1$. Then there exists a prefix-free machine $M$ such that for any pair $(\sigma,n) \in W$ there is a $\tau \in 2^{<\omega}$ such that $|\tau| = n$ and $M(\tau) = \sigma$.

The Kraft-Chaitin theorem allows us to build machines merely by specifying in a recursively enumerable way how short we want descriptions of strings to be. When we make use of this theorem, we will call the set $W$ that we construct a *Kraft-Chaitin set* and the pairs $(\sigma,n)$ in $W$, *requests*. The condition that $\sum_{(\sigma,n)\in W} 2^{-n} < 1$ is merely some reasonability condition on our requests, that we don’t ask for too many short descriptions of different strings. As long as we don’t get greedy, the machine we want exists. When we build a request set, we will refer to this sum as the *mass* or *weight* of the set.

We note that if $1 < \sum_{(\sigma,n)\in W} 2^{-n} < \infty$ we cannot build a machine that exactly satisfies our requests, but we can ensure we are only off by an additive constant. For the smallest $d$ such that $\frac{c}{2^d} < 1$ the sum $\sum_{(\sigma,n)\in W} 2^{-n-d}$ is less than 1, and so for the recursively enumerable set $\hat{W}$
given by $\hat{W} = \{(\sigma, n + d) : (\sigma, n) \in W\}$ we have $\sum_{(\sigma, m) \in W} 2^{-m} < 1$. This gives us the following corollary to the Kraft-Chaitin theorem, which will be much more convenient to use.

**Corollary 1.3.2.** Let $W \subset 2^{<\omega} \times \omega$ be a recursively enumerable set such that $\sum_{(\sigma, n) \in W} 2^{-n} < \infty$. Then there exists a prefix-free machine $\mathcal{M}$ and a constant $c$ such that for any pair $(\sigma, n) \in W$ there is a $\tau \in 2^{<\omega}$ such that $|\tau| = n + c$ and $\mathcal{M}(\tau) = \sigma$.

A machine is just a partial recursive function, so we can have machines that query the bits of a real as an oracle and reals with sufficient power may be able to drastically reduce the complexities of strings. We can assume our universal decoding machine $U$ can consider all the machines that query an oracle, and so we can use $K^A(\sigma)$ to denote the shortest description of $\sigma$ given by $U$ when it has oracle access to $A$. For a real $A$, we use $\mathcal{M}^A$ to denote the machine $\mathcal{M}$ relative to oracle $A$. For a finite string $\alpha$, we use $\mathcal{M}^\alpha$ to denote the machine relative to the oracle $\alpha \triangleleft 0^\infty$. Note that $U$ is still prefix-free, so relative to any oracle we still have $\mu(\text{dom}(U^A)) \leq 1$. The Kraft-Chaitin theorem also has an oracle version which we can use to control the complexities of strings relative to other reals.

**Theorem 1.3.3.** Let $W \subset 2^{<\omega} \times \omega \times 2^{<\omega}$ be a recursively enumerable set such that for any $A \in 2^\omega$ the sum $\sum_{(\sigma, n, \alpha) \in W, \alpha < A} 2^{-n} < c < \infty$. Then there exists a prefix-free machine $\mathcal{M}$ and a constant $d$ such that for any triple $(\sigma, n, \alpha) \in W$, there is a $\tau \in 2^{<\omega}$ such that $|\tau| = n + d$ and for every $A \succ \alpha$, $\mathcal{M}^A(\tau) = \sigma$.

Here enumerating a request into the set $W$ can be thought of as asking for a description of $\sigma$ of length $n$ relative to any oracle with initial segment $\alpha$. The weight condition ensures that we do not ask for the domain of the machine relative to any real to be too large.

### 1.4 Lowness Notions

Kolmogorov complexity is a measure of the information content of finite binary strings, but there are two common ways to extend it to a measure of information content of reals. First, one can look at the *initial segment complexity* of a real $A$, i.e., how the quantity $K(A \upharpoonright n)$ behaves as a function of $n$. Second, one can examine the *compressive power* of $A$ used as an oracle, i.e., how the quantity $K(\sigma) - K^A(\sigma)$ behaves as a function of $\sigma$. This second quantity can be thought of as a measure of $A$’s information ‘about $\sigma$’; $K(\sigma)$ is the amount of information needed to describe $\sigma$ without using an oracle and $K^A(\sigma)$ is the amount needed for $A$ to describe $\sigma$, so the difference is the information $A$ already has about $\sigma$. This idea plays a role in the field of mutual information, which is discussed in more detail in Chapter 4.

Each measure of information content for reals has an associated notion of the ‘minimal possible’ amount of information. In terms of initial segment complexity, a real is called $K$-trivial if its initial segments contain no more information than their length, which is the
least amount of information a finite string can have. In terms of compressive power, a real
is called \textit{low for }K \textit{if it can provide no more than a constant advantage in compression over}
the empty set when used as an oracle. More formally,

\textbf{Definition 1.4.1.} A real \(A\) is \textit{\(K\)-trivial} if there is some constant \(c\) such that for all \(n \in \omega\),
\[ K(A \upharpoonright n) \leq K(\bar{n}) + c, \]
where \(\bar{n}\) is the string of \(n\) zeroes, and

\textbf{Definition 1.4.2.} A real \(A\) is \textit{low for }\(K\) if there is some constant \(d\) such that for all \(\sigma \in 2^{<\omega}\),
\[ K(\sigma) \leq K^A(\sigma) + d. \]

In practice, we will often write \(n\) for \(\bar{n}\) when there will be no confusion.

Besides having natural definitions, these two concepts are nicely behaved in terms of
the sets of reals they define. For instance, Nies [12] showed that the \(K\)-trivial reals are
exactly the reals that are low for \(K\) (they now tend to be referred to just as \(K\)-trivial since
this definition was studied first, unless particular attention is being called to their lowness
for \(K\)), Chaitin [4] showed that there are only countably many such reals and that each is
computably approximable, and Downey, Hirschfeldt, Nies, and Stephan [5] showed that they
are closed under effective join (\(A \oplus B\), the real that has the bits of \(A\) as its even bits and
the bits of \(B\) as its odd bits, is \(K\)-trivial if both \(A\) and \(B\) are). Additionally, they are closed
downwards under Turing reducibility (\(A \leq_T B\) if there exists a Turing machine that will
answer questions about membership in \(A\) if given oracle access to the bits of \(B\); this closure
follows easily from the definition of lowness for \(K\)). We will show that these properties fail
in general for certain weakenings of these notions.

\section{1.5 Weak Lowness Notions}

There several possible ways to weaken \(K\)-triviality and lowness for \(K\), but we focus on a
very natural one: replacing the constants in the definitions with slow-growing functions.

\textbf{Definition 1.5.1.} For a function \(f : 2^{<\omega} \rightarrow \omega\), a real \(A\) is \textit{low for }\(K\) \textit{up to }\(f\) \textit{if for all }\(\sigma \in 2^{<\omega}\)
\[ K(\sigma) \leq^+ K^A(\sigma) + f(\sigma). \]

\textbf{Definition 1.5.2.} For a function \(g : \omega \rightarrow \omega\), a real \(A\) is \textit{\(K\)-trivial up to }\(g\) \textit{if for all }\(n \in \omega\)
\[ K(A \upharpoonright n) \leq^+ K(\bar{n}) + g(n) \]

We denote the set of reals low for \(K\) up to \(f\), \(\mathcal{L}K(f)\) and the set of reals \(K\)-trivial up
to \(g\), \(\mathcal{K}T(g)\). These notions are more interesting when the given functions are \textit{orders}, that is, unbounded and nondecreasing. However, we have many possibilities for orderings on the
domains of these functions with respect to which this could be true. We list some of the
options below.
Definition 1.5.3.
A function $f : 2^{<\omega} \to \omega$ is a length-order if it is unbounded and is nondecreasing with respect to the order $\sigma \leq \tau \iff |\sigma| \leq |\tau|$. 

A function $f : 2^{<\omega} \to \omega$ is a $\prec$-order if it is unbounded along any infinite chain of end extensions and is nondecreasing with respect to the order $\sigma \leq \tau \iff \sigma \preceq \tau$.

A function $f : 2^{<\omega} \to \omega$ is a $K$-order if it is unbounded and is nondecreasing with respect to the order $\sigma \leq \tau \iff K(\sigma) \leq K(\tau)$.

Definition 1.5.4.
A function $g : \omega \to \omega$ is a standard order if it is unbounded and is nondecreasing with respect to the standard ordering on $\omega$.

A function $g : \omega \to \omega$ is a $K$-order if it is unbounded and is nondecreasing with respect to the ordering $n \leq m \iff K(\bar{n}) \leq K(\bar{m})$.

The $K$-orders on $\omega$ and $2^{<\omega}$ are perhaps more intuitive as bounds for lowness for $K$ and $K$-triviality, since we may wish to only allow initial segment complexity or compressive power of a real to increase for those segments or strings that are themselves more complex.

1.6 Finite-to-one Approximations

We now have a zoo of competing notions of order, and we would like some way to prove results that are independent of our choice of notion. With this in mind, we distill out the relevant property that these orders have in common and work in this larger class of functions.

Definition 1.6.1. A finite-to-one approximation of a function $f : X \to Y$ is a recursive function $\phi : X \times \omega \to Y$ such that:

1. $\lim_{s \to \infty} \phi(x, s) = f(x)$
2. For all $y \in Y$ there are only finitely many $x \in X$ such that there exists $s \in \omega$ with $\phi(x, s) = y$.

An $f$ that has a finite-to-one approximation is finite-to-one approximable.

We will often write $\phi_s(x)$ for $\phi(x, s)$. Since we require that these finite-to-one approximations be recursive, it is clear that any finite-to-one approximable function is $\Delta^0_2$.

It turns out that being finite-to-one approximable is in a certain sense equivalent to being an order, as the following lemmas demonstrate.
Lemma 1.6.2. For any total $\Delta^0_2$ order $f : X \to \omega$ (in any of the senses defined above, so $X$ is either $\omega$ or $2^{<\omega}$), $f$ is finite-to-one approximable.

Proof. Let $f$ be a $\Delta^0_2$ order from $X$ to $\omega$, with respect to an ordering denoted $\ll$, and with recursive approximation $f_s$. $\ll$ as a subset of $X \times X$ is either recursive (if $f$ is the standard order on $\omega$, or a length- or $\ll$-order on $2^{<\omega}$) or $\Delta^0_2$ (if $f$ is a $K$-order). Let $\ll_s$ be a recursive approximation to $\ll$. We will say that $f_s$ is a $\ll_s \upharpoonright x$-order if $f_s$ is nondecreasing with respect to $\ll_s$ on the elements $\ll_s$-less than $x$.

We now want to find a recursive approximation to $f$ that is finite-to-one. First, we define a recursive sequence of stages by $t_0 = 0$ and $t_i = \text{the first stage } t > t_{i-1} \text{ such that } f_t$ is a $\ll_i \upharpoonright x$-order, where $x$ is the length-lexicographically $i$th element of $X$. We note that since $f$ is total, it must eventually converge on the first $i$ elements of $X$, and since it is an $\ll$-order, when $f_s$ and $\ll_s$ have converged on these elements $f$ must be a $\ll_s$-order on them. Thus, $t_i$ is always defined, and since $t_i > t_{i-1}$, $(t_i)_{i \in \omega}$ is an infinite sequence.

Now we define

$$g_i(x) = \begin{cases} f_{t_i} & \text{if } x \text{ is one of the length-lexicographically first } i\text{-many elements of } X, \\ |x| & \text{otherwise.} \end{cases}$$

The notation $|x|$ denotes the length of a binary string or the absolute value of a natural number.

Note that $g_i(x)$ is recursive. For any $x \in X$, $x$ has some position in the length-lexicographic ordering of $X$ and after this stage $g_i(x) = f_{t_i}(x)$ for all $i$. Since $(t_i)$ is an infinite subsequence of $\omega$, we have that for any $x$, $\lim_{i \to \infty} g_i(x) = f(x)$, so $g_i$ is a recursive approximation to $f$.

It remains to show that $g_i$ is a finite-to-one approximation. Suppose this were not the case. Then for some $c \in \omega$ there are infinitely many $x \in X$ such that each has at least one $i$ with $g_i(x) = c$. We let $G = \{ x \in X : \exists i g_i(x) = c \}$. We are assuming $G$ is infinite. We now construct a $\ll$-increasing sequence of elements of $X$, not necessarily effectively.

First, let $x_0$ be an element of $X$ such that infinitely many elements $\ll$-above $x_0$ are in $G$. If $\ll$ is $<$ then we can take $x_0$ to be $\langle \rangle$ and otherwise each element has only finitely many elements $\ll$-less than or incomparable to it, so we can take $x_0$ to be anything. Now, let $x_j \in X$ be some $x \gg x_{j-1}$ such that $x_j$ is $\ll$-below infinitely many elements of $G$. Here we may need to use the compactness of $2^\omega$ if $\ll$ is $\prec$. By definition, $(x_j)$ is $\ll$-increasing and every $x_j$ has infinitely many elements of $G$ $\ll$-above it.

Now, for each $x_j$ we eventually reach a stage $s_j$ such that $\ll_{s_j}$ has converged on elements up to $x_j$. After this stage, any $y$ with $y \gg x_j$ will also have $y \gg, x_j$. For each stage $s$, only finitely many $x$ can have $g_s(x) = c$, so it must be the case that there are infinitely many stages where new elements are added to $G$. Moreover, from the argument above it follows that there must be infinitely many stages where elements $\ll$-above $x_j$ are added to $G$. For any of these stages $s$ after $s_j$, it must be the case that $f_s(x_j) \leq c$, since for the value of $g$ to change on $y \gg x_j$, $f_s$ must be an order up to $y$. Thus, for each $i$ there are infinitely many
stages where \( f_s(x_j) \leq c \). Now, for each \( j \), either there are also infinitely many stages with \( f_s(x_j) > c \), so \( f \) is not total, or \( f(x_j) \leq c \). But \( (x_j) \) is a \( \ll \)-increasing sequence, so since \( f \) is a \( \ll \)-order its values must be unbounded along \( (x_j) \), which gives us a contradiction. Thus, \( G \) must be finite, and so the approximation \( g_i \) to \( f \) must be finite-to-one.

We have a similar lemma in the other direction.

**Lemma 1.6.3.** If \( f : X \to \omega \) is total and finite-to-one approximable, then there is a total \( \Delta^0_2 \) order (in any of these senses) \( g : X \to \omega \) such that \( g(x) \leq f(x) \) for all \( x \in X \) (\( f \) majorizes \( g \)).

**Proof.** We let \( \ll \) be one of the above senses of order on \( X \) and let \( (f_s) \) be a finite-to-one approximation of \( f \). We define \( g_t(x) = \min\{ f_t(y) : x \ll y \text{ and } y \text{ is one of the length-lexicographically first } t \text{-many elements of } X \} \), if \( x \) is one of the length-lexicographically first \( t \)-many elements of \( X \) and \( g_t(x) = |x| \) otherwise.

It is clear that if it converges, \( g(x) \leq f(x) \), since the \( f_t(x) \) values are included in the set over which we take a minimum. To show that \( g \) is total, we note that since \( f_s \) is a finite-to-one approximation there can be only finitely many values of \( f \) that ever get \( f \)-values less than \( n \). For any \( y \) that is not \( \ll \) one of these \( x \), \( g(y) = n \), and since this holds for any \( n \), \( g \) is unbounded. Additionally, once an \( x \) gets a \( g_s \)-value, any \( y \ll x \) by definition gets a \( g_s \)-value no greater than \( g_s(x) \). Since this holds for stages when \( \ll \) has converged on \( y \) and \( x \), \( g \) must be nondecreasing with respect to \( \ll \).

Lemmas 1.6.2 and 1.6.3 allow us to set aside issues of ‘naturalness’ of notions of order. Having a finite-to-one approximation is the essential property of all these notions, and so as long as we only appeal to this property we can prove statements that will apply to any of these notions.

Given that so many notions of order coincide in this way with finite-to-one approximability, it may be unclear why a notion of order is even necessary. It might be possible that just being finite-to-one in the limit suffices to have a finite-to-one approximation and we can ignore issues of decreasingness relative to particular orderings. Unfortunately, this is not the case. We include in the appendix a construction of a finite-to-one \( \Delta^0_2 \) function that has no finite-to-one approximation. Many of the results in the following sections are stated in terms of \( \Delta^0_2 \) orders since the proof techniques rely on the existence of finite-to-one approximations. The existence of this pathological function does not mean these results cannot be strengthened to apply more generally, but different techniques will need to be found.
We could, of course, examine functions other than $\Delta^0_2$ orders, but the situation becomes more complicated. For instance, we might want to look at orders at higher levels in the arithmetic hierarchy. However, Baartse and Barmpalias [2] showed that there is a $\Delta^0_3$ order $g : \omega \to \omega$ that is a so-called ‘gap function’ for $K$-triviality, that is, being $K$-trivial up to $g$ is equivalent to being $K$-trivial. From this $g$ it is easy to construct a $\Delta^0_3$ gap function for lowness for $K$, so results about either of these weaker lowness notions in higher-complexity settings will be much more complicated.

We might instead want to consider functions with slightly weaker requirements than being an order. For instance, we might allow functions that are unbounded but have finite lim inf. The sets of reals that are $K$-trivial or low for $K$ up to such functions have already been studied in different contexts. Being $K$-trivial up to a function with finite lim inf is equivalent being infinitely often $K$-trivial, a notion that has been studied by Barmpalias and Vlek in [3]. Being low for $K$ up to a function with finite lim inf is equivalent to being weakly low for $K$, which Miller has shown is equivalent to being low for $\Omega$ ($A$ is low for $\Omega$ if $\mu(\text{dom}(U))$ is a Martin-Löf random real relative to $A$), along with some other results, in [11].
Chapter 2

Initial-Segment Complexity

2.1  A failure of Nies’s Theorem in the weaker case

The first theorem we prove is a counterexample to Nies’s theorem that lowness for \( K \) and \( K \)-triviality coincide, in this weaker sense. When we introduce slow-growing bounds on the initial segment complexity and compressive power of a real, it is not clear what we should be looking for as a notion of coincidence, since the bounds have different domains and will necessarily be different functions (even applying some recursive bijection between \( 2^{<\omega} \) and \( \omega \)). The following theorem tells us that there cannot be any general sense in which we can identify lowness for \( K \) up to some \( f \) with \( K \)-triviality up to an associated \( g \).

Theorem 2.1.1. There is a recursive finite-to-one approximable \( f : 2^{<\omega} \to \omega \) such that for any total \( \Delta^0_2 \) order \( g : \omega \to \omega \), \( \mathcal{L}K(f) \neq \mathcal{K}T(g) \). In fact, for \( f(\sigma) = \lfloor |\sigma|/3 \rfloor \) no \( \Delta^0_2 \) order \( g : \omega \to \omega \) satisfies \( \mathcal{K}T(g) \subseteq \mathcal{L}K(f) \).

Proof. It will suffice to show that there is an \( f \) such that for any \( \Delta^0_2 \) \( g \) that is an order on \( \omega \) there is a real \( A \) (which depends on \( g \)) such that \( A \in \mathcal{K}T(g) \) but \( A \notin \mathcal{L}K(f) \). That is, we need to show that for our chosen \( f \) and any \( \Delta^0_2 \) order \( g \),

- \( \exists c \forall n \ K(A \upharpoonright n) \leq K(\bar{n}) + g(n) + c \)
- \( \forall d \exists \sigma_d \ K(\sigma_d) > K^A(\sigma_d) + f(\sigma_d) + d \).

By Lemma 1.6.2 we know that any \( \Delta^0_2 \) order \( g \) is finite-to-one approximable. We can assume without loss of generality that we are given a finite-to-one approximation, \( g_s \), to \( g \). We will construct \( A \) by recursive approximation to satisfy the requirements

\[ R_i : \text{ For all } n \text{ with } g(n) = i, K(A \upharpoonright n) \leq K(\bar{n}) + i + c \]
for all \( i \in \omega \), and

\[ N_i : \text{ There is a } \sigma_i \in 2^{<\omega} \text{ such that } K(\sigma_i) > K^A(\sigma_i) + f(\sigma_i) + i \]
for all \( i \geq 1 \). Note that we do not need to look for a \( \sigma_0 \), since if any \( d \) witnesses lowness for \( K \) up to \( f \) then any larger \( e \) must also witness it. It suffices to show there is a cofinite set of numbers that fail to witness lowness for \( K \) up to \( f \), and the proof will go more smoothly if we skip the case \( d = 0 \).

To meet the \( R_i \) requirements we build alongside \( A \) a Kraft-Chaitin set \( M \) into which we put requests for descriptions of initial segments of \( A \) of the desired lengths. To meet the \( N_i \) requirements we build alongside \( A \) an oracle Kraft-Chaitin set \( L \) into which we put requests for descriptions of tentative \( \sigma_i \)'s relative to our approximation to \( A \), hoping that the universal machine \( U \) will be unable to give descriptions that are as short. We use a set of movable markers, denoted \( \gamma(i) \) to keep track of where increases in \( g \) occur (that is, we will try to define \( \gamma(i) \) to be a number where \( g(n) \) takes a value larger than \( i \), and use \( \gamma_s(i) \) to denote the position of \( \gamma(i) \) at stage \( s \). We use \( \sigma_i,s \) to denote our choice for \( \sigma_i \) at stage \( s \). To simplify the construction, we will make the commitment now that \( f(\sigma_i) = i \) for all \( i \) (we are allowed to choose our \( f \) so that we can meet our \( N_i \) requirements). We will use \( A_i', \gamma_i'(s) \), etc. to denote the values of objects between these two parts in a given stage. We now give the construction.

**Stage 0:** \( A_0 = \emptyset, M_0 = \emptyset, L_0 = \emptyset, \gamma_0(0) = 0 \), all other \( \gamma_0(i) \) are undefined, \( \sigma_{i,0} = 0^{3i} \) for all \( i \).

**Stage** \( s + 1 \):

1. If there is an \( i \) and an \( n \leq s + 1 \) with \( n \geq \gamma_s(i) \) and \( g_{s+1}(n) < i \) then \( \gamma_s(i) \) is no longer marking a point where \( g \) becomes greater than \( i \).

   a) Let \( j \) be the least such \( i \). Put \( \gamma_s(j) \) into \( A_s \) and call the new set \( A'_s \).

   b) Move \( \gamma_s(j) \) to the smallest number \( m \) that has not appeared anywhere in any earlier stage in the construction that has \( g_{s+1}(m) \geq j \), and call it \( \gamma'_s(j) \). Put a request \( (\sigma_{j,s}, j, A'_s \upharpoonright \gamma'_s(j)) \) into \( L_s \).

   c) Repeat 1b) for all \( i \) that satisfy \( j < i \leq s + 1 \). That is, move \( \gamma_s(i) \) and put the corresponding request for \( \sigma_{i,s} \) into \( L_s \). The resulting set is \( L'_s \).

   d) For all \( i \) less than \( j \), let \( \gamma'_s(i) = \gamma_s(i) \)

2. Compute \( K_{s+1}(\sigma_{i,s}) \) for all \( i \leq s + 1 \). If there is an \( i \leq s + 1 \) with \( K_{s+1}(\sigma_{i,s}) \leq K_{A'_s}(\sigma_{i,s}) + f(\sigma_{i,s}) + i \), then \( \sigma_{i,s} \) is no longer satisfying requirement \( N_i \). Recall that \( f(\sigma_{i,s}) = i \) and that our requests into \( L \) ensure that \( K_{A'_s}(\sigma_{i,s}) \leq^+ i \), so really we need to check whether \( K_{s+1}(\sigma_{i,s}) \leq 3i \).
a) Let \( k \) be the least such \( i \). Put \( \gamma'_s(k) \) into \( A'_s \) to get \( A_{s+1} \). Let \( \sigma_{k,s+1} = \sigma_{k,s} + 1 \) (in binary).

b) Move \( \gamma'_s(k) \) to the smallest number \( m \) not used yet in the construction that has 
\( \phi_{s+1}(m) \geq k \), and call it \( \gamma_{s+1}(k) \). Put a request \( (\sigma_{k,s+1}, k, A_{s+1} \uparrow \gamma_{s+1}(k)) \) into \( L'_s \) to get \( L_{s+1} \).

c) For all \( i \) with \( k < i \leq s + 1 \), let \( \sigma_{i,s+1} = \sigma_{i,s} \) and repeat 3b.).

3. Now, to ensure we meet the \( R_i \) requirements we need to add descriptions to \( M \). For every \( m \), let \( i(m) \) be the greatest \( i \) such that \( \gamma_{s+1}(i) \leq m \). Then for every \( m \leq s + 1 \) put a request \( \left( A_{s+1} \uparrow m, K_{s+1}(m) + i(m) \right) \) into \( M_{s+1} \), unless an identical request is already in \( M_s \) (i.e \( A \uparrow m \) and \( K(m) \) have not changed at stage \( s + 1 \)).

Let \( A = \bigcup_s A_s \), \( M = \bigcup_s M_s \), \( L = \bigcup_s L_s \). This ends the construction. Now we verify that the constructed objects have the desired properties. First we verify that the \( M \) and \( L \) that we construct Kraft-Chaitin sets, so the machines they specify actually exist.

**Lemma 2.1.2.** \( \mu(\text{dom}(M)) \leq 2 \).

**Proof.** We weigh the mass that is requested in \( M \) against the mass of the domain of \( U \), which we know is bounded by 1. Requests that go into \( M \) are each associated with a length \( n \) of an initial segment of \( A_s \), so let us first fix an \( n \). Let \( \tau_0, \tau_1, \ldots, \tau_k \) be the strings that are used as shortest descriptions of \( \bar{n} \) at different stages by the universal machine \( U \) in its approximation of \( K(\bar{n}) \), so that \( |\tau_k| \) is the length of the actual shortest description. Then \( U(\tau_j) \downarrow \bar{n} \) for all \( j \leq k \). The first time a request is put into \( M \) on \( n \)’s behalf (at stage \( n \)) there is some number, say \( l \), of markers at positions less than \( n \). If there are \( l \) markers, then the greatest marker is \( \gamma_n(l) \), so the length of that request is \( |\tau_j| + l \) for some \( j \leq k \), and so a bound on the mass contributed to \( \text{dom}(M) \) by the first request is \( 2^{-|\tau_j| - l} \). Markers only ever move upwards, but before any marker below \( n \) moves we might be forced to put another request into \( M \) on \( n \)’s behalf to account for a change in \( K_n(\bar{n}) \). If \( K_n(\bar{n}) \) changes, it is because \( U_n \) has converged on one of the other \( \tau_j \)’s, so we can bound the mass spent on \( n \) before the markers below it move by \( \sum_{j=0}^{k} 2^{-|\tau_j| - l} \). Once a marker below \( n \) moves, say at stage \( t \), \( i_t(n) \) will be strictly less than \( l \), since \( \gamma_t(l) \) will be moved to some number larger than any seen yet in the construction. This means that when we pay into \( M \) again for \( n \) the most we can pay before another marker moves is bounded by \( \sum_{j=0}^{k} 2^{-|\tau_j| - l + 1} \). There are only \( l \) markers that are ever less than \( n \), and we can only be forced to pay for \( n \) again when one of these \( l \) markers move (we are already assuming we are paying for all possible \( K_n(\bar{n}) \) changes). Thus, the most that is paid into \( M \) on behalf of \( n \) is \( \sum_{i=0}^{l} \sum_{j=0}^{k} 2^{-|\tau_j| - i} \). This sum is bounded by \( 2 \cdot \sum_{j=0}^{k} 2^{-|\tau_j|} \). Now, the \( \tau_j \)’s are all distinct elements in the domain of \( U \), and as
we range over all $n$, the collection of $\tau$'s that we consider must also be distinct, since they give different outputs when input into $U$. Thus, the amount of mass requested from $M$ for all $n$ is bounded by $2 \cdot \sum_{\tau \in \text{dom}(U)} 2^{-|\tau|} = 2 \cdot \text{dom}(U) \leq 2 \cdot \text{dom}(U) \leq 2$

Now we need to show that the oracle Kraft-Chaitin set we constructed also has an a priori bound on its mass relative to each oracle. For a real $X$, we use $L(X)$ to denote the set $\{ (\sigma, n, \alpha) \in L : \alpha \prec X \}$.

**Lemma 2.1.3.** For any real $X$, $\mu(\text{dom}(L(X))) \leq 1$.

**Proof.** This follows easily from the construction. The only oracles that are used in requests to $L$ are initial segments of approximations to $A$, and for a given $A_s$ there is at most one request of length $i$ for any $i \in \omega$ (a request to describe $\sigma_{i,s}$). Thus, for a given oracle the amount of mass requested in $L$ is bounded by $\sum_{i=1}^{\infty} 2^{-i} \leq 1$ (there is no $\sigma_0$).

By the preceding lemmas we know that $M$ and $L$ are both Kraft-Chaitin sets, so the machines they witness actually exist. Now we must show that $A$ satisfies the desired properties.

**Lemma 2.1.4.** There is a $c$ such that for every $n$, $K(A \upharpoonright n) \leq K(n) + g(n) + c$.

**Proof.** By the Kraft-Chaitin theorem, we know there is a machine $M$ such that $K_M(\sigma) \leq \min\{ l : (\sigma, l) \in M \}$, and so by the universality of $U$ we know there is a $c$ such that $K(\sigma) \leq \min\{ l : (\sigma, l) \in M \} + c$. Suppose we are at a stage $s$ in the construction such that $A_s \upharpoonright n = A \upharpoonright n$. Then no marker that was placed at a number less than $n$ ever moves again (any such move would need a number less than $n$ to go into $A_s$). Let $i$ be the greatest number such that $\gamma_s(i) \leq n$. Since $\gamma_s(i)$ never moves again, we have $\gamma(i) = \gamma_s(i)$. Now, at the stage $t$ when $\gamma(i)$ was placed at $\gamma(i)$ we had $g_t(\gamma(i)) \geq i$. We must have that $g_t(\gamma(i)) \geq i$ for every $t > t'$ and every $k > \gamma(i)$ or else $\gamma_t(\gamma(i))$ would move, and so we must have $g(\gamma(i)) \geq i$ for every $t' > t$. Thus, we have $g(n) \geq i$. Then at a stage $s' > s$ such that $K_{\sigma'}(\bar{n})$ has converged to its final value, we put a request $(A \upharpoonright n, K(\bar{n}) + i)$ into $M$ if it is not there already, and so we know $K(A \upharpoonright n) \leq K(\bar{n}) + i + c \leq K(\bar{n}) + g(n) + c$.

**Lemma 2.1.5.** For every $i$ there is a $\sigma_i$ such that $K(\sigma_i) > K^A(\sigma_i) + f(\sigma_i) + i$.

**Proof.** It suffices to find an explicit $\sigma_i$ for all $i > 0$. In the course of the construction we put challenges to $U$ into $L$ in the form of descriptions of $\sigma_i$'s of length $i$ from initial segments of $A$. When a challenge is met by $U$ (i.e., $\sigma_i$ gets a description of length less than $K^A(\sigma_i) + f(\sigma_i) + i$) we change $A_s$ by putting in the length of the use of the previous request to $L$ and put into $L$ a request for a length-$i$ description of a different $\sigma_i$ from the new $A_{s+1}$. If we commit to having $f(\sigma_i) = i$ and we use $L$ to ensure $K^A(\sigma_i)$ is at most $i$, then for $U$ to meet a $\sigma_i$-challenge it must find a description of $\sigma_i$ of length no more than $3i$. This adds at least $2^{-3i}$ much mass to the domain of $U$. We know the measure of the domain of $U$ is bounded by $1$, so, as long as we have at least $2^{3i}$-many $\sigma_i$'s with $f(\sigma) = i$, we can issue more
challenges than $U$ can meet. This is true of the $f$ we have chosen, $f(\sigma) = \lceil |\sigma|/3 \rceil$, so we can ensure that every $i$ fails to witness that $A$ is low for $K$ up to $f$.

This ends the proof of Theorem 2.1.1.

Theorem 2.1.1 tells us that the equivalence between having no initial-segment complexity and having no compressive power breaks down in the case of having even very little of either. We can find reals with arbitrarily slow-growing initial segment complexity (i.e., reals in $\mathcal{KT}(g)$ for slow-growing $g$) that fail to satisfy some slow-growing bound on compressive power (i.e., they are not in $\mathcal{LK}(f)$). There are recursive orders $f$ such that $\mathcal{LK}(f)$ is not equivalent to $\mathcal{KT}(g)$ for any $\Delta^0_2$ (and so in particular any recursive) order $g$. In particular, we can interpret $f : 2^{<\omega} \to \omega$ as a function on $\omega$ by applying some recursive bijection between $2^{<\omega}$ and $\omega$ (so it will still be recursive), and we will have $\mathcal{LK}(f) \not\subseteq \mathcal{KT}(f)$, which stands in contrast to Nies’s result that $\mathcal{LK}(0) = \mathcal{KT}(0)$. We show in the beginning of Chapter 3 that there is some sense where the reverse inclusion does hold, and so we do get some sense of one direction of Nies’s Theorem.

### 2.2 Finding a universal witness

We can strengthen the result in Theorem 2.1.1 by finding a single $A$ that works for all $\Delta^0_2$ orders $g$, with a significantly more complicated proof. First we expand our notation.

**Definition 2.2.1.** $\mathcal{KT}(\Delta^0_2)$ is the set of reals that are $K$-trivial up to every $\Delta^0_2$ order $g : \omega \to \omega$. That is,

$$\mathcal{KT}(\Delta^0_2) = \bigcap_{g \text{ a } \Delta^0_2 \text{ order}} \mathcal{KT}(g).$$

**Theorem 2.2.2.** There is a recursive $f : 2^{<\omega} \to \omega$ such that $\mathcal{KT}(\Delta^0_2) \not\subseteq \mathcal{LK}(f)$.

**Proof.** The construction is similar to the one used in the proof of Theorem 2.1.1 although it is substantially more complicated. Again, it suffices to find a real $A$ to witness the separation, but now we must ensure that $A$ is $K$-trivial up to all $\Delta^0_2$ orders. By Lemma 1.6.2 it suffices to ensure that $A$ is $K$-trivial up to all finite-to-one approximable functions. We are trying to construct an $A$ which satisfies

- For any finite-to-one approximable $g$, $\exists c_g \forall n \ K(A \upharpoonright n) \leq K(n) + g(n) + c_g$, and
- $\forall d \exists \sigma_d K(\sigma_d) > K^A(\sigma_d) + f(\sigma_d) + d$.

The basic idea from the earlier construction will be the same; the added complexity comes from trying to run this construction for all finite-to-one approximations simultaneously. The Kraft-Chaitin sets we construct to witness the desired properties of $A$ must be recursively enumerable, so we can’t use knowledge as to which recursive approximations are actually
finite-to-one and total. The best we can do is to guess. To that end, we use \((\phi_e(n,s))_{e \in \omega}\) as an effective listing of all partial recursive 2-place functions, and then we actually construct a tree \(T\) of all possible guesses regarding the behaviors of all the \(\phi_{e,s}\). The \(A\) we are looking for will be the path through \(T\) that follows all the correct guesses. The strategy for keeping \(A\) out of \(\mathcal{L}K(f)\) is unchanged and we can use the same \(f(\sigma) = |\sigma|/3\) and the same \(\sigma_i\)’s. We will build our tree \(T\) by recursive approximation. At certain points in the construction it will be necessary to kill certain finite branches through \(T\), which will mean to make a commitment to never add nodes above these branches.

We will need a more sophisticated set of markers, since we have infinitely many \(\phi_{e,s}\)’s whose behavior we need to mark. For a string \(\alpha \in 2^{<\omega}\), we say \(e\) is a member of \(\alpha\) if \(\alpha(e) = 1\). We will use markers \(\gamma(\alpha)\) to mark places where \(\phi_{e,s}\) takes large enough values for all \(e\) that are members of \(\alpha\). Note that we will not be able to place some of these markers if, for instance, one of the \(\phi_{e,s}\) with \(e\) a member of \(\alpha\) is actually a constant function with some small value. These \(\gamma(\alpha)\) will do double duty as marking where the tree \(T\) branches, that is, the nodes \(\gamma_s(\alpha)\) will be the nodes \(\sigma\) such that both \(\sigma \uparrow 0\) and \(\sigma \downarrow 1\) are in \(T_s\). We introduce these branching nodes into \(T\) to allow for guesses as to the behavior of the \(\phi_{e,s}\)’s, and will interpret paths with a ‘1’ after the \(e\)th branching node as guessing that \(\phi_{e,s}\) is a total finite-to-one approximation and those with a ‘0’ as guessing that it is not.

To simplify notation, for a string \(\alpha \in 2^{<\omega}\) we will use \(\psi_{\alpha,s} = \min_{e \in \alpha} \phi_{e,s}\). This \(\psi_{\alpha}\) keeps track of the rate of growth of all the \(\phi_e\) such that \(e\) is a member of \(\alpha\), and we will use it to decide where to put new markers and branches above the path that follows \(\alpha\) through the first \(|\alpha|\)-many branching nodes (i.e., that guesses that exactly the members of \(\alpha\) are finite-to-one and total).

Since \(L\) was an oracle Kraft-Chaitin machine in the preceding construction, we can treat it exactly the same and merely put in requests for many more oracles (all the possible \(A\)’s). However, the \(M\) from the preceding construction was working with a particular \(g\), so now we will need a collection \((M_e)_{e \in \omega}\) of Kraft-Chaitin sets, with \(M_e\) witnessing the \(K\)-triviality of \(A\) up to \(\phi_e\).

One final point of concern is that we will now have potentially many different initial segments of a given length for the paths through \(T\), each of which could be the true \(A\). To ensure that we make \(A\) \(K\)-trivial up to \(\phi_e\), we will need to pay into \(M_e\) for each of these paths, which could drive the mass in \(M_e\) too high. Luckily, we have control of when \(T\) branches as long as we guarantee it will do so infinitely often, so we can push the branching nodes high enough that when we pay for the same length on multiple paths we pay at a much reduced rate. We let \(b_0 = 0\), \(b_i = 2i\) for all \(i > 0\) and then work to ensure that, for \(|\alpha|\geq i\), the path that follows \(\alpha\) through \(T\) only branches again at a point when \(\psi_{\alpha} \geq b_i\). We also round all values of \(\phi_{e,s}(n)\) down to the largest \(b_i\) no larger than \(\phi_{e,s}(n)\), to make keeping track of the mass we pay easier. Clearly, if we satisfy \(K(A \upharpoonright n) \leq K(\bar{n}) + b_i + c_e\) for \(b_i \leq \phi_e(n)\) then we have satisfied \(K(A \upharpoonright n) \leq K(\bar{n}) + \phi_e(n) + c_e\).
CHAPTER 2. INITIAL-SEGMENT COMPLEXITY

Now the requirements we are trying to meet are

\( B_\alpha: \) The path through \( T \) that follows \( \alpha \) through the branching nodes branches again at a level, \( n \), where \( \psi_\alpha(n) \geq b_{|\alpha|} \)

for all \( \alpha \in 2^{<\omega} \),

\( R_i^e: \) For all \( n \) with \( b_i \leq \phi_e(n) < b_{i+1} \), \( K(A \upharpoonright n) \leq K(\bar{n}) + b_i + c_e \)

for all \( i, e \in \omega \) with \( e \leq i \), and

\( N_i^\alpha: \) There is a \( \sigma_i \in 2^{<\omega} \) such that \( K(\sigma_i) > K^{\gamma(\alpha)}(\sigma_i) + f(\sigma_i) + i \)

for all \( i > 0 \) and \( |\alpha| = i \).

We order these requirements \( B_{\langle \rangle}, R_{00}, B_{\langle 0 \rangle}, B_{\langle 1 \rangle}, N_{1}^{\langle 0 \rangle}, N_{1}^{\langle 1 \rangle}, R_{01}, R_{11}, B_{\langle 00 \rangle}, \ldots \) The construction will be an injury construction, and we give the strategies for meeting each of the requirements.

A \( B_\alpha \) requirement will require attention at a stage \( s \) if there is not a living branching node \( \tau \) above the path that follows \( \alpha \) through \( T_s \) with \( \psi_{\alpha,s}(|\tau|) \geq b_{|\alpha|} \).

The strategy for satisfying \( B_\alpha \) is

1. Search for an \( n \) such that \( \psi_{\alpha,s}(n) \geq b_{|\alpha|} \)

2. Extend the path that follows \( \alpha \) with a string of 0s to a length \( n' + 1 \) where \( n' \) has not been used yet in the construction (so it will be greater than \( n \)). Put the marker \( \gamma(\alpha)_{s+1} \) at the node on this branch of length \( n' + 1 \). Put both extensions of this node into \( T_{s+1} \).

An \( R_i^e \) requirement will require attention at a stage \( s \) if there is an \( n \) with \( b_i \leq \phi_e(n) < b_{i+1} \) and a living path \( \tau \) through \( T_s \) of length at least \( n \) and an \( \alpha \) such that \( e \) is a member of \( \alpha \) and \( \gamma(\alpha)_s \leq \tau \) such that there is not a request in \( M_{e,s} \) for a description of \( \tau \upharpoonright n \) of length no more than \( K_s(n) + b_i \). The strategy for satisfying \( R_i^e \) is

1. For all such \( n \) and \( \tau \), for the longest \( \alpha \) such that \( \gamma(\alpha)_s \leq \tau \) and \( e \) is a member of \( \alpha \), put the request \( (\tau \upharpoonright n, K_s(n) + b_{|\alpha|}) \) into \( M_{e,s+1} \).

An \( N_i^\alpha \) requirement will require attention at a stage \( s \) if the marker \( \gamma(\alpha)_s \) is defined and there is not a request for a description of \( \sigma_{i,s} \) of length \( i \) from the oracle \( \gamma(\alpha)_s \) in \( L_{s+1} \). The strategy for satisfying \( N_i^\alpha \) is

1. Put the request \( (\sigma_{i,s}, i, \gamma(\alpha)_s) \) into \( L_{s+1} \).

**Stage 0:** \( T_0 = \emptyset, L_0 = \emptyset, M_{e,0} = \emptyset \) for all \( e, \sigma_{i,0} = 0^{|i|}, \gamma(\langle \rangle)_0 = \langle \rangle \), all other \( \gamma(\alpha)_0 \) are undefined.

**Stage** \( s + 1: \)
1. Compute $\phi_{e,s+1}(n), K_{s+1}(n),$ and $K_{s+1}(\sigma_{i,s})$ for all $e, n, i \leq s + 1$

2. If there are an $n$ and an $\alpha$ such that $|\gamma(\alpha)s| < n$ and $\psi_{\alpha,s+1}(n) < b|\alpha|$, then $\gamma(\alpha)s$ is no longer marking a point after which $\psi_{\alpha}$ is greater than $b|\alpha|$, so for the length-lexicographically first $\alpha$

   a) Kill all branches of the tree above $\gamma(\alpha)$.

   b) Let $\gamma(\alpha)s^{-}$ be the initial segment of $\gamma(\alpha)s$ of length $|\gamma(\alpha)s| - 1$ and put $\gamma(\alpha)s^{-}1$ into $T_{s+1}$ as a living node. Note that since $\gamma(\alpha)s$ always ends in a 0 and is always a node of length at least 2 longer than any number seen earlier in the construction, $\gamma(\alpha)s^{-}1$ will not have been used before this point.

3. If there is an $i$ such that $K(\sigma_{i,s}) \leq 3i$, then $U$ has defeated the challenge posed by $\sigma_{i,s}$ so for the least such $i$,

   a) For all $\alpha$ with $|\alpha| = i$, $\gamma(\alpha)s$ was being used as an oracle for a short description of $\sigma_{i,s}$ which we no longer need, so kill all branches of $T_s$ above these $\gamma(\alpha)s$'s.

   b) Let $\gamma(\alpha)s^{-}$ be the initial segment of $\gamma(\alpha)s$ of length $|\gamma(\alpha)s| - 1$ and put $\gamma(\alpha)s^{-}1$ into $T_{s+1}$ as a living node.

   c) Let $\sigma_{i,s+1} = \sigma_{i,s} + 1$ (in binary)

4. For the highest priority requirement that requires attention of the first $s + 1$ many requirements, run $s + 1$-many steps of its strategy.

5. Repeat 4.) for any of the first $s + 1$-many requirements that require attention, in order of increasing priority

This ends the construction. We let $T = \bigcup_{s} T_s$, $M_e = \bigcup_{s} M_{e,s}$, $L = \bigcup_{s} L_s$. Now we must show that there is a path through $T$ that has the properties we desire. Unfortunately in this construction, as in life, not all of our requirements can be satisfied. Because the tree contains all guesses as to the behavior of all the $\phi_e$'s, some branches will wait forever for a particular $\phi_e$ to take a value that it will never take, while others will introduce branching nodes and then destroy them infinitely often. Luckily, all we need is a single path that is in $KT(\Delta^0_2)$ but not in $LK(f)$ (We note that it is not difficult to produce more than this. If we introduce extra branching nodes between the guessing nodes and adjust the sequence $(b_i)$ accordingly we can get a perfect set of such reals. A construction along these lines is given in the proof of Theorem 3.1.2). We let $A \in [T]$ be the path through $T$ such that for every $\gamma(\alpha) \prec A$, $\gamma(\alpha)s^{-}1 \prec A \iff \phi_{|\alpha|,s}$ is finite-to-one and total, that is, $A$ guesses correctly at all of the branching nodes it goes through. Now all that remains to prove is that relative to $A$ the requirements are satisfied, and that the Kraft-Chaitin sets we constructed have bounded measure.

We will call a string $\alpha \in 2^{<\omega}$ correct if $\alpha(e) = 1 \iff \phi_{e,s}$ is finite-to-one and total.
Lemma 2.2.3. The requirements $B_\alpha$ and $N_t^\alpha$ for correct $\alpha$ and $R_i^e$ for $e$ such that $\phi_{e,s}$ is finite-to-one and total are all eventually satisfied.

Proof. We start with the $B_\alpha$ and $N_t^\alpha$ requirements. Note that these requirements only need to act once to become satisfied until some action of the construction injures them. Thus, it suffices to show that they will be injured only finitely often. The two possible sources of injury to these requirements are actions of the construction at step 2.) or at step 3.) for a $\beta \preceq \alpha$ and a $j \leq i$.

First, let us fix a $j \leq i$ and show there can be only finitely many injuries from this source. Any injury from step 2.) for $j$ at a stage $s$ in the construction occurs because the universal machine $U$ has found a description of $\sigma_{j,s}$ of length at most $3j$. This adds $2^{-3j}$ much mass to the domain of $U$, which we know has measure bounded by 1. Thus, there can be no more than $2^{3j}$-many injuries from this source.

Now, let us fix a $\beta \preceq \alpha$ and show there can be only finitely many injuries from step 2.). A injury for $\beta$ at step 2.) of a stage $s$ in the construction occurs because there is an $n$ with $|\gamma(\beta)| < n$ and $\psi_{\beta,s+1}(n) < b_{|\beta|}$, which means there is some $e$ that is a member of $\beta$ with $\phi_{e,s+1}(n) < b_{|\beta|}$. Since $\alpha$ is correct, $\beta$ must also be correct, so for any $e$ that is a member of $\beta$ we must have that $\phi_{e,s}$ is finite-to-one and total. By its finite-to-oneness, we know that $\phi_{e,s+1}(n) < b_{|\beta|}$ can only happen for finitely many pairs $(n,s)$, so $\phi_{e,s}$ can only cause $\psi_{\beta,s}$ to drop finitely often. Then, since $|\beta|$ is finite, $\psi_{\beta}$ can only drop finitely often, and so can only cause finitely many injuries.

Thus, each $B_\alpha$ and $N_t^\alpha$ requirement for correct $\alpha$’s is only injured finitely often, after which point as long as its strategy can be carried out it will be satisfied. For $N_t^\alpha$ this is trivial, as all it needs to do is put a request into $L_s$. For $B_\alpha$ we again need to appeal to the correctness of $\alpha$: since $\alpha$ is correct there will be a pair $(m,t)$ such that for every $n > m$ and $s > t$, $\psi_{\alpha,s}(n) \geq b_{|\alpha|}$, so $B_\alpha$ will eventually find an $n$ that works and be able to act.

Finally, we need to show that the $R_i^e$ requirements for the right $e$’s are satisfied. Since $\phi_{e,s}$ is finite-to-one there is some number $m$ such that for all $n > m$ and all $s$ we have $\phi_{e,s}(n) > b_{i+1}$. That is, there are only finitely many $n$ with $b_i \leq \phi_{e,s}(n) < b_{i+1}$ for any $s$. Eventually we will come to a stage $t$ in the construction such that for all $|\gamma(\alpha)| < m$, $\gamma(\alpha)_t = \gamma(\alpha)$, that is, these $\gamma(\alpha)$’s are never moved again. At this stage $t$ the tree up to level $m$ has settled down, so the possible initial segments of $A$ of length no more than $m$ are all in $T_t$. At some later stage $t'$, $\phi_{e,t'}(n)$ and $K_{t'}(n)$ have converged to their final values for all $n < m$ and after this stage when $R_i^e$ can act at most once for each $n$ after which point it is satisfied.

Now we need to show the measures of the domains of our Kraft-Chaitin sets have a priori bounds.

Lemma 2.2.4. For every $e$, $\mu(\text{dom}(M_e)) \leq 4$.

Proof. This proof is similar to the proof of Lemma 2.1.2. We keep track of the mass paid into $M_e$ on behalf of a fixed $n$, using a fixed description $\rho$ of $n$. We obtain a very rough
upper bound by bounding the mass that would go into $M_e$ if we paid for every path of length $n$ in some $T_s$, even those on branches that guess that $\phi_{e,s}$ is not finite-to-one and total. For a given path $\tau$ of length $n$, at each stage $s$ there is some maximal $\alpha$ such that $\gamma(\alpha) \leq \tau$, and so to satisfy $R^e_i$ we will put a request $(\tau, |\rho| + b|\alpha|)$ into $M_{e,s}$, assuming $\phi_{e,s}(n)$ has not dropped below $b|\alpha|$ and caused an injury. Since we are fixing $\rho$ for now, we can assume we only pay into $M_{e,s}$ on behalf of $n$ when there is a change in the tree below $n$, which only happens when a $\gamma(\alpha)$ marker moves up and makes a new $\alpha'$ maximal such that $\gamma(\alpha') \leq \tau$ for some $\tau$ of length $n$ in $T_s$. Each time a new $\alpha'$ becomes maximal in this sense we may have to pay into $M_{e,s}$ for a new initial segment of our tree, and the amount we pay in is $2^{-|\rho| - b|\alpha'|}$. Thus, we can bound the total mass paid with $\rho$ on behalf of $n$ by summing this term over all $\alpha'$. We know there can be at most 2 strings for which $\langle \rangle$ is maximal (if $\gamma(\langle 0 \rangle)$ and $\gamma(\langle 1 \rangle)$ both move above $n$), at most 4 strings for which an $\alpha$ of length 1 is maximal, at most 8 for which an $\alpha$ of length 2 is maximal, and so on. So, for a given $i$ we can be forced to pay into $M_{e,s}$ on behalf of $n$ at a rate of $b_i$ at most $2^{i+1}$-many times. Thus, the total amount paid with $\rho$ is bounded by $\sum_{i=0}^{\infty} 2^{i+1} \cdot 2^{-|\rho|-b_i}$. We can rewrite this sum as $2^{-|\rho|} \sum_{i=0}^{\infty} 2^{i+1-b_i}$, which, using $b_i = 2i$ is less than $4 \cdot 2^{-|\rho|}$. We can now sum over all $\rho$ ever used as a shortest description (of any $n$), and we find the total mass paid into $M_e$ is no more than $4 \cdot \sum_{\rho \in \text{dom}(U)} 2^{-|\rho|} \leq 4$.

Note that Lemma 2.2.4 holds for all $M_e$, not just those for which $\phi_{e,s}$ is finite-to-one and total. This construction in fact ensures that for every path $B$ through $T$, $B$ is $K$-trivial up to all those $\phi_e$ that $B$ guesses are finite-to-one and total. If $B$ is incorrect and guesses, say, that some constant $\phi_e$ is in fact finite-to-one, and then becomes an isolated path through $T$ (after some point no branching nodes will be stable, since we will not have $\phi_e > b_i$), then $M_e$ witnesses that $B$ is $K$-trivial in the standard sense.

**Lemma 2.2.5.** For every real $X$, $\mu(\text{dom}(L(X))) \leq 1$.

**Proof.** As in the simpler case, this follows easily. For any path $\beta$ through $T_s$ at any stage, there is at most one request for a description of length $i$ using $\beta$ as an oracle for each $i$. Thus, the total mass relative to $\beta$ is no more than $\sum_{i=1}^{\infty} 2^{-i} = 1$.

Because the relevant requirements are satisfied and the Kraft-Chaitin sets we construct do actually produce prefix-free machines, the $A$ that guesses correctly through $T$ will in fact be in $\mathcal{KT}(\Delta^0_2)$ but not in $\mathcal{LK}(f)$. This completes the proof of Theorem 2.2.2.

Note that the only property we used of the function $f$ was that it had sufficiently many preimages for each $i$. In our construction this was $8^i$-many, but this was merely chosen for convenience. However, for sufficiently quickly growing functions (for example, $f(\sigma) = K(\sigma)$)
this construction can’t go through because every real $X$ is in $\mathcal{LK}(f)$. Whether there is a precise cut-off in growth rate between these two cases, and what the growth rate might be, are still open questions at this time.

2.3 Downwards closure

It is easy to see that lowness for $K$ as a property of reals is closed downwards under $\leq_T$, since a real can simulate the compression algorithms of any real it can compute. From this and Nies’s theorem it follows that $K$-triviality is also closed downwards. The same argument shows that $\mathcal{LK}(f)$ is closed downwards for any $f$, but since we showed above that Nies’s theorem does not hold in this weaker case, it does not necessarily follow that $\mathcal{KT}(g)$ is closed downwards in general. In fact, this is not the case, and this section explores various results on the interaction of computational power and bounded initial segment complexity. We begin with the simplest case.

**Theorem 2.3.1.** There is a recursive order $g$ such that $\mathcal{KT}(g)$ is not closed downwards under $\leq_T$.

**Proof.** We need an order $g$, a real $A \in \mathcal{KT}(g)$, a real $B \notin \mathcal{KT}(g)$, and a Turing functional $\Psi$ such that $\Psi^A = B$. Regarding $g$, our only concern is that it grows slowly enough, and we can define $g$ once we have decided what the rate of growth should be. The $A$ and $B$ that we construct will be recursively enumerable. As in the proofs above, we will construct a Kraft-Chaitin set $M$ alongside our reals to witness $A \in \mathcal{KT}(g)$. That is, $M$ will be used to ensure

$$\exists c \forall n \ K(A \upharpoonright n) \leq K(\bar{n}) + g(n) + c.$$  

The idea behind the construction will be to have changes at large numbers in $A$ (where $g$ values are large enough that we can afford to pay into $M$ for new descriptions) correspond to changes at small numbers in $B$ (where $g$ values are too small for the universal machine to pay for too many new descriptions). This case is simple enough that we can specify many values at the start. For convenience we will let $n_0 = 0$, $n_i = \sum_{j=1}^{i} 8^j$ for $i > 0$ and declare that $g(n_i) = i$ for all $i$. This is all the information about $g$ that will be important, and so for concreteness we can let $g(n) = g(n_i)$ for $n_{i-1} < n \leq n_i$. This $g$ is clearly a recursive order. Note also that $n_i$ can be obtained recursively from $i$, so $K(\bar{n}_i) \leq^+ K(\bar{i}) \leq^+ i$. We now give the construction.

**Stage 0:** $A_0 = \emptyset$, $B_0 = \emptyset$, $M_0 = \emptyset$, $\Psi_0(n) \uparrow$ for all $n$.

**Stage** $s + 1$:

1. Compute $K_{s+1}(n)$ and $K_{s+1}(B_s \upharpoonright n_i)$ for all $n \leq n_{s+1}$ and all $i \leq s + 1$. 

2. For the least $i \leq s + 1$ such that $K_{s+1}(B_s \downarrow n_i) \leq 3i$, if there is one
   
   a) Put the greatest $n$ such that $n_{i-1} < n \leq n_i$ and $n \notin B_s$ into $B_s$ to get $B_{s+1}$.
   
   b) Put the greatest $m$ such that $n_{3i-1} < m \leq n_{3i}$ and $m \notin A_s$ into $A_s$ to get $A_{s+1}$.

3. Otherwise, let $B_{s+1} = B_s$ and $A_{s+1} = A_s$.

4. For each $n \leq n_{s+1}$, put a request $(A_{s+1} \downarrow n, K_{s+1}(\bar{n}) + g(n))$ into $M_{s+1}$, if there is not an identical request already there.

5. For each $i \leq s + 1$ let $\Psi_{s+1}^{A_{s+1} [n_{3i}]} \downarrow n_i \downarrow = B_{s+1} \downarrow n_i$.

Now we let $A = \bigcup_s A_s$, $B = \bigcup_s B_s$, $\Psi = \bigcup_s \Psi_s$, and $M = \bigcup_s M_s$. This completes the construction. We now verify that the constructed objects have the desired properties.

**Lemma 2.3.2.** $B \notin \mathcal{KT}(g)$.

**Proof.** We need to show for each $i$ there is an $n$ such that $K(B \downarrow n) > K(\bar{n}) + g(n) + i$, i.e. a witness to $i$’s failure to be a constant of $K$-triviality up to $g$ (again, it suffices to show this for $i > 0$, which is more convenient for the proof). We will use the $n_i$ from the construction as the witness for $i$. We have chosen our $g$ and $(n_i)_{i \in \omega}$ so that $g(n_i) = i$, and as noted above $K(n_i) \leq^+ i$, so it suffices to show that for each $i$, $K(B \downarrow n_i) > 3i$. We show that this holds for an arbitrarily chosen $i$.

At step 2.) of each stage $s > i$ of the construction we will check whether $K_s(B_s \downarrow n_i) \leq 3i$. If it is, we change $B_{s+1}$ below $n_i$, causing the universal machine $U$ to have wasted the mass it paid to describe $B_s \downarrow n_i$. Since $U$’s description of $B_s \downarrow n_i$ had length no more than $3i$, at least $2^{-3i}$ much mass went into the domain of $U$ to describe this initial segment of the old $B$. Thus, as long as we can make this change to $B$ at least $2^{3i}$-many times the domain of $U$ will not be able to contain short enough descriptions of all these different possible initial segments of $B$, and we will eventually find some $B_t \downarrow n_i$ that does not have a $U$-description of length less than or equal to $3i$. By our choice of $(n_i)_{i \in \omega}$ there are exactly $2^{3i}$-many numbers between $n_{i-1}$ and $n_i$, and each of these can be used to introduce a $B$-change below $n_i$ and will never be used by any other $n_j$ (there may be a change below a lower $n_j$ when $i$ is not the least such that $B_s \downarrow n_i$ has a too short description). Hence, the construction will eventually defeat every $i$.

**Lemma 2.3.3.** $\Psi^A = B$.

**Proof.** This follows easily from the construction. Facts about $\Psi$ are enumerated as $\Psi_{s+1}^{A_{s+1} [n_{3i}]} \downarrow n_i \downarrow = B_{s+1} \downarrow n_i$ (we refer to facts of this sort as axioms), and any change in $B$ below $n_i$ is accompanied by a change in $A$ below $n_{3i}$.

**Lemma 2.3.4.** $A \in \mathcal{KT}(g)$. 


Proof. Through the course of the proof we enumerate requests \( (A_s \upharpoonright n, K_s(n) + g(n)) \) into \( M \), so it suffices to show that the mass we put into \( M \) is bounded to ensure that the machine \( \mathcal{M} \) derived from \( M \) witnesses \( A \)'s \( K \)-triviality up to \( g \). First we fix some \( n \) and a \( \mathbb{U} \)-description \( \sigma \) of \( n \) and bound the mass put into \( M \) on \( n \)'s behalf using \( \sigma \).

Let \( i \) be the greatest such that \( n_{3i} < n \leq n_{3(i+1)} \). We consider two cases. In the first case, \( n \leq n_{3(i+1)}-1 \). Then any change in \( A \) below \( n \) is a change below \( n_{3i} \), so it must correspond to a change in \( B \) below \( n_i \). There can be no more than \( n_i \)-many such changes, and so the most we can pay into \( M \) on behalf of \( n \) would be to pay in \( 2^{-|\sigma|-g(n)} \) for each of these changes. Now, since \( n > n_{3i} \), \( g(n) > 3i \), so it will suffice to find a bound for \( n_i \cdot 2^{-|\sigma|-3i} \).

In the second case, \( n > n_{3(i+1)}-1 \) so there may be changes in \( A \) below \( n \) that originate as changes below \( n_{3(i+1)} \). Thus, there may be as many as \( n_{i+1} \)-many such changes. On the other hand, in this case we know that \( g(n) = 3(i+1) \) since we have \( n_{3(i+1)}-1 < n \leq n_{3(i+1)} \), so we are trying to bound \( n_{i+1} \cdot 2^{-|\sigma|-3(i+1)} \). But this is just the term from the first case with a higher index, so it will suffice to find a bound for \( n_{i} \cdot 2^{-|\sigma|-3i} \) in terms of \( i \).

We recall that \( n_i = \sum_{j=1}^{i} 8^j \), so we are trying to bound
\[
\sum_{j=1}^{i} 8^j \cdot 2^{-|\sigma|-3i}.
\]
We can rewrite this as
\[
\sum_{j=1}^{i} 2^{3j} \cdot 2^{-|\sigma|-3i} = 2^{-|\sigma|} \sum_{j=1}^{i} 2^{3j-3i}.
\]
We can reindex to express this as
\[
2^{-|\sigma|} \sum_{k=0}^{i-1} 2^{-3k}.
\]
and the sum here is bounded by \( \frac{8}{7} \). Thus, \( n_i \cdot 2^{-|\sigma|-3i} \leq \frac{8}{7} 2^{-|\sigma|} \). We can now sum over all \( \sigma \) used as a shortest description for any \( n \), and find that the total mass of requests in \( M \) is bounded by \( \sum_{\sigma \in \text{dom}(\mathbb{U})} \frac{8}{7} 2^{-|\sigma|} \leq \frac{8}{7} \). This is an \textit{a priori} bound on the mass of requests in \( M \), so \( M \) is in fact a Kraft-Chaitin set and the associated machine \( \mathcal{M} \) does exist. \( \mathcal{M} \) witnesses that \( A \) is \( K \)-trivial up to \( g \).

This completes the proof of Theorem 2.3.1.

\[\square\]

2.4 Strengthenings

The proof of Theorem 2.3.1 actually gives us something much stronger than that \( \mathcal{K}_T(g) \) is not closed downwards under \( \leq_T \). We give a series of corollaries based on some observations.
First we define truth-table reducibility, a stronger notion of computation than Turing reducibility. For two reals \( A \) and \( B \), \( A \) is truth-table reducible to \( B \) (\( A \leq_{tt} B \)) if there is an algorithm for computing the bits of \( A \) by querying the bits of \( B \), and moreover the positions to be queried in \( B \) to answer whether \( A(n) = 0 \) are must be specified before any query is made. Intuitively, one needs to be able to decide whether \( A(n) = 0 \) by looking at a ‘truth-table’ of the relevant bits of \( B \). We note that the functional \( \Psi \) that we constructed actually gives a truth-table reduction of \( B \) to \( A \), since we can specify beforehand the use of the computations. The first \( n_i \) bits of \( B \) can be computed from the first \( n_{3i} \) bits of \( A \). This gives us the first corollary.

**Corollary 2.4.1.** \( \mathcal{KT}(g) \) is not closed downwards under \( \leq_{tt} \).

Next, we note that as constructed it is also possible to compute \( A \) from \( B \). A change in \( A \) only happens to account for a change we make to \( B \), so we can just as easily compute \( A \upharpoonright n_{3i} \) from \( B \upharpoonright n_i \).

**Corollary 2.4.2.** There are reals \( A \), \( B \) and a recursive order \( g \) such that \( A \equiv_{tt} B \) and \( A \in \mathcal{KT}(g) \) but \( B \notin \mathcal{KT}(g) \).

**Proof.** We define a Turing functional \( \Theta \) by approximation. \( \Theta_0(n) \uparrow \) for all \( n \), and for each \( s \), for each \( i \leq s \), \( \Theta_{3i}^B \upharpoonright n_i \downarrow = A_s \upharpoonright n_{3i} \). We note that since the use of the computation is specified in advance this actually gives a truth-table reduction, so with Corollary 2.4.1 the statement follows.

We can generalize further by releasing control of \( B \) and working with arbitrary recursively enumerable reals.

**Corollary 2.4.3.** For the recursive order \( g \) defined above, for any recursively enumerable \( B \) there is an \( A \equiv_{tt} B \) with \( A \in \mathcal{KT}(g) \).

**Proof.** For a given recursively enumerable \( B \), we carry out the same construction as in the proof of Theorem 2.3.1, but instead of monitoring \( K_s(B \upharpoonright n_i) \) and changing \( B \) when this quantity drops too low, we monitor the enumeration of \( B \) and wait for a change in \( B_s \) below \( n_i \). When this happens, we change \( A \) below \( n_{3i} \) by putting in the number below \( n_{3i} \) that corresponds to the number that went into \( B \). The definitions of \( \Psi \) and \( \Theta \) remain unchanged. Since \( B \) is recursively enumerable, we still have \( n_i \) as a bound on how many changes we can be forced to make in \( A \) below \( n_{3i} \), and so the verification lemmas follow as before.

Note that if instead of specifying \( g \) in the statement of Corollary 2.4.3 we stated that there existed some such \( g \) the corollary would be trivial, since there are \( g \) that grow fast enough that \( \mathcal{KT}(g) = 2^\omega \). We can further relax the requirement that \( B \) be recursively enumerable to merely \( \Delta_0^2 \). This proof will require more work, since we will have to adjust the use of the \( \Psi \) computations.
Corollary 2.4.4. For the recursive order $g$ defined above, for any $\Delta^0_2$ real $B$ there is an $A \equiv^tt B$ with $A \in KT(g)$.

Proof. We now have to account for as many as $2^{n_i}$-many potential initial segments of length $n_i$ for $B$. We will have to allow $A$ to be $\Delta^0_2$ as well (instead of just recursively enumerable), and then $A$ can follow any changes $B$ makes. In the proof of Theorem 2.3.1 we used the bits in the interval $(n_{3i} - 8^i, n_{3i}]$ in $A$ to track the changes in the bits in the interval $(n_{i-1}, n_i]$ in $B$. We must do something similar in this case, except we must move this interval higher up so that we will be able to afford to pay into $M$ for all the possible initial segments of $A$ this will create. For the sakes of convenience and clarity, we give a revised construction here. We maintain the same sequence $n_i = \sum_{j=1}^{i} 8^j$.

**Stage 0**: $A_0 = \emptyset$, $M_0 = \emptyset$, $\Psi_0(n) \uparrow$, $\Theta_0(n) \uparrow$ for all $n$.

**Stage $s + 1$**:

1. Compute $B_{s+1}(n)$ for all $n \leq n_{s+1}$.
2. For the least $i$ such that $B_s \upharpoonright n_i \neq B_{s+1} \upharpoonright n_i$, replace the interval $(n_{n_i} - 8^i, n_{n_i}]$ in $A_s$ with the bits from the interval $(n_{i-1}, n_i]$ from $B_{s+1}$ to get $A_{s+1}$.
3. For each $n \leq n_{s+1}$, put a request $(A_{s+1} \upharpoonright n, K_{s+1}(\bar{n}) + g(n))$ into $M_{s+1}$.
4. For each $i \leq s + 1$ let $\Psi^{A_{s+1}|n_{n_i}}_{s+1} \upharpoonright n_i \downarrow = B_{s+1} \upharpoonright n_i$ and let $\Theta^{B_{s+1}|n_{n_i}}_{s+1} \upharpoonright n_{n_i} \downarrow = A_{s+1} \upharpoonright n_{n_i}$.

Again we let $A = \bigcup_s A_s$ and $M = \bigcup_s M_s$, $\Psi = \bigcup_s \Psi_s$, and $\Theta = \bigcup_s \Theta_s$, and we now verify that the desired properties hold.

Lemma 2.4.5. $A \equiv^tt B$.

Proof. As above, this follows easily from the construction. We build $A$ by approximation so that $A$ changes below $n_{n_i}$ only in response to changes in $B$ below $n_i$ and in response to every one of these changes. Since the action of $A$ is always to copy exactly the corresponding interval from $B$, if $B$ moves back to an initial segment it had used before $A$ will move back to the matching initial segment and the axioms will already have been enumerated into the definitions of $\Psi$ and $\Theta$. As before, the uses of these functionals are specified beforehand, so the reductions are $tt$-reductions.

Lemma 2.4.6. $A \in KT(g)$.

Proof. As before, we put all the relevant requests into $M$ over the course of the construction, so all that remains is to show that the mass of these requests is bounded. We fix some $n$ and some description $\sigma$ of $n$ from $U$ and argue for a bound on the mass that goes into $M$ on behalf of $n$ using $\sigma$. 

Let \( i \) be such that \( n_{n_i} < n \leq n_{n_i+1} \). As before, we have two cases. In the first, we consider \( n < n_{n_i+1} - 8^{i+1} \). Then \( n \) is outside of the interval that is affected by changes to \( B \) in the interval \( (n_i, n_{i+1}] \), so there are at most \( 2^{n_i} \)-many possible initial segments of \( A \) of length \( n \). Since \( n > n_{n_i}, g(n) > n_i \), and so the mass paid into \( M \) on behalf of \( n \) using \( \sigma \) is no more than \( 2^{n_i} \cdot 2^{-|\sigma|-n_i} \). This is clearly bounded by \( 2^{-|\sigma|} \).

In the second case, we have \( n \geq n_{n_i+1} - 8^{i+1} \), so \( n \) could be affected by changes to \( B \) in the interval \( (n_i, n_{i+1}] \). There are then \( 2^{n_{i+1}} \)-many possible initial segments of \( A \) of length \( n \) that may need to be paid for. Now, \( n_{n_i+1} - 8^{i+1} < n_{n_i+1} - 8^{i+1} \) by definition of the sequence \( (n_i) \), and so in this case we have that \( g(n) = n_{i+1} \). Thus, we again have that the mass that goes into \( M \) is bounded by \( 2^{n_{i+1}} \cdot 2^{-|\sigma|-n_{i+1}} \leq 2^{-|\sigma|} \). We can now sum over all \( \sigma \) used as shortest descriptions, and we see that the total mass of \( M \) is bounded by the mass of the domain of \( U \), which is no more than 1.

This ensures that the machine \( \mathcal{M} \) requested by \( M \) exists, and this machine witnesses that \( A \in \mathcal{K}T(g) \).

\( B \) was an arbitrary \( \Delta^0_2 \) real given to us, so there is nothing to verify about it, and so we have completed the proof of Corollary 2.4.4

There is one final generalization we can take in this direction. The only property we really used of \( g \) was in the initial proof of Theorem 2.3.1 where it had to grow slowly enough that we could force \( B \) to be excluded from \( \mathcal{K}T(g) \). Now that we have moved to arbitrary \( B \), the growth rate of \( g \) is of no consequence, so we can expand our theorem to work for any \( \Delta^0_2 \) order. Unfortunately, once we lose control of \( g \) we can no longer specify in advance the uses of the Turing functionals we will build, so we will no longer be able to ensure a truth-table reduction. Additionally, we will no longer be able to guarantee that \( B \geq_T A \), since \( A \) will be able to compute the growth rate of \( g \), which may be beyond the computational power of \( B \).

**Theorem 2.4.7.** For any \( \Delta^0_2 \) order \( g : \omega \to \omega \) and any \( \Delta^0_2 \) real \( B \) there is a \( \Delta^0_2 \) real \( A \geq_T B \) such that \( A \in \mathcal{K}T(g) \).

**Proof.** First we note that the purposes of the sequence \( (n_i) \) in the later proofs was essentially to keep track of where \( g \) attained values high enough that we were willing to pay into \( M \) for changes at those levels. We will want to do the same thing here, but we will have to monitor the approximation to \( g \) and follow where these points move. Using Lemma 1.6.2 it suffices to show the statements for finite-to-one approximable functions, so we may assume the \( g \) we are given has such an approximation \( g_s \). We can further assume that at each stage \( g_s \) is a standard order on \( \omega \). We will use the \( (n_i) \) now as movable markers, with \( n_i \) marking a point such that \( g_s(n_i) \geq 2i \) for \( i > 0 \) and \( n_0 \) staying at 0. To avoid the need to use intervals, we will code \( B \) into \( A \) bit by bit, with the value \( A(n_i) \) giving the value of the \( i \)th bit of \( B \). A heavily modified version of the earlier construction works as follows.
CHAPTER 2. INITIAL-SEGMENT COMPLEXITY

Stage 0: \( A_0 = \emptyset, M_0 = \emptyset, \Psi_0(m) \uparrow, \Theta_0(m) \uparrow \) for all \( m, n_{0,0} = 0 \) all other \( n_{i,0} \) are undefined.

Stage \( s + 1 \):

1. Compute \( B_{s+1}(n) \) for all \( n \leq s + 1 \) and \( g_{s+1}(m) \) for all \( m < n_{s,s} \).

2. If there is an \( i < s + 1 \) such that \( g_{s+1}(n_{i,s}) < 2i \),
   - a) For the least such \( i \), let \( n \) be the smallest number that is larger than anything used so far in the construction such that \( g_{s+1}(n) \geq 2i \), put \( n_{i,s} - 1 \) into \( A_{s+1} \), and move \( n_{i,s+1} \) to \( n + 1 \).
   - b) Repeat 2a) for all \( j \) with \( i < j \leq s \), regardless of whether or not \( g_{s+1}(n_{j,s}) < 2j \), in order from smallest to largest.

3. Let \( n \) be the smallest number that is larger than anything used so far in the construction such that \( g_{s+1}(n) \geq 2(s + 1) \) and set \( n_{s+1,s+1} \) equal to \( n + 1 \).

4. For each \( i < s + 1 \), set \( A_{s+1}(n_{i,s+1}) = B_{s+1}(i) \).

5. For each \( n \leq n_{s+1,s+1} \), put a request \( (A_{s+1} \upharpoonright n, K_{s+1}(\vec{n}) + g_{s+1}(n)) \) into \( M_{s+1} \).

6. For each \( i \leq s + 1 \) let \( \Psi_{s+1}^{A_{s+1}}(n_{i,s+1}) \upharpoonright i \downarrow = B_{s+1} \upharpoonright i \).

This completes the constructions. We set \( A = \bigcup_s A_s, M = \bigcup_s M_s, \Psi = \bigcup_s \Psi_s \), and \( n_i = \lim n_{i,s} \). Again, we verify that the desired properties hold.

Lemma 2.4.8. \( A \geq_T B \).

Proof. As before, we have that changes in \( A \) below \( n_{i,s} \) correspond to changes in \( B \) below \( i \). Whenever we move a marker \( n_{i,s} \) we first put \( n_{i,s} - 1 \) into \( A_{s+1} \) to change \( A \) below the use of any computation from \( n_{i,s} \). This allows us to enumerate new axioms using \( n_{i,s+1} \) as the new use, since we may need to make changes at the marker positions to keep up with changes in \( B \). Note that we place the markers so that there is always at least 1 empty space between them that can be used for this purpose.

Eventually we reach a stage \( s \) such that \( n_{i,s} = n_i \), since \( g_s \) is a finite-to-one approximation. After this stage changes in \( B \) below \( i \) always change \( A \) below \( n_i \), which is the use of the old computation, so new axioms can be enumerated into \( \Psi \) (these new axioms for \( \Psi \) might in fact have already been used in earlier stages, but in this case they don’t contradict the current state).

\[ \square \]

Lemma 2.4.9. \( A \in \mathcal{KT}(g) \)
CHAPTER 2. INITIAL-SEGMENT COMPLEXITY

Proof. Once again, $M$ will be our witness to $K(A \mid n) \leq^+ K(n) + g(n)$ once we show that $M$ is a Kraft-Chaitin set. As before, we fix an $n$ and a description $\sigma$ of $n$ from $U$ and bound the mass paid into $M$ on behalf of $n$ using $\sigma$. This construction is now more complicated in that the $n_{i,s}$ markers move, so the number of markers (and thus the bound we have on $g(n)$) will be changing over the course of the construction. We fix an $i$ and find a bound for the amount paid into $M$ on behalf of $n$ using $\sigma$ while there are $i$ active markers below $n$.

Let $[r,t]$ be the largest interval such that there are $i$ active markers below $n$ between stages $r$ and $t$ of the construction. We note that markers only ever move upwards, so there can only be one such interval. During these stages there are $i$-many places below $n$ that $A_s$ could change: the first $i$ markers $n_{j,s}$. Since the values of $A$ at these markers correspond to the values of the first $i$-many bits of $B$ and $B$ is $\Delta^0_2$ there are $2^i$ many possible initial segments of $A$ of length $n$ that we might see during the interval $[r,t]$. Now, since $n > n_{i,s}$ for all $s \in [r,t]$, we have that $g_s(n) \geq 2i$ for all these $s$. Thus, the amount we may have to pay into $M$ using $\sigma$ on behalf of $n$ while there are $i$ active markers below $n$ is bounded by $2^i \cdot 2^{-|\sigma| - 2i} = 2^{-|\sigma| - i}$. If we now sum over all $i$, we get that the total mass paid into $M$ on behalf of $n$ using $\sigma$ is no more than $\sum_i 2^{-|\sigma| - i} \leq 2 \cdot 2^{-|\sigma|}$. Summing over all $\sigma$ in $U$ we get that the total mass paid into $M$ is bounded by $\sum_{\sigma \in \text{dom}(U)} 2 \cdot 2^{-|\sigma|} \leq 2$. This is an a priori bound on the mass of $M$, so it is a Kraft-Chaitin set and the machine $M$ exists.

$A$ is clearly $\Delta^0_2$ by construction. This completes the proof of Theorem 2.4.7.

Theorem 2.4.7 resembles Theorem 2.1.1 in that for any $\Delta^0_2$ order $g$ each gives us a real $A$ in $\mathcal{KT}(g)$ with a particular property. The final result of this section will be to prove an analog of Theorem 2.2.2, building a single $A$ that works for all $\Delta^0_2$ orders $g$. We will have to combine the proof methods from this section with the technique from the proof of Theorem 2.2.2 in building a tree $T$ of guesses about the behaviors of $\phi_{e,s}$ and picking the path through $T$ that guesses correctly at every node. We recall that $\mathcal{KT}(\Delta^0_2) = \bigcap_{g \text{ a } \Delta^0_2 \text{ order}} \mathcal{KT}(g)$.

A will be able to compute which of the $\phi_{e,s}$ are finite-to-one approximations, which may be beyond the computational power of $B$. One happy side effect of building a tree this way is that we are no longer required to have $B$ be $\Delta^0_2$. We can build paths through our tree that will work for any $B$.

**Theorem 2.4.10.** For any real $B$ there is an $A \geq_T B$ such that $A \in \mathcal{KT}(\Delta^0_2)$.

**Proof.** By Lemma 1.6.2, it suffices to build an $A$ that is Turing above $B$ and that is $K$-trivial up to every finite-to-one approximation. As in the proof of Theorem 2.2.2 we do not know a priori which $\phi_{e,s}$ are finite-to-one approximations, and, since we need our Kraft-Chaitin set $M$ to be recursively enumerable, we build a tree $T$ and use the branching nodes to mark guesses as to whether the $\phi_{e,s}$ ’s are finite-to-one approximations. To reaffirm our earlier definition, to kill a node is to make the commitment never to build above it in the tree.
We will use a system of markers $\gamma(\alpha)$ to mark where the values of corresponding $\phi_{e,s}$'s are large enough to place another branching node, which we will use to guess the behavior of the next $\phi_{e,s}$. As in the proof of Theorem 2.4.7 we will also want to use the markers to hold the values of the bits of $B$, so the tree around these nodes will be slightly more complicated. We will have to introduce another kind of branching node which will keep track of which value is in the $i$th bit of $B$. Once we put a marker $\gamma(\alpha)$ at a node to make it a guessing node, both successors of that node, $\gamma(\alpha)^1_s$ and $\gamma(\alpha)^0_s$, will also both be in $T_s$, and a path taking one or the other of these nodes will correspond to guessing whether $\phi|\alpha|$ is or is not a finite-to-one approximation. After this branching we will immediately introduce another branching, so that $\gamma(\alpha)^1_s$, $\gamma(\alpha)^0_s$, $\gamma(\alpha)^1_s$, and $\gamma(\alpha)^0_s$ will all be in $T_s$. The value of a path through $\gamma(\alpha)$ at $|\gamma(\alpha)|+2$ will correspond to the $|\alpha|$th bit of $B$. We distinguish between the two kinds of branching nodes as either guessing nodes or coding nodes. There is one final way for both extensions of a node to appear in $T$, which is when we kill the tree above $\gamma(\alpha)$ and start building above $\gamma(\alpha)^-1$. This makes $\gamma(\alpha)^-1$ a branching node in $T$, but at no stage of the construction are both nodes above $\gamma(\alpha)^-1$ alive (and thus in $T$ the subtree above one of these nodes is finite), so we will not need to refer to these as branching nodes.

The fact that not all of our branching nodes are guessing nodes complicates our keeping track of the guesses that paths are making. A path that follows $\alpha$ through the first $|\alpha|$-many branching nodes is only making guesses about the first $|\alpha|/2$-many $\phi_{e,s}$’s, since only the even-numbered branching nodes correspond to guesses. To make the notation simpler we complicate the definitions slightly, and say that $e$ is a guessing member of $\alpha$ if and only if $\alpha(e) = 1$ and $e$ is even. We denote this $e \in g \alpha$. We then keep the notation from the earlier proof that $\psi_{\alpha,s}(n) = \min\{\phi_{e,s}(n) : e \in g \alpha\}$. Note that $\psi_{\alpha}$ considers up to $|\alpha|/2$ many $\phi_{e,s}$’s.

We will need to ensure we branch at a rate that forces these values to be large enough that we can afford to pay for multiple initial segments of the same length into $M$. We use the sequence $c_i = 3^i$ as values we will wait for before branching.

We again build a separate Kraft-Chaitin set $M_e$ to witness $A$’s $K$-triviality up to $\phi_e$. $M_e$ will only take requests to describe initial segments that are on the tree above nodes that guess that $\phi_{e,s}$ is a total finite-to-one approximation, so for the correct path through $T$ these $M_e$ together will witness that $A \in KT(\Delta^0_2)$.

Ensuring that $A$ is Turing-above $B$ will be handled after the construction. Essentially, we will just take the right $A$ that follows the bits of $B$ through its coding nodes. We will then show that $A$ can then compute where those levels are, so it can query its own values there to obtain information about $B$.

The requirements we are trying to meet are

\[ B_\alpha : \text{The path through } T \text{ that follows } \alpha \text{ through the branching nodes branches twice more at a point } n \text{ where } \psi_{\alpha} \geq c_{|\alpha|} \]

for all $\alpha \in 2^{<\omega}$ with $|\alpha|$ even,
$R^e_i$ : For all $n$ with $c_i \leq \phi_e(n) < c_{i+1}$, $K(A \upharpoonright n) \leq^+ K(n) + c_i$

for all $i, e \in \omega$ with $e \leq i$, and

We order these requirements $B_{\langle \rangle}, R^0_0, B_{\langle 00 \rangle}, B_{\langle 01 \rangle}, B_{\langle 10 \rangle}, R^0_1, R^1_1, B_{\langle 0000 \rangle}, \ldots$. As above, the construction will be an injury construction, and we give the strategies for meeting each of the requirements.

A $B_\alpha$ requirement will require attention at a stage $s$ if there is not a living branching node $\tau$ above the path that follows $\alpha$ through $T_s$ with $\psi_\alpha, s(|\tau|) \geq c_{|\alpha|}$. The strategy for satisfying $B_\alpha$ is

1. Search for an $n$ such that $\psi_\alpha, s(n) \geq c_{|\alpha|}$

2. Extend the path that follows $\alpha$ with a string of 0s to a length $n' + 1$ where $n'$ has not been used yet in the construction (so it will be greater than $n$). Put the marker $\gamma(\alpha)_s + 1$ at the node on this branch of length $n' + 1$. Put all four extensions of this node of length $n' + 3$ into $T_{s+1}$.

An $R^e_i$ requirement will require attention at a stage $s$ if there is an $n$ with $c_i \leq \phi_{e,s}(n) < c_{i+1}$ and there is a living path $\tau$ through $T_s$ of length at least $n$ and there is an $\alpha$ such that $e$ is a guessing member of $\alpha$ and $\gamma(\alpha)_s \leq \tau$ such that there is not a request in $M_{e,s}$ for a description of $\tau \upharpoonright n$ of length less than $K_s(n) + c_i$. The strategy for satisfying $R^e_i$ is

1. For all such $n$ and $\tau$, for the longest $\alpha$ such that $\gamma(\alpha)_s \leq \tau$ and $e$ is a guessing member of $\alpha$, put the request $(\tau \upharpoonright n, K_s(\bar{n}) + c_{|\alpha|})$ into $M_{e,s+1}$.

We now give the construction, which will call these subroutines as necessary.

Stage 0: $T_0 = \emptyset$, $M_0 = \emptyset$, $\gamma(\langle \rangle)_0 = \langle \rangle$ and $\gamma(\alpha)_0$ undefined for all other $\alpha$.

Stage $s+1$:

1. Compute $\phi_{e,s+1}(n)$ for $e, n \leq s + 1$.

2. If there are an $n$ and an $\alpha$ such that $|\gamma(\alpha)_s| < n$ and $\psi_{\alpha, s+1}(n) < c_{|\alpha|}$, then $\gamma(\alpha)_s$ is no longer marking a point after which $\psi_\alpha$ is greater than $c_{|\alpha|}$, so for the length-lexicographically first $\alpha$

   a) Kill all branches of the tree above $\gamma(\alpha)_s$.

   b) Let $\gamma(\alpha)_{s-}$ be the initial segment of $\gamma(\alpha)_s$ of length $|\gamma(\alpha)_s|-1$ and put $\gamma(\alpha)_{s-1}^-$ into $T_{s+1}$ as a living node. Note that again by construction $\gamma(\alpha)_s$ always ends in a 0 and is always a node of length at least 2 longer than any number seen earlier in the construction, hence $\gamma(\alpha)_{s-1}^-$ will not have been used before this point.
c) Repeat 2a) and 2b) for all other such \( \alpha \) with \( \gamma(\alpha)_s \) still living, in length-lexicographic order.

3. For the highest priority requirement that requires attention of the first \( s + 1 \) many requirements, run \( s + 1 \)-many steps of its strategy.

4. Repeat 3.) for any of the first \( s + 1 \)-many requirements that require attention, in order of increasing priority.

This completes the construction. As usual, we let \( T = \bigcup_s T_s \), \( M_e = \bigcup_s M_{e,s} \), and \( \gamma(\alpha)_s = \lim_s \gamma(\alpha)_s \). As in the proof of Theorem 2.2.2, the verification will be substantially more complicated in that several of the requirements will actually not be satisfied. We will define some terms for ease of use.

Let us call a string \( \alpha \) correct if for all \( n \leq |\alpha| \), if \( n = 2m \) is even then \( \alpha(n) = 1 \) if and only if \( \phi_{m,s} \) is a total finite-to-one approximation. That is, \( \alpha \) is correct about the behavior of the \( \phi_{e,s} \)'s that it makes guesses about. We need to show that the requirements that are relevant to building correct paths through \( T \) are all satisfied, and that the mass we put into \( M_e \) to satisfy the \( R_e^i \) requirements is bounded.

**Lemma 2.4.11.** The requirements \( B_\alpha \) for correct \( \alpha \) and \( R_e^i \) for \( e \) such that \( \phi_{e,s} \) is finite-to-one and total are all eventually satisfied.

**Proof.** First we argue that each of these requirements can be subject to at most finitely many injuries. An injury to any \( B_\alpha \) requirement only occurs when for some \( e \in g \alpha \) the value of \( \phi_{e,s}(n) \) drops below \( c_{|\alpha|} \) for some \( n > |\gamma(\alpha)_s| \). When this happens, we respond by moving \( \gamma(\alpha)_{s+1} \) to a higher level. Now, since there are only finitely many \( e \in g \alpha \) if there were infinitely many such injuries then by the Pigeonhole Principle there would be at least one \( e \in g \alpha \) that was responsible for infinitely many. If this were true there would be infinitely many \( n \) such that for some \( t \), \( \phi_{e,t}(n) < c_{|\alpha|} \), so \( \phi_{e,s} \) would not a finite-to-one approximation. Then \( e \in g \alpha \) is a contradiction to \( \alpha \)'s correctness.

For \( R_e^i \) requirements, since \( \phi_{e,s} \) is a total finite-to-one approximation we will eventually reach some stage \( s \) where \( \phi_{e,s} \) has converged on all \( n \) such that \( \phi_{e}(n) < c_{i+1} \). At this stage all the \( n \) that \( R_e^i \) will ever be concerned about have been found. Let the largest of these \( n \) be \( n' \). Then injuries to \( R_e^i \) can only occur either when \( K_s \) changes for one of these \( n \), but this happens only finitely often, or when there is a change in \( T \) below \( n' \) on some branch that is guessing that \( \phi_{e,s} \) is a total finite-to-one approximation. Each of these changes moves some marker to a point larger than \( n' \), and, since there were only finitely many markers at positions lower than \( n' \) at stage \( s \), this can only happen finitely often.

Now we need to show that once these requirements are no longer injured they will be able to act to satisfy themselves. Again, we rely on the correctness of \( \alpha \).

For \( B_\alpha \) requirements, the strategy waits until it finds an \( n \) such that \( \psi_{\alpha,s}(n) \geq c_\alpha \), and then extends a branch in the tree to this height and branches twice. Since every \( e \in g \alpha \) is in
fact a total finite-to-one approximation, there will exist an \( n \) for which \( \psi_\alpha(m) \geq \alpha \) for all \( m > n \), and so eventually \( B_\alpha \) will find such an \( m \) and act and be satisfied.

For \( R^e \) requirements, the strategy puts requests into \( M_e \). After it is no longer injured, it needs to act at most once for each \( n \) with \( \phi_e(n) < \alpha \) and there are only finitely many of these so it is eventually satisfied.

Now we know the \( R^e \) requirements for \( \phi_{e,s} \) that are total finite-to-one approximations are satisfied, but this happens just by placing the relevant requests into \( M_e \). Unless we have that the mass of these requests is bounded the Kraft-Chaitin Theorem will not give us a machine \( M_e \) that satisfies these requests. Once we have shown there is a bound on the mass, \( M_e \) will witness that any path that guesses correctly that \( \phi_{e,s} \) is a total finite-to-one approximation will be in \( \mathcal{KT}(\phi_{e,s}) \).

**Lemma 2.4.12.** For all \( e \), \( \mu(\text{dom}(M_e)) \leq 4 \).

*Proof.* The proof is similar to that of Lemma 2.2.4. Let us start by fixing some \( e \). As above, we will bound the amount paid into \( M_e \) for an arbitrary \( n \) using some shortest description \( \sigma \) of \( n \) from \( U \). We may use the total number of paths in \( T \) of length \( n \) as a rough bound on the number of times we will need to pay into \( M_e \) on behalf of \( n \), even though we will only pay on those that guess that \( \phi_{e,s} \) is a total finite-to-one approximation. For any path \( \tau \) of length \( n \) in \( T \), when \( \tau \) is alive there is some maximal \( \alpha \) such that \( \gamma(\alpha) \preceq \tau \). Now, a given \( \alpha \) can be maximal for at most 4 such \( \tau \)'s since \( T_s \) will branch twice immediately after \( \gamma(\alpha) \) and then no more until \( \gamma(\alpha') \) for some \( \alpha' > \alpha \), at which point \( \alpha' \) would be maximal. A change in \( T_s \) below \( n \) moves at least one of these markers to some new height above \( n \), at which point a smaller \( \alpha \) becomes maximal. Recall that \( \gamma(\alpha) \) are only placed for \( \alpha \) of even length.

For each \( \alpha \) of even length, we know that \( \gamma(\alpha) \) is placed at a node such that \( \psi_\alpha(|\gamma(\alpha)|) \geq c^{|\alpha|} \), and so we know the rate at which we will pay into \( M_e \) for \( \tau \) for which \( \alpha \) is maximal is \( 2^{-c^{|\alpha|}} \). Here we use the fact that we only pay for paths with \( e \in g \alpha \). Now all that remains is to add up the total mass that could be paid into \( M_e \) when any \( \alpha \) is maximal such that \( \gamma(\alpha) < n \). This gives us the sum

\[
\sum_{|\alpha| \text{ even}} 4 \cdot 2^{-|\sigma|} 2^{-c^{|\alpha|}}.
\]

Since there are \( 2^{2i} \)-many \( \alpha \) of length \( 2i \) this can be rewritten as

\[
4 \cdot \sum_{i=0}^{\infty} 2^{2i} \cdot 2^{-|\sigma| - c_i}.
\]

Recall that \( c_i = 3i \) and this sum reduces to \( 4 \cdot 2^{-|\sigma|} \sum_{i=0}^{\infty} 2^{-i} \). The sum here is bounded by 1, so we can bound the mass paid into \( M_e \) on behalf of \( n \) using \( \sigma \) by \( 4 \cdot 2^{-|\sigma|} \). As usual, we
now sum over all $\sigma$ in the domain of the universal machine to find a bound of the mass paid into $M_e$ for any $n$ using any $\sigma$. We get that this mass is bounded by $\sum_{\sigma \in \text{dom}(U)} 4 \cdot 2^{-|\sigma|} \leq 4$, and we are done.

This lemma gives us that a path through $T$ that guesses correctly that $\phi_{e,s}$ is a total finite-to-one approximation will be $K$-trivial up to $\phi_e$. Thus, paths through $T$ that guess correctly about the behavior of every $\phi_{e,s}$ (i.e. that are correct) will be $K$-trivial up to every $\Delta^0_2$ order. Now all that remains is to show that given a real $B$ we can find a path through $T$ that computes $B$ and that is correct about all these guesses.

**Lemma 2.4.13.** For any $B$, there is an $A \in [T]$ such that $A \geq_T B$ and $A$ is correct.

**Proof.** Suppose we are given $B$. We define a sequence of strings $\alpha_i$. Let $\alpha_i$ be such that $|\alpha_i| = 2i$ and for all $n < 2i$, if $n = 2m + 1$ then $\alpha_i(2m + 1) = B(m)$ and if $n = 2m$ then $\alpha_i(n) = 1$ if and only if $\phi_{n,s}$ is a total finite-to-one approximation. Since $\alpha_i$ is correct and $|\alpha_i|$ is even, $B_{\alpha_i}$ is a requirement in our construction that is eventually satisfied. This means that we will eventually place the marker $\gamma(\alpha_i)$ on some node that follows $\alpha_i$ through the first $2i$ many branching nodes, where it will remain for the rest of the construction. Now $\alpha_i \prec \alpha_{i+1}$ for all $i$, so $\gamma(\alpha_i)$ must necessarily be an initial segment of $\gamma(\alpha_{i+1})$ for all $i$. Then we can let $A = \bigcup \gamma(\alpha_i)$ and this is well-defined. Now, $A$ follows $\alpha_i$ through $T$ for every $i$, and each $\gamma(\alpha_i)$ is correct about its guesses, so $A$ must also be correct about its guesses. From this and the previous lemma, we know that $A \in KT(\Delta^0_2)$. All that remains is to show that $A \geq_T B$.

First, we know that the bits of $B$ are encoded somewhere in $A$, since they are the odd bits of the $\alpha_i$’s. $A$ follows $\alpha_i$ through the branching nodes of $T$, so $A(|\gamma(\alpha_i)| + 2) = B(i)$. Now we need to show that $A$ can find $\gamma(\alpha_i)$ for all $i$. Fortunately, this is easy. $A$ can simulate the construction of $T$ and the approximation to $\gamma(\alpha_i)$, $\gamma(\alpha_{i+1})$. When we move a $\gamma(\alpha)$ we first kill the tree above $\gamma(\alpha)$, and then start building above $\gamma(\alpha)$, so it can retrieve the $i$th bit of $B$.

This was the final step in the proof of Theorem 2.4.10.

Theorem 2.4.10 is as far as we will generalize Theorem 2.3.1. We will end this section with a few remarks on the proof.

First, we note that the paths through $T$ in $KT(\Delta^0_2)$ corresponding to different choices of $B$ are all unique, since they contain the bits of $B$ at certain coding locations. Since there is such a path for any $B$, we have that there are uncountably many elements of $KT(\Delta^0_2)$.
Moreover, the subtree $T'$ of $T$ that consists of just those nodes $\tau$ such that for every $\gamma(\alpha) \leq \tau$, $\alpha$ is correct is a perfect tree of reals in $\mathcal{K}T(\Delta^0_2)$. We state this result as a corollary.

**Corollary 2.4.14.** $\mathcal{K}T(\Delta^0_2)$ contains a perfect set of reals.

Second, we note that the paths through $T$ that we construct to compute $B$ may have much higher Turing degree than is necessary. By the same process that $A$ uses to compute the bits of $B$, $A$ can deduce which $\phi_{e,s}$ are total finite-to-one approximations and the index set $\{e : \phi_{e,s} \text{ is a total finite-to-one approximation} \}$ is $\Pi^0_3$-complete (see Appendix B). Of course, for certain $B$ there may be much less complicated $A$ (for example, if $B$ is recursive then $A = B$ is in $\mathcal{K}T(\Delta^0_2)$).

Theorem 2.4.10 shows that not only is $\mathcal{K}T(\Delta^0_2)$ not closed downwards under $\leq_T$ but there is not even an upper bound on the Turing degrees that contain elements of $\mathcal{K}T(\Delta^0_2)$. We also note that Theorem 2.4.10 implies a stronger version of Theorem 2.2.2, which said that there was a recursive $f$ such that $\mathcal{K}T(\Delta^0_2)$ is not contained in $\mathcal{L}K(f)$. The only important property of this $f$ was its growth rate (in fact, that there were sufficiently many preimages for each $i$) and so a similar proof technique would work for a variety of functions. From Theorem 2.4.10 and the fact that $\mathcal{L}K(f)$ is closed downwards under $\leq_T$ we get the following dichotomy without having to worry about adjusting the earlier proof.

**Corollary 2.4.15.** For any function $f : 2^{<\omega} \to \omega$ either $\mathcal{L}K(f) = 2^\omega$ or $\mathcal{K}T(\Delta^0_2) \not\subseteq \mathcal{L}K(f)$.

**Proof.** If there is an $B$ such that $B \notin \mathcal{L}K(f)$, then by Theorem 2.4.10 there is an $A \geq_T B$ such that $A \in \mathcal{K}T(\Delta^0_2)$. Now, since $\mathcal{L}K(f)$ is closed downwards under $\leq_T$ and $B \notin \mathcal{L}K(f)$, $A$ cannot be in $\mathcal{L}K(f)$. Thus, there is an $A$ that is in $\mathcal{K}T(\Delta^0_2)$ but not in $\mathcal{L}K(f)$. \[\square\]

We know there are reals that are not in $\mathcal{K}T(\Delta^0_2)$, so in particular $\mathcal{K}T(\Delta^0_2) \neq 2^\omega$. This gives us the following corollary, which demonstrates that it is impossible to capture this notion of low initial segment complexity with any notion of low compressive power.

**Corollary 2.4.16.** There is no collection $F$ of functions $2^{<\omega} \to \omega$ such that $\mathcal{L}K(F) = \mathcal{K}T(\Delta^0_2)$, where $\mathcal{L}K(F) = \bigcap_{f \in F} \mathcal{L}K(f)$.

We end this chapter by giving some examples of results that follow from the preceding theorems, using particular choices of $g$.

First we note that $K(n)$ is trivially a $K$-order on $\omega$ and it is clearly finite-to-one approximable. Thus, for any rational $\epsilon$, $[\epsilon K(n)]$ is also a finite-to-one approximable order on $\omega$. For any $\epsilon$, a real $A$ is in $\mathcal{K}T([\epsilon K(n)])$ if and only if for every $n$, $K(A \upharpoonright n) \leq^+ K(n) + [\epsilon K(n)] = [(1+\epsilon)K(n)]$. $\mathcal{K}T([\epsilon K(n)])$ then gives us a set of reals that are ‘within $\epsilon$’ of being $K$-trivial. The theorems of this chapter then give us the following facts about this set. Note that for any $\epsilon$ there is a rational $\delta < \epsilon$, so we do not need to worry about the rationality of $\epsilon$ in these statements.
Corollary 2.4.17. For any $\epsilon$, there is a perfect set of reals that are within $\epsilon$ of being $K$-trivial (in the above sense).

Corollary 2.4.18. For any $\epsilon$ and any $\Delta^0_2$ real $A$, there is a $\Delta^0_2$ real $B \equiv_T A$ that is within $\epsilon$ of being $K$-trivial (in the above sense).

Moreover, since each of these $\lfloor \epsilon K(n) \rfloor$ is $\Delta^0_2$, the reals in $\mathcal{KT}(\Delta^0_2)$ must be within every $\epsilon$ of being $K$-trivial, that is, for every $\epsilon$, for every $n$ $K(A \upharpoonright n) \leq^+ \lfloor (1 + \epsilon)K(n) \rfloor$. Intuitively perhaps one would expect those reals that are arbitrarily close to being $K$-trivial to behave like the reals that are in fact $K$-trivial, but by the following corollaries we know this is not the case.

Corollary 2.4.19. There is a perfect set of reals that are within $\epsilon$ of being $K$-trivial for every $\epsilon$.

Corollary 2.4.20. For any real $B$ there is an $A \geq_T B$ such that $A$ is within $\epsilon$ of being $K$-trivial for every $\epsilon$.

Alternatively, we could have used $\log(K(\bar{n}))$ to get the same result.
Chapter 3

Compressive Power

3.1 A Perfect Set

Chapter 2 has focused mainly on weak notions of initial segment complexity, and we turn now to examine weak notions of compressive power. Several of the techniques from earlier proofs will appear, along with some new techniques for dealing with compressive power. Theorem 2.1.1 and its generalizations should give us cause to expect that proofs will be more complicated here, since it shows that making reals $K$-trivial up to some $g$ in general will not suffice to make them low for $K$ up to a given $f$. We note that, in contrast, the converse of this idea does hold: we can force reals to be $K$-trivial up to $g$ by making them low for $K$ up to the right choice of $f$. This is an analog of the direction of Nies’s Theorem that we did not disprove in the previous chapter.

**Lemma 3.1.1.** For any function $g : \omega \to \omega$ there is a function $\hat{g} : 2^{<\omega} \to \omega$ such that $LK(\hat{g}) \subseteq KT(g)$. If $g$ is finite-to-one approximable, then $\hat{g}$ can be taken to be finite-to-one approximable as well.

**Proof.** Define $\hat{g}(\sigma) = g(|\sigma|)$. If $X$ is in $LK(\hat{g})$, then by definition for all $\sigma \in 2^{<\omega}$, $K(\sigma) \leq^+ K^X(\sigma) + \hat{g}(\sigma)$. In particular, if we take $\sigma = X \upharpoonright n$ for some $n$, then we have $K(X \upharpoonright n) \leq^+ K^X(X \upharpoonright n) + \hat{g}(X \upharpoonright n) = K^X(X \upharpoonright n) + g(n)$. Now, describing the first $n$ bits of $X$ using $X$ as an oracle is no harder than describing the number $n$, even without an oracle, so we get $K^X(X \upharpoonright n) \leq^+ K(\bar{n})$. Putting these inequalities together, we get $K(X \upharpoonright n) \leq^+ K(\bar{n}) + g(n)$, and this holds for all $n$, since $n$ was chosen arbitrarily. This is the definition of $X \in KT(g)$.

If $g$ has a finite-to-one approximation $g_s$, then the approximation to $\hat{g}$ given by $\hat{g}_s(\sigma) = g_s(|\sigma|)$ will also be finite-to-one.

The preceding lemma means that we will be able to use theorems about low compressive power to derive certain results about low initial-segment complexity. Note that according to the discussion at the end of the last chapter, for $\Delta^0_2$ functions $g$ it will necessarily be the case that as long as $LK(\hat{g}) \neq 2^\omega$, $KT(g) \nsubseteq LK(\hat{g})$. 

\[\square\]
The main result of this chapter will be a construction of a perfect set in $\mathcal{LK}(\Delta^0_2)$. We will also show that $\mathcal{LK}(\Delta^0_2)$ and $\mathcal{LK}(f)$ for finite-to-one approximable functions $f$ are not closed under effective join. We recall that $K$-triviality and lowness for $K$ as properties of reals are both closed under effective join, a result of Downey, Hirschfeldt, Nies, and Stephan [5]. This then further separates the weak notions from the standard ones.

We begin with the simplest version of the theorem, building a perfect set for a single $\Delta^0_2$ order.

**Theorem 3.1.2.** For any $\Delta^0_2$ order $f : 2^{<\omega} \to \omega$ there is a perfect $\Pi^0_1$ set, $P$, of reals that are in $\mathcal{LK}(f)$. Moreover, for any real $A$ there are reals $B_0, B_1 \in P$ such that $B_0 \oplus B_1 \geq_T A$. If $A \geq \emptyset$ we have $B_0 \oplus B_1 \equiv_T A$.

**Proof.** First, we note that since $f$ is an order, by Lemma 1.6.2 it is finite-to-one approximable. We let $e$ be the index of a finite-to-one approximation of $f$, so $\phi_e = f$ and $\phi_{e,s}$ satisfies the condition that for all $n$, there are only finitely many $\sigma$ for which there exists an $s$ with $\phi_{e,s}(\sigma) = n$. We will construct a perfect tree $T \subset 2^{<\omega}$ by finite approximation such that every path through $T$ is in $\mathcal{LK}(\phi_e)$. Since $T$ is recursive the set of paths through $T$, $[T]$ will be a $\Pi^0_1$ set of reals.

The general idea of the tree will be similar to constructions from the last chapter. We are trying to build reals that all satisfy some bounding condition on their compressive power, so alongside $T$ we will construct a Kraft-Chaitin set $L$ with the goal of ensuring $K(\sigma) \leq^+ K^A(\sigma) + \phi_e(\sigma)$ for all $A \in [T]$ by enumerating requests into $L$. As before, there is the risk that putting all these requests into $L$ will push the measure of its domain too high, and the solution will be similar to the technique from the last chapter. Since the problem is that the number of possible oracles whose compressive power we are trying to bound is growing as our tree branches, we try to control where the branching levels are and make sure they only happen when the values of $\phi_e$ are high enough. Here, of course, things get more complicated because $\phi_{e,s}$ is a function on $2^{<\omega}$, not on $\omega$, so it will not be as simple as finding a height after which $\phi_{e,s}$ is large enough. However, we know that $\phi_{e,s}$ is a finite-to-one approximation, so for any $i$ there are only finitely many $\sigma$ that ever take values less than $i$. The work of the construction will be to ensure that short descriptions of these $\sigma$ only occur using as oracles initial segments of $T$ below a certain branching level, so that we will be able to control the number of times we pay for mass at the rate $2^{-i}$. Additionally, as in the proof of Theorem 2.4.10, we will use branchings as coding locations to ensure that for any real we can find a pair of branches that together compute it.

We define some terminology that we will use throughout the proof. To *kill* a node in our partially constructed tree is to make a commitment to never put nodes above it in our tree. After we kill a node, it is *dead*; before it is killed it is *living*. As long as a number $n_i$ is associated to some $R_i$ strategy it is called a *branching level*, and nodes at that height, *branching nodes*. When there is an injury and the value of $n_i$ changes, the old value is no longer a branching level, and the old nodes are no longer branching nodes. The coding locations will be the numbers $n_i + 1$ for the $n_i$ that the construction settles on. We will write $n_{i,s}$ for the value of $n_i$ at stage $s$.  


As in earlier proofs, we will define a sequence \((c_i)_{i \in \omega}\) of integers that will be the cutoffs for adding new branches to our tree. In this case, we will use \(c_0 = 0\) and \(c_i = 4^i\) for \(i > 0\).

We will need to meet the requirements

\(R_i:\) There is a level where all paths through \(T\) branch for the \((i + 1)\)st time

for \(i > 0\), and

\(S_i:\) For all \(\sigma \in 2^{<\omega}\) with \(c_i \leq \phi_e(\sigma) < c_{i+1}, K(\sigma) \leq^+ K^A(\sigma) + c_i\) for all \(A \in [T]\)

for all \(i \in \omega\).

We will order the requirements \(S_0, R_0, S_1, R_1, \ldots\).

Note that \(S_i\) will be concerned with various \(\sigma\)'s over the course of the construction as the values of \(\phi_e, s\) change. We will say that \(S_i\) is \(s\)-responsible for \(\sigma\) if \(c_i \leq \phi_e, s(\sigma) < c_{i+1}\) and \(\sigma\) is one of the length-lexicographically first \(s\) elements of \(2^{<\omega}\), and we will let \(B(i, s) = \{\sigma : S_i\) is \(s\)-responsible for \(\sigma\}\). Note that \(B(i, s)\) is always finite by the finite-to-oneness of \(\phi_e, s\).

We now give the subroutines associated with satisfying the requirements and for dealing with injuries to them.

An \(R_i\) requirement requires attention at a stage \(s\) if it does not have a number \(n_i\) associated to it. To satisfy an \(R_i\) requirement that requires attention,

1. Pick a new \(n\) bigger than anything seen yet in the construction

2. Let \(T_{s+1} = T_s \cup \{\alpha \triangledown \beta \triangledown j : \alpha\) is a living leaf node in \(T_s, |\alpha \triangledown \beta| = n, j = 0\) or \(1, \) and \(\beta(k) = 0\) for all \(k \leq |\beta|\}\}

3. Let \(n_{i,s+1} = n\). It is now associated to \(R_i\) and is a branching level.

4. \(L_{s+1} = L_s\)

An \(S_i\) requirement requires attention at a stage \(s\) if there are a \(\sigma\) that \(S_i\) is \(s\)-responsible for and a living node \(\alpha\) in \(T_s\) such that \(K(\sigma) < K^\alpha(\sigma) + \phi_e(\sigma)\) is less than the shortest description of \(\sigma\) in \(L_s\) and \(K^\alpha(\sigma) < K_s(\sigma)\) (otherwise there is no need to act). This means that we are not guaranteeing the inequality \(K(\sigma) \leq K^A(\sigma) + \phi_e(\sigma)\) for any \(A \in [T]\) that extends \(\alpha\). The strategy for meeting \(S_i\) when it requires attention is

1. Let \(v^\alpha_s(\tau)\) be the use of the computation \(U^\alpha_s(\tau) = \sigma\) giving the new shorter description for the lexicographically least pair \((\alpha, \sigma)\) that is causing \(S_i\) to require attention.

2. If \(v^\alpha_s(\tau) \leq n_{i,s}\) (or \(n_{i,s}\) not defined):
   This is fine. The level \(n_i\) is the level where we branch for the \((i + 1)\)st time, so if our new description appears before that level, we can make the adjustments to keep up with this change.
a) Let $T_{s+1} = T_s$.

b) Put a new request $(\sigma, K^\alpha_s(\sigma) + c_i)$ into $L_s$ to get $L_{s+1}$.

3. If $v^\alpha_s(\tau) > n_{i,s}$:
   This is a problem. Since $\sigma \in B(i, s)$, we want to only have descriptions of $\sigma$ appearing before the tree branches $i + 1$ many times, but we have some living node $\alpha$ of $T$ with length greater than $n_i$ which gives a new shorter description of $\sigma$. So,

a) Injure $R_i$ and run the Injury Subroutine on it.

b) Let $T_{s+1} = T_s$ and $L_{s+1} = L_s$.

The Injury Subroutine for $R_i$ at stage $s$ is:

1. Find the node $\alpha$ of $T_s$ at level $n_i$ and the string $\gamma$ such that $\alpha \downarrow \gamma$ is a living leaf node of $T_s$ and that maximizes $\sum_{\tau: U_\downarrow^\gamma(\tau) \downarrow, U^\gamma_\uparrow(\tau) \uparrow} 2^{-|\tau|}$. If there is more than one such pair, choose the leftmost.

2. For all living nodes $\beta$ at level $n_i$ keep the leaf node $\beta \downarrow \gamma$ alive; kill all other nodes above $\beta$.

3. Set all $R_k$ for $k \geq i$ to requiring attention (i.e. disassociate from each $R_k$ its $n_k$).

The idea of the Injury Subroutine is to salvage what we can of the tree when we find that we are failing in our goal of having all $\sigma$ with $c_i \leq \phi_e(\sigma) < c_{i+1}$ get all their descriptions on initial segments of the tree before it branches $i + 1$-many times. We know we will have to move the $i + 1$st branching level up, but in order to make the most of this situation we keep alive the branch on which the most mass has converged. This essentially ‘locks in’ at least as much mass as was used by the universal machine to make this injury, and prevents it from being used against us again at a later stage. We keep all our branches identical except at the branching nodes so that we can use the branching levels as coding locations.

Now using these subroutines, we give the skeleton of the construction:

**Stage 0:** Let $T_0 = \emptyset$, $L_0 = \emptyset$, $B(i, 0) = \emptyset$, $n_{i,0}$ undefined for all $i$.

**Stage $s+1$:**

1. Calculate $\phi_{i,s+1}(\sigma)$ and $K^\alpha_{s+1}(\sigma)$ for all living branches $\alpha$ in $T_s$ and the length-lexicographically first $(s + 1)$-many $\sigma$’s. From this, adjust $B(i, s + 1)$ where necessary.

2. If one of the first $s + 1$-many requirements requires attention, execute the strategy to satisfy the one with highest priority, performing the Injury Subroutine if necessary.

This is all there is to the construction. We let $T = \bigcup_s T_s$, $L = \bigcup_s L_s$, $n_i = \lim_s n_{i,s}$, and we are now left to verify that it works as desired. We start by showing the requirements all eventually become satisfied.
Lemma 3.1.3. Each $R_i$ requirement is injured only finitely often.

Proof. For a given $i$, there are only finitely many $S_j$ requirements that come before $R_i$ in the ordering of the requirements, and these are the only ones that can cause an injury to $R_i$. Each injury caused by an $S_j$ at a stage $s$ is due to some $\sigma$ that $S_j$ is $s$-responsible for, and since $\phi_{e,s}$ is a finite-to-one approximation, there are only finitely many $\sigma$ that $S_j$ is ever responsible for. Thus, only finitely many $\sigma$ can cause an injury to $R_i$. For a proof by induction, fix $i$ and assume all $R_j$ with $j < i$ are injured only finitely often. We fix some $\sigma$ that could cause an injury to $R_i$ (i.e., some $S_j$ with $j < i$ has responsibility for $\sigma$), and show that it can cause only finitely many injuries. Since $i$ is least such that $R_i$ gets injured infinitely often, it must be the case that $S_i$ has $s$-responsibility for $\sigma$ for infinitely many $s$.

Now, $\phi_{e,s}(\sigma)$ will eventually settle, so we can assume we are at some stage $s$ after which this has happened, so $c_i \leq \phi_{e,s}(\sigma) < c_{i+1}$. Let us also assume that at stage $s$ we have $K_s(\sigma) = K(\sigma)$, that is, $K$ has reached its final value on $\sigma$, and that all requirements $R_j$ for $j < i$ will never be injured again. The string $\sigma$ will cause an injury to the requirement $R_i$ at stage $t > s$ whenever we find a new shorter description of $\sigma$ using more than $n_{i,t}$-many bits of a living path in $T_i$. This provokes a run of the Injury Subroutine. When the Injury Subroutine runs we find the branch above $n_{i,t}$ on which the most mass has converged and keep this branch alive above all nodes at height $n_{i,t}$, killing all other branches. When this happens, either the branch $\alpha$ on which the new shorter description of $\sigma$ converged is kept alive, or it is killed. In either case the amount of mass that converged on the branch that was selected for being the most massive must be at least $2^{-K(\sigma)}$, since to cause an injury the new shorter description of $\sigma$ must be shorter than $K(\sigma)$. Thus, a run of the Injury Subroutine adds at least $2^{-K(\sigma)}$ much mass to one of the living branches below $n_{i,t+1}$.

Now, there are always $2^i$ many living paths to level $n_{i,t}$. Since no earlier $R_j$ requirement is ever injured, for each stage $t > s$ the $2^i$-many living paths at level $n_{i,t+1}$ are finite extensions of the $2^i$-many living paths at $n_{i,t}$. Each injury that $\sigma$ causes to $R_i$ adds at least $2^{-K(\sigma)}$ much mass to the domain of the universal machine that uses one of these paths as an oracle. Since the new oracles are finite extensions of the old ones, the mass that converges along them is never lost. For each oracle, the mass of the domain of the universal machine relative to that oracle is bounded by 1, so there can be no more than $2^i \cdot 2^{K(\sigma)}$ many injuries caused by $\sigma$.

Since only finitely many $\sigma$ can injure $R_i$ and each one can only do so finitely often, $R_i$ can only be injured finitely often.

Lemma 3.1.4. Every requirement is eventually satisfied.

Proof. By the preceding lemma, each requirement is injured only finitely often, so we just need to ensure that once this has stopped happening it can act and be satisfied. For the $R_i$ requirements this is trivial. For an $S_i$ requirement, eventually we will reach a stage where $S_i$ has $s$-responsibility for all the $\sigma$ it will ever have responsibility for. When it has stopped injuring $R_i$, it may need to act for each $\sigma$, putting requests for shorter descriptions of $\sigma$ into $L$. Each time it puts a request into $L$, the length of next description of $\sigma$ it may have to
respond to decreases by at least 1. Since there can’t be descriptions of negative length, this can happen only finitely often for each $\sigma$. Thus, it will act at most finitely often before being satisfied.

We now know that the requirements are satisfied, but the $S_i$ requirements are satisfied just by putting requests into $L$, without considering what effect this has on the weight of $L$. We need to show that this mass is in fact bounded. This part of the proof is significantly more difficult than the corresponding part of the proofs from the last chapter.

First, we define the subtree

$$T' = \{ \alpha \in T : \alpha \text{ is not killed in any run of the Injury Subroutine} \}.$$ 

$T'$ is the living subtree of $T$. Since any $(\sigma, l)$ goes into $L$ only when a new description of $\sigma$ converges on some living node of $T$, we may divide $L$ into two parts:

$$L' = \{ (\sigma, l) \in L : (\sigma, l) \text{ was put into } L \text{ in response to a description converging on an initial segment in } T' \}$$

$$L'' = \{ (\sigma, l) \in L : (\sigma, l) \text{ was put into } L \text{ in response to a description converging on an initial segment not in } T' \}.$$ 

In essence, $L'$ is the part of $L$ that is really necessary to the proof: the descriptions of those $\sigma$ that get short descriptions on paths through $T$, while $L''$ is the part of $L$ that is wasted mass, paid in for paths which are eventually killed. We will bound the mass of these two sets separately.

First, we may make some assumptions that will increase the mass but allow the proofs to run more smoothly. In particular, instead of monitoring $\sigma$’s and reacting when new short descriptions of $\sigma$’s emerge on the living tree, we can monitor the action of $U$ and assume that any time $U^\alpha(\tau)$ converges for a new $\tau$ it is a description of some $\sigma$ we were monitoring. That is, $U$ converging on a new $\tau$ with oracle $\alpha$ either causes us to put some request $(U^\alpha(\tau), |\tau| + c_i)$ into $L$ for the corresponding $i$, or causes an injury to some $R$. We call a string $\alpha$ exact for a string $\tau$ if $U^\alpha(\tau) \downarrow$ and the use of this computation is $v^\alpha(\tau) = |\alpha|$. That is, $\alpha$ is exactly the oracle needed for this computation. Here we make use of standard conventions on uses of computations.

We bound the mass of $L$ by instead bounding the quantity

$$\Delta = \sum_{(\alpha, \tau, i) : \alpha \in T \text{ is exact for } \tau \text{ and } \alpha \text{ is alive at a stage } s \text{ when } U^\alpha_s(\tau) \downarrow \text{ and } c_i \leq \phi_{c_i}(U^\alpha_s(\tau)) < c_{i+1}} 2^{-|\tau| - c_i}.$$
Clearly the mass of $L$ is less than $\Delta$, and if we define $\Delta'$ and $\Delta''$ similarly but with the $T'$ and $T''$ where necessary, then the masses of $L'$ and $L''$ are less than $\Delta'$ and $\Delta''$, respectively. We start with $\Delta'$

**Lemma 3.1.5.** $\Delta' \leq 2$

*Proof.* We count the mass contributed by each segment of the living subtree, $T'$. Since there are no injuries to the paths of $T'$, we know that our $n_{i,s}$ are always at their final values $n_i$.

First, for each $\nu$ in $2^{<\omega}$, let $\alpha(\nu)$ be the node in $T'$ such that $|\alpha(\nu)| = n_{|\nu|}$ and for all $i < |\nu|$, $\alpha(\nu)(n_i + 1) = \nu(i)$. That is, $\alpha(\nu)$ is the partial path through $T'$ that follows $\nu$ at the branching levels.

Now define $Q(\nu) = \{\tau \in 2^{<\omega} : \mathbb{U}^{\alpha(\nu)}(\tau) \downarrow \text{ and } \mathbb{U}^{\alpha(\nu')}(\tau) \uparrow \text{ for } \nu' < \nu\}$. This is the set of strings on which $\alpha$ converges, using as an oracle the segment of $\alpha(\nu)$ that does not include any shorter $\alpha(\nu')$. Note that each pair $(\alpha, \tau)$ such that $\alpha$ is exact for $\tau$ contributes exactly one $\tau$ to exactly one $Q(\nu)$ (the one for the least $\nu$ such that $\alpha < \alpha(\nu)$).

Finally, we define the mass that is contributed to $\Delta'$ for each $Q(\nu)$. Let $m(\nu) = \sum_{\tau \in Q(\nu)} 2^{-|\tau|}$. Here $m(\nu)$ is the measure of the strings that converge as descriptions along the path $\alpha(\nu)$ with use of computation $\nu^{\alpha(\nu)}(\tau)$ at least $n_{|\nu|-1}$ (here let $n_{-1} = 0$). Since these convergences do not cause injuries, we must have that $\phi_{e,s}(\mathbb{U}_s^{\alpha(\nu)}(\tau)) > c_{|\nu|}$ for $\tau \in Q(\nu)$.

We recall that $c_0 = 0$ and $c_i = 4^i$ for $i > 0$, and note that $c_i \geq i$ for every $i$, so we have $\phi_{e,s}(\mathbb{U}_s^{\alpha(\nu)}(\tau)) \geq |\nu|$ for all $\tau \in Q(\nu)$.

Thus, for each pair $(\alpha, \tau)$ with $\alpha$ exact for $\tau$, we put at most $2^{-|\tau|-|\nu|+1}$ much mass into $\Delta'$. Here the extra factor of 2 accounts for the fact that we may have to pay in for multiple decreasing values of $\phi_{e,s}$ as this converges to its final value. This can’t do more than double the mass paid for each $\tau$. It follows that the total amount of mass that we put into $\Delta'$ for all $\tau$ in $Q(\nu)$ is $2 \cdot m(\nu) \cdot 2^{-|\nu|}$. Since every pair $(\alpha, \tau)$ with $\alpha$ exact for $\tau$ causes $\tau$ to go into exactly one $Q(\nu)$, we can bound the total mass paid into $\Delta'$ by summing over all the $Q(\nu)$.

This gives us

$$\Delta' \leq \sum_{\nu \in 2^{<\omega}} \frac{2 \cdot m(\nu)}{2^{|\nu|}}.$$

Now, it is clear that any $m(\nu)$ must be less than 1, since it is the measure of some subset of the domain of the universal machine relative to some oracle (namely, $\alpha(\nu)$). However, since $\alpha(\nu)$ extends $\alpha(\nu')$ for $\nu \succ \nu'$, it is also true that for any $\nu$ we have $\sum_{\nu' \leq \nu} m(\nu') \leq 1$. If we add this sum for every $\nu$ of a given length $n$ we get the double sum $\sum_{|\nu| = n} \sum_{\nu' \leq \nu} m(\nu') \leq 2^n$. Now, we note that for any $\nu'$ of length less than $n$ there are $2^{n-|\nu'|}$-many $\nu'$’s of length $n$ with $\nu' \prec \nu$, so in this sum each term $m_{\nu'}$ for $\nu'$ of length less than $n$ is counted $2^{n-|\nu'|}$-many times. Thus, we can rewrite it as the single sum $\sum_{|\nu'| \leq n} 2^{n-|\nu'|} m(\nu')$. Since we know this sum is bounded by $2^n$, we can divide every term by $2^n$ to get the inequality $\sum_{|\nu'| \leq n} 2^{-|\nu'|} m(\nu') \leq 1$. Since this
sum is bounded by 1 for any \( n \), the limit of these partial sums, \( \lim_{n \to \infty} \sum_{\nu' \leq n} 2^{-|\nu'|} m(\nu') \leq 1 \) must also be no greater than 1. This sum is \( \frac{1}{2} \) our bound on \( \Delta' \), so we have that \( \Delta' \leq 2 \).

Now we need to find a bound on \( \Delta'' \). This will be more difficult, because the tree can be changing as we pay into \( L'' \). To this end, we first prove a lemma that will help us bound the mass that is wasted by an injury.

**Lemma 3.1.6.** For any injury to requirement \( R_i \) at stage \( s \) in the construction, the amount that has been paid into \( \Delta \) on the paths above \( n_{i,s} \) (those kept and those killed) is no more than \( \frac{1}{2} \cdot \frac{1}{2} c_i \) times the mass, \( m \), that converges on the path chosen by the run of Injury Subroutine for this injury.

**Proof.** Let us fix an \( R_i \) and \( s \) such that the Injury Subroutine is run for \( R_i \) at stage \( s \). A run of the Injury Subroutine will find the path \( \alpha \) of length \( n_{i,s} \) and the string \( \gamma \) such that \( \alpha \dashv \gamma \) is a living leaf node in \( T_s \) and such that \( m = \sum_{\tau} 2^{-|\tau|} \cdot \phi_{e,s}(U_{\tau}(\gamma)) \) is maximal. It then keeps this and every other living path that follows \( \gamma \) from height \( n_{i,s} \) alive and kills all other nodes above \( n_{i,s} \). What we claim in this lemma is that

\[
m \cdot 2^{-c_i - 1} \geq \sum_{\eta \text{ a living leaf node}} \sum_{\tau: U_{\tau}(\gamma) \downarrow, U_{\eta}(\tau) \uparrow} 2^{-|\tau| - \phi_{e,s}(U_{\tau}(\gamma))} \cdot 2.
\]

Here again the extra 2 in the sum is to account for \( \phi_{e,s} \)'s converging down to its value at stage \( s \) possibly forcing us to pay for the same \( \tau \) at a slightly higher rate. We now prove that this bound holds.

First, note that at stage \( s \) there is some amount of tree \( T_s \) above level \( n_{i,s} \) with, say, \( k \) many branching levels above and including \( n_{i,s} \). This gives us \( 2^k \)-many choices for \( \gamma \), each of which is divided by the branching levels into \( k \) segments. We will call the rank of one of these segments the number of branchings below it and above and including \( n_{i,s} \), so before the injury there are \( 2^{i+l} \)-many living segments of rank \( l \) on \( T_s \) above \( n_{i,s} \). We know the most mass that can have converged on any of these segments is \( m \), since \( m \) is the mass that has converged along the most massive \( \gamma \). This will give us a very rough upper bound.

Now, for any mass that has converged on one of the living segments of rank \( l \) above \( n_{i,s} \), the \( \phi_{e,s} \) values of the \( U \) images of the strings that are converging must be at least \( c_i + l \) (there are \( i + l \)-many branchings below segments of rank \( l \)) or we would have had an earlier injury and a different \( T_s \). This means with our overestimate of \( m \) much mass on each segment, we get an upper bound for what we pay into \( \Delta \) of

\[
\sum_{l=1}^{k} \frac{m \cdot (2^{i+l})}{2^{c_i + l}} \cdot 2.
\]
The upper bound we are trying to get is \( \frac{m}{2^{c_i+1}} \) so we want
\[
\frac{m}{2^{c_i+1}} \geq \sum_{l=1}^{k} \frac{m \cdot (2^{i+l})}{2^{c_i+l}} \cdot 2.
\]

We have the same \( m \) on both sides so we can cancel these out and show that
\[
\frac{1}{2^{c_i+1}} \geq \sum_{l=1}^{k} \frac{2^{i+l+1}}{2^{c_i+l}}
\]
or, equivalently, that
\[
1 \geq \sum_{l=1}^{k} \frac{2^{c_i+i+l+2}}{2^{c_i+l}}.
\]

We recall that \( 1 \geq \sum_{l=1}^{k} \frac{1}{2^l} \), so we now need only show that
\[
\frac{1}{2^l} \geq \frac{1}{2^{c_i+l-c_i-l-2}}.
\]

This reduces to showing that \( \forall i \geq 0 \forall l \geq 1 \) \([c_i+l \geq c_i + i + 2l + 2]\). Recall that \( c_0 = 0 \) and \( c_i = 4^i \) for all \( i > 0 \) and then this follows by elementary computations.

Thus, the amount of mass paid into \( \Delta \) that could be wasted by this injury (i.e., that could end up in \( \Delta'' \)) is bounded by \( \frac{m}{2^{c_i+1}} \), where \( m \) is the mass that converges on the most massive path above \( n_{i,s} \).

With this bound on mass wasted by injuries, we can finally find an a priori bound for \( \Delta'' \).

**Lemma 3.1.7.** \( \Delta'' \leq 2 \)

**Proof.** Any mass that goes into \( \Delta'' \) does so because a description of some \( \sigma \) converged on a path \( \alpha \) through \( T_s \) that was killed by a subsequent run of the Injury Subroutine. Thus, we can bound \( \Delta'' \) by bounding the contributions to it by each injury to an \( R_i \), for which we will use Lemma 3.1.6. As it will be important, we recall again that \( c_0 = 0 \) and \( c_i = 4^i \) for \( i > 0 \).

A run of the Injury Subroutine for \( R_i \) at stage \( s \) finds the most massive path above \( n_{i,s} \) and fixes that much mass on one of the \( 2^i \) new living branches below the new \( n_{i,s+1} \). Note that there are always \( 2^i \) living paths below \( n_{i,t} \). As a simple example, consider the case \( i = 0 \). Since there are no earlier \( R \) requirements, \( R_0 \) will not be disturbed by injuries to other \( R_i \)'s, and each injury to \( R_0 \) will settle some extension of the previous stem of \( T \) to be the new stem. Each of these will settle some more mass on the one path below \( n_{0,s} \).

Let \( k \) be the number of injuries to \( R_0 \) during the construction (we know by Lemma 3.1.3 that there are only finitely many). If we let \( m \) be the mass that converges along the final stem
of the tree, that is, on the initial segment of $T'$ below $n_0$, we can split this into $m_1, m_2, \ldots m_k$, where each $m_j$ is the amount of mass that converges on the segment of the final stem that is settled by the $j$th injury to $R_0$. Since $m$ is the mass that converges into the domain of $\mathcal{U}$ for a given oracle, $m$ can be no greater than 1. Clearly $m = \sum_j m_j$. We can then use Lemma 3.1.6 to bound the amount of mass that goes into $\Delta''$ while $R_0$ is settling. The $j$th injury to $R_0$ wastes no more than $\frac{m_j}{\frac{2}{3}+\frac{1}{t}} = \frac{m_j}{2}$. Thus, the total amount wasted by injuries to $R_0$ is no more than $\sum_{j=1}^k \frac{m_j}{2} \leq \frac{1}{2}$.

In general, of course, there will be injuries to lower priority requirements that will complicate matters. As the simplest possible example of this, consider $R_1$. There will always be 2 branches below $n_{1,s}$, and each injury to $R_1$ will find the most massive living path above $n_{1,s}$ and attempt to fix that much mass onto one of the two branches below $n_{1,s+1}$. However, a subsequent injury to $R_0$ only chooses one of these two paths, so it could waste mass that injuries to $R_1$ have fixed on the other. The mass that is on the living branches killed by this $R_0$-injury will be accounted for by $R_0$, but there was also mass wasted by $R_1$ injuries in fixing the mass that $R_0$ is wasting. If we look at what happens in the construction between the $j-1$st and $j$th injuries to $R_0$, we see that at most $m_j$ much mass can converge on either of the paths below $n_{1,s}$, since this is the most mass on any path when the $j$th injury to $R_0$ occurs. The sum of the masses of the most massive branches for each run of the Injury Subroutine for $R_1$ in this time interval can not exceed $2m_j$, since at most $m_j$ much mass can be fixed on each path. By Lemma 3.1.6, the amount of mass wasted by these injuries must then be bounded by $2 \cdot \frac{m_j}{\frac{2}{3}+\frac{1}{t}} = \frac{2m_j}{2t}$. We can now sum over injuries to $R_0$ and get that the total amount of mass paid into $\Delta''$ to account for injuries to $R_1$ during the time when $R_0$ is settling is no more than $\sum_{j=1}^k \frac{m_j}{2t} = \frac{1}{2t}$.

Now there may be more injuries to $R_1$ after $R_0$ has stopped being injured, but for the mass wasted this way we can argue as we did for $R_0$, since the tree below $n_0$ will never change after this point. Each run of the Injury Subroutine for $R_1$ after this point fixes some mass on one of the 2 paths below $n_{1,s}$, and for increasing $s$ the new paths are end extensions of the old ones. It is enough to use a rough bound of 1 on the mass that converges on either of these paths after $n_0$ converges, so again by Lemma 3.1.6 we get that the most mass that can go into $\Delta''$ due to these injuries is $2 \cdot \frac{1}{2t} = \frac{1}{t}$. Thus, the total mass that is wasted by injuries to $R_1$ is no more than $\frac{1}{2t} + \frac{1}{2t} = \frac{1}{t}$.

The argument for any $i > 0$ will be similar. We can imagine the $2^i+1$ living paths between $n_{l,s}$ and $n_{l+1,s}$ as reservoirs for mass (these reservoirs are essentially approximations to the $Q(\nu)$ from Lemma 3.1.5). As injuries occur the $n_{l,s}$ move up the tree, and so these reservoirs can lose their mass, either pouring it into lower reservoirs or spilling it out on the ground. In particular, an injury to $R_i$ moves $n_i$ up but doesn’t move $n_{i-1}$ (again, let $n_{-1} = 0$). All the mass that was in the reservoirs (on the paths) that are kept alive then goes into the $2^i$ reservoirs that start at $n_{i-1}$ (i.e., the $Q(\nu)$, for $|\nu| = i$). Unfortunately, we don’t know how much mass will be in the reservoirs that are poured into lower reservoirs, and how much will
be spilled out. All we know is that for whichever reservoir above \( n_{i,s} \) has the most mass, at least that much mass must be on the path that is kept alive by the run of the Injury Subroutine, so at least that much mass ends up one of \( R_i \)'s reservoirs. This is exactly why we set up the Injury Subroutine to work like it does. To achieve a gross upper bound, we will assume an injury saves only as much mass as is in the most massive reservoir above it and spills out all the rest. Lemma 3.1.6 gives us a bound on the mass that is wasted by this injury that is proportional to the mass that is added to this reservoir. Now if we keep track of how much mass is put into \( Q(\nu)_s \)-reservoirs by runs of the Injury Subroutine for \( R_{|\nu|} \) we can get a bound on how much total mass is wasted by these runs.

Let us consider the worst possible case for some \( R_i \), that is, the most mass that can ever go through its reservoirs. When \( R_0 \) is still active, say between the \( j-1 \)th and \( j \)th injuries to \( R_0 \), we could have \( R_i \) fill each of its \( 2^i \)-many reservoirs with \( m_j \) much mass, then have an injury to \( R_{i-1} \) take the mass from just one of these reservoirs and empty all the rest. \( R_i \) could then fill all its reservoirs with \( m_j \) much mass again, only to have an injury to \( R_{i-1} \) repeat this and fill a different one of its reservoirs. This could happen until all of \( R_{i-1} \)'s and \( R_i \)'s reservoirs had \( m_j \) much mass, at which point \( R_{i-2} \) could be injured and take the mass for only one higher reservoir and waste the rest. This can keep happening, with enough mass going through \( R_i \)'s reservoirs to fill all the reservoirs for \( R_j \)'s with \( 0 < j < i \), at which point \( R_0 \) can have its \( j \)th injury and take \( m_j \) much mass and empty all the rest. If we keep track of all the times \( R_i \) will have to fill one of its own reservoirs with \( m_j \) much mass in this worst case, we get \( \prod_{p=0}^{i} 2^p \). This gives us that \( m_j \prod_{p=0}^{i} 2^p \) much mass goes through \( R_i \)'s reservoirs. If we assume that all of this mass comes in due to injuries \( R_i \) is causing (some might come in just by new descriptions converging on the \( \alpha(\nu)_s \), in which case, \( R_i \) won’t put any mass into \( \Delta'' \)), the amount wasted by injuries to \( R_i \) in this time interval is \( \frac{m}{2^i+1} \prod_{p=0}^{i} 2^p \). This gives us that the total amount wasted by \( R_i \) in the construction before \( n_0 \) settles is

\[
\sum_{j=1}^{k} \frac{m_j}{2^j+1} \prod_{p=0}^{i} 2^p = \sum_{j=1}^{k} m_j \frac{2^{2^j+i}}{2^j+1} \leq \frac{m^2 2^{2^i+i}}{2^i+1} \leq \frac{2^{2^i+i}}{2^i+1}.
\]

Now, of course, the worst that can happen after this is that \( R_i \) needs to fill all its own reservoirs with enough mass enough times to fill each of \( R_1 \)'s reservoirs with total mass up to 1 (an easy upper bound). This can cost at most

\[
\prod_{p=1}^{i} \frac{1}{2^p} \prod_{p=0}^{2^p} = \frac{2^{2^i+i}}{2^i+1} \leq \frac{2^{2^i+i}}{2^i+1}.
\]

After this, \( R_i \) might be forced to fill all of \( R_2 \)'s reservoirs with 1 total mass each, and then each of \( R_3 \)'s, and so on, until no lower priority requirement gets injured and it can fill all of its own reservoirs in peace. This is just summing over the lower index in the product, so the total amount wasted by injuries to \( R_i \) is bounded by
We can replace the lower bound of the product with 0 to get an upper bound, since this is less than
\[ \sum_{l=0}^{i} \prod_{p=0}^{i} \frac{1 \cdot 2^p}{2^{c_i+1}}. \]

Using \( c_0 = 0 \) and \( c_i = 4^i \) for \( i > 0 \), it follows from some calculus that this last term in less than \( 2^{-i} \). This is a bound on the amount of mass that goes into \( \Delta'' \) due to injuries to \( R_i \). Thus, summing over all \( i \), we get that the amount of mass that goes into \( \Delta'' \) due to any injury is no more than 2, which was our goal.

\[ \square \]

Lemmas 3.1.5 and 3.1.7 together give us that \( \Delta \leq 4 \), and since \( \Delta \) was a bound on the mass of our Kraft-Chaitin set, we have shown that this is an actual Kraft-Chaitin set. The machine that corresponds to \( L \) will by construction be a witness to the lowness for \( K \) up to \( \phi_e \) (and so, up to \( f \)) of every path through \( T \). Since the living subtree of \( T \) is by construction a perfect set (the \( R_i \) requirements introduce all the branching levels) we have built our perfect set \( \mathcal{P} \) of elements of \( \mathcal{L} \mathcal{K}(f) \). We note again that since \( T \) is recursive, \([T]\) is \( \Pi^0_1 \).

Finally, we want to show that we can join paths through \( T \) to compute any real. This follows from our use of branching levels as coding locations. Since each run of the Injury Subroutine for some \( R_i \) keeps identical paths above the branching level \( n_i \), the only places that paths through \( T \) will differ is at the level right after the branching levels. Thus, for a given real \( A \), there are paths \( B_0 \) and \( B_1 \) through \( T \) such that \( B_0 \) follows the bits of \( A \) at the branching levels and \( B_1 \) follows the bits of \( \bar{A} \), the complement of \( A \). Then \( B_0 \oplus B_1 \) can merely compare the two reals \( B_0 \) and \( B_1 \) and output the values of \( B_0 \) at the places where they differ.

Note that as long as \( A \geq_T \emptyset' \), \( A \) can run the construction of \( T \) and will be able to compute the stages when the \( n_{i,s} \) converge to their final values. This is enough to compute where the coding locations of a path through \( T \) must be and what the values are away from the coding locations, so to compute \( B_0 \) or \( B_1 \) it can insert either its own bits or the bits of its complement into the coding locations. This gives us that \( A \equiv_T B_0 \oplus B_1 \) in this case.

\[ \square \]

One interesting feature of the proof of Theorem 3.1.2 is that we get a single machine that witnesses the lowness for \( K \) up to \( f \) of all the elements of \( \mathcal{P} \), so the constant for \( K(\sigma) \leq^+ K^A(\sigma) + f(\sigma) \) is the same for every real in \( \mathcal{P} \), which is stronger than what we needed. This is another break from the behavior of traditional lowness for \( K \). In the classic case, it is a result of Chaitin [4] that there are only finitely many reals that are low for \( K \) using a given constant, while in the weaker case we can find a perfect set of reals that all use the same constant.
3.2 Examples

As for many of the results of Chapter 2, we can apply Theorem 3.1.2 for a particular choice of $f$ to get some interesting results. First, we discuss applications to effective dimension. The effective dimension of a real is another measure of its complexity. Effective Hausdorff dimension and effective packing dimension were originally defined in terms of martingales, but they have equivalent definitions that will be more applicable in our framework. Essentially, the effective dimension tells us what proportion of the bits are necessary to describe the real, asymptotically.

**Definition 3.2.1.** (Mayordomo [10]) The *effective Hausdorff dimension* of a real $S$ is

$$\dim(S) = \liminf_{n \to \infty} \frac{K(S \upharpoonright n)}{n}$$

(Athreya, Hitchcock, Lutz, and Mayordomo [1]) The *effective packing dimension* of a real $S$ is

$$\text{Dim}(S) = \limsup_{n \to \infty} \frac{K(S \upharpoonright n)}{n}$$

We can relativize each of these notions to a given real by allowing the universal machine access to that real as an oracle when computing Kolmogorov complexity, and we can denote the relativized version $\dim^X$ and $\text{Dim}^X$.

Each notion to effective dimension has a related notion of *lowness* for dimension.

**Definition 3.2.2.**
A real $X$ is *low for effective Hausdorff dimension* if for every real $A$, $\dim^X(A) = \dim(A)$.

A real $X$ is *low for effective packing dimension* if for every real $A$, $\text{Dim}^X(A) = \text{Dim}(A)$.

We use $\mathcal{LHD}$ for the set of reals that are low for Hausdorff dimension and $\mathcal{LPD}$ for the set of reals that are low for packing dimension. We can use Theorem 3.1.2 to construct a perfect set of reals that are low for both notions of dimension.

**Corollary 3.2.3.** The intersection $\mathcal{LHD} \cap \mathcal{LPD}$ contains a perfect $\Pi^0_1$ set of reals. Additionally, $\mathcal{LHD} \cap \mathcal{LPD}$ does not form a Turing ideal and in fact any real above $0'$ is Turing-equivalent to the join of two reals in $\mathcal{LHD} \cap \mathcal{LPD}$.

**Proof.** Apply Theorem 3.1.2 to $f(\sigma) = \log(|\sigma|)$. This $f$ is certainly total on $2^{<\omega}$ and finite-to-one approximable so we will get a perfect set $\mathcal{P}$ of reals that are in $\mathcal{LK}(\log|\sigma|)$. This gives us that each $X$ in $\mathcal{P}$ has $K(\sigma) \leq^+ K^X(\sigma) + \log|\sigma|$ for all $\sigma$. For any real $R$ we have $K^R(\sigma) \leq^+ K(\sigma)$, since the universal oracle machine can simulate the universal oracle-free machine, so for the $X$’s we build we get $K(\sigma) - \log|\sigma| \leq^+ K^X(\sigma) \leq^+ K(\sigma)$.
Then for any $A$ and $n$, we can consider $\sigma = A \upharpoonright n$ and get $K(A \upharpoonright n) - \log |A \upharpoonright n| \leq K^X(A \upharpoonright n) \leq K(A \upharpoonright n)$. Dividing through by $n$, we get

$$\frac{K(A \upharpoonright n)}{n} - \log \frac{n}{n} \leq \frac{K^X(A \upharpoonright n)}{n} \leq \frac{K(A \upharpoonright n)}{n}.$$

Now if we take the limit as $n \to \infty$ the $\log \frac{n}{n}$ term goes to 0, so we get that $\lim \inf \frac{k^X(A \upharpoonright n)}{n} = \lim \inf \frac{K(A \upharpoonright n)}{n}$ for every $A$, and so $X$ is low for effective Hausdorff dimension. The same equality holds for lim sup so $X$ is also low for effective packing dimension. This observation that the reals in $\mathcal{L}K(\log |\sigma|)$ are all low for effective Hausdorff and effective packing dimensions was first made by Hirschfeldt and Weber in [7].

A perfect set of reals low for effective Hausdorff dimension and effective packing dimension was independently constructed by Lempp, Miller, Ng, Turetsky, and Weber [8].

We can choose other functions $f$ to get other interesting results. For instance, it is clear that $K(\sigma)$ is a finite-to-one approximable $K$-order on $2^{<\omega}$, and so for any rational $\epsilon$ so is $[\epsilon K(\sigma)]$. If we consider a real $A$ in the set $\mathcal{L}K([\epsilon K(\sigma)])$, by definition we get that for any $\sigma$, $K(\sigma) \leq K^A(\sigma) + [\epsilon K(\sigma)]$. Reorganizing this inequality gives us that for any $\sigma$ $[(1-\epsilon)K(\sigma)] \leq [K^A(\sigma)]$, that is, $A$ is within $\epsilon$ of being low for $K$. Our Theorem 3.1.2 then shows that for any $\epsilon$, being within $\epsilon$ of lowness for $K$ is a much weaker notion. By the density of the rationals, we can ignore the requirement that $\epsilon$ be rational in the statement of this corollary.

**Corollary 3.2.4.** For any $\epsilon$ there is a perfect set of reals that are within $\epsilon$ of being low for $K$ (in the above sense), and the set of these reals is not closed under effective join.

### 3.3 A Universal Tree

It should perhaps be no surprise, given much of the content of Chapter 2, that we can generalize Theorem 3.1.2 to build a tree that works for all $\Delta^0_2$ orders on $2^{<\omega}$. As in that context the proof will be significantly more complicated, and so we will not be able to get the same strictness in our result. The generalization we get is the following theorem.

**Theorem 3.3.1.** There is a perfect set $Q$ such that $Q \subseteq \mathcal{L}K(\Delta^0_2)$. Moreover, for any real $A$, there exist $B_0, B_1 \in Q$ such that $A \leq_T B_0 \oplus B_1$.

**Proof.** As always, we use Lemma 1.6.2 to reduce showing membership in $\mathcal{L}K(\Delta^0_2)$ to showing lowness for $K$ up to all finite-to-one approximations from $2^{<\omega}$ to $\omega$. We use $(\phi_{\epsilon,s})$ as an effective listing of recursive approximations of functions from $2^{<\omega}$ to $\omega$.

The proof of this result will combine the techniques from Theorem 3.1.2 with those from Theorems 2.2.2 and 2.4.10. We will build a tree $T$ with a similar injury construction to
ensure that descriptions of strings with low $\phi_e$ values all converge before too many branchings have occurred while also using branching nodes to record guesses as to the behavior of the $\phi_{e,s}$’s. Taking only the paths that guess correctly whether each $\phi_{e,s}$ is a total finite-to-one approximation will give us a perfect subtree of $T$ that will have every path in $LK(\Delta_0^2)$. Again, using this guessing strategy is necessary because the Kraft-Chaitin sets we construct must be recursively approximable, so we can not use knowledge of which $\phi_{e,s}$ are finite-to-one approximations in their construction.

The general form of the construction will be similar to that of Theorem 3.1.2. Alongside $T$ we construct a collection of Kraft-Chaitin sets $(L_e)$, each of which will attempt to witness that every path through $T$ that guesses that $\phi_{e,s}$ is a finite-to-one approximation is low for $K$ up to $\phi_e$. We will use some of our branching nodes to record the guesses of paths that go through them, but since we are trying to construct a perfect set rather than just one real in $LK(\Delta_0^2)$, we will also have use some branching nodes not to record guesses but to keep our subtree growing. We will call the first kind of branching nodes guessing nodes and the second kind coding nodes, since they will eventually be used as coding locations for showing that we can join paths through $T$ above any given real. Note that unlike the constructions in Chapter 2 we will be attempting to keep paths through our subtree identical except for at coding locations, so we will also be able to talk about branching levels and coding levels.

We will modify our constant sequence from Theorem 3.1.2 using our subtree growing. We will call the first kind of branching nodes guessing nodes in $LK$ through them, but since we are trying to construct a perfect set rather than just one real in $LK(\Delta_0^2)$, we will also have use some branching nodes not to record guesses but to keep our subtree growing. We will call the first kind of branching nodes guessing nodes and the second kind coding nodes, since they will eventually be used as coding locations for showing that we can join paths through $T$ above any given real. Note that unlike the constructions in Chapter 2 we will be attempting to keep paths through our subtree identical except for at coding locations, so we will also be able to talk about branching levels and coding levels.

We will modify our constant sequence from Theorem 3.1.2 using $c_i = 4^{i+1}$.

For a given $\alpha \in 2^{<\omega}$, we will say $e$ is a member of $\alpha$ if $\alpha(e) = 1$ and denote this $e \in \alpha$. We will say that $\alpha$ is correct if for every $e \leq |\alpha|$, $e$ is a member of $\alpha$ if and only if $\phi_{e,s}$ is a total finite-to-one approximation. For each $\alpha$ we define a function that guesses that $\alpha$ is correct: $\psi_{\alpha,s}(\sigma) = \min\{\phi_{e,s}(\sigma) : e \in \alpha\}$.

The requirements that we will try to meet are:

$R_\alpha$: For all paths through $T$ that follow $\alpha$ at the guessing nodes, there is a level where they all branch twice more

for all $\alpha \in 2^{<\omega}$, and

$S_i^e$: For all $\sigma$ with $c_i \leq \phi_e(\sigma) < c_{i+1}$, $K(\sigma) \leq^+ K^A(\sigma) + c_i$ for all $A \in [T]$ for all $i, e \in \omega$ with $i \geq e$.

Note that we only have $S_i^e$ requirements for $i \geq e$. This prevents $\phi_e$ from injuring the tree below the guessing node for $e$.

We order the requirements $R_{\langle \rangle}, S_0^0, R_{\langle 0 \rangle}, R_{\langle 1 \rangle}, S_1^0, S_1^1, R_{\langle 00 \rangle}, R_{\langle 01 \rangle}, R_{\langle 10 \rangle}, R_{\langle 11 \rangle}, S_2^0, S_2^1, S_2^2, \ldots$.

The various $S_i^e$ requirements will be concerned with different $\sigma$ throughout the construction as the approximations to the $\phi_{e,s}$ settle. We will say $S_i^e$ is $(e,s)$-responsible for $\sigma$ if at stage $s$ we have $c_i \leq \phi_{e,s}(\sigma) < c_{i+1}$ and $\sigma$ is one of the length-lexicographically first $s$ elements of $2^{<\omega}$. Note that for each $e$ for a given string $\sigma$ and stage $s$ at most one $S_i^e$ is $(e,s)$-responsible for $\sigma$. Analogously, we let $B(e,i,s) = \{\sigma : S_i^e$ is $(e,s)$-responsible for $\sigma\}$. Note that for finite-to-one approximations $\phi_e$ this set is always finite for any $i$ and $s$, but in general it need not be. As in the last proof, we will want to keep track of our guessing
levels and to this end we will use a collection of markers $n_{\alpha,s}$. Each $n_{\alpha,s}$ will mark the level where the paths that follow $\alpha$ through the first $|\alpha|$-many guessing nodes of $T_s$ branch for the $|\alpha|+2$nd time, and this will be the next guessing level. As before, to kill a node is to make a commitment to never add nodes above it into $T$. Nodes in $T_s$ that have not been killed are living.

We now give the strategies for satisfying each of our requirements.

An $R_\alpha$ requirement requires attention at a stage $s$ if the guessing level $n_{\alpha,s}$ is not defined. The strategy for meeting $R_\alpha$ is

1. Let $\alpha^- \prec \alpha$ be such that $|\alpha^-| = |\alpha|-1$ (i.e., the immediate predecessor of $\alpha$).
2. Let $n$ be some number larger than any seen before in the construction.
3. For every living leaf node, $\eta$, of $T_s$ that follows $\alpha$ through the first $|\alpha|$-many guessing nodes (these $\eta$ should have $|\eta| = n_{\alpha^-,s} + 2$ in the construction), add the path $\eta \cap \beta \cap j \cap k$ to $T_s$ to get $T_{s+1}$, where $|\eta| + |\beta| = n$, $\beta(i) = 0$ for all $i$ where it is defined, for every combination of $j$ and $k$ from $0, 1$.
4. Let $n_{\alpha,s+1} = n$. This is now a guessing level.

An $S_e$ requirement requires attention at a stage $s$ if there is a $\sigma$ that it is $(e,s)$-responsible for (so $c_i \leq \phi_{e,s}(\sigma) < c_{i+1}$ and $\sigma$ is one of the length-lexicographically first $s$-many strings) and there is a living partial path $\gamma$ in $T_s$ that, if it goes through at least $e$-many guessing nodes then it takes the ‘1’ branch after the $e$th one, and we have $K_\gamma^s(\sigma) + \phi_{e,s}(\sigma)$ is less than the shortest description of $\sigma$ in $L_{e,s}$. This means that the shorter description of $\sigma$ is on a path that either has not reached a guessing node for $\phi_{e,s}$ or is guessing that it is a finite-to-one approximation, so we will need to act. The strategy for meeting this requirement is

1. Find the length-lexicographically least $\sigma$ and for this $\sigma$ the length-lexicographically least $\gamma$ that are causing $S_e^i$ to require attention. By the choice of these as length-lexicographically least, we must have that the use of the computation $U_3^\gamma(\tau) \downarrow = \sigma$ that is causing $S_e^i$ to act is $|\gamma|$.
2. For this $\gamma$, let $\alpha$ be maximal such that $\gamma$ follows $\alpha$ through the guessing nodes of $T_s$. In other words, $\alpha$ is the collection of guesses that are being made on the path $\gamma$.
3. If $|\alpha| < e$, then $\gamma$ has not guessed about the behavior of $\phi_{e,s}$, but we know $i \geq e$, so we can afford to pay for a description of $\sigma$ on this part of the tree anyway. Put a request $(\sigma, K_\gamma^s(\sigma) + c_i)$ into $L_{e,s}$ to get $L_{e,s+1}$.
4. Otherwise, $|\alpha| \geq e$, and since $S_e^i$ requires attention, we must have $\alpha(e) = 1$. We now consider whether $|\alpha| \leq i + 1$, in order to check whether $\gamma$ has too many guessing nodes and will cause us to injure the tree. If $|\alpha| \leq i + 1$, then we have not yet branched for the $2(i+1)$th time (where the $(i+1)$th guessing node would be) so we can pay. Put a request $(\sigma, K_\gamma^s(\sigma) + c_i)$ into $L_{e,s}$ to get $L_{e,s+1}$. 
5. Otherwise, $|\alpha| > i + 1$, so $\gamma$ is longer than the $(i+1)$st guessing level. Since $\phi_{e,s}(\sigma) < c_{i+1}$, this is too high, so
   
   a) Injure $R_{\alpha|i+1}$ and run the Injury Subroutine for it.

   b) Let $T_{s+1} = T_s$, $L_{e,s+1} = L_{e,s}$.

   The Injury Subroutine for an $R_\alpha$ strategy at stage $s$ is

1. Find the living node $\eta$ at height $n_{\alpha,s}$ and the string $\rho$ such that $\eta \Dash\rho$ is a living leaf node of $T_s$, $\eta$ follows $\alpha$ through the guessing nodes of $T_s$, and $\sum_{\tau:U_{\eta,s}^\rho(\tau)\downarrow,U_{\eta,s}^\rho(\tau)\uparrow} 2^{-|\tau|}$ is maximal. If there is more than one pair $(\eta, \rho)$, choose the leftmost.

2. For every node $\zeta$ at height $n_{\alpha,s}$ that follows $\alpha$ through the first $|\alpha|$-many guessing nodes, keep $\zeta \Dash \rho$ alive. Kill every other extension of $\zeta$. Set all $R_\beta$ for $\beta \geq \alpha$ to requiring attention (i.e. set $n_{\beta,s+1}$ to be undefined).

The skeleton of the Construction is

**Stage 0:** Set $T_0 = \emptyset$, $L_{e,0} = \emptyset$ and $B(e,i,0) = \emptyset$ for every $e$ and $i$, $n_{\alpha,0}$ undefined for all $\alpha$.

**Stage** $s + 1$:

1. Compute $\phi_{e,s+1}(\sigma)$ and $K_{s+1}^\alpha(\sigma)$ for all living branches $\alpha$ in $T_s$, the first $s + 1$-many $\sigma$’s, and $e \leq s + 1$. Adjust $B(e,i,s+1)$ as necessary.

2. In order of priority, run the strategy for each of the first $s + 1$-many requirements that require attention, including executing the Injury Subroutine as necessary.

Now let $T = \bigcup_s T_s$, $L_e = \bigcup_s L_{e,s}$, $n_\alpha = \lim_s n_{\alpha,s}$. This completes the construction. The verification follows.

As in the proofs of Theorems 2.2.2 and 2.4.10, we will not be able to ensure that all requirements are satisfied, but only those that are correct about their guesses. We would like to show that the subtree of $T$ generated by all the correct guesses about the $\phi_{e,s}$ is perfect and that every path through it is in $\mathcal{LK}(\Delta_0^2)$. First, we need to show that the relevant requirements are satisfied. We call $\alpha$ correct if for every $e \leq |\alpha|$, $e \in \alpha$ if and only if $\phi_{e,s}$ is a total finite-to-one approximation.

**Lemma 3.3.2.** For all correct $\alpha$, $R_\alpha$ is injured only finitely often.

**Proof.** To derive a contradiction, let us assume there is some correct $\alpha$ such that $R_\alpha$ is injured infinitely often. Assuming this, we can take $\alpha$ to be minimal in length among such strings. Now, there are finitely many $S_i^e$ that come before $R_\alpha$ in the ordering of requirements, and these are the only requirements that can injure $R_\alpha$. The only $S_i^e$ requirements that can injure
CHAPTER 3. COMPRESSIVE POWER

53

$R_{\alpha}$ are those for which $e \in \alpha$, and since $\alpha$ is correct, these must correspond to total finite-to-one approximations. Each such $S^e_i$, then, only ever has $(e, s)$-responsibility for finitely many $\sigma$, so there must be at least one $\sigma$ that is the cause of infinitely many injuries to $R_{\alpha}$.

We now repeat the argument of Lemma 3.1.3. Let us assume we are at a stage $s$ such that $\phi_{e,s}(\sigma)$ and $K_s(\sigma)$ have settled and such that no $R_{\beta}$ for $\beta < \alpha$ will ever be injured again. Since $\sigma$ causes infinitely many more injuries to $R_{\alpha}$, it must be the case that $S^e_{|\alpha|-1}$ has $(e, t)$-responsibility for $\sigma$ for all $t \geq s$. Let $i = |\alpha| - 1$.

Now, each run of the Injury Subroutine for $R_{\alpha}$ at some stage $t$ will keep at least the most massive branch that follows $\alpha$ alive and kill the living branches that follow $\alpha$ that do not look identical to this most massive branch above $n_{\alpha,t}$. There are always $2^{\omega|\alpha|}$ many living nodes at height $n_{\alpha,t}$ that follow $\alpha$ through the guessing nodes, and a run of the Injury Subroutine for $R_{\alpha}$ fixes some amount of mass as converging along on of these paths. Since no earlier $R_{\beta}$ will ever be injured again, these branches are extended at each injury to get the nodes at height $n_{\alpha,t+1}$, so earlier nodes are initial segments of later ones. Since $K(\sigma)$ has converged, each injury must be caused by our finding a description of $\sigma$ of length less than $K(\sigma)$, so at least $2^{-K(\sigma)}$ much mass must converge on one of the paths that follow $\alpha$ through the guessing nodes, above $n_{\alpha,t}$. This mass is never lost due to these paths changing below the use of the computation, and there are only finitely many possible oracles that we are building, so the measure of the domain of $U$ relative to one of these oracles is infinite. This is a contradiction.

Thus, $R_{\alpha}$ can only be injured finitely often.

Lemma 3.3.3. For all correct $\alpha$ and all $e$ such that $\phi_{e,s}$ is a total finite-to-one approximation, the requirements $R_{\alpha}$ and $S^e_i$ are all eventually satisfied.

Proof. Note that each of the first $s$ requirements that require attention at a stage $s$ are allowed to act at that stage. This prevents requirements that will require attention infinitely often from blocking the action of better-behaved requirements. Requirements that act infinitely often will not interfere with the part of the tree we are interested in (the correctly guessing subtree), so we will be able to ignore their actions for the most part.

By the preceding lemma, each $R_{\alpha}$ for correct $\alpha$ is only injured finitely often. Eventually, it will never be injured again, at which point it can act to place $n_{\alpha}$ and the branching nodes above it.

For the $S^e_i$ requirements, since $\phi_{e,s}$ is a finite-to-one approximation, $S^e_i$ will only ever be $(e, s)$-responsible for finitely many $\sigma$. When one of these $\sigma$ causes $S^e_i$ to require attention, either a request for a shorter description of $\sigma$ is put into $L_{e,s}$, or there is an injury to some $R_{\alpha}$. Any such injury is to an $R_{\alpha}$ with $|\alpha| = i + 1$. There are then $2^{2i+1}$-many paths that $S^e_i$ can add mass to by causing an injury at a stage $s$ ($2^i$ many strings $\alpha$ of length $i + 1$ that have $\alpha(e) = 1$ and $2^{i+1}$-many coding levels between the guessing levels for these $n_{\alpha,s}$).

Let us assume we are at a stage $s$ where $K_s(\sigma) = K(\sigma)$, so when $S^e_i$ requires attention it will be because a description of $\sigma$ of length less than $K(\sigma)$ has converged on some path
through $T_s$ that it was monitoring. Each injury caused by $\sigma$ then adds at least $2^{-K(\sigma)}$ much mass to at least one of the $2^{2i+1}$-many paths that $S^e_i$ can add mass to. Now, we are not assuming that there will be only finitely many injuries to earlier requirements, so we do not know that these paths will be end extensions of earlier ones, but an injury to an earlier requirement caused by some $S^d_j$ (so $j < i$) must then put at least $2^{-K(\sigma)}$ much mass on one of the $2^{2j+1}$ paths that it affects, which will be an initial segment of at least 4 of the paths that $S^e_i$ can add mass to. Since there are only finitely many requirements before $S^e_i$ in the ordering, the lowest of these can only carry out this mass stealing injury finitely often before its oracles can hold no more mass, then the one above that can do the same finitely often, and so on. Eventually no lower priority requirement will be able to cause injuries after $\sigma$ causes an injury for $S^e_i$, and then $S^e_i$ will be able to cause finitely many more such injuries before its oracles are also too massive.

When it can cause no more injuries with $\sigma$ and it requires attention due to $\sigma$, $S^e_i$'s action will be putting a request into $L_{e,s}$ for a description of $\sigma$ of the smallest length it has seen, and then it will never require attention due to $\sigma$ again. Since it is responsible for only finitely many $\sigma$, eventually it will be satisfied.

The proof of Lemma 3.3.3 shows that for any $S^e_i$, any $\sigma$ that it is ever $(e,s)$-responsible can only be the source of finitely many injuries from $S^e_i$. Thus, the only $S^e_i$ that cause infinitely many injuries are those that become $(e,s)$-responsible for infinitely many $\sigma$ over the course of the construction. This is exactly why the hypothesis that $\phi_{e,s}$ is a finite-to-one approximation is necessary.

Lemma 3.3.3 gives us that the $R_\alpha$ for $\alpha$ correct are all satisfied, and since each of these requirements places a coding level above the branching level $n_\alpha$, the subtree $T^*$ of $T$ that consists of all paths that follow correct $\alpha$ through the guessing nodes of $T$ will be a perfect tree. The set of infinite paths through this subtree is then the perfect set we are looking for. We know that every $S^e_i$ requirement for a $\phi_{e,s}$ that is a total finite-to-one approximation will eventually be satisfied, so if the $L_e$ constructed by these requirements have bounded mass, then we will have that all the paths in our set are also in $\mathcal{LK}(\phi_e)$ for each such $e$, so they are in $\mathcal{LK}(\Delta^0_2)$.

We now come to the part of the proof of Theorem 3.3.1 where we need to find a bound for the mass that we put into the $L_e$ so that we can apply the Kraft-Chaitin Theorem. For the purposes of bounding the mass of the $L_e$, it will be easier to set aside this correct subtree and instead prove lemmas analogous to those used in the proof of Theorem 3.1.2.

Since we now have an $L_e$ for each $e$, it will make the proof simpler if we examine the subtrees where the $S^e_i$ requirements are active separately for each $e$. We define this subtree by letting

$$T_{e,s} = \{ \sigma \in T_s : \text{if } \alpha \text{ is maximal such that } \sigma \text{ follows } \alpha \text{ through the guessing nodes of } T_s, \text{ then either } |\alpha| < e \text{ or } \alpha(e) = 1 \},$$
and then letting \( T_e = \bigcup_s T_{e,s} \).

If an \( S^e_\alpha \) requirement requires attention at a stage \( s \), it can only be because of some description of some \( \sigma \) converging on a living path through \( T_{e,s} \). Thus, all nodes in \( T \) that cause mass to go into \( L_e \) are actually nodes in \( T_e \), so we can bound the mass of \( L_e \) by considering only these nodes. As before, we will split this tree into its living subtree and its dead branches.

\[
T_e' = \{ \sigma \in T_e : \sigma \text{ is never killed by a run of the Injury Subroutine} \}.
\]

\[
T_e'' = \{ \sigma \in T_e : \sigma \text{ is killed by a run of the Injury Subroutine} \}.
\]

We make the same simplifying assumptions on the construction, that at any stage \( s \) if we have a string \( \tau \) and a living node \( \gamma \) in \( T_s \) such that \( \cup^s \gamma (\tau) \downarrow \) but \( \cup^{s-1} \gamma (\tau) \uparrow \) then \( \tau \) is a shorter description of some \( \sigma \) we were monitoring and so the relevant \( S^e_\alpha \)'s will have to either pay for the description \((\sigma, K_s(\sigma) + c_i)\) into \( L_e \) or will cause an injury to some \( R_{\alpha} \).

We recall that \( \gamma \) is exact for \( \tau \) if \( \cup^\gamma (\tau) \downarrow \) and if the use of this computation is \( v^\gamma (\tau) = |\gamma| \).

We define an analogous \( \Delta'_e \) and \( \Delta''_e \) to use in finding a bound for \( L_e \).

\[
\Delta'_e = \sum_{(\gamma, \tau, i) : \gamma \in T'_e \text{ is exact for } \tau, \gamma \text{ is alive at a stage } s \text{ when } \cup^s \gamma (\tau) \downarrow, c_i \leq \phi_{e,s}(\cup^s \gamma (\tau)) < c_i+1} 2^{-|\tau| - c_i}
\]

\[
\Delta''_e = \sum_{(\gamma, \tau, i) : \gamma \in T''_e \text{ is exact for } \tau, \gamma \text{ is alive at a stage } s \text{ when } \cup^s \gamma (\tau) \downarrow, c_i \leq \phi_{e,s}(\cup^s \gamma (\tau)) < c_i+1} 2^{-|\tau| - c_i}
\]

We note that \( \Delta'_e + \Delta''_e \geq \sum_{(\sigma, n) \in L_e} 2^{-n} \), which is the mass of \( L_e \). We now find bounds for these sums.

**Lemma 3.3.4.** For any \( e \), \( \Delta'_e \leq 2 \).

**Proof.** We proceed similarly to the proof of Lemma 3.1.5. First we fix an \( e \). We define the strings \( \gamma^e(\alpha, \beta) \) to be the nodes in \( T'_e \) such that \( \gamma^e(\alpha, \beta) \) follows \( \alpha \) through the guessing nodes and \( \beta \) through the coding nodes of \( T'_e \), and \( |\gamma^e(\alpha, \beta)| \) is maximal such that \( \gamma^e(\alpha, \beta) \) does not go through any more branching nodes. Since \( e \) is fixed, we will suppress the superscript \( e \).

Note that \( \gamma(\alpha, \beta) \) only makes sense for \( |\alpha| - 1 \leq |\beta| \leq |\alpha| \), since every guessing node is followed immediately by the next coding node. We will have, then, that if \( |\beta| = |\alpha| \) then \( |\gamma(\alpha, \beta)| = n_{\alpha} \) and if \( |\beta| = |\alpha| - 1 \) then \( |\gamma(\alpha, \beta)| = n_{\alpha} - + 1 \). Recall that we use \( \sigma^- \) to denote the immediate predecessor of \( \sigma \). We will extend this notation slightly and use \( (\alpha, \beta)^- \) to denote \( (\alpha, \beta^-) \) if \( |\beta| = |\alpha| \) and \( (\alpha^-, \beta) \) if \( |\beta| = |\alpha| - 1 \).

We have additional conditions on \( \gamma(\alpha, \beta) \) that follow from our working in \( T'_e \). First, it must be the case that if \( |\alpha| > e \) then \( \alpha(e) = 1 \), or following \( \alpha \) would take us out of \( T'_e \).
Second, $\phi_{e,s}$ might not be a finite-to-one approximation, so it may be the case that $S^e_i$ causes infinitely many injuries to some $R_\alpha$ (for $R_\alpha$ to be injured like this, we must have $\alpha(e) = 1$). In this case $n_{\alpha,s}$ will not converge. This will be fine for our purposes, since in this case the infinite paths through $T^\prime_e$ that follow $\alpha$ through the guessing nodes are isolated. We abuse notation significantly, and let $\gamma(\alpha, \beta)$ for these $\alpha$ be the infinite isolated path through $T^\prime_e$ that follows $\alpha$ through the guessing nodes and $\beta$ through the coding nodes.

With this caveat, we can define $Q(\alpha, \beta) = \{ \tau : U^{\gamma(\alpha,\beta)}(\tau) \downarrow \text{ and } U^{\gamma(\alpha,\beta)}(\tau) \uparrow \text{ for any } \hat{\alpha} \preceq \alpha, \hat{\beta} \preceq \beta \}$. $Q(\alpha, \beta)$ is then the set of strings with $\alpha$ and $\beta$ minimal on which the universal machine converges using $\gamma(\alpha, \beta)$ as an oracle. Note that for any pair $(\tau, \gamma)$ such that $\gamma \in T^\prime_e$, $U^{\gamma}(\tau) \downarrow$, and $\gamma$ is exact for $\tau$, $\tau$ appears in exactly one $Q(\alpha, \beta)$ (in particular, for $\alpha, \beta$ maximal such that $\gamma \preceq \gamma(\alpha, \beta)$).

Now as above we define $m(\alpha, \beta) = \sum_{\tau \in Q(\alpha, \beta)} 2^{-|\tau|}$ to be the mass that converges along the path between $\gamma(\hat{\alpha}, \hat{\beta})$ and $\gamma(\alpha, \beta)$, where $(\hat{\alpha}, \hat{\beta}) = (\alpha, \beta)^{-}$. This is the mass the $S^e_i$’s will have to pay into $L_e$ for, although they will pay in at a rate of $2^{-c_i}$.

We are currently only dealing with $T^\prime_e$, the subtree that is living through the whole construction, so there will never be injuries to this tree. This means we can assume for a $\tau \in Q(\alpha, \beta)$ that $\phi_{e,s}(U^{\gamma(\alpha,\beta)}(\tau))$ is always at least $c_{|\alpha|-1}$ for the stage $s$ when we will pay into $L_{e,s}$ for $\tau$ (if it caused an injury, $n_\alpha$ would be greater than $|\gamma(\alpha, \beta)|$, which contradicts the definition).

This means we can bound $\Delta^e_n$ by the sum $\sum_{(\alpha, \beta)} m(\alpha, \beta) \cdot 2^{-c_{|\alpha|-1}+1}$.

We can rewrite this sum as $\sum_{(\alpha, \beta) \text{ with } |\alpha|=|\beta|} [m(\alpha, \beta) + \frac{m(\alpha, \beta^{-})}{2}] 2^{-c_{|\alpha|-1}+1}$, since each of $\alpha$, $\beta^{-0}$ and $\alpha$, $\beta^{-1}$ will contribute a $\frac{m(\alpha, \beta^{-})}{2}$, which will together give us $m(\alpha, \beta^{-})$.

Note that since we have $\gamma(\hat{\alpha}, \hat{\beta}) \preceq \gamma(\alpha, \beta)$ for $\hat{\alpha} \preceq \alpha$, $\hat{\beta} \preceq \beta$, it must be the case that for any $(\alpha, \beta)$, $\sum_{\hat{\alpha} \preceq \alpha, \hat{\beta} \preceq \beta, |\alpha|=|\beta|_{\hat{\alpha} \preceq \beta} [m(\hat{\alpha}, \hat{\beta}) + \frac{m(\alpha, \beta^{-})}{2}] \leq 1$, since this is mass that converges along a single oracle for the universal machine (we could even remove the 2 from the denominator of the fraction and this would be true).

Now, we are trying to bound $\sum_{(\alpha, \beta) \text{ with } |\alpha|=|\beta|} [m(\alpha, \beta) + \frac{m(\alpha, \beta^{-})}{2}] 2^{-c_{|\alpha|-1}+1}$. This is the limit as $n \to \infty$ of the sum $\sum_{(\alpha, \beta) : |\alpha|=|\beta| \leq n} [m(\alpha, \beta) + \frac{m(\alpha, \beta^{-})}{2}] 2^{-c_{|\alpha|-1}+1}$. We show that each of these finite sums is less than 2.

First, we note that for $\hat{\alpha}$ and $\hat{\beta}$ with $|\hat{\alpha}|=|\hat{\beta}|=k$, there are $2^{(n-k)}$ many pairs $(\alpha, \beta)$ with $|\alpha|=|\beta|=n$ such that $\hat{\alpha} \prec \alpha$ and $\hat{\beta} \prec \beta$, so we can rewrite this sum as

$$\sum_{(\alpha, \beta) : |\alpha|=|\beta|=n} \sum_{\hat{\alpha} \preceq \alpha, \hat{\beta} \preceq \beta, |\hat{\alpha}|=|\hat{\beta}|} [m(\hat{\alpha}, \hat{\beta}) + \frac{m(\alpha, \beta^{-})}{2}] 2^{-c_{|\alpha|-1}+1} 2^{-2(n-|\hat{\alpha}|)}.$$
CHAPTER 3. COMPRESSIVE POWER

Recall that \( c_i = 4^{i+1} \), so \( 2^{|\hat{\alpha}| - c_{|\hat{\alpha}| - 1} } \) is less than 1 for all \( \hat{\alpha} \). Hence, this sum is bounded by

\[
\sum_{(\alpha, \beta):|\alpha|=|\beta|=n} \sum_{\hat{\alpha} \leq \alpha, \hat{\beta} \leq \beta, |\hat{\alpha}|=|\hat{\beta}|} [m(\hat{\alpha}, \hat{\beta}) + \frac{m(\hat{\alpha}, \hat{\beta}^{-})}{2}] 2^{1-2n} =
\]

\[
2^{1-2n} \sum_{(\alpha, \beta):|\alpha|=|\beta|=n} \sum_{\hat{\alpha} \leq \alpha, \hat{\beta} \leq \beta, |\hat{\alpha}|=|\hat{\beta}|} [m(\hat{\alpha}, \hat{\beta}) + \frac{m(\hat{\alpha}, \hat{\beta}^{-})}{2}]
\]

This inner sum is bounded by 1, as discussed above, so the whole sum is bounded by \( 2^{1-2n} \sum_{(\alpha, \beta):|\alpha|=|\beta|=n} 1 \). There are \( 2^n \) many pairs \((\alpha, \beta)\) with \(|\alpha|=|\beta|=n\), so this sum is bounded by 2.

Since all the partial sums are bounded by 2, the limit \( \sum_{(\alpha, \beta):|\alpha|=|\beta|} [m(\alpha, \beta) + \frac{m(\alpha, \beta^{-})}{2}] 2^{-c_{|\alpha| - 1} + 1} \) must also be no more than 2, and this sum is larger than \( \Delta_e'' \).

\[ \Box \]

All that remains now is to find a bound for \( \Delta_e'' \). As in the simpler case, we first start with a lemma bounding the mass wasted by each injury.

**Lemma 3.3.5.** For any injury to requirement \( R_\alpha \) at a stage \( s \), the amount of mass that has gone into \( \Delta_e \) from branches that are affected by this injury (i.e., that follow \( \alpha \) through the first \(|\alpha|\)-many guessing nodes in \( T_s \)) is no more than \( \frac{1}{2^{|\alpha|-1}} \) times the mass, \( m \), that has converged on the branch that is chosen as most massive by the run of the Injury Subroutine.

**Proof.** This proof follows the same basic argument as that of Lemma 3.1.6. Fix a run of the Injury Subroutine for \( R_\alpha \). At the time of this injury, we have constructed some amount of the tree above \( T_{\alpha, s} = \{ \sigma \in T_s : \sigma \) follows \( \alpha \) through the first \(|\alpha|\)-many guessing nodes of \( T_s \} \).

The nodes of \( T_{\alpha, s} \) at height \( n_{\alpha, s} \) are the guessing nodes associated with \( R_\alpha \), and these are the nodes above which we look for maximal mass. There are \( 2^{|\alpha|} \) many of them, since there are \( 2^{|\alpha|} \) many branching nodes they go through, but they must follow \( \alpha \) at the guessing nodes.

Now, say we have built \( k \)-many more branching nodes above \( n_{\alpha, s} \) in \( T_{\alpha, s} \). This gives us \( 2^{|\alpha|+k} \) many segments between branching nodes above \( n_{\alpha, s} \). We call the rank of such a segment the number of guessing nodes below it, counting the first node of the segment if it is a guessing node. Since \( n_{\alpha, s} \) is a guessing level, the nodes at this height are guessing nodes, and there are \( 2^{|\alpha|} \) many of them. Above these there are \( 2^{|\alpha|+1} \)-many segments of rank \(|\alpha|+1 \) before the next coding level, then another \( 2^{|\alpha|+2} \)-many segments of rank \(|\alpha|+1 \) before the next guessing level. For rank \(|\alpha|+2 \), we have \( 2^{|\alpha|+3} + 2^{|\alpha|+4} \)-many. In general, then, we will have \( 3 \cdot 2^{|\alpha|+2i-1} \)-many segments of rank \(|\alpha|+i \) for \( 1 \leq i \leq (k - 2)/2 \).

Since there was not an earlier injury to any of these branching nodes (they are still alive, after all), we know that the \( \phi_{e,s} \) values are either less than \( e \) (so there is no requirement that is \((e, s)\)-responsible for the corresponding \( \sigma \)) or are high enough to not cause injury. What
this means is that for any mass that converges on the segment above \( i \) guessing nodes (so on a segment of rank \( i \)), if we pay into \( L_e \) for it, has \( \phi_{e,s} \) value at least \( c_i \). We may in fact pay in up to twice this, since we can have paid in for arbitrarily many larger values of \( \phi_{e,s} \) while it was settling down. We take this into account by multiplying by 2.

Now we can bound the mass paid into \( L_{e,s} \) by assuming in the worst case that \( m \) much mass converged on each segment above height \( n_{\alpha,s} \) and was paid into \( L_{e,s} \) at a rate of \( 2^{-c_{i-1}} \) where \( i \) is the rank of that segment. Summing over all ranks, from \(|\alpha| + 1\) to \(|\alpha| + (k - 2)/2\), we get a bound on this mass of

\[
\sum_{i=1}^{(k-2)/2} m \cdot 3 \cdot 2^{(|\alpha|+2i-1)} \cdot 2^{c_{|\alpha|+i-1}} \cdot 2,
\]

and we want to show this sum is less than \( m \cdot 2^{-|\alpha|-1} \). We can divide through by \( m \), and we are trying to prove the inequality

\[
3 \sum_{i=1}^{(k-2)/2} 2^{(|\alpha|+2i-c_{|\alpha|}+i-1)} \leq 2^{-c_{|\alpha|-1}}.
\]

We can collect all the powers of 2 into the sum to get

\[
3 \sum_{i=1}^{(k-2)/2} 2^{(|\alpha|+2i+c_{|\alpha|}-c_{|\alpha|}+i-1)} \leq 1.
\]

We will have this if we can show the sum is less than \( \frac{1}{4} \), so let us show that we can bound the \( i \)th term of the sum by \( 2^{-i-2} \), since \( \sum_{i=1}^{\infty} 2^{-i-2} = \frac{1}{4} \). That is, we wish to show

\[
2^{(|\alpha|+2i+c_{|\alpha|-1}-c_{|\alpha|}+i-1)} \leq 2^{-i-2}.
\]

It is enough to show the inequality in the exponent, so we want

\[
|\alpha|+2i + c_{|\alpha|-1} - c_{|\alpha|+i-1} \leq -i - 2.
\]

We recall that \( c_j = 4^{j+1} \), and rewrite this as

\[
|\alpha|+3i + 4^{|\alpha|} + 2 \leq 4^{|\alpha|+i}.
\]

Note that we need this to hold for all \( i > 0 \), \(|\alpha| > 0\). Dividing by \( 4^{|\alpha|} \) gives us

\[
\frac{|\alpha|}{4^{|\alpha|}} + \frac{3i}{4^{|\alpha|}} + 1 + \frac{2}{4^{|\alpha|}} \leq 4^i.
\]

The left hand side is maximized when \(|\alpha| = 1\), but for any \( i \) the right hand side still dominates it, so the inequality holds.

We have now proved that an injury to \( R_\alpha \) can waste no more than \( 2^{-c_{|\alpha|-1}} \) times as much mass as converges on the path chosen by the run of the Injury Subroutine.
At this point we are ready to find a bound for $\Delta''_e$.

**Lemma 3.3.6.** For every $e$, $\Delta''_e \leq 1$.

**Proof.** We follow the argument of Lemma 3.1.7. We can fix an $e$ at the start, since none of our arguments will depend on it. As in the proof of Lemma 3.1.7, we wish to think of the segments of our tree $T_s$ at stage $s$ as being reservoirs for mass. These are like the $Q(\alpha, \beta)$ we used in the proof of Lemma 3.3.4, but we consider them as depending on $T_s$, not the subtree $T_{e,s}$. Additionally, since we are always dealing with our tree at a finite stage, we will not have to worry about some of our segments being isolated infinite paths.

We will use the segments between guessing nodes as reservoirs, ignoring the fact that there are branchings at coding nodes except to double our number of reservoirs. This does mean that two reservoirs can share a prefix, but this can only help us. If mass ends up in the prefix shared by two reservoirs, then it counts toward filling both of them. That is, at a stage $s$, we consider the segments between $\gamma(\alpha^-, \beta^-)_s$ and $\gamma(\alpha, \beta)_s$ for any pair $(\alpha, \beta)$ with $|\alpha| = |\beta| \geq 1$ as a reservoir for mass. When there is an injury to some $R_\alpha$, the injury subroutine will choose the branch that follows $\alpha$ through the guessing nodes of $T_s$ with the most mass above $n_\alpha$, keep this and all paths identical to it above $n_\alpha$ alive, and kill all others that follow $\alpha$ through the guessing nodes. The mass of this most massive path then ends up in one of the reservoirs for an $\alpha'$ with $|\alpha'| < |\alpha|$, and mass from other reservoirs may either end up in some lower reservoir or be poured out. Because we are only interested in an upper bound, we can assume that only the mass from the most massive reservoir is poured into a lower reservoir by a run of the Injury Subroutine, and all other reservoirs at that level have their mass emptied out.

We are currently trying to bound the mass that is wasted over the course of the construction, and so we use Lemma 3.3.5 to tie this mass to the mass that is saved by the run of the Injury Subroutine that wastes it. As before, we just need to keep track of how much mass can ever move through the reservoirs for a given $R_\alpha$ injury. Even this is more detail than we need; an injury to $R_\alpha$ only affects the reservoirs corresponding to $R_\alpha$, but for the purposes of achieving a bound we can assume an injury to any $R_\alpha$ with $|\alpha| = n$ affects the reservoirs for all these $R_\alpha$. Now we just need to bound how much mass can go through the reservoirs at a certain height, knowing that an injury at one level saves the mass from the most full reservoir and empties the rest.

First, we count the reservoirs at each level. The marker $n_0$ never moves, since there is no $S^e_t$ that has lower priority than $R_0$, so we don’t need to worry about anything before that level. There is a guessing node at $n_0$, and a coding level above that before the next guessing level, so there are 4 reservoirs at the first level. These get mass added to them by injuries to $R_{(0)}$ and $R_{(1)}$. Above each of these reservoirs there is a guessing node and a coding node, so there are 16 reservoirs at the next level, and these get mass from injuries to $R_\alpha$ for $|\alpha| = 2$. Following this pattern, there are $4^i$ many reservoirs that gain mass from injuries to $R_\alpha$ for $|\alpha| = i$.

We now consider how much mass may have to pass through the reservoirs at height $i$. In the worst case, all the mass that ends up in any lower reservoir may go through one of the
reservoirs at level \( i \), and in the impossibly worse case, each of these lower reservoirs could end up with total mass equal to 1. So, the \( 4^i \) many reservoirs could all fill up, then be injured to fill one of the \( 4^{i-1} \)-many reservoirs one level lower, have this process repeat to fill all these lower reservoirs, and then have an injury move that mass to a reservoir two levels lower. They can have to filter enough mass down to fill the 4 lowest reservoirs, then to fill the next 16, and so on until all lower reservoirs are full and they can fill themselves without fear of losing their mass.

Then the total mass that could ever have to move through the reservoirs at level \( i \) is

\[
\sum_{k=1}^{i} \prod_{j=k}^{i} 4^j
\]

Now, it is injuries to \( R_\alpha \) for \(|\alpha| = i\) that move mass into the reservoirs at level \( i \). We know from Lemma 3.3.5 that for any such injury the amount of mass paid into \( L_{e,s} \) that is wasted by the run of the Injury Subroutine is no more than \( 2^{-c |\alpha|-1} \) times the mass that converges on the path that is chosen as most massive, which is exactly the mass that we know goes into a lower reservoir. Thus, in the worst case all the mass that goes through reservoirs at level \( i \) enters these reservoirs due to an injury to some requirement, and so the most that could be paid into \( L_{e,s} \) due to injuries to \( R_\alpha \) with \(|\alpha| = i\) is

\[
\sum_{k=1}^{i} \prod_{j=k}^{i} 4^j 2^{-c_{i-1}}.
\]

We can always take the largest product, and bound this by

\[
\sum_{k=1}^{i} \prod_{j=1}^{i} 4^j 2^{-c_{i-1}}.
\]

Now \( \prod_{j=1}^{i} 4^j = 4^{\sum_{j=1}^{i} j} = 4^{\frac{i^2+i}{2}} = 2^{i^2+i} \). Thus, our sum reduces to

\[
\sum_{k=1}^{i} 2^{i^2+i} 2^{-c_{i-1}} = i \cdot 2^{i^2+i-c_{i-1}}.
\]

\( i \) is always less than \( 2^i \), so we know this term is less than \( 2^{i^2+2i-c_{i-1}} \). This is a bound on the mass wasted by injuries to \( R_\alpha \) with \(|\alpha| = i\). Since we want the total mass wasted by injuries to all \( \alpha \) to be finite, we show that this term is bounded by \( 2^{-i} \). That is, we need

\[
2^{i^2+2i-c_{i-1}} \leq 2^{-i},
\]
or

\[
i^2 + 2i - c_{i-1} \leq -i,
\]
which is

\[ i^2 + 3i \leq c_{i-1}. \]

Recalling that \( c_n = 4^{n+1} \) for all \( n \), we are now trying to prove that \( i^2 + 3i \leq 4^i \) for all \( i > 0 \), but this follows easily from elementary computations.

Thus, \( \Delta'' \), the total mass in \( L_e \) wasted by any injury, is no more than \( \sum_i 2^{-i} = 1 \).

We note that the proof of the preceding lemma did not use \( e \) anywhere. The same bound works for all \( e \), so we actually have that every \( L_e \) is a Kraft-Chaitin set. For \( e \) that are not finite-to-one and are causing infinitely many injuries to part of \( T \), this set just witnesses that the degenerate paths they create through \( T \) are low for \( K \).

We consider now the subtree \( T^* \) of \( T \) that consists of all nodes that guess correctly at every guessing node, that is, \( \sigma \in T^* \) if and only if for the maximal \( \alpha \) such that \( \sigma \) follows \( \alpha \) through the guessing nodes of \( T \), for every \( e \leq |\alpha| \), \( e \in \alpha \leftrightarrow \phi_{e,s} \) is a total finite-to-one approximation. Since every guess of every path through \( T^* \) is correct, in particular each path guesses that \( \phi_{e,s} \) is a total finite-to-one approximation when it is, so \( S^e_i \) will ensure that \( L_e \) witnesses that that path is in \( \mathcal{L}K(\phi_e) \). Since all the \( \alpha \) that give guesses through the guessing nodes of \( T^* \) are correct, the \( R_{\alpha} \) requirements will only be injured finitely often, so the coding nodes they put into \( T^* \) will make \( T^* \) a perfect tree.

As in the proof of Theorem 3.1.2, any two paths through \( T^* \) differ only at the coding levels, by the work of the Injury Subroutine. Thus, given any real \( A \) we can construct paths through \( T^* \) such that one follows \( A \) at the coding nodes and the other follows \( \bar{A} \). Then the join of these two paths can compute \( A \) by finding the places where the two paths differ and reading off \( A \)'s bits from the first path.

This completes the proof of Theorem 3.3.1.

By the discussion at the beginning of this chapter, any real in \( \mathcal{L}K(\Delta^0_2) \) is also in \( \mathcal{KT}(\Delta^0_2) \), so this theorem automatically gives us the same statement for the reals that are \( K \)-trivial up to every \( \Delta^0_2 \) order. This gives us another proof of Corollary 2.4.14.

We already have something stronger than that for any real there are two elements of \( Q \) that join above it. By Theorem 2.4.10 for any real \( B \) there is an element of \( \mathcal{KT}(\Delta^0_2) \) that computes \( B \).

We can also now extend our corollary to Theorem 3.1.2 regarding reals that are within \( \epsilon \) of being low for \( K \). Recall that \( A \) being within \( \epsilon \) of low for \( K \) means \( \lfloor (1-\epsilon)K(\sigma) \rfloor \leq K^A(\sigma) \) for all \( \sigma \).

**Corollary 3.3.7.** There is a perfect set of reals that for every \( \epsilon \) are within \( \epsilon \) of being low for \( K \), and this set of reals is not closed under effective join.


Chapter 4

Mutual Information

4.1 Information Content

We turn now to a discussion of mutual information of reals, applying the results from the preceding chapters. Historically, many of the results from these chapters were first discovered as they applied to the specific case of mutual information, and so this presentation is in some sense counterchronological.

To start, we will need to define what exactly it means for reals to share information, and to do this we first need to define what we mean by information. There are many possible notions we could be trying to capture. For instance, by information we might mean information about other reals. We could say that a real ‘knows about’ another real just in case the first computes the second. We could use any one of a number of notions of computation, but in any case the mutual information of a pair of reals would just be the (Turing-, tt-, etc.) ideal generated by the pair. For various notions of computation these ideals have been studied either more or less, and they are often interesting subjects of study.

However, we will be interested in studying not a real’s information about other reals, but rather its information about finite objects, in particular finite binary strings. For finite strings we already have a notion of the information content: prefix-free Kolmogorov complexity. When we relativize the universal machine to a real oracle $A$, we get a notion of the information content of strings relative to $A$, and then we can take this difference between the standard and the relative complexity of a string $\sigma$ to be the information $A$ has about $\sigma$. Intuitively, if there are $K(\sigma)$-many bits of information in $\sigma$, but $A$ only needs $K^A(\sigma)$-many bits to describe $\sigma$, the remaining bits of information in $\sigma$, $K(\sigma) - K^A(\sigma)$ must be the information $A$ already has about $\sigma$.

This notion of information about strings, together with the framework we have set up earlier, gives us a nice way to talk about information content of reals (in this sense) and to compare reals based on this quantity. By definition, a real $A$ is low for $K$ up to a function $f$ if for every $\sigma \in 2^{<\omega}$, $K(\sigma) \leq^+ K^A(\sigma) + f(\sigma)$. Rewriting this equation, we get that $A \in \mathcal{LK}(f)$ if and only if for every $\sigma \in 2^{<\omega}$, $K(\sigma) - K^A(\sigma) \leq^+ f(\sigma)$, or in other words, if and only if
f(σ) is a bound on the information A has about σ for every σ, up to an additive constant. We can consider the information A has about σ as a function of σ and formalize this with a definition, and then make a few observations.

**Definition 4.1.1.** The *information content* of σ in A is $\text{IC}^A(\sigma) = K(\sigma) - K^A(\sigma)$.

**Observation 4.1.2.** $\text{IC}^A$ is $\Delta_2^0$. Thus, if $A$ is $\Delta_0^0$, then so is $\text{IC}^A$.

**Proof.** $K(\sigma)$ and $K^A(\sigma)$ both have recursive approximations relative to $A$ that are monotone nonincreasing.

**Observation 4.1.3.** For any $A$, $A \in \mathcal{L}K(\text{IC}^A)$

**Proof.** For any $\sigma$, $K(\sigma) \leq [K(\sigma) - K^A(\sigma)]$.

**Observation 4.1.4.** $A$ is low for $K$ if and only if $\sup \text{IC}^A < \infty$.

**Proof.** $\sup \text{IC}^A \leq c < \infty$ if and only if for all $\sigma$, $K(\sigma) - K^A(\sigma) \leq c$, that is $K(\sigma) \leq K^A(\sigma)$.

For any non-low-for-$K$ real $A$ we have $\sup \text{IC}^A = \infty$, so sup does not give us much in terms of separating reals by their information content. However, there is a way to use IC to compare the information content of reals. For two reals $A$ and $B$, $A$ has no more information than $B$ does about finite strings if we have $A \in \mathcal{L}K(\text{IC}^B)$ (that is, $K(\sigma) \leq K^A(\sigma) + K(\sigma) - K^B(\sigma)$ for all $\sigma$). In other words, $\mathcal{L}K(\text{IC}^B)$ is the collection of reals with no more information about finite strings than $B$. Note that this says nothing about these reals having the same information, all we know is that the quantities of information are the same. For example, if $A$ and $B$ are mutually Martin-Löf random (that is, $K^A(B \upharpoonright n) \geq K^A(n)$ and $K^B(A \upharpoonright n) \geq K^B(n)$, for all $n$), we may have $K^A((A \oplus B) \upharpoonright 2n) = K^B((A \oplus B) \upharpoonright 2n)$, but in some sense $A$ has information about the even bits of this string while $B$ has information about the odd bits. Thus, to capture the information about finite strings that is common to both reals we need to do something further.

There have been many suggestions for what the definition is that actually captures this concept of mutual information about finite objects, and unfortunately so far none has emerged as clearly superior. We will work with some of the suggested definitions (both suggested by Levin in [9]). First,

**Definition 4.1.5.** The *mutual information* of reals $A$ and $B$ is

$$I(A : B) = \log \sum_{(\sigma, \tau)} 2^{K(\sigma) - K^A(\sigma) + K(\tau) - K^B(\tau) - K(\sigma, \tau)}.$$  

Here $K(\sigma, \tau)$ is the prefix-free Kolmogorov complexity of the pair $(\sigma, \tau)$. One way to interpret this definition is that $K(\sigma) - K^A(\sigma)$ is the information content of $\sigma$ in $A$, $K(\tau) - K^B(\tau)$ is the information content of $\tau$ in $B$, and $-K(\sigma, \tau)$ is a measure of how much $\sigma$
and $\tau$ have in common ($K(\sigma, \tau)$ is larger for less related strings). We are then summing for all pairs $(\sigma, \tau)$ a measure of how much $A$ knows about $\sigma$ times a measure of how much $B$ knows about $\tau$ times a measure of how related $\sigma$ and $\tau$ are. The sum is inside a log so that for finite $A$ and $B$ the definition reduces to the mutual information of finite strings $I(\sigma : \tau) = K(\sigma) + K(\tau) - K(\sigma, \tau)$. The sum here has infinitely many terms, so the main question for a pair $(A, B)$ is whether $I(A : B)$ is infinite or finite. If finite, the particular value will depend on the precise coding of the universal machine.

An alternative definition suggested by Levin is

**Definition 4.1.6.** The simplified mutual information of reals $A$ and $B$ is

$$I^s(A : B) = \log \sum_{\sigma} 2^{K(\sigma) - K^A(\sigma) - K^B(\sigma)}.$$ 

Note that the simplified mutual information is just the mutual information restricted to those pairs of strings with $\sigma = \tau$. Thus, it is always the case that $I^s(A : B) \leq^+ I(A : B)$. However, it is unknown whether the reverse inequality holds. That is, it is unknown whether the terms for related but not identical $\sigma$ and $\tau$ can contribute enough to the sum to make it infinite while the sum over the identical terms is finite. The following observations follow easily from the definitions.

**Observation 4.1.7.** For any $A$, $B$, $I(A : B) \leq^+ I(B : A)$ and $I^s(A : B) \leq^+ I^s(B : A)$.

**Observation 4.1.8.** If $A \geq_T B$, then for every $C$, $I(A : C) \geq^+ I(B : C)$ and $I^s(A : C) \geq^+ I^s(B : C)$.

**Observation 4.1.9.** If $A$ is low for $K$, then for every real $B$, $I(A : B) < \infty$.

**Proof.** By definition, if $A$ is low for $K$ then there is some constant $c$ such that for every $\sigma$, $K(\sigma) - K^A(\sigma) \leq c$. This gives us that $I(A : B) = \log \sum_{(\sigma, \tau)} 2^{K(\sigma) - K^A(\sigma) + K(\tau) - K^B(\tau) - K(\sigma, \tau)} \leq C + \log \sum_{(\sigma, \tau)} 2^{K(\tau) - K^B(\tau) - K(\sigma, \tau)}$. It is a result of Levin in Gács [6] that for any $\sigma$ and $\tau$, $K(\sigma, \tau) =^+ K(\tau) + K(\sigma|\tau, K(\tau))$, where $K(\sigma|\tau, K(\tau))$ is the conditional complexity of $\sigma$ given $\tau$ and $K(\tau)$, that is, the length of the shortest description of $\sigma$ by a machine that has access to $\tau$ and $K(\tau)$. This gives us that $K(\tau) - K(\sigma, \tau) =^+ -K(\sigma|\tau, K(\tau))$ so we can bound $I(A : B)$ up to a constant by $\log \sum_{(\sigma, \tau)} 2^{-K^B(\tau) - K(\sigma|\tau, K(\tau))}$. We can factor out the term that depends only on $\tau$ and rewrite our sum as $\log \sum_\tau 2^{-K^B(\tau)} \sum_\sigma 2^{-K(\sigma|\tau, K(\tau))}$. The inner sum here is bounded by the measure of the domain of the universal machine relative to an oracle that contains the information $\tau$ and $K(\tau)$, so it must be less than 1 for any $\tau$, and the sum $\sum_\tau 2^{-K^B(\tau)}$ is bounded by the measure of the domain of the universal machine relative to $B$, so it is also less than 1. Thus, $I(A : B) < \infty$. \qed
We actually have something stronger than Observation 4.1.9 in that having finite mutual information with every other real actually characterizes being low for $K$. The harder direction is due to Hirschfeldt, Reimann, and Weber in [7].

**Theorem 4.1.10.** A real $A$ is low for $K$ if and only if for any real $B$, $I(A : B) < \infty$.

**Proof.** We show that if $A$ is not low for $K$, then any $B \geq_T A \oplus 0'$ will have infinite mutual information with $A$. Let $B \geq_T A \oplus 0'$.

Now, since $A$ is not low for $K$, for each $c$ there is a string $\sigma_c$ such that $K(\sigma_c) - K^A(\sigma_c) > c$. Since $B$ is above $0'$ it can compute $K(\sigma)$ for all $\sigma$, and since it is above $A$, it can approximate $K^A(\sigma)$ from above. Thus, $B$ can recursively enumerably find such a $\sigma_c$ for each $c$. Relative to $B$, then, we can construct a Kraft-Chaitin set $M$ by enumerating $(\sigma_c, c)$ into $M$ where $\sigma_c$ is the string for which we find $K(\sigma) - K^A(\sigma) > c$ at the earliest stage. Now via $M$ we have that for these $\sigma_c$, $K^B(\sigma_c) \leq c$. Thus $K(\sigma_c) - K^B(\sigma_c) \geq K(\sigma_c) - c$. The exponents in our sum for the terms given by $(\sigma_c, c)$ are then $K(\sigma) - K^A(\sigma) + K(\sigma) - K^B(\sigma) - K(\sigma, \sigma) \geq c + K(\sigma) - c - K(\sigma, \sigma) = K(\sigma, \sigma) - K(\sigma, \sigma)$. Now, for any $\tau$, it is the case that $K(\tau) = K(\tau, \tau)$ (witnessed by the machine that copies the action of the universal machine, but doubles every output, and the machine that copies the action of the universal machine and if the output is a pair, outputs the first element). Thus, we have for these $\sigma_c$ the exponent of the term $(\sigma_c, \sigma_c)$ in our sum is up to a constant, 0. Since there are infinitely many of these strings (one for each $c$), the sum will be infinite.

Note that in the proof of Theorem 4.1.10 we only used terms in the sum that were pairs of identical strings, so in fact being low for $K$ is also equivalent to having finite simplified mutual information with every other real. The equivalence between lowness for $K$ and having finite mutual information with every real gives some support to this being a proper definition of mutual information. The low for $K$ reals are supposed to be those with the least possible information, and we have an explicit way in which that property is demonstrated using mutual information.

To further explore the relationship between lowness for $K$ (and $K$-triviality) and this definition of mutual information, we examine a real’s mutual information with itself. The self-information of a real $A$ is the quantity $I(A : A)$, and the simplified self-information is $I^s(A : A)$. This approach then gives us a notion of what it means for a real to have little information content in our sense of information: its self-information is as low as possible.

**Definition 4.1.11.**

A has finite self-information if $I(A : A) < \infty$.

A has finite simplified self-information if $I^s(A : A) < \infty$.

The first result in comparing this notion of ‘least possible information’ and the traditional notion of lowness for $K$ is the trivial observation that follows from the equivalence above.

**Observation 4.1.12.** If $A$ is low for $K$, then $A$ has finite self-information.
The proof of Theorem 4.1.10 shows that any non-low-for-$K$ real $A$ has infinite mutual information with any real that computes $A \oplus \emptyset'$, so we have the following corollary.

**Corollary 4.1.13.** Any $A \geq_T \emptyset'$ does not have finite self-information.

We also note that since the infinite sum in the proof uses only pairs $(\sigma_c, \sigma_c)$, this holds for simplified self-information as well.

We say that a real $A$ is Martin-L"of random if for all $n$, $K(A \upharpoonright n) \geq n + c$, that is, if describing the initial segments of $A$ is as complicated as just giving the bits one by one. Equivalently, $A$ is Martin-L"of random if it does not lie in any effectively presented null set (a null set given by the intersection of a collection $(U_n)$ of uniformly $\Sigma^0_1$ sets of strings such that $\mu(U_n) \leq 2^{-n}$), so the set of Martin-L"of random reals has measure 1. The following observation, then, shows that having finite self-information is a rare property.

**Observation 4.1.14.** If $A$ is Martin-L"of random then $I_s(A : A) = \infty$, so $A$ does not have finite self-information.

**Proof.** Let $A$ be Martin-L"of random. Then there is some $c$ such that for all $n$, $K(A \upharpoonright n) \geq n - c$. Now, any real can compress its own initial segments as easily as it can describe their lengths, so $K^A(A \upharpoonright n) \leq K(\bar{n})$. Considering the $(A \upharpoonright n, A \upharpoonright n)$-terms of the sum that gives simplified self-information, we see the exponents of the summands for these terms is $K(A \upharpoonright n) - 2K^A(A \upharpoonright n)$. Using the above inequalities, this is up to a constant at least $n - 2K(\bar{n})$. If we consider only those $n$ that have $n = 2^m$ for some integer $m$, we have that $K(\bar{n}) \leq K(\bar{m}) \leq m$. Thus, the simplified self-information of $A$ is at least (up to an additive constant) $\log \sum_m 2^{2^m - 2m}$, and this sum diverges.

The proof of Theorem 4.1.10 suggests a way in which having finite self-information could differ from being low for $K$: finding the strings that a real $A$ has some amount of information about requires the computational power of the halting problem. Reals with sufficiently little computing power may have information about some strings (i.e., might not be low for $K$), but might not know which strings these are.

The first result that there actually are reals for which this is the case was an explicit construction by Hirschfeldt and Weber in [7] of a recursively enumerable real that had finite self-information but was not low for $K$. Although they did not use the terminology, the structure of their proof was essentially to find a function $f : 2^{<\omega} \to \omega$ such that being low for $K$ up to $f$ implies having finite self-information and then to construct a real that was not low for $K$ but was low for $K$ up to $f$.

We include a version of their proof here.

**Theorem 4.1.15 (Hirschfeldt, Weber).** There is a finite-to-one approximable function $f_{HW}$ such that $\sum \sigma, \tau 2^{f_{HW}(\sigma) + f_{HW}(\tau) - K(\sigma, \tau)} < \infty$.

**Proof.** The function $f_{HW}$ built using an auxiliary function, which we describe below.
Lemma 4.1.16. There is a finite-to-one approximable function $h$ such that $\sum_{\sigma} 2^{2h(\sigma)-K(\sigma)} < \infty$.

Proof. First, we define an approximation to a sequence of numbers $(s_t)$ by $s_{0,t} = 0$ for all $t$ and $s_{i,t}$ is the first stage $s$ such that $\mu(\text{dom}(U_s)) - \mu(\text{dom}(U_{s-1})) \leq 2^{-i}$. Now, for any $\sigma$ at any stage $t$, if $K_t(\sigma) < \infty$ then there is a least $r_{\sigma,t} \leq t$ such that $K_{r_{\sigma,t}}(\sigma) = K_t(\sigma)$ (the first stage that $\sigma$ received its current shortest description). We define

$$h_t(\sigma) = \begin{cases} |\sigma| & \text{if } K_t(\sigma) = \infty \\ i & \text{such that } s_{i,t} < r_{\sigma,t} \leq s_{i+1,t} \text{ otherwise,} \end{cases}$$

and we let $h(\sigma) = \lim h_t(\sigma)$. It is clear that eventually the $s_i$ and $r_\sigma$ all converge, so $h(\sigma)$ does as well. To show that our approximation to $h$ is finite-to-one, we note that for a fixed $i$, we eventually reach a stage $t$ such that $s_{i+1,t}$ has converged to its final value. At this stage, only finitely many $\sigma$ have ever received values less than or equal to $i$ (only finitely many have received descriptions from $U$, and of the others only finitely many have $|\sigma| < i$). Moreover, any $\sigma$ that receives a description from $U$ after stage $s_{i+1}$ cannot take an $h$-value less than $i + 1$.

It remains to show that $h$ has the desired property. First, if $h(\sigma) = i$, then $r_\sigma > s_i$, so there is a $\tau$ that is used as the shortest description of $\sigma$ such that $U(\tau) = \sigma$ but $U_{s_i}(\tau) \uparrow$. Since $\tau$ is a shortest description of $\sigma$, we have that $|\tau| = K(\sigma)$. Since $U$ only converges on $\tau$ after $s_i$, we must have that the sum $\sum_{\sigma: h(\sigma) = i} 2^{-K(\sigma)}$ is bounded by $\mu(\text{dom}(U)) - \mu(\text{dom}(U_{s_i}))$ (we add $2^{-K(\sigma)}$ much mass to the domain of $U$ when $U(\tau) \downarrow$, and this happens after stage $s_i$). By definition we know that this quantity is bounded by $8^{-i}$. Now, summing over all $i$, we get that

$$\sum_{\sigma} 2^{2h(\sigma)-K(\sigma)} \leq \sum_i \sum_{h(\sigma) = i} 2^{2i-K(\sigma)} \leq \sum_i 2^{2i} \sum_{h(\sigma) = i} 2^{-K(\sigma)} \leq \sum_i 4^i8^{-i} \leq \sum_i 2^{-i} \leq 2.$$

Now we define the Hirschfeldt-Weber function, $f_{HW}$. We let $\langle \rangle$ be the length-lexicographic ordering on $2^{<\omega}$, and define

$$f_{HW}(\tau) = \min_{\sigma \leq \tau} h(\sigma).$$

First we note that since $h$ is finite-to-one approximable then so is $f_{HW}$. Now we need to show that $\sum_{\sigma,\tau} 2^{f_{HW}(\sigma)+f_{HW}(\tau)-K(\sigma,\tau)} < \infty$. First, we note that by symmetry,

$$\sum_{\sigma,\tau} 2^{f_{HW}(\sigma)+f_{HW}(\tau)-K(\sigma,\tau)} \leq 2 \sum_{\sigma} \sum_{\tau \geq \sigma} 2^{f_{HW}(\sigma)+f_{HW}(\tau)-K(\sigma,\tau)}.$$
CHAPTER 4. MUTUAL INFORMATION

Now, for $\tau \geq \sigma$, $f_{HW}(\tau) \leq h(\sigma)$, so we can bound this sum by

$$2\sum_{\sigma} 2^{2h(\sigma)} \sum_{\tau \geq \sigma} 2^{-K(\sigma, \tau)}.$$

Using the same result of Levin in Gács [6] we can replace $K(\sigma, \tau)$ by $K(\sigma) + K(\tau|\sigma, K(\sigma))$ up to an additive constant. Our inner sum then becomes

$$\sum_{\tau \geq \sigma} 2^{-K(\sigma) - K(\tau|\sigma, K(\sigma)) + c}.$$

Factoring out the purely $\sigma$ terms, we rewrite our bounding sum as

$$2C \sum_{\sigma} 2^{2h(\sigma) - K(\sigma)} \sum_{\tau \geq \sigma} 2^{-K(\tau|\sigma, K(\sigma))}.$$

Now, the inner sum here is finite since it is bounded by the measure of the domain of the universal machine relative to the string $\sigma \triangle K(\sigma)$, and the outer sum is finite by the relevant property of $h$. Hence, $\sum_{\sigma, \tau} 2^{f_{HW}(\sigma) + f_{HW}(\tau) - K(\sigma, \tau)} < \infty$.

The Hirschfeldt-Weber function $f_{HW}$ is a sufficient bound on the information content function of a real to ensure that that real has finite self-information. That is, if for a real $A$, for all $\sigma$, $IC^A(\sigma) \leq f_{HW}(\sigma)$, then $A$ has finite self-information. This, and the proof that $LK(0) \neq LK(f_{HW})$ give us a separation between notions of information. Intuitively, being low for $K$ is having actually no information (or only constantly much), while a real can have 'some amount' of information and as long as this is no more than $f_{HW}$-much that real will not be able to tell that it has any information, so its self-information will be finite. If it is not actually low for $K$, then a sufficiently powerful oracle will be able to extract its information, by finding the relevant strings. $f_{HW}$ is a positive amount of information that is 'not enough' to be distinguishable from no information without additional computational power. The study of reals with finite self-information is then the study of reals that think they have no information, and this includes both those that have no information and those that have information but do not know it.

After showing that finite self-information did not coincide with lowness for $K$, Hirschfeldt and Weber leave some open questions about how the set of reals with finite self-information behaves, including whether all such reals had to be $\Delta^0_2$. We can show that this is not the case, using the results from Chapter 3. All that is necessary is to note that the Hirschfeldt-Weber $f_{HW}$ is finite-to-one approximable, and then we can apply Theorem 3.1.2. A perfect set is uncountable, and there are only countably many $\Delta^0_2$ reals.

**Theorem 4.1.17.** There is a perfect set of reals that have finite self-information, and for any real $A$, there are two reals $B_0$ and $B_1$ with finite self-information such that $B_0 \oplus B_1 \geq_T A$. 
Theorem 3.1.2 applied to $f_{HW}$ gives us more than just that there are non-$\Delta^0_2$ reals with finite self-information. We also know that the set of these reals does not form a Turing ideal, because it is not closed under effective join. This is to be expected, though, since there is no reason that the information that is coded into a real and inaccessible to it should be inaccessible to another oracle. We have a certain other sense of closure in this set, in that all the reals with finite self-information have pairwise finite mutual information.

**Theorem 4.1.18.** If $I(A : A) < \infty$ and $I(B : B) < \infty$, then $I(A : B) < \infty$.

**Proof.** We show the convergence of $I(A : B)$ by bounding it by the sum of convergent sums. First, recall that $I(A : B) = \log \sum_{(\sigma, \tau)} 2^{K(\sigma) - K^A(\sigma) + K(\tau) - K^B(\tau) - K(\sigma, \tau)}$. Note also that finiteness of self-information implies finiteness of simplified self-information, so by hypotheses on $A$ and $B$ the sums $\sum_\sigma 2^{K(\sigma) - K^A(\sigma)}$ and $\sum_\sigma 2^{K(\sigma) - K^B(\sigma)}$ converge.

We want to partition $2^{<\omega} \times 2^{<\omega}$ into 2 sets, such that the sum of the terms in $I(A : B)$ over each set is bounded by one of the sums we have that converge. We partition based on the inequality $K(\sigma) - K^A(\sigma) \leq K(\tau) - K^B(\tau)$, that is, on whether $B$ knows more about $\tau$ than $A$ knows about $\sigma$.

If $K(\sigma) - K^A(\sigma) \leq K(\tau) - K^B(\tau)$, then the $(\sigma, \tau)$ term in the definition of $I(A : B)$ is less than or equal to $2^{K(\tau) - K^B(\tau) + K(\tau) - K^B(\tau) - K(\sigma, \tau)}$. By the symmetry of information result of Levin in Gács [6] that $K(\sigma, \tau) = K(\tau) + K(\sigma | \tau, K(\tau))$ for all $\sigma, \tau$, our term is up to a constant $2^{K(\tau) - 2K^B(\tau)}$. If we sum over all pairs $(\sigma, \tau)$ with $K(\sigma) - K^A(\sigma) \leq K(\tau) - K^B(\tau)$, we get

$$\sum_{(\sigma, \tau): K(\sigma) - K^A(\sigma) \leq K(\tau) - K^B(\tau)} 2^{K(\tau) - 2K^B(\tau) - K(\sigma | \tau, K(\tau))}.$$ 

We can factor out the purely $\tau$ terms to get

$$\sum_{\tau} 2^{K(\tau) - 2K^B(\tau)} \sum_{\sigma: K(\sigma) - K^A(\sigma) \leq K(\tau) - K^B(\tau)} 2^{-K(\sigma | \tau, K(\tau))}.$$ 

The inner sum here is bounded by $\sum_\sigma 2^{-K(\sigma | \tau, K(\tau))}$, which is bounded by the measure of the universal machine with oracle $\tau^* K(\tau)$, so it must be less than 1. Thus, our sum is bounded by the sum $\sum_\tau 2^{K(\tau) - 2K^B(\tau)}$, which the the simplified self-information of $B$, so we know it is finite.

If we have $K(\sigma) - K^A(\sigma) > K(\tau) - K^B(\tau)$, then the $(\sigma, \tau)$ term in the definition of $I(A : B)$ is bounded by $2^{K(\sigma) - K^A(\sigma) + K(\sigma) - K^A(\sigma) - K(\sigma, \tau)}$, and through the same process we can get that the sum over all terms with $K(\sigma) - K^A(\sigma) > K(\tau) - K^B(\tau)$ is bounded by the simplified self-information of $A$.

Each pair $(\sigma, \tau)$ has either $K(\sigma) - K^A(\sigma) \leq K(\tau) - K^B(\tau)$ or $K(\sigma) - K^A(\sigma) > K(\tau) - K^B(\tau)$, so a term bounding it appears in one of the 2 convergent sums. Also, each term in each convergent sum is used as a bound for the term in $I(A : B)$ for exactly one pair $(\sigma, \tau)$.
Thus, we have in general that the inequality $I(A : B) \leq I^*(A : A) + I^*(B : B)$ holds, and in particular when both terms on the right hand side are finite, the left hand side must be as well.

The proof of Theorem 4.1.18 shows that $I(A : B) \leq I^*(A : A) + I^*(B : B)$ for any reals $A$ and $B$. In particular, for $A = B$ we get that $I(A : A) \leq 2I^*(A : A)$, so if $A$ has finite simplified self-information then it also has finite self-information. This answers a very particular case of the question as to whether mutual information and simplified mutual information coincide.

### 4.2 Further Directions

Currently there are very few other results concerning this notion of mutual information. The only characterization we have of having finite self-information is the definition we started with. We know that being in $\mathcal{LK}(f_{HW})$ implies that a real has finite self-information, and Hirschfeldt and Weber have shown that reals that have finite self-information are jump-traceable (that is, there is a bound on the number of guesses required to pick out the value of $\phi^A(e)$) and therefore generalized low$_1$, however we do not have a necessary condition in terms of IC for a real to have finite self-information, nor a sufficient condition in terms of jump-traceability. Finding either would hopefully make these reals much easier to work with.

**Question 4.2.1.** Is it possible to characterize the reals with finite self-information in terms of IC? Is it possible to characterize them in terms of jump-traceability?

Another open question about finite self-information is how this notions interacts with effective dimension. So far almost nothing is known in this direction. Having finite self-information and being low for effective dimension both follow from having a particular bound on IC$^A$ (either $f_{HW}$ or $\log|\sigma|$), so there may be some way to relate the two notions, but neither of these conditions is necessary and the two functions that are involved are quite different.

**Question 4.2.2.** Does every real with finite self-information have effective dimension 0?

**Question 4.2.3.** What is the relation between having finite self-information and being low for effective dimension?

The longest standing open question in this field is whether mutual information and simplified mutual information coincide. So far all the results we have hold for both notions, and inequality in one direction is trivial. However, the other direction has so far proved intractable. We do have the partial result that self-information and simplified self-information do coincide, up to finiteness.

**Question 4.2.4.** Is $I(A : B) \leq^+ I^*(A : B)$ for all reals $A, B$?
There are also much open space for exploration in the more general field of these weak lowness notions. As above, for $\Delta^0_2$ orders, $\mathcal{L}K(f)$ and $\mathcal{K}T(g)$ are uncountable, but for certain choices of $f$ and $g$ these set contain only reals with some ‘lowness’ property, while others are sufficiently fast-growing that these sets are all of $2^\omega$. Currently there is not a characterization of the functions $f$ for which not all reals are low for $K$ (or $K$-trivial) up to $f$. The best possible result would be a sharp bound in growth-rate that separates the functions that give all of $2^\omega$ from those that give strict subsets. There are similar questions in terms of measure and category.

**Question 4.2.5.** Can we characterize the functions $f, g$ for which $\mathcal{L}K(f)$ and $\mathcal{K}T(g)$ are not $2^\omega$; have measure less than 1; or are meager?

Further, there are questions related to the interactions of these weak lowness notions in the Turing degrees. For instance, we know there are degrees (any degree above $0'$, for instance) such that no real in that degree can be low for $K$ up to $f_{HW}$ or any slower-growing function, so we have a lower bound on IC for reals in these degrees. However, there are also reals that are low for $K$ up to all $\Delta^0_2$ orders $f$, so there are degrees without such a lower bound. The differences between these types of degrees (where they can occur in the Turing degrees, what the lower bounds can be, how they interact) are all interesting questions.

There are other reducibility notions that are more directly related to initial segment complexity and compressive power, so there may be interesting results about these weak notions under these reducibilities. We say that $A$ is $K$-reducible to $B$ ($A \leq_K B$) if for all $n$, $K(A \upharpoonright n) \leq^+ K(B \upharpoonright n)$ and $A$ is $LK$-reducible to $B$ ($A \leq_{LK} B$) if for all $\sigma$, $K^B(\sigma) \leq^+ K^A(\sigma)$. $K$-reducibility orders reals based on their initial segment complexity and $LK$-reducibility orders them by compressive power. By definition, we know that $\mathcal{K}T(g)$ is closed downwards under $\leq_K$ and $\mathcal{L}K(f)$ is closed downwards under $\leq_{LK}$, but not much else in this field has been explored.

**Question 4.2.6.** What are the relations between almost $K$-triviality and almost lowness for $K$ and the various reducibility notions?
Appendix A: A finite-to-one $\Delta^0_2$ function with no finite-to-one approximation

Here we give a construction of a $\Delta^0_2$ function that is finite-to-one but not finite-to-one approximable. For the sake of clarity we will construct it as a function $\omega \to \omega$, but the same proof will apply to functions on $2^{<\omega}$.

**Lemma 4.2.7.** There is a $\Delta^0_2$ function $f : \omega \to \omega$ that is finite-to-one but not finite-to-one approximable.

**Proof.** We recall Lemma 1.6.3, which states that any finite-to-one approximable function majorizes (that is, is pointwise above) a $\Delta^0_2$ order. It then suffices to build a $\Delta^0_2$ $f$ that it is finite-to-one but fails to majorize any $\Delta^0_2$ order (in the standard sense of order on $\omega$). We define $f$ by recursive approximation, diagonalizing against majorizing those recursive approximations that give orders on $\omega$. We do not know which approximations these will be, so we develop a uniform strategy that will try to defeat any $\phi_{e,s}$. The construction of $f$ is as follows.

First, we devote to each $e$ an infinite disjoint set of natural numbers that will be used to ensure $f$ does not majorize $\phi_e$, if it is an order. For concreteness, let $P(e) = \{q^{e+1} : q \text{ is prime}\}$. We let $f_s(n) = n$ for all $s$ for any $n$ which is not a nonzero power of a prime. The approximation $f_s$ will converge so that at most one element of $P(e)$ is not mapped to itself, and that element will be sent to $e$. We now define $f$’s action on $P(e)$. We define a function $n(e,s) =$ the largest $n \leq s$ such that $\phi_{e,s}$ is nondecreasing up to $n$.

**Stage 0:** $f_0(n) = n$ for all $n \in P(e)$.

**Stage** $s + 1$:

1. If there is an $n$ that is greater than the largest $p^e$ such that for some stage $t < s + 1$, $f_t(p^e) = e$ (or 0, if there is no such $p^e$) and $\phi_{e,s+1} \upharpoonright n$ is nondecreasing and takes a value larger than $e$ and $f_s \upharpoonright n$ majorizes $\phi_{e,s+1} \upharpoonright n$, then
a) For the smallest element $q^e$ of $P(e)$ that is greater than $n$ for the largest such $n$, set $f_{s+1}(q^e) = e$.

b) For any other element $p^e$ of $P(e)$ set $f_{s+1}(p^e) = p^e$.

2. Otherwise, let $f_{s+1}(p^e) = f_s(p^e)$ for all $p^e \in P(e)$.

We now need to verify that the $f$ we construct is total, finite-to-one, and fails to majorize any $\Delta^0_2$ order. The first two properties are clear from the construction. For any $n$ that is not in any $P(e)$, $f_s(n) = n$ at all stages of the construction, so this value converges. For a $p^e \in P(e)$, $f_s$ initially maps $p^e$ to $p^e$, it may at some stage start mapping it instead to $e$ and at some still later stage start mapping it to $p^e$ again, after which its value never changes. Thus, the value converges. For finite-to-oneness, for each value $e$ at most 2 elements are sent to $e$ by $f$ ($e$ and possibly some $p^e$ if it is used to prevent majorizing $\phi^e$ in the limit).

To show that $f$ fails to majorize any $\Delta^0_2$ order, let $\phi^e$ converge to a $\Delta^0_2$ order. Then at some stage $s$, $\phi^e_{s,s}$ will have converged to its final values on an initial segment of $\omega$ of some length $n$ and will take a value greater than $e$. Either this initial segment includes the largest $p^e$ such that $f_t(p^e) = e$ for some $t < s$, or it does not. If not, there is an even later stage $s'$ and an even longer initial segment such that $f_{s'}$ is nondecreasing that does include this $p^e$. This initial segment must still have $f_{s'}$ take a value greater than $e$ on it. The only way for the largest number $p^e$ to change in the interim is for an initial segment of at least as long as $p^e$ to be found with the relevant qualities and for the construction to act, so we can assume we are at the first such stage where this is true for the current $p^e$. If at this stage $f_{s'}$ majorizes $\phi^e_{s,s'}$, we will find the least element $q^e$ of $P(e)$ that is greater than the length of the longest nondecreasing initial segment of $\phi^e_{s,s'}$ and set $f_{s'}(q^e) = e$. Since $q^e$ is greater than an initial segment of $\omega$ on which $\phi^e_{s,s}$ has already converged and has taken value greater than $e$, $\phi^e_{s,s}$ can never again be nondecreasing on an initial segment that includes $q^e$ and also be majorized by $f_s$ since $\phi^e_{s,s}(q^e)$ must be bigger than $e$ for $\phi^e_{s,s}$ to appear nondecreasing. Thus, we will not have to act again and $f$ will not majorize $\phi^e$.

The function $f$ constructed this way is no more than two-to-one, yet it has the property that any way to try to approximate it recursively must have a value which is infinitely often the image of new elements. What makes its behavior particularly perverse is that the values that have two elements in their preimage actually do satisfy the property that over the course of the approximation their preimage is finite, at least for the approximation given in the proof. The outputs that are problematic are ones that, in the end, are only mapped to by themselves.
Appendix B: The index set of finite-to-one approximations is $\Pi^0_3$-complete

For the sake of completeness, we include a proof of the arithmetical complexity of the index set of total, finite-to-one approximations. The reader who is unfamiliar with arithmetical complexity and completeness may refer to [13] for a review of these notions. First, let us define $\text{FOA} = \{ e : \phi_{e,s} \text{ is a total finite-to-one approximation} \}$.

**Lemma 4.2.8.** $\text{FOA}$ is $\Pi^0_3$-complete.

**Proof.** To show that $\text{FOA}$ is $\Pi^0_3$, we observe that $e \in \text{FOA}$ if and only if $\forall x \exists y \forall z \forall s[\text{if } z > y \text{ then } \phi_{e,s}(z) > x]$ (finite-to-oneness) and $\forall x \exists y \exists t \forall s[\text{if } s > t \text{ then } \phi_{e,s}(x) = y]$ (totality). Each of these statements is $\Pi^0_3$, so the conjunction is as well.

To show completeness, let $P$ be an arbitrary $\Pi^0_3$ predicate. We wish to show that there is a recursive function $f$ such that $a \in P \iff f(a) \in \text{FOA}$. By definition of $\Pi^0_3$, $P$ is given by $a \in P \iff \forall x \exists y \forall z R(a, x, y, z)$ for a recursive predicate $R$. Now, the index set $\text{FIN} = \{ e : W_e \text{ is finite} \}$ is $\Sigma^0_2$-complete, where $W_e$ is the $e$th recursively enumerable set. Thus, there is a recursive function $g$ such that we can replace the $\Sigma^0_2$ part of this definition with "$W_g(a, x)$ is finite". We will define $f$ by recursion, using this function $g$. We use $p_i$ to denote the $i$th prime number. $f(a)$ gives the index of the recursive approximation given by

$$\phi_{f(a),s}(n) = \begin{cases} x & \text{if } n = p_i^j \text{ and } i \in W_{g(a, x)} \text{ at stage } s, \\ n & \text{otherwise.} \end{cases}$$

Clearly $\phi_{f(a),s}$ is total. We need to check that $a \in P \iff f(a) \in \text{FOA}$. First, if $a \in P$, then for every $x$, $W_{g(a, x)}$ is finite. That is, for each $x$, there are only finitely many elements that are ever in $W_{g(a, x)}$, and so only finitely many that ever get sent to $x$ by $\phi_{f(a),s}$, so $f(a) \in \text{FOA}$. If $a \notin P$, then for some $x$, $W_{g(a, x)}$ is infinite, and so $\phi_{f(a)}$ sends infinitely many elements to $x$ in the limit. Then clearly $f(a) \notin \text{FOA}$. $\square$
Bibliography


