Computably Extendible Order Types

James Robert Kishore Gay

Submitted in accordance with the requirements for the degree of Doctor of Philosophy

The University of Leeds
Department of Pure Mathematics

May 2016
The candidate confirms that the work submitted is his own and that appropriate credit has been given where reference has been made to the work of others.

This copy has been supplied on the understanding that it is copyright material and that no quotation from the thesis may be published without proper acknowledgement.

©2016 The University of Leeds and James Robert Kishore Gay

The right of James Robert Kishore Gay to be identified as Author of this work has been asserted by him in accordance with the Copyright, Designs and Patents Act 1988.
Abstract

In this thesis we consider, from a computability perspective, the question of what order-theoretic properties of a partial order can be preserved under linear extension. It is well-known that such properties as well-foundedness or scatteredness can be preserved, that is, given any well-founded partial order you can find a well-founded linear extension and mutatis mutandis for scattered partial orders.

An order type $\sigma$ is extendible if a partial order that does not embed $\sigma$ can always be extended to a linear order that does not extend $\sigma$. So for example “given any well-founded partial order, you can find a well-founded linear extension” is equivalent to saying that $\omega^*$ is extendible. The extendible order types were classified by Bonnet [3] in 1969.

We define notions of computable extendibility and then apply them to investigate the computable extendibility of three commonly used order types, $\omega^*$, $\omega^* + \omega$ and $\eta$.

In Chapter 2 we prove that given a computably well-founded computable partial order, you can find a computably well-founded $\omega$-c.e. linear extension, and further that this result doesn’t hold for $n$-c.e. for any finite $n$. In Chapter 3 we show how to extend these results for linearisations of computable partial orders which do not embed $\zeta = \omega^* + \omega$. In Chapter 4 we prove the analogous results for scattered partial orders.
In memory of Barry Cooper
Acknowledgements

I must first thank my parents for all of their support during this process. I would not have got this far without the love and encouragement of my friends and family, so I thank them for that.

I would also like to thank my supervisor Barry Cooper for all of his help and advice, my collaborator Charles Harris for our many useful conversations over the last couple of years, and his valuable comments on this thesis and Immi Halupczok and Dugald Macpherson for their kind assistance in the later stages of my PhD.

I would like to thank the Leverhulme Trade Charities Trust who partially funded my research.

The research in this thesis has been carried out in collaboration with S. Barry Cooper and Charles Harris. The proofs of Theorems 4.14 and 3.5 were my individual work, the proofs of the other main results, Theorems 2.14, 3.9, 3.11, 4.18 and Corollaries 2.19 and 4.22 were joint work.
Contents

Abstract ................................................................. iii
Dedication ............................................................... v
Acknowledgements .................................................... vii
Contents ................................................................. ix
List of figures ........................................................... xiii

1 Introduction ......................................................... 1
  1.1 Notations and Conventions .................................... 1
  1.2 Computability Theory ........................................... 2
  1.3 The Ershov Hierarchy ............................................. 6
  1.4 Priority Arguments .............................................. 9
  1.5 Order Theory .................................................... 12
  1.6 Extensions ....................................................... 14
  1.7 Reverse Mathematics ........................................... 19
  1.8 Thesis Outline .................................................. 22
## 2 Well-founded orderings

2.1 Background .................................................. 25

2.2 The upper bound ............................................. 31
   2.2.1 Requirements ......................................... 32
   2.2.2 Strategy ................................................. 33
   2.2.3 Construction ............................................ 34
   2.2.4 Verification ............................................. 35

2.3 Computable extendibility .................................. 37

2.4 The lower bound ............................................. 38

## 3 The order type $\zeta$

3.1 Background .................................................. 53

3.2 The lower bound ............................................. 55

## 4 Scattered orderings

4.1 Background .................................................. 65

4.2 The upper bound ............................................. 74
   4.2.1 Requirements ......................................... 75
   4.2.2 Strategy ................................................. 75
   4.2.3 Construction ............................................ 77
   4.2.4 Verification ............................................. 78

4.3 The lower bound ............................................. 80
List of figures

2.1 Construction of the linear extension ........................................ 27
2.2 The first four elements of $P_e$, added at stage $e + 1$ ..................... 39
2.3 After the first good stage for $e$ ............................................. 41
2.4 After the second good stage for $e$ ............................................ 42
2.5 After a general good stage for $e$ ............................................. 43
2.6 After the first stage at which a pair is 2-computed .......................... 45
2.7 After the second stage at which a pair is 2-computed ....................... 47
2.8 After the first pair is 3-computed ............................................ 50
3.1 The first eight elements of $P_e$ added at stage $e + 1$ ..................... 57
4.1 Coding of finite binary strings on a binary tree ............................. 71
4.2 Coding of indices on a binary tree ........................................... 72
4.3 The first four elements of $P_e$, added at stage $e + 1$ ..................... 82
4.4 After the first good stage for $e$ ............................................. 83
4.5 Coding of finite binary strings on a binary tree ............................. 84
4.6 Coding of indices on a binary tree . . . . . . . . . . . . . . . . . . . . 84
4.7 After the first few stages . . . . . . . . . . . . . . . . . . . . . . . . . 85
4.8 After the first pair is 3-computed . . . . . . . . . . . . . . . . . . . . 89
Chapter 1

Introduction

1.1 Notations and Conventions

We begin by setting out the notation and conventions that we will be working in, in order to avoid confusion with some of the historical conventions which may differ from the modern standards. We will mostly follow the conventions laid out in Cooper [10], Soare [49] and other standard texts.

We denote the set of natural numbers by $\mathbb{N} = \{0, 1, 2, \ldots\}$ and its order type under the usual $\leq$ ordering by $\omega$. We will use lower-case Roman characters to denote natural numbers, and upper-case Roman characters to denote subsets of $\mathbb{N}$. We will use lower-case Greek characters to denote order types, $\omega$ for the order type of the natural numbers, $\eta$ for the order type of the rational numbers, and a star $^*$ to denote a reverse type, so for example $\omega^*$ denotes the reverse of the order type of the natural numbers or the order type of the negative integers. We will use $\equiv$ to denote isomorphism between order types.

Given a (possibly partial) function $f$, we use the notation $f(n) \downarrow$ if $f$ is defined at
We use the standard bijective pairing function $\langle \cdot, \cdot \rangle : \mathbb{N} \times \mathbb{N} \to \mathbb{N}$, $\langle x, y \rangle = \frac{x^2 + 2xy + y^2 + 3x + y}{2}$, and if necessary we can nest it to get a bijection from $\mathbb{N}^i \to \mathbb{N}$ for any $i$.

We use the standard listing $\{W_e\}_{e \in \mathbb{N}}$ of computably enumerable sets, where by the Normal Form Theorem, $W_e$ is the domain of $\varphi_e$, the $e$th partial computable function in a standard computable listing of partial computable functions $\{\varphi_e\}_{e \in \mathbb{N}}$. We have the standard approximations $\{W_e[s]\}_{e,s \in \mathbb{N}}$ associated to the $W_e$. $W_e[s]$ denotes the set of natural numbers enumerated into $W_e$ by the end of stage $s$ of a computation. Turing functionals will be denoted by upper case Greek letters.

### 1.2 Computability Theory

Computability Theory, also known as Recursion Theory, is based in the study of effectiveness. Intuitively we think of an algorithm as an effective process for calculating a function or deciding a question like whether a number is in a specific set. What it means for a process to be effectively calculable, and whether some algorithms are in some sense more effectively calculable than others, are questions we ask in Computability theory.

The notion of effectively calculable can be formalised in different ways, including “general recursive”\(^1\) (Gödel and Herbrand 1931 [17]), “$\lambda$-definable” (Church 1932 [6])

\(^1\)This notion was also just called “recursive”, particularly by Church and Kleene after 1934/5.
Chapter 1. Introduction

1936 [8] 1940 [9]², and “computable”³ (Turing 1936 [52]). These different models of computation were proved to be equivalent by Church and Turing [7] and the idea that they capture the intuitive notion of effectively calculable is known as the Church-Turing Thesis, stated for example in Turing’s PhD thesis [53].

Turing computability, using Turing’s “automatic machines”⁴ provides a useful way of talking about computable functions. We assume familiarity with the technical details of Turing machines and their use, further details can be found in the texts listed at the end of this section.

The functions we consider in computability theory are conventionally functions of natural numbers, with a domain of \( A \subseteq \mathbb{N} \) and a codomain of \( \mathbb{N} \). It is useful to distinguish the case when \( A = \mathbb{N} \) and functions which are defined on all natural numbers are called total functions, and partial functions otherwise. When we talk about the computability of a subset \( A \subseteq \mathbb{N} \), we identify the set with its characteristic function, the function \( f_A : \mathbb{N} \rightarrow \{0, 1\} \) such that

\[
f_A(n) = \begin{cases} 
1 & \text{if } n \in A \\
0 & \text{if } n \notin A.
\end{cases}
\]

And then the computability of \( A \) is defined as the computability of \( f_A \). Given the usual equivalence between subsets of \( \mathbb{N} \) and binary reals, this also allows us to talk about the computable complexity of individual real numbers, in terms of the oracle strength needed to compute them.

²The original system published in 1932 was shown to be inconsistent by Kleene and Rosser [26], so in 1936 Church published what is now known as the untyped lambda calculus, and in 1940 the simply typed lambda calculus.

³This notion is now known as Turing computable

⁴Turing [52] used the terminology “automatic machines” or “a-machines”. We now know these constructs as “Turing machines”.
The oracle Turing machine, introduced by Turing in 1939 [53] allows us to talk about functionals. Functionals are partial or total functions which range over functions of numbers and return numbers, in an effective sense. The functional $F(\alpha_0, \ldots, \alpha_n, \vec{x})$ is a \textit{partial computable functional} if it can be obtained from partial functions $\alpha_0, \ldots, \alpha_n$ and the initial functions by composition, primitive recursion and $\mu$-recursion, where the partial functions $\alpha_0, \ldots, \alpha_n$ correspond to oracles in the Turing machine implementation of the functional.

Each partial computable functional $F(\alpha_0, \ldots, \alpha_n, \vec{x})$, considered as a functional of a variable $\alpha_0$, is computable uniformly\(^5\) in $\alpha_1, \ldots, \alpha_n, \vec{x}$, and so we can construct an effective listing of partial computable functionals $\{F_e(\alpha_0)\}_{e \in \mathbb{N}}$. This allows us to list all possible Turing machines, since we can assign a Gödel number $n$ to a Turing machine $\Psi$, then list the machines $\Psi_1, \Psi_2, \ldots$, which compute $F_m$ for some $m$. For ease of use, we say that if $n$ is not the Gödel number of a Turing machine, then $\Psi_n$ is a Turing machine running the empty program.

Given such a list of Turing machines, Turing showed that we can find a partial computable functional $F(x, y) = \Psi_x(y)$, which we can use to simulate any of the Turing machines in the list. We call such a machine a \textit{Universal Turing Machine}.

We write $\Psi(B) = A$ to mean that the Turing machine $\Psi$ can compute the set $A$ when given the set $B$ as an oracle, and we use this notion to define the relation $A \leq_T B$ if there exists a Turing machine $\Psi$ such that $\Psi(B) = A$, and we call the relation $\leq_T$ “Turing reducible to”. The relation $\leq_T$ induces an equivalence relation $\equiv_T$ on sets of natural numbers; $A \equiv_T B$ if $A \leq_T B$ and $B \leq_T A$ (so there are Turing functionals $\Psi, \Psi'$ such that $\Psi(A) = B$ and $\Psi'(B) = A$). We call the equivalence classes of this relation \textit{Turing degrees} as defined by Post [39]. Formally, the Turing degree of a set $A$ is

\(^5\)By which we mean that the computation is the same, just with different inputs.
$\deg(A) = \{X \subseteq \mathbb{N} \mid X \equiv_T A\}$, often denoted by $a$.

Further, as shown by Post [39] there is an induced partial ordering $\leq$ on the Turing degrees obtained by setting $a \leq b$ if there is some $A \in a$ and $B \in b$ such that $A \leq_T B$.

We can use this partial ordering to define the Turing degree structure $D = (D, \leq)$, which is a major object of study in Computability Theory.

The list of Turing machines allows us to define the halting set $K = \{x \mid \Psi_x(x) \text{ is defined}\}$, which is not computable. Using the list of Turing machines, we can define a corresponding list of sets $W_e = \{n \mid \Psi_e(n) \text{ halts}\}$, which gives us the standard listing of what are called the computably enumerable sets. Clearly $K = W_e$ for some $e \in \mathbb{N}$.

Kleene and Post [25] relativised the halting set, $K^A = \{x \mid \Psi_x(A; x) \downarrow\}$. This notion defines the jump operator, $K^A$ is $A'$, the jump of $A$, and the $n + 1$th jump of $A$ is $A^{n+1} = K^{A^n} = (A^n)'$. It naturally follows that the jump of a degree is $a' = \deg(A')$ and the jump function $f : D \rightarrow D$, $f : a \mapsto a'$ is a well defined and strictly increasing function on the Turing degrees. Of particular interest is the degree $0'$, the degree of the jump of a computable set, which is also the degree of $K$.

The arithmetical hierarchy of sets of natural numbers classifies the complexity of the formulae needed to define such sets. A formula with only bounded quantifiers is in both the classes $\Sigma_0^0$ and $\Pi_0^0$. Then we inductively define the hierarchy as follows, in the manner due to Kleene [22] and (independently) Mostowski [33], using the Prenex Normal Form of Kuratowski and Tarski [27].
• A formula is in the class $\Sigma^0_n$ if it is logically equivalent to a formula of the form $\exists \bar{x} \varphi(\bar{x})$, where $\varphi(\bar{x})$ is in the class $\Pi^0_{n-1}$.

• A formula is in the class $\Pi^0_n$ if it is logically equivalent to a formula of the form $\exists \bar{x} \varphi(\bar{x})$, where $\varphi(\bar{x})$ is in the class $\Sigma^0_{n-1}$.

Then a set is in the class $\Sigma^0_n$ if it can be defined by a $\Sigma^0_n$ formula, and similarly for $\Pi^0_n$.

The intersection of the classes $\Sigma^0_n$ and $\Pi^0_n$ is called $\Delta^0_n$. The Turing computable sets are exactly those in class $\Delta^0_1$, and the computably enumerable sets are exactly those in $\Sigma^0_1$. The arithmetical hierarchy on degrees is defined in the obvious way, a degree is in $\Sigma^0_n$ if it contains a set in $\Sigma^0_n$ and the same for $\Pi^0_n$. In this thesis, the superscript 0 will be assumed, and for simplicity of notation will be suppressed.

The degrees below (Turing reducible to) $0'$ are called the local degrees, and are exactly those in the class $\Delta_2$ (by Post’s Theorem [38]). In general the $\Delta_{n+1}$ sets are precisely those Turing reducible to $0^{(n)}$. The sets in the local degrees will be of particular interest in this thesis, and we will need to define a finer-grained hierarchy on these sets.

### 1.3 The Ershov Hierarchy

The Ershov hierarchy, also known as the difference hierarchy, or the hierarchy of $\alpha$-c.e. sets, characterises the $\Delta_2$-definable sets by exploiting the Shoenfield Limit Lemma, which approximates $\Delta_2$ sets in a limit computable way.

**Lemma 1.1** (Shoenfield Limit Lemma, 1959 [45]).

A set $A$ is $\Delta_2$ if and only if there is a computable binary function $f : \mathbb{N}^2 \to \{0, 1\}$ such that for all $n \in \mathbb{N}$ there are cofinitely many stages $s$ at which $\chi_A(n) = f(n, s)$, (where $\chi_A$ is the characteristic function of $A$) that is that $\lim_{s \to \infty} f(n, s)$ exists and is equal to $\chi_A(n)$. 
Chapter 1. Introduction

The $n$-c.e. hierarchy for finite $n$ was introduced by Putnam (1965 [41]) and Gold (1965 [18]), then it was extended to transfinite $\alpha$-c.e. by Ershov (1968-70 [13] [14] [15]).

Intuitively we think of a c.e. set as a set which can be approximated by a function $A(s, x) : \mathbb{N}^2 \to \{0, 1\}$, where $A(0, x) = 0$ for all $x$, i.e. we start by guessing that $x \notin A$ at stage 0, then if at some time $t$ we enumerate $x$ into $A$, we have $A(t, x) = 1$, and $A(t', x) = 1$ for all $t' > t$. Notice that the approximation changes at most once for a given $x$ as $s$ increases.

If we allow the approximation to change more than once, we can approximate a larger class of sets, and for each $n$, we call the class of sets which can be approximated by allowing the approximation as discussed above to change up to $n$ times for each $x$, the class of $n$-c.e. sets. This informally defines the finite levels of the Ershov hierarchy, and we can give a formal definition as follows.

**Definition 1.2.**

A set $A \subseteq \mathbb{N}$ is $n$-computably enumerable if either $n = 0$ and $A = \emptyset$ or $n > 0$ and there are c.e. sets $R_0 \supseteq R_1 \supseteq \cdots \supseteq R_{n-1}$ such that,

$$A = \left\lfloor \frac{n-1}{2} \right\rfloor \bigcup_{i=0}^{\lfloor \frac{n-1}{2} \rfloor} (R_{2i} \setminus R_{2i+1}) \quad \text{(if $n$ is odd then set $R_n = \emptyset$).}$$

Or equivalently, if there is a computable binary function $f : \mathbb{N}^2 \to \{0, 1\}$ such that for all $x$, $f(0, x) = 0$ and $A(x) = \lim_{s \to \infty} f(s, x)$, and $|\{s \mid f(s + 1, x) \neq f(s, x)\}| \leq n$.

Following the notation introduced by Ershov, we call the class of $n$-c.e. sets level $\Sigma_n^{-1}$ of the Ershov hierarchy, with the complements of those sets constituting level $\Pi_n^{-1}$ and the intersection of $\Sigma_n^{-1}$ and $\Pi_n^{-1}$ is called $\Delta_n^{-1}$.

It is obvious that $\Sigma_n^{-1} \subset \Sigma_{n+1}^{-1}$, since any $n$-c.e. set is also an $(n + 1)$-c.e. set.
But the inclusion is strict because for every $n > 0$ there is an $(n + 1)$-c.e. set which is not $n$-c.e., and which has a complement which is $(n + 1)$-c.e. and not $n$-c.e., so $\Sigma_{n}^{c.e} \subsetneq \Sigma_{n+1}^{c.e}$ and $\Pi_{n}^{c.e} \subsetneq \Pi_{n+1}^{c.e}$. Then obviously we have the hierarchy theorem that for all $n > 0$, $\Sigma_{n}^{c.e} \cup \Pi_{n}^{c.e} \subsetneq \Sigma_{n+1}^{c.e} \cap \Pi_{n+1}^{c.e}$.

We define $\Sigma_{\omega}^{c.e}$, the first infinite level of the Ershov hierarchy, in a similar way.

**Definition 1.3.**

A set $A \subseteq \mathbb{N}$ belongs to level $\Sigma_{\omega}^{c.e}$ of the Ershov hierarchy, if there is an $\omega$-sequence of uniformly c.e. sets $\{R_e\}_{e \in \omega}$ such that $A = \bigcup_{n=0}^{\infty}(R_{2n+1} \setminus R_{2n})$.

Then $\Pi_{\omega}^{c.e}$ is the class of complements of sets in $\Sigma_{\omega}^{c.e}$, and $\Delta_{\omega}^{c.e} = \Sigma_{\omega}^{c.e} \cap \Pi_{\omega}^{c.e}$, and we describe the sets in $\Delta_{\omega}^{c.e}$ as $\omega$-c.e. The $\omega$-c.e. sets can be characterised in a number of different ways, which may be more or less useful depending on the situation. We present some of these characterisations below, following Arslanov [1].

**Theorem 1.4 (Arslanov [1]).**

Let $A \subseteq \mathbb{N}$, then the following are equivalent:

1. $A$ is $\omega$-c.e.
2. There exists an $\omega$-sequence of uniformly c.e. sets $\{R_e\}_{e \in \omega}$ such that $\bigcup_{e \in \mathbb{N}} R_e = \mathbb{N}$ and $A = \bigcup_{n=0}^{\infty}(R_{2n+1} \setminus R_{2n})$.
3. There exist computable functions $f, g$ such that for all $s, x \in \mathbb{N}$, $A(x) = \lim_{s \to \infty} g(s, x)$ and $|\{s \mid g(s, x) \neq g(s + 1, x)\}| \leq f(x)$.
4. There is a partial computable function $\varphi$ such that for all $x \in \mathbb{N}$, $A(x) = \varphi(\mu t(\varphi(t, x) \downarrow), x)$.
5. $A$ is tt-reducible to $0'$. 
For further background on Computability Theory we refer the reader to standard texts such as Cooper [10], Nies [35], Odifreddi [36] [37] and Soare [49]. For further details on the Ershov hierarchy we refer to Arslanov [1], Ershov [13] [14] [15] and Stephan, Yang and Yu [50].

1.4 Priority Arguments

Priority arguments are one of the key tools of computability theory, used in proofs in all areas of computability. They are used to construct a structure (or several structures) with some particular properties in a computable way. The idea is to break the property down into infinitely many requirements, and then come up with a strategy to satisfy each requirement individually, and then a method to combine all of the strategies. In order to deal with clashes when combining the strategies, we put a priority ordering on the requirements, so that if at any point two or more requirements are in conflict then the requirement with highest priority takes precedence. Once the construction ends and the argument is completed, we have a verification to show that the argument is sound, and the construction satisfies the properties that we want.

Priority arguments originated with Kleene and Post [25] who gave the first use of this method in the proof of Theorem 1.5.

Theorem 1.5 (Kleene and Post 1954 [25]).
There exist incomparable degrees below $0'$

Note that in this argument each requirement is satisfied without injuring any other requirements. This is the simplest type of priority argument, a more complicated argument allows requirements to injure requirements of lower priority, by which we mean cause them to become unsatisfied if they have been satisfied before the higher priority
requirement. This type of priority argument was independently developed by Friedberg [16] and Muchnik [34] to prove Theorem 1.6.

**Theorem 1.6** (Friedberg 1957 [16] and Muchnik 1956 [34]).

There exist incomparable c.e. degrees.

There is an important distinction to be drawn between arguments which allow requirements to be injured infinitely often, and arguments which only allow finite injury. The theorems in this thesis will only use finite injury priority arguments, so we will not go into detail on how infinite injury works. To give an example of the use of the finite injury priority method related to linear orderings, we prove the following well-known theorem, following the proof as presented by Downey [11]. This is not the simplest proof, it is possible to prove this result without using the priority method, but we hope this will be illustrative of how it works.

We first need to define the concept of a mathematical structure (in this case a linear order) being 1-computable. A structure $A$ is 1-computable if any logical sentence with one quantifier can be effectively decided in the structure. Note that this is stronger condition than $A$ being computable, which just means that the order relation is computable.

**Theorem 1.7** (Folklore, see for example Downey [11] Theorem 2.11).

There is a computable linear ordering $A$ of order type $\omega$ with $S(x)$, the successor function, not computable. Hence, $A$ is not 1-computable since the adjacency relation is not computable.

**Proof.** We build a computable linear order $A$ as the union of finite linear orderings $\bigcup_s A_s$. The two properties that we want $A$ to have are that it has order type $\omega$ and $S(x)$ is not computable in $A$. We can split each property into infinitely many requirements. Labelling the elements of $A$ as $a_0, a_1, a_2, \ldots$, and using a standard computable listing of partial computable functions $\{\varphi_e\}_{e \in \mathbb{N}}$, we get the requirements as follows, for each $e \in \mathbb{N}$.
$N_e$: $e$ has finitely many predecessors.

$P_e$: $\varphi_e$ is not the successor function on $A$.

Where clearly the $N$ requirements ensure $A$ has order type $\omega$, and the $P$ requirements ensure that $S(x)$ is not computable.

In order to satisfy $P_e$, we choose some $x = x(e)$ and wait for a stage $s$ such that $\varphi_{e,s}(x) \downarrow$. If this never happens then $\varphi_e$ is not total and $P_e$ is satisfied. If there is some $s$ such that $\varphi_{e,s}(x) \downarrow$ then we place a new point between $x$ and $\varphi_e(x)$, ensuring that $\varphi_e(x) \neq S(x)$. This strategy for satisfying $P_e$ is called the basic module for $P_e$.

This has the potential to cause problems for $N_i$, as if infinitely many $P$ requirements are allowed to place new points below $a_i$ then $N_i$ will not be satisfied. In order to avoid this, we use the priority ordering $N_0 \succ P_0 \succ N_1 \succ P_1 \succ N_2 \succ P_2 \cdots$. Thus only finitely many $P$ requirements have higher priority than a given $N$ requirement $N_i$, and so only finitely many can place a new point below $a_i$. If we ensure that each $P_e$ with $e <_N i$ only injures $N_i$ finitely many times, that is only places finitely many new points below $a_i$, then $N_i$ will be satisfied.

The strategy we use to combine the basic modules for the $P_e$ is just to ensure that we choose $x(e)$ so that it is not below $a_i$ for $i \leq_N e$. Then $P_e$ will not add a predecessor of $a_i$ if $i \leq_N e$. In the construction we will say that $P_e$ requires attention at stage $s + 1$ if either $P_e$ has not chosen an element $x(e)$, or $P_e$ has chosen $x(e)$ and $\varphi_{e,s}(x(e)) \downarrow$ and $\varphi_{e,s}(x(e)) = S(x(e))$.

The construction proceeds by stages. At Stage 0, we set $A_0 = \{a_0\}$. At the start of stage $s + 1$, we have $\leq_{A_s}$ as some linear order on the set $\{a_0, \ldots, a_s\}$. We search for the least $e \leq s$ such that $P_e$ requires attention. We know that there will be at least one
$P_e$ with $e \leq s$ which requires attention, as we cannot have chosen $x(e)$ for $e = s$. If $P_e$ requires attention because there is no $x(e)$, then define $A_{s+1}$ by placing $a_{s+1}$ above all $a_i$ for $i < N_s + 1$ and declare that $x(e) = a_{s+1}$. Alternatively, if $P_e$ requires attention because $\varphi_{e,s}(x(e)) \downarrow$ and $\varphi_{e,s}(x(e)) = S(x(e))$ then define $A_{s+1}$ by placing $a_{s+1}$ between $x(e)$ and $S(x(e))$. Then proceed to Stage $s + 2$.

It is clear that each $P_e$ requires attention at most twice, each time adding one element to the linear order. Hence, each $N_i$ requirement will be satisfied because only finitely many points can be added as predecessors to $a_i$, since only $P_e$ for $e < i$ can add such points. This means that $N_i$ can only be injured finitely often, and therefore all the requirements will be satisfied, and $A$ will have the desired properties.

\[ \square \]

1.5 Order Theory

The theory of ordering is based on the highly intuitive notion of some numbers being bigger than others. If we abstract this notion to a general notion of an order on a set, we can define an ordering $P$ as a set $P$ equipped with a binary relation $\leq_P$ with the following properties. We choose here for convenience to use a non-strict ordering, which conforms to the usual notion of $\leq$, rather than a strict order which conforms to the notion of $<$.

- Reflexivity, $\forall p \in P, p \leq_P p$.
- Antisymmetry, $\forall p, q \in P$, if $p \leq_P q$ and $q \leq_P p$ then $p = q$.
- Transitivity, $\forall p, q, r \in P$, if $p \leq_P q$ and $q \leq_P r$ then $p \leq_P r$.

These properties define a partial order, and if we further require that any two elements are comparable, then we define a linear order, also known as a total order.
In general partial orders are defined on any set, but for the purposes of this thesis we will only be using countable domains, in particular \( \mathbb{N} \) or subsets of \( \mathbb{N} \).

An order type is a equivalence class under order isomorphism of partial orders, though we will often refer to it as if it is a partial order of that type. Order types will be denoted by lower case Greek letters, for example, \( \omega \) is the order type of the natural numbers. Ordinals are a subset of order types, and we will use an extension of ordinal arithmetic on order types.

If \( \sigma \) and \( \tau \) are order types, then \( \sigma + \tau \) is the order type formed by concatenation of \( \sigma \) and \( \tau \), and \( \sigma \tau \) is the order type formed by replacing every point in \( \tau \) by a copy of \( \sigma \). Also we denote the reverse ordering of \( \sigma \) by \( \sigma^* \).

There are several properties of orderings which we will be concerned with in this thesis and which I will introduce now. The most well-known property is that of being well-founded, this means that every non-empty chain of \( P \) has a \( \leq_P \)-minimal element. Equivalently, we say that an ordering is well-founded if it contains no infinite descending chain, that is, it does not embed \( \omega^* \), the order type of the negative integers.

Another key property of orderings is that of being scattered. An ordering is scattered if it contains no dense chain, with density here being defined as follows. If \( \mathcal{P} = (P, \leq_P) \) is a partial ordering, and given a chain \( Q \subseteq P \), then \( Q \) is dense in \( P \) if for all \( p, q \in Q \) with \( p \leq_P q \), there is some \( r \in Q \) such that \( p <_P r <_P q \). Equivalently we can say an ordering is scattered if it does not embed \( \eta \), the order type of the rational numbers.

We say a linear order \( \mathcal{L} \) is indecomposable if whenever we can write \( \mathcal{L} = A + B \) for some linear orderings \( A, B \), either \( A \) or \( B \) embeds \( \mathcal{L} \). We say \( \mathcal{L} \) is right-indecomposable if whenever \( \mathcal{L} = A + B \) and \( B \) is not empty, then \( B \) embeds \( \mathcal{L} \). Similarly we say \( \mathcal{L} \) is
left-indecomposable if whenever $L = A + B$ and $A$ is not empty, then $A$ embeds $L$.

We call a partial order $\mathcal{P} = (P, \leq_P)$ computable, if $P$ is a computable subset of $\mathbb{N}$ and $\leq_P$ is a computable subset of $\mathbb{N}^2$, using a suitable coding. Then we can classify non-computable orderings by the complexity of the order relation, as all orderings we will discuss in this thesis will have a computable set as the domain, in fact we will mostly consider orderings on $\mathbb{N}$.

1.6 Extensions

Given a partial order $\mathcal{P} = (P, \leq_P)$, we say that $\mathcal{L} = (L, \leq_L)$ is a linear extension, or linearisation of $\mathcal{P}$, if $L$ is a linear order, $P = L$ and $\leq_P \subseteq \leq_L$. We know that due to an old theorem of Szpilrajn, linear extensions always exist, and in fact this can be effectivised.

**Theorem 1.8** (Szpilrajn 1930 [51]).
Every partial order has a linear extension.

**Theorem 1.9** (Folklore, see for example Downey 1998 [11]).
Every computable partial order has a computable linear extension.

There has been a lot of work in the area of preservation under linearisation of various properties of partial orders such as those discussed above. Considerable work was done here by Bonnet, Pouzet, Jullien, Galvin, McKenzie and others (see [5]) on the question of whether partial orders without a countable subordering of a certain order type must always have a linear extension which also has no suborderings of that type. If an order type can be avoided in such a way, then we call it extendible.

**Definition 1.10.**
If $\alpha$ is a countable order type, we say $\alpha$ is extendible if any partial order $\mathcal{P}$ which does not embed $\alpha$ can be extended to a linear order $\mathcal{L}$ which does not embed $\alpha$. 
This culminated in the complete classification of countable extendible order types by Bonnet in 1969 [3]. This includes the order type $\omega^*$, the avoidance of which makes a partial order well-founded, and the order type $\eta$, the avoidance of which makes a partial order scattered.

**Definition 1.11.**

We define a family of order types, by transfinite induction over the countable ordinals, i.e. the ordinals below $\omega_1$.

- $\pi_1 = \omega$.
- For a successor ordinal $\alpha + 1 < \omega_1$, $\pi_{\alpha+1} = \pi_\alpha^* \omega$.
- For a limit ordinal $\lambda < \omega_1$, $\pi_\lambda = \sum_{\alpha < \lambda} \pi_\alpha^*$.

**Theorem 1.12** (Bonnet 1969 [3]).

A countable order type $\alpha$ is extendible if and only if:

- $\alpha \equiv \eta$, or
- $\alpha \equiv \pi_\nu$ for some $\nu < \omega_1$, or
- $\alpha \equiv \pi_\nu^*$ for some $\nu < \omega_1$, or
- $\alpha \equiv \pi_\nu^* + \pi_\nu$ for some $\nu < \omega_1$.

We must note that the definition of partial order that was used here is not the same as ours, in particular Bonnet considered partial orders over any set as the domain, not just partial orders on countable domains. If we restrict to partial orders on a countable domain, then we get a different classification, as proved by Jullien [21]. We use the formulation of Theorem 1.15 given by Montalbán [32] rather than Jullien’s original theorem, but it is equivalent, as Montalbán showed.
Definition 1.13.
A linear order type $\alpha$ is weakly extendible, when $\alpha$ is countable and any countable partial order that does not embed $\alpha$ can be extended to a linear order that does not embed $\alpha$.

Definition 1.14. A segment $B$ of a linear ordering $L = A + B + C$ is essential if whenever we have $L \leq A + B' + C$ for some linear ordering $B'$, then it must be the case that $B \leq B'$.

Theorem 1.15 (Jullien 1969 [21]).
A scattered linear ordering $L$ is extendible if and only if it does not have an essential segment $B$ such that

- $B = R + Q$, where $R$ is right-indecomposable and $Q$ is left-indecomposable, or
- $B = 2$.

There has been a lot of work carried out on linear extensions of partial orders, other than concerning extendible order types. We give some examples here to show the context for the work on extendible types. We are still concerned here with classes of partial orders which have particular properties, and whether or not they can be guaranteed to have linear extensions with those properties, or possibly other properties.

In some unpublished notes, Miller [31] proved some interesting results about the computational power of scattered extensions, in particular the existence of scattered partial orders such that any scattered extensions can compute various sets.

Definition 1.16.
Given sets $A$ and $B$, a separator of $A$ and $B$ is a superset of $A$ which is disjoint from $B$.

Theorem 1.17 (Miller 2015 [31]).
Let $A, B$ be disjoint $\Sigma^1_1$ sets. There is a scattered computable partial order such that any scattered linear extension computes a separator of $A$ and $B$. 
Theorem 1.18 (Miller 2015 [31]).
Let $A, B$ be disjoint c.e. sets. There is a scattered computable partial order such that any linear extension either computes a suborder of itself of type $\eta$ or computes a separator of $A$ and $B$.

Theorem 1.19 (Miller 2015 [31]).
Let $A, B$ be disjoint $\Sigma^0_2$ sets. There is a scattered computable partial order such that any (classically) scattered linear extension computes a separator of $A$ and $B$.

Pouzet and Rival [40] investigated extensions of partial orders with the family of properties around chain-completeness.

Definition 1.20.
We say a linearly ordered subset (chain) of a partial order is saturated if no element of the partial order between two elements of the chain can be added to the subset without losing the property of being linearly ordered. A stronger property is a chain being maximal, which is when no element can be added without losing the property of being linearly ordered, even at the top or bottom of the subset.

Definition 1.21.
We say a partial order $\mathcal{P}$ is chain-complete if every maximal chain of $\mathcal{P}$ is complete, that is, every subchain of a maximal chain has both an infimum and a supremum. We say $\mathcal{P}$ is locally chain-complete if every interval in $\mathcal{P}$ is chain-complete. If $\mathcal{P}$ is a chain and it is locally chain-complete then we call it locally complete.

Theorem 1.22 (Pouzet and Rival 1981 [40]).

- Every locally chain-complete ordered set in which all antichains are finite has a locally complete linear extension.
- Every countable, locally chain-complete partially ordered set has a locally complete linear extension.
In response to an unpublished question of Łoś, Rutkowski found some necessary conditions and some sufficient conditions for a partial order to have a linear extension of order type \( \eta \).

**Theorem 1.23** (Rutkowski 1996 [44]).

- If \( P \) is a partial order which has a linear extension of order type \( \eta \), then \( P \) contains an infinite antichain or a nontrivial dense saturated chain.

- If \( P \) is a countable partially ordered set which contains a maximal chain \( C \) of order type \( \eta \), such that for each \( p \in C \), the set \( \{ q \in P \mid \forall c \in C, (c <_P p \iff c <_P q \land p <_P c \iff q <_P c) \} \) is either equal to \( \{p\} \) or has an extension of order type \( \eta \), then \( P \) has a linear extension of order type \( \eta \).

- If \( P \) is a countable partially ordered set which satisfies the property that if \( \mathcal{X}, \mathcal{Y} \) are finite antichains in \( P \) such that no element of \( \mathcal{X} \) is \( \leq_P \)-above any element of \( \mathcal{Y} \), then there exists an element \( p \in P \) which is neither in the upward cone of \( \mathcal{X} \) or the downward cone of \( \mathcal{Y} \). Then \( P \) will have a linear extension of order type \( \eta \). Moreover \( \leq_P \) is the intersection of all such extensions.

Slaman and Woodin further investigated the question of when a partial order has a linear extension of order type \( \eta \), and showed that it has no easy answer.

**Definition 1.24.** If \( X \) is a subset of \( \mathbb{N} \), then consider \( \{e\}^X \) as a suborder of \( \omega^{<\omega} \) and define

\[
\mathcal{L}(X) = \{e \mid \{e\}^X \text{ is a partial order on } \mathbb{N} \text{ and has a linearisation of order type } \eta\}.
\]

Then let \( \mathcal{L} = \{\langle e, X \rangle \mid e \notin \mathcal{L}(X)\} \).

**Theorem 1.25** (Slaman and Woodin 1998 [48]).

There is a computable function \( f \) such that for all \( e, X, \{e\}^X \) is a well-founded subtree of \( \omega^{<\omega} \) if and only if \( f(e) \notin \mathcal{L}(X) \).

By a theorem of Kleene [24], the collection of well-founded subtrees of \( \omega^{<\omega} \) is \( \Pi^1_1 \)-complete hence, \( \mathcal{L} \) is a complete \( \Sigma^1_1 \)-predicate, and therefore not a Borel set.
In model theory, Ramakrishnan and Steinhorn [42] extending work of Macpherson and Steinhorn [30] proved the following theorem concerning classes of first-order structures in which definable partial orders have definable linear extensions.

**Theorem 1.26** (Ramakrishnan and Steinhorn 2014 [42]).

Let $M = (M, <, \ldots)$ be a first-order structure which is weakly o-minimal, o-minimal, quasi-o-minimal, well-ordered, or elementarily equivalent to a structure with one of those properties, in which the symbol $<$ is a linear order on $M$. Then if $P = (P, \prec)$ is a partial order on a subset $P \subseteq M^n$ for some $n$ which is definable in $M$, then $P$ will have a linear extension definable in $M$.

### 1.7 Reverse Mathematics

There has also been a lot of recent interest in the proof-theoretic content of the theorems about extendibility of order types, investigated using the methods of reverse mathematics. The idea is to determine, for a given theorem, the minimal axiomatic system which is sufficient to prove it. In particular we consider the strength of set comprehension needed to prove the theorem. A comprehension scheme is the collection of axioms $\exists X \forall n (n \in X \leftrightarrow \varphi(n))$, for all formulae $\varphi$ in some particular class.

It is the size of the class of formulae needed which we use to classify the strength of the theorem. Typically classes used are defined by allowable quantifier depth, we consider for example $\Delta^0_1$-comprehension and $\Pi^1_1$-comprehension amongst others, but other measures of formula complexity are also used.

As a base, we use the finite axioms of Peano Arithmetic (i.e. PA without the induction schema) along with the $\Sigma^0_1$-induction scheme, which means that for all $\Sigma^0_1$ formulae $\varphi$, $(\varphi(0) \land \forall x (\varphi(x) \rightarrow \varphi(x + 1))) \rightarrow \forall n \varphi(n)$.
On top of this base system, Simpson [47] describes five successively stronger systems, which are generally known as the **Big Five** (see, for example, Hirschfeldt [19]). These systems are all frequently used in reverse mathematics. The list below gives the axioms which need to be added to the base system to get each of the Big Five, and are listed in order of strictly increasing proof-theoretic strength.

It turns out that many classical theorems are equivalent to one of the above Big Five systems, and much work in reverse mathematics involves proving that a given theorem is weaker, stronger or equivalent to one of the above. Significant work has been done on placing the theorems which state the extendibility of certain key order types in the reverse mathematics classification, although precise equivalences have not been found in all cases.

- **Recursive comprehension axiom** \( \text{RCA}_0 \) \( \exists X \forall n (n \in X \leftrightarrow \varphi(n)) \) for \( \Delta^0_1 \) formulae \( \varphi \).

- **Weak König’s Lemma** \( \text{WKL}_0 \) Every infinite binary tree has an infinite path.

- **Arithmetical comprehension axiom** \( \text{ACA}_0 \) \( \exists X \forall n (n \in X \leftrightarrow \varphi(n)) \) for arithmetical formulae \( \varphi \).

- **Arithmetical transfinite recursion** \( \text{ATR}_0 \) Any arithmetical operator can be iterated tranfinitely along any countable well-order.

- **\( \Pi^1_1 \) comprehension axiom** \( \Pi^1_1\text{-CA}_0 \) \( \exists X \forall n (n \in X \leftrightarrow \varphi(n)) \) for \( \Pi^1_1 \) formulae \( \varphi \).

The research that has been conducted into the question of determining the proof-theoretic strength of the statement of the extendibility of various order types can be summarised as follows.
Chapter 1. Introduction

Theorem 1.27 (Downey, Hirschfeldt, Lempp and Solomon 2003 [12]).

- “$\omega$ is extendible” is provable in ACA$_0$.
- “$\omega$ is extendible” proves WKL$_0$ over RCA$_0$.
- “$\omega$ is extendible” is not provable in WKL$_0$.

Theorem 1.28 (Downey, Hirschfeldt, Lempp and Solomon 2003 [12]).

- “$\eta$ is extendible” is provable in $\Pi^1_1$-CA$_0$.
- “$\eta$ is extendible” is not provable in WKL$_0$.

Theorem 1.29 (J. Miller 2015 [31]).

- “$\eta$ is extendible” implies WKL$_0$ over RCA$_0$.
- “$\eta$ is extendible” implies ATR$_0$ over $\Sigma^1_1$-CA$_0$.

Theorem 1.30 (Montalbán 2006 [32]).

- “$\eta$ is extendible” is provable in ATR$^*$.

Theorem 1.31 (Downey, Hirschfeldt, Lempp and Solomon 2003 [12]).

- “$\zeta$ is extendible” is equivalent to ATR$_0$ over RCA$_0$.

Another interesting reverse mathematical result in the field of extendible order types involves Jullien’s classification of weakly extendible order types (Theorem 1.15) and Fraïssé’s conjecture. We recall Jullien’s result from Theorem 1.15 and the statement of Fraïssé’s conjecture below.

---

$^6$Downey et. al. actually proved that “$\eta$ is extendible” is provable in $\Pi^1_1$-CA$_0$, which was then improved to $\Pi^1_1$-CA$_0$ with Becker, in the same paper.

$^7$ATR$_*$ is the system ATR$_0$ with $\Sigma^1_1$-induction.

$^8$Proved by Laver 1971 [28].
**Theorem 1.32** (Fraïssé’s conjecture, Laver 1971 [28]).

The class of countable linear orderings, quasiordered by the relation of embeddability, contains no infinite descending chain and no infinite antichain.

**Theorem 1.33** (Montalbán 2006 [32]).

Jullien’s theorem is equivalent to Fraïssé’s conjecture over RCA\(_0\) + \(\Sigma^1_1\)-IND.

It should be noted that Shore [46] had previously proved that Fraïssé’s conjecture is proof-theoretically strong, indeed it implies ATR\(_0\) over RCA\(_0\), so Montalbán’s result shows that Jullien’s theorem is also strong.

### 1.8 Thesis Outline

In this thesis we consider the computable content of Bonnet’s Theorem (Theorem 1.12), extending previous work of Rosenstein, Statman and Kierstead [43], Downey, Hirschfeldt, Lempp and Solomon [12] and Cooper, Lee and Morphett [29].

The question is whether given a computable partial order which does not computably embed some extendible computable order type, what is the lowest level of complexity at which we can always find a linear extension which also does not computably embed that order type?

In Chapter 2 we look at well-founded orderings, i.e. orderings which do not embed \(\omega^*\), and improve the lower bound proved by Rosenstein and Kierstead to show that the upper bound previously proved by Cooper, Lee and Morphett is the best possible result.

**Theorem 1.34** (Theorem 2.8, Theorem 2.14, Corollary 2.19).

If \(\mathcal{P}\) is a computably well-founded computable partial order, then \(\mathcal{P}\) has a computably well-founded \(\omega\)-c.e. linear extension. However, there is a computably well-founded
computable partial order $\mathcal{P}$ which has no computably well-founded $n$-c.e. linear extension for any $n \geq 1$.

In Chapter 3 we consider the order type $\zeta = \omega^* + \omega$ and prove an analogous result for $n$-c.e. extensions, but then show that this in fact extends to $\omega$-c.e. and $\Delta_2$ extensions.

**Theorem 1.35** (Theorem 3.9, Theorem 3.5).
There is a computable partial order $\mathcal{P}$ which does not computably embed $\zeta$ and has no $n$-c.e. linear extension which does not computably embed $\zeta$ for any $n \geq 1$. There is a computable partial order $\mathcal{P}$ which does not computably embed $\zeta$ and has no $\omega$-c.e. linear extension which does not computably embed $\zeta$. There is a computable partial order $\mathcal{P}$ which does not computably embed $\zeta$ and has no $\Delta_2$ linear extension which does not computably embed $\zeta$.

In Chapter 4 we consider the case of scattered orderings, i.e. orderings which do not embed $\eta$. We show that the order type $\eta$ behaves similarly to $\omega^*$.

**Theorem 1.36** (Theorem 4.14, Theorem 4.18, Corollary 4.22).
If $\mathcal{P}$ is a computably scattered computable partial order, then $\mathcal{P}$ has a computably scattered $\omega$-c.e. linear extension. However, there is a computably scattered computable partial order $\mathcal{P}$ which has no computably scattered $n$-c.e. linear extension for any $n \geq 1$.
Chapter 2

Well-founded orderings

We consider what is known about linearisations of well-founded partial orders, and the computable content of those theorems in the Ershov hierarchy. We look at the previously proved bounds on the complexity of linearisations which preserve well-foundedness, and prove a new lower bound, closing the gap to give a sharp result.

2.1 Background

We know from the Szpilrajn Extension Theorem (Theorem 1.8) that any partial order has a linear extension, so it is an obvious line of inquiry to ask whether a partial order which has a particular order-theoretic property must have a linear extension which shares that property. The most well known order theoretic property is probably that of well-foundedness. It is often defined as the property that any suborder will have minimal elements, but here we will use the following equivalent definition.

Definition 2.1.
A partial order $\mathcal{P}$ is well-founded if there is no infinite descending sequence under $\leq_{\mathcal{P}}$, i.e. $\mathcal{P}$ contains no subordering of type $\omega^*$. 
Classically, it is known that well-foundedness can be preserved by linearisation, due to a result of Bonnet. This is relatively easy to see, as we demonstrate below.

**Theorem 2.2** (Bonnet 1969 [3]).

Every well-founded partial order $\mathcal{P} = (P, \leq_P)$ has a well-founded linear extension. I.e. $\omega^*$ is extendible.

*Proof.* Inductively define a partition of $P$ as follows: let $P_0$ be the set of $\leq_P$-minimal elements of $P$, then let $P_\alpha$ be the set of $\leq_P$-minimal elements of $P \setminus \bigcup_{\beta < \alpha} P_\beta$. This process will terminate for some $P_\delta = \emptyset$. Clearly each $P_\alpha$ is an antichain in $\mathcal{P}$, so let $\mathcal{L}(P_\alpha)$ be a well-founded linear extension of $P_\alpha$, then $\sum_{\alpha < \delta} \mathcal{L}(P_\alpha)$ is a well-founded linear extension of $\mathcal{P}$. \qed

Kierstead and Rosenstein [43] demonstrated a version of Bonnet’s result for computable orderings, but at the same time Rosenstein and Statman [43] showed that a version with a computable version of the well-foundedness property does not hold. We use the notion of computable well-foundedness, as introduced by Rosenstein.

**Theorem 2.3** (Kierstead and Rosenstein 1984 [43]).

Every well-founded computable partial order has a well-founded computable linear extension.

*Proof.* Suppose $\mathcal{P} = (P, \leq_P)$ is a computable partial order, and that it is well-founded. Then we construct a computable linear extension $\mathcal{L} = (P, \leq_L)$ of $\mathcal{P}$ as follows.

Suppose at the start of stage $n$ we have already defined whether $a \leq_L b$ holds for all $a, b < \mathbb{N}$, and $\leq_{L,n-1}$ is a linear ordering of $\{1, 2, \ldots, n - 1\}$ which extends $\leq_P|_{n-1}$. At stage $n$ we must decide where $n$ should be placed in the ordering that we are constructing. We choose to place $n$ as high as possible whilst extending the ordering of $\mathcal{P}$, that is we place it immediately below the $\leq_{L,n-1}$-smallest number which is $\leq_P$-greater
than \( n \).

In Figure 2.1 we depict the linear order on the first \( n - 1 \) elements, and have circled the elements which are \( \preceq_P \)-greater than \( n \) and squared the elements which are \( \preceq_P \)-less than \( n \). The arrow then shows where \( n \) will be placed in the linearisation.

![Figure 2.1: Construction of the linear extension](image)

It is clear that this method will produce a computable linear extension of \( P \). Now suppose that \( L \) is not well-founded, so there exists some infinite descending chain \( x_0 >_L x_1 >_L x_2 >_L \cdots \). Without loss of generality we can assume that \( x_0 <_N x_1 <_N x_2 <_N \cdots \) and we know that for all \( i <_N j \) we must have either \( x_i >_P x_j \) or \( x_i \not\preceq_P x_j \). Since we assume that \( P \) is well-founded, by Ramsey’s theorem we can assume that \( x_i \not\preceq_P x_j \) for all \( i, j \), by ignoring some of the sequence if necessary.

Therefore, for each \( i > 0 \) we know that \( x_i \not\preceq_P x_0 \), \( x_i >_N x_0 \) and \( x_i \not<_L x_0 \), and hence there must be some \( y_i \) such that \( y_i <_N x_0 \) and \( x_i \not<_P y_i \preceq_L x_0 \). Since each \( y_i <_N x_0 \) there must be some \( y_0 <_N x_0 \) such that \( x_i <_P y_0 \preceq_L x_0 \) for infinitely many \( x_i \), and hence we may assume that \( y_0 \) is chosen minimally with this property, and by eliminating other \( x_i \) we may thus assume that \( x_i <_P y_0 \preceq_L x_0 \) for all \( i > 0 \).

If we continue to inductively define \( y_i \) for all \( i \in \mathbb{N} \), we build a sequence \( y_0, y_1, y_2, \ldots \) such that for all \( i, j \in \mathbb{N} \) such that \( i >_N j \), we have \( x_i <_P y_j \preceq_L b_j \) and that all the \( y_j \) are chosen minimally (i.e. such that there is no \( z <_N y_j \) with \( x_i <_P z \) for infinitely many \( x_i \)). We know that none of the \( y_i \) are equal and in fact \( y_0 <_N y_1 <_N y_2 <_N \cdots \), because of the minimality and since \( y_i \not<_L x_i \not<_P y_j \) for all \( i >_N j \).
Since \( y_{j+1} <_L x_{j+1} <_P y_j \), we have \( y_{j+1} <_L y_j \). If \( y_{j+1} \upharpoonright_P y_j \), then since \( y_{j+1} \upharpoonright_N y_j \), there must be some \( z \upharpoonright_N y_j \) such that \( y_{j+1} <_P z <_L y_j \). But then \( x_i <_P y_{j+1} <_P z <_L y_j \) for all \( i \upharpoonright_N j + 1 \), which contradicts the minimality of the choice of \( y_j \). Hence, \( y_{j+1} <_P y_j \) for all \( j \), which contradicts the well-foundedness of \( \mathcal{P} \).

Therefore, no such chain \( x_0 >_L x_1 >_L x_2 >_L \cdots \) exists and \( \mathcal{L} \) must be well-founded. \( \square \)

This theorem is only partially effective. We can also effectivise the embedding of \( \omega^* \), by using the following definition instead of just wellfoundedness.

**Definition 2.4.**

A partial order \( \mathcal{A} \) is defined to be *computably well-founded* if there is no computable infinite descending sequence under \( \leq_A \), i.e. \( \mathcal{A} \) contains no computable subordering of type \( \omega^* \). Equivalently, if any computable suborder has a least element.

**Theorem 2.5** (Rosenstein and Statman 1984 [43]).

There is a computably well-founded computable partial order which does not have a computably well-founded computable linear extension.

**Proof.** The counterexample that we construct is a computable binary tree \( \mathcal{T} \) known as a Kleene Tree [23]. We start by computably generating the full downward-branching binary tree, labelling the nodes by natural numbers coding binary strings, as in Figures 4.11 4.11. And simultaneously enumerating all computably enumerable sets \( W_e \). If at some time we have enumerated the first \( e \) elements of \( W_e \) and they form a descending chain in the tree, then we “kill” the node corresponding to the \( e \)th element of \( W_e \), by which we mean that no new nodes will be added below it, so that the tree below that point remains finite. Note that this construction doesn’t halt at a finite stage, because at each level of the tree there are more nodes than c.e. sets that can possibly kill nodes at that level. So the tree constructed will be infinite.
This tree, whilst by König’s Lemma, clearly contains an infinite descending sequence, by construction cannot contain a computable infinite descending sequence, and is therefore computably well-founded.

Now, let \( \mathcal{L} \) be any computable linear extension of \( \mathcal{T} \). We will show that \( \mathcal{L} \) is not computably well-founded by constructing a computable infinite descending chain in \( \mathcal{L} \).

Let \( x_0 \) be the root of the tree, and \( P_0 \) be the set of immediate \( \mathcal{T} \)-predecessors of \( x_0 \). At the beginning of stage \( n \), we assume we have already defined a sequence of elements \( x_0, x_1, \ldots, x_{n-1} \) and a set \( P_{n-1} \). We further assume as part of the induction hypothesis that the sequence \( x_0, x_1, \ldots, x_{n-1} \) forms a terminal part of the tree, that is if \( y \) is above \( x_i \) in the tree, then \( y = x_j \) for some \( j < i \), and that \( P_{n-1} \) consists of all the of the immediate \( \mathcal{T} \)-predecessors of all of the \( x_i \), apart from \( x_0, x_1, \ldots x_{n-1} \).

Now, we define \( x_n \) to be the \( \mathcal{L} \)-greatest element of \( P_{n-1} \), and \( P_n \) to be \( P_{n-1} \) without \( x_n \) but with the \( \mathcal{T} \)-predecessors of \( x_n \) added. We now argue that the sequence \( x_0, x_1, \ldots \) has order type \( \omega^* \) in \( \mathcal{L} \).

Clearly all \( x_i \) are distinct, since \( x_i \notin P_i \) but \( x_{i+1} \in P_i \). Suppose that \( x_i <_L x_{i+1} \), then \( x_{i+1} \notin P_{i-1} \), as otherwise it would have been chosen as \( x_i \), so \( x_{i+1} \) must be a \( \mathcal{T} \)-predecessor of \( x_i \), and therefore \( x_{i+1} <_T x_i \), which is a contradiction of \( \mathcal{L} \) being a linear extension of \( \mathcal{T} \). We therefore have \( x_{i+1} <_L x_i \) for all \( i \) and hence \( \mathcal{L} \) is not computably well-founded.

The Kleene Tree is a natural example, though it is by no means the only way of constructing an infinite computable binary tree with no computable path, and indeed any such object would suffice to prove the theorem. It should be noted that this counter-example also gives the result for computably enumerable linearisations, because
a computably enumerable linear order with a computable domain, will in fact just be computable. Although Rosenstein did not claim this corollary, it immediately follows from Theorem 2.5.

**Corollary 2.6.**
There is a computably well-founded computable partial order which does not have a computably well-founded *computably enumerable* linear extension.

Theorem 2.5 is the first negative result, showing that there is a partial order which does not allow preservation of the given property under computable linear extension. This establishes the negative lower bound, and prompts us to look for a positive upper bound. Rosenstein and Statman also gave a positive result, that is they gave an upper bound on the complexity of the computation necessary to obtain a computably well-founded linear extension of a computably well-founded computable partial order.

**Theorem 2.7** (Rosenstein and Statman 1984 [43]).
A computably well-founded computable partial order has a computably well-founded $\Delta_2$ linear extension.

*Proof*. Given some computably well-founded computable partial order $\mathcal{P}$, we construct a linearisation $\mathcal{L}$ which is computably well-founded. In order for $\mathcal{L}$ to be computably well-founded, it suffices to show that no c.e. set $W_e$ enumerates an infinite descending sequence in $\mathcal{L}$.

We can ensure this for each $W_e$ in turn if we can ask an oracle if $W_e$ is infinite. If $W_e$ is finite it will clearly not enumerate an infinite descending sequence, but if it is infinite then we may need to construct $\mathcal{L}$ in such a way as to avoid an $\omega^*$-sequence. Of course if $W_e$ is infinite then we will have the opportunity to do this, because since $W_e$ cannot enumerate an $\omega^*$-sequence in $\mathcal{P}$, it must contain infinitely many $\mathcal{P}$-incomparable suborderings, which can be enumerated into $\mathcal{L}$ to not build an infinite descending
sequence.

“Is $W_e$ infinite?” is a $\Pi_2$ question, but in fact we can use a simpler $\Sigma_1$ oracle which answers the question “Does $W_e$ have another element?”, and hence the construction of $L$ in order to be computably well-founded is computable in a $\Sigma_1$ set, or in other words, $\mathcal{P}$ has a $\Delta_2$ linearisation which is computably well-founded.

## 2.2 The upper bound

Downey, in his 1998 survey article on Computability and Linear Orderings [11], asked whether this $\Delta_2$ bound of Rosenstein and Statman could be improved. In Lee’s PhD thesis [29], he presented a result, jointly with Cooper and Morphett, which improves the bound using the Ershov difference hierarchy.

Rosenstein’s argument in the proof of Theorem 2.7 uses a $0'$ oracle, but it is unclear on whether there is a computable bound on the number of times the oracle is queried, i.e. whether the construction is tt-reducible to $0'$ and therefore $\omega$-c.e. (c.f. Theorem 1.4). Cooper, Lee and Morphett re-cast this construction as a full-approximation priority argument, giving an improved bound of $\omega$-c.e.

The proof we give here is a slight modification of the Cooper, Lee and Morphett proof, in particular we use a stronger restraint to make the proof easier to adapt for results in later chapters.

**Theorem 2.8** (Cooper, Lee and Morphett 2011 [29]).

A computably well-founded computable partial order has a computably well-founded $\omega$-c.e. linear extension.

**Proof.** Let $\mathcal{P} = (\mathbb{N}, \leq_{\mathcal{P}})$ be a computably well-founded computable partial order. We
build a linearisation $L$ of $\mathcal{P}$ as the limit of a uniformly computable sequence of finite linear orderings $(L_s = (\mathbb{N}|_s, \leq_{L,s}))_{s \in \omega}$, such that the limit $\lim_{s \to \infty} \leq_{L,s} = \leq_L$ exists and is a linear extension of $\leq_P$.

By examining the construction, we shall show there is a computable bound on the number of changes to each element of $\leq_{L,s}$, that is the set of $s$ such that $\leq_{L,s}$ does not agree with $\leq_{L,s+1}$ on given $a, b \in \mathbb{N}$ is bounded by a computable function in $a$ and $b$ (and so in $(a, b)$). Hence, the limit $\leq_L = \lim_{s \to \infty} \leq_{L,s}$ is $\omega$-c.e.

Let $W_e$ be the $e$-th computably enumerable set, under the standard listing, and let $x_{e0}, x_{e1}, x_{e2}, \ldots$ be the elements of $W_e$ in the order they are enumerated. Clearly, if $(y_i)_{i \in \omega}$ is a computable sequence, then there is some $e$ such that $y_i = x_{ei}$ for all $i$.

To ensure that $L$ is computably well-founded, it suffices to ensure that no sequence $(x_i^e)_{i \in \omega}$ defines an $\omega^*$-sequence in $L$. As in the proof of Theorem 2.7 we achieve this for each infinite $W_e$ by finding some $x_{i}^e, x_{i+1}^e \in W_e$ such that $x_i^e <_P x_{i+1}^e$ or $x_i^e |_P x_{i+1}^e$ and $x_i^e >_L x_{i+1}^e$ so we can change the order in the linearisation. Then $W_e$ cannot define an $\omega^*$-sequence as it contains an ascending pair.

If $W_e$ is infinite then such $x_i^e, x_{i+1}^e$ must exist as otherwise $W_e$ would define an $\omega^*$-sequence in $\mathcal{P}$, contradicting the assumption that $\mathcal{P}$ is computably well-founded.

### 2.2.1 Requirements

The construction will satisfy the following requirements, for $e \in \omega$,

- $N$: $L$ is a linear extension of $\mathcal{P}$.
- $R_e$: $W_e$ does not define an $\omega^*$-sequence in $\leq_L$. 

To satisfy the requirements we place them in a finite injury construction, with the requirements in the priority ordering $N \succ R_0 \succ R_1 \succ R_2 \succ \cdots$.

### 2.2.2 Strategy

We ensure that $\mathcal{L}$ is a linear extension of $\mathcal{P}$ by defining at every stage $\mathcal{L}_s$ to be a linear extension of $\mathcal{P} \restriction_s$. Then to ensure that the limit $\lim_{s \to \infty} \mathcal{L}_s$ exists, we restrain the requirement $R_e$ from acting on $\leq_{L,s} \restriction_e$. So only finitely many requirements can modify any part of $\leq_{L,s}$, and since we will show each requirement can only act finitely often, the limit exists. This also gives a computable bound on the number of changes, and hence that the ordering $\leq_L$ is $\omega$-c.e.

For each requirement $R_e$, we set a restraint threshold $t_e[s]$, which is the $\mathbb{N}$-greatest number which $R_e$ needs preserved from future alteration in order to ensure it remains satisfied, at the beginning of stage $s$. We also define $T_e[s] = \{ n \in \mathbb{N} \mid n \leq \max \{ t_{e'}[s] \mid e' < e \} \}$, which is the portion of $\leq_{L,s}$ which $R_e$ is not permitted to alter, in order to avoid injuring higher priority requirements.

To satisfy $R_e$, we look for elements $x_i^e, x_{i+1}^e \geq_{\mathbb{N}} e$, such that either they are $\leq_P$-comparable and $x_i^e <_P x_{i+1}^e$ or they are $\leq_P$-incomparable and such that we can make them an ascending pair in $\mathcal{L}$ without affecting $\leq_{L,s+1}$ on $T_e[s]$, and therefore without injuring the restraint of any higher priority requirement, in particular such that there is no element of $T_e[s]$ between them. We will argue that if $W_e$ is infinite, then we will eventually find some suitable pair of elements.

In the first case, if we find a pair of elements which prevent $W_e$ from enumerating an infinite descending chain, then we say that $R_e$ has been discharged, and move on.
In the second case, we plummet the element $x_i^e$, called the **plummet witness**, $\leq_{L,s}$ below $x_{i+1}^e$ as follows. Suppose that $x_{i+1}^e = c_k \leq_{L,s-1} \cdots \leq_{L,s-1} c_1$ is a list in $L_{s-1}$ of the finite set

$$\{ c \mid c \in L_{s-1} \text{ and } x_j^e \leq_{L,s-1} c <_{L,s-1} x_i^e \text{ and } c \not\leq_P x_i^e \}$$

and notice that clearly $c_n \mid_P x_i^e$ for all $c_n$ (and therefore $c_n$ is also $\leq_P$-incomparable with any element $\leq_P$-comparable to $x_i^e$), since $\leq_{L,s-1}$ extends $\leq_P$ on the elements already enumerated.

Now, for any $c_1 \leq_{L,s-1} d \leq_{L,s-1} x_i^e$, we must have that $d \leq_P x_i^e$ but by definition $c_1 \not\leq_P x_i^e$ and therefore $c_1 \not\leq_P d$. Hence, $c_1$ can be moved past $d$. Denote by $x_i^e + 1$ the immediate successor of $x_i^e$ in $L_{s-1}$ if it exists and move $c_1$ until $x_i^e \leq_{L,s} c_1 \leq_{L,s} x_i^e + 1$. Then by the same argument, we can repeat the process for all $c_i$ to get $x_i^e \leq_{L,s} c_k \leq_{L,s} \cdots \leq_{L,s} c_1 \leq_{L,s} x_i^e + 1$. If the successor $x_i^e + 1$ does not exist then just move $c_1$ to immediately above $x_i^e$ and then proceed as if $c_1 = x_i^e + 1$.

Notice that since $x_j^e \leq_{L,s-1} x_i^e$ at stage $s - 1$ and $x_j^e \in \{ c_1, \ldots, c_k \}$, at the end of stage $s$ we have $x_i^e \leq_{L,s} x_j^e$, and hence $W_{e,s}$ is not a descending sequence. Now set $t_e[s+1] = \max\{ x_i^e, x_{i+1}^e \}$. After we have completed the plummeting, we say that $R_e$ is discharged.

### 2.2.3 Construction

At stage 0, set $t_e[0] = 0$ for all $e$.

At stage $2s + 1$ we enumerate the number $s$ into $L_s$, as high as possible whilst ensuring that $L_{2s+1}$ is a linear extension of $P |_s$, and so the transitive closure of $L_{2s+1} \cup P$ is a partial order, because of the “high as possible” constraint.
We say that \( R_e \) requires attention at stage \( s \) if the elements of \( W_{e,s} \) describe the start of an \( \omega^* \)-sequence in \( \leq_L \), and \( R_e \) can be satisfied at stage \( s \), i.e. there are two elements \( x^e_i, x^e_{i+1} \in N \) \( T_e[s] \) which are \( \leq_P \)-incomparable, greater than \( e \) and there is no \( n \leq T_e[s] \) between \( x^e_i \) and \( x^e_{i+1} \) in \( L[s] \).

At stage \( 2s + 2 \) we find the least \( e \) such that \( R_e \) requires attention (if no such \( e \) exists then end the stage and go on to stage \( 2s + 3 \)). Then take the elements \( x^e_i, x^e_{i+1} \) as described above and plummet the element \( x^e_i \leq_L 2s+2 \) below \( x^e_{i+1} \) by defining \( x^e_i \leq_L 2s+2 \) \( x^e_{i+1} \) as described above. Set \( t_e[2s + 1] = \max\{x^e_i, x^e_{i+1}\} \) and \( t_e'[2s + 2] = 0 \) for all \( e' \geq N e \). This ensures that \( W_e \) does not define an \( \omega^* \)-sequence, because it contains at least one ascending pair \( x^e_i, x^e_{i+1} \), but \( L_{2s+2} \) is still a linear extension of \( P \mid_s \).

### 2.2.4 Verification

**Lemma 2.9.**

\( L \) is a linear extension of \( P \).

*Proof.* At every stage \( L_s \) is defined to be a linear extension of \( P \mid_s \), and any finite initial segment of the linearisation is fixed after a finite number of stages, so the limit \( \lim_{s \to \infty} L_s = L \) exists, and is a linear extension of \( P \). \( \square \)

**Lemma 2.10.**

\( L \) is computably well-founded.

*Proof.* If \( L \) were not computably well-founded then there would be some computable subchain of \( L \) which had order type \( \omega^* \). And so there would be some computably enumerable set \( W_e \) which enumerated this subchain as an \( \omega^* \)-sequence, and so the requirement \( R_e \) would not be satisfied.
Let $s_0$ be a stage such that $T_e[s]$ is fixed for all $s \geq s_0$, which exists because the requirements with higher priority can only act a finite number of times. We can show this by induction on $e$. Obviously $R_0$ only acts at most once, so we have the base case, then assuming for induction that the requirements $R_{e'}$ for $e' < e$ act finitely often, we see that $R_e$ must act at most once more than the combined number of actions of all $R_{e'}$, which will also be finite. Hence, $s_0$ exists. We now argue that $R_e$ is satisfied after $s_0$.

We suppose that $W_e$ is infinite, because if $W_e$ is finite then $R_e$ is trivially satisfied. If the finite part of $W_e$ which has been enumerated by stage $s_0$ does not define a descending sequence, then $R_e$ will be satisfied, so suppose it does. Then if $R_e$ is never satisfied, the enumeration of $W_e$ after stage $s_0$ must go on to define a $\omega^*$-sequence, and never require attention, because if it did then $R_e$ would act to satisfy itself with highest priority, since all requirements of higher priority are satisfied by stage $s_0$. Therefore it must be the case that there is no pair $x^e_i, x^e_{i+1} \geq_{N} T_e[s_0]$ which is incomparable in $P$, greater than $e$ and such that there is no $n \leq_{N} T_e[s_0]$ between $x^e_i$ and $x^e_{i+1}$ in $L[s_0]$.

$T_e[s_0]$ is finite, and so then there will be an $\leq_L$-least element and since $W_e$ is infinite and descending there will be an infinite subset of $W_e$ which is $\leq_L$-below all elements of $T_e[s_0]$. This infinite subset then defines a computable $\omega^*$-sequence in $P$, which contradicts the fact that $P$ is computably well-founded.

Hence, all requirements $R_e$ are satisfied and $L$ is computably well-founded. \hfill \Box

**Lemma 2.11.**

$L$ is $\omega$-c.e.

**Proof.** If $R_e$ acts at stage $s$, it will remain satisfied and will not act again at a subsequent stages, unless it is injured by a requirement with higher priority. Hence, any requirement
will act at most $2^e$ times. Since $R_e$ cannot make changes to $\leq_{L,s}$ for numbers $\leq e$, $\leq_{B,s} \upharpoonright e$ can change at most $2^e - 1$ times, and so changes in $\leq_{L}$ are computably bounded, and $L$ is $\omega$-c.e.

This completes the proof of Theorem 2.8.

\section{Computable extendibility}

We now consider extendibility from a computable point of view, beginning by defining the appropriate notions of computable extendibility.

\textbf{Definition 2.12.}

A computable order type $\alpha$ is \textit{computably extendible} if any computable partial order $\mathcal{P}$ which does not computably embed $\alpha$ can be extended to a computable linear order $\mathcal{L}$ which does not computably embed $\alpha$.

A computable order type $\alpha$ is \textit{computably 2-c.e.-extendible (resp. $\Delta_2$-extendible, $\omega$-c.e.-extendible, $n$-c.e.-extendible)} if any computable partial order $\mathcal{P}$ which does not computably embed $\alpha$ can be extended to a 2-c.e. (resp. $\Delta_2$, $\omega$-c.e., $n$-c.e.) linear order $\mathcal{L}$ which does not computably embed $\alpha$.

Theorems 2.5, 2.7 and 2.8 above can now be re-stated more concisely using this terminology.

\textbf{Theorem 2.13.}

$\omega^*$ is not computably extendible. \hspace{1cm} [Rosenstein and Statman 1984 [43]]

$\omega^*$ is computably $\Delta_2$-extendible. \hspace{1cm} [Rosenstein and Statman 1984 [43]]

$\omega^*$ is computably $\omega$-c.e.-extendible. \hspace{1cm} [Cooper, Lee and Morphett 2011 [29]]
2.4 The lower bound

We now show that Cooper, Lee and Morphett’s result is in fact the best possible, and the bound cannot be lowered to $n$-c.e. for any finite $n$, because we can exhibit a counterexample, for any given $n$ of a computably well-founded computable partial order, which cannot be extended to a computably well-founded $n$-c.e. linear extension. We will initially prove this in the case of $n = 2$ (note that the case $n = 1$ is just Corollary 2.6) and then show how to generalise that case to any $n$.

**Theorem 2.14.**

There exists a computably well-founded computable partial order $P$ which has no computably well-founded $2$-c.e. linear extension. That is, $\omega^*$ is not computably 2-c.e.-extendible.

**Proof.** We construct as a witness a computable partial order $P = (\mathbb{N}, \leq_P)$ as the disjoint union of sub-partial orderings $P_e = (P_e, \leq_P|P_e)$ such that each $P_e$ forms a connected component of $P$, and in particular every element of $P_e$ is incomparable to every element of every other $P_f$ where $e \neq f$. We will assume a computable listing of 2-c.e. sets, $\{R_e\}_{e \in \mathbb{N}}$, where $R_{(i,j)} = W_i \setminus W_j$, and an associated computable approximation $R_{(i,j)}[s] = W_i[s] \setminus W_j[s]$. Then the purpose of $P_e$ is to show that if $R_e$ is a linearisation of $\leq_P$, then there exists a computable $\omega^*$-sequence in $R_e$. By construction, such a sequence will be made up of elements of $P_e$.

We will construct $P_e$ in such a way that any $\omega^*$-sequence in $P_e$ computes the halting set $K$, and is therefore not computable. Since there are no $\leq_P$-comparabilities between the components and each component is computably well-founded, we have that $P$ as a whole is computably well-founded.

At each stage $s$ of the construction we define an approximation $P[s]$ with domain
some finite initial segment of \( \mathbb{N} \), we also compute the approximations \( R_e[s] \) for \( e < s \) and an approximation \( K[s] \) of the halting set, to be used in the construction. At stage 0, \( P[0] = \emptyset \), then at stage \( s + 1 \) we add up to four new elements to each \( P_e[s] \) with \( e < s \) and four new elements to \( P_s[s] \). When a number \( n \) enters the construction at a stage \( s \), all of the \( \leq P \)-comparabilities of \( n \) with \( k < n \) are set at stage \( s \) and do not change at any later stage. So each \( P_e \) is computable because if we wish to compute \( \leq P \) relative to a pair of numbers \( (n, m) \) then it suffices to run the construction until stage \( s = \max\{n, m\} + 1 \) and the approximation to \( \leq P \) relative to \( (n, m) \) at stage \( s \) will be the true value, and hence \( \mathcal{P} \) is computable.

We now fix some \( e \in \mathbb{N} \) and consider the construction of \( P_e \). At all stages \( s \leq e \), \( P_e[s] = \emptyset \), then at stage \( s = e + 1 \), four new elements are added to define \( P_e[s + 1] \), with the comparabilities \( b_{-1} <_P y <_P a_{-1} \) and \( b_{-1} <_P x <_P a_{-1} \) as shown in Figure 2.2. We define \((b_{-1}, a_{-1})\) to be the Level 1 active interval and also to be the Level 2 active interval.

![Figure 2.2: The first four elements of \( P_e \), added at stage \( e + 1 \)](image)

The labels \( y \) and \( x \) are temporary and will be reused many times during the construction for different elements before they are given a permanent label.
Definition 2.15.

We call a stage $s > e + 1$ a good stage for $e$ if $R_e[s]$ linearises $P_e[s]$. The construction of $P_e$ will only proceed at good stages for $e$. Note that the set of good stages for $e$ is computable.

Definition 2.16.

If $s$ is a good stage for $e$, then for $m, n \in P_e[s]$ we say that $m R_e n$ is computed at stage $s$ if $\langle m, n \rangle \in R_e[s]$. Since $s$ is good for $e$ and $R_e[s]$ is therefore a linear order, this obviously means that $\langle n, m \rangle \notin R_e[s]$.

Definition 2.17.

- We say that $m R_e n$ is 1-computed at stage $s$ if it is computed at stage $s$ and there is no earlier good stage at which $n R_e m$ is computed.
- We say that $m R_e n$ is 2-computed at stage $s$ if it is computed at stage $s$ and there is an earlier good stage at which $n R_e m$ is 1-computed.

Note that for a pair $m, n \in P_e$, $n R_e m$ can only be computed at a good stage for $e$, so from now on when we say that $m R_e n$ is computed (or 1-computed, 2-computed, etc) at stage $s$ we mean by definition that $s$ is a good stage for $e$.

For some $R_e$ it may be the case that $m R_e n$ is 1-computed at stage $s$ and there has been an earlier stage $t < s$ such that $(n, m) \in R_e[t]$, and it may even be the case that $(m, n) \in R_e[t]$ as well. However, this just means that by definition $t$ is not a good stage for $e$, and hence neither $m R_e n$ or $n R_e m$ was computed at stage $t$.

We now consider the simple case in which there is no stage $s$ and pair $k, l \in P_e$ such that $k R_e l$ is 2-computed at stage $s$. In effect we are initially proving the case for $n = 1$, which is a reproof of Corollary 2.6. We will then consider the case where there
are 2-computed pairs, and show how the construction nests.

Now, suppose that \( s \) is the first good stage for \( e \), and without loss of generality suppose that \( y \mathrel{R_e} x \) is 1-computed at stage \( s \). Then relabel \( x \) as \( b_0 \) and \( y \) as \( c_0 \), and we add four new elements \( a_0, d_0, y, x \) into \( P_e[s + 1] \) such that \( b_0 <_P y <_P a_0 <_P a_{-1}, b_0 <_P x <_P a_0 <_P a_{-1} \) and \( b_{-1} <_P d_0 <_P c_0 \), as shown in Figure 2.3, and we define the Level 1 active interval to be \( (b_0, a_0) \).

![Figure 2.3: After the first good stage for \( e \)](image)

At subsequent stages \( s \), the construction will continue in the interval \( (b_0, a_0) \) for as long as \( 0 \notin K[s] \). However, if at stage \( s \), \( 0 \in K[s] \) then \( (d_0, c_0) \) will become the Level 1 active interval and the construction will restart there.

If we assume that \( 0 \notin K[s] \) for the time being, then at the next good stage \( s' \), if without loss of generality \( m_0 \mathrel{R_e} n_0 \) is 1-computed at stage \( s' \) then relabel \( n_0 \) as \( b_1 \) and \( m_0 \) as \( c_1 \), and we add four new elements \( a_1, d_1, y, x \) into \( P_e[s' + 1] \) such that \( b_1 <_P y <_P a_1 <_P a_0, b_1 <_P x <_P a_1 <_P a_0 \) and \( b_0 <_P d_1 <_P c_1 \), as shown in Figure 2.4, and we define the Level 1 active interval to be \( (b_1, a_1) \).

We will then continue the construction in the same way at the subsequent good
stages for \( e \) in the interval \((b_1, a_1)\), unless 1 is enumerated into \( K \), at which time the construction will move to the interval \((d_1, c_1)\), and we will define that as the new Level 1 active interval, or if 0 is enumerated into \( K \) in which case the construction moves to \((d_0, c_0)\) as mentioned above.

![Figure 2.4: After the second good stage for \( e \)](image)

**Definition 2.18.**

When the elements \( a_i \) are added we say they are *active*, and we say that \( a_0 \) is 0-coloured, \( a_1 \) is 1-coloured etc., referring to the number they are testing for membership in \( K \). If \( i \) is enumerated into \( K \), then the construction interval will move and all of the \( j \)-coloured elements, for \( j > i \) will be deactivated, and new \( j \)-coloured elements will be defined as the construction continues.

During the construction there may be up to \( 2^i \) indices \( n \) at each Level of the construction such that \( a_n \) is \( i \)-coloured, but at a given stage \( s \) there is at most one \( n \) at each Level such that \( a_n \) is \( i \)-coloured and active, for each number \( i \).

If \( i \) is enumerated into \( K[s] \), and there is some \( a_n \in P_e[s] \) which is \( i \)-coloured and active, then the Level 1 active interval will change, as mentioned above, and any \( a_m \) which are \( j \)-coloured, for \( j > i \), will no longer be in the Level 1 active interval and are said to
become inactive.

So for example if 0 is enumerated into $K$ then any $a_n$, $n \geq 1$ which have been defined will all be deactivated, the construction moves to the interval $(d_0, c_0)$ and restarts with a new $a_{n+1}$ which is active and 1-coloured.

When the Level 1 active interval is redefined due to a number being enumerated into $K[s]$, we add two new elements to the new Level 1 active interval $(d_{i+1}, c_{i+1})$ and $<_P$-incomparable to each other, and call them $y$ and $x$.¹

Then at the next good stage $t$ for $e$, if without loss of generality $yRe x$ is 1-computed at stage $t$ then relabel $x$ as $b_{n+1}$ and $y$ as $c_{n+1}$, where $n + 1$ is the least index for which $a_{n+1}, b_{n+1}, c_{n+1}$ have not been defined, and we add four new elements $a_{n+1}, d_{n+1}, y, x$ into $\mathcal{P}_e[t + 1]$ such that $b_{n+1} <_P y <_P a_{n+1} <_P c_{i+1}$ and $b_{n+1} <_P x <_P a_{n+1} <_P c_{i+1}$ and $d_{i+1} <_P d_{n+1} <_P c_{n+1}$, as shown in Figure 2.5, and we define the Level 1 active interval to be $(b_{n+1}, a_{n+1})$. Note that $a_{n+1}$ is active with colour $j + 1$, where $j$ is the number which was enumerated into $K$ and caused the Level 1 active interval to be redefined.

![Figure 2.5: After a general good stage for $e$](image)

¹If there are already numbers in the construction labelled $y$ and $x$ then we can ignore them as they are no longer relevant to the construction.
The construction in the case where no pair is 2-computed is then a simple nesting of the above process. It should be noted that the sequence of $a_n$ defined will be such that $a_{n+1} \text{Re} a_n$ for all $n$, i.e. they form a descending sequence in $\text{Re}$, and that if $a_n$ is active and has colour $k$, then the Level 1 active interval will be within (or all of) either $(b_n, a_n)$ or $(d_n, c_n)$.

In particular, if the Level 1 active interval lies within $(b_n, a_n)$ and $k$ is seen to be in $K$, then the Level 1 active interval will be redefined to be $(d_n, c_n)$, and any $a_p$ which have been defined for $p > n$ will be deactivated. Although given the nested nature of the construction, some of those may have already been deactivated at an earlier stage.

If there are only finitely many good stages, then $\mathcal{P}_e$ will be finite, and the necessary conditions for the theorem trivially hold, as it contains no $\omega^*$-sequence, and $\text{Re}$ is not a linearisation of $\mathcal{P}_e$. Indeed if $\text{Re}$ is a linearisation of $\mathcal{P}_e$ then there will be infinitely many good stages for $e$, at each of which either two or four elements were added to $\mathcal{P}_e$, and hence $\mathcal{P}_e$ would be infinite.

If $\mathcal{P}_e$ is infinite, still under the supposition that there is no pair which is ever 2-computed, then the $a_n$ form an infinite descending sequence in $\text{Re}$, and for each $i \in \mathbb{N}$ there is exactly one $a_n$ which is $i$-coloured and active at all good stages for $e$ after some finite stage $s_i$. We call this the test witness for $i$.

Suppose then that there is some $\omega^*$ sequence $\{x_n\}_{n \in \mathbb{N}}$ in $\mathcal{P}_e$. If there is any $k$ such that $b_0 <_P x_k$ then $0 \notin K$ and if there is any $k$ such that $x_k <_P c_0$ then $0 \in K$. Only one of these can be true, since $b_0 <_P y$ and $z <_P c_0$ implies that $y|_P z$, but all $x_n$ must be $<_P$-comparable, and since elements are only put in the intervals $(b_0, a_0)$ and $(d_0, c_0)$. Therefore, if we know the location of the sequence (or in fact any element of the sequence), relative to $b_0$ and $c_0$, we know whether $0 \in K$ or $0 \notin K$. 

Then by simulating the construction above, we can find the test witness for 1, and depending on where we can find an element of \( \{x_n\}_{n \in \mathbb{N}} \), compute whether \( 1 \in K \). And so on for any number \( i \) we can use the position of the \( x_n \) to compute whether \( i \in K \) and hence there is an algorithm defined uniformly in \( e \) which computes \( K \) with oracle \( \{x_n\}_{n \in \mathbb{N}} \). So any \( \omega^* \) sequence \( \{x_n\}_{n \in \mathbb{N}} \) in \( P_e \) computes \( K \) and hence \( P_e \) is computably well-founded.

If \( R_e \) linearises \( P_e \), then we can find a computable \( \omega^* \)-sequence in \( R_e \). In fact by construction, \( \{a_n\}_{n \in \mathbb{N}} \) is a computable descending sequence in \( R_e \), which will be infinite as we know that \( R_e \) linearising \( P_e \) means that \( P_e \) will be infinite, and therefore \( R_e \) is not computably well-founded. This concludes the proof of the simpler case in which there is no pair which is ever 2-computed.

Now, suppose that in fact there is at least one pair \( n, m \in P_e \) such that \( nR_e m \) is 2-computed at some stage \( s \). Note that without loss of generality we can in fact assume that the 2-computed pair is \( c_n, b_n \) for some \( n \), because if any pair is 2-computed then there will be some \( c_n, b_n \) that is 2-computed, by inspection of the construction. Thus during the construction below we only need to search for the least index \( k \) such that \( b_k R_e c_k \) is 2-computed at stage \( s + 1 \).

![Figure 2.6: After the first stage at which a pair is 2-computed](image-url)
So, suppose we are at the first stage at which some \( c_n, b_n \) is 2-computed, and suppose that \( n \) is the least index of a 2-computed pair. Then we relabel \( c_n \) as \( b'_0, b_n \) as \( c'_0 \) and add four new numbers \( a'_0, d'_0, y', x' \) to \( P_e[s + 1] \) such that \( b'_0 <_P y' <_P a'_0 <_P a_{-1}, b'_0 <_P x' <_P a'_0 <_P a_{-1} \) and \( b_{-1} <_P d'_0 <_P c'_0 \), as shown in Figure 2.6. Then we define \((b'_0, a'_0)\) to be the Level 1 active interval and also the Level 2 active interval.

It is important to note here that \( a'_0 \) is active and 0-coloured, and all previous active elements have been deactivated at this stage. So if at some stage \( s' > s \) we have that \( 0 \in K[s'] \), then both the Level 1 active interval and the Level 2 active interval will move to \((d'_0, c'_0)\).

The Level 1 construction then restarts from scratch in the new Level 1 interval, building a new \( a_n \) sequence until such a time as there is another stage \( t \) at which some pair is 2-computed. Assuming that \( m \) is the least index of such a 2-computed pair \( b_m, c_m \), we redefine \( b_m \) as \( c'_1 \) and \( c_m \) as \( b'_1 \) and add \( a'_1, d'_1, x', y' \) such that \( b'_1 <_P y' <_P a'_1 <_P a'_0, b'_1 <_P x' <_P a'_1 <_P a'_0 \) and \( b'_0 <_P d'_1 <_P c'_1 \) as shown in Figure 2.7. (Assuming that \( 0 \notin K[s - 1] \), if \( 0 \in K[s - 1] \) then replace \( a'_0, b'_0 \) by \( d'_0, c'_0 \) respectively.)

The Level 2 active interval is now redefined to be \((b'_1, a'_1)\), and \( a'_1 \) is 1-coloured and active.
It should be noted here that $a'_1$ is directly below $a'_0$, and not below any other elements which may have been added at stages between $s$ and $t$. Those elements are now irrelevant, and incomparable to the rest of the construction. Any coloured elements apart from $a'_0$ and $a'_1$ are deactivated.

Then the Level 1 construction restarts as before in the new Level 1 active interval $(b'_1, a'_1)$ (or $(d'_1, c'_1)$ if $1 \in K[t + 1]$). We see that if there are only finitely many stages at which the Level 2 active interval is redefined, then from some stage $s$ all activity in the construction will be at Level 1, and we get the same outcomes as in the first case. Namely, if $R_e$ is a linearisation of $P_e$, then it contains a computable $\omega^*$-sequence $\{a_n\}_{n \in \mathbb{N}}$ in the eventually permanently active Level 2 interval. And any $\omega^*$-sequence in $P_e$ will compute $K$, by performing the same analysis as earlier but within the eventually permanently active Level 2 interval, since we know that the construction outside of that interval will be finite and hence there will be an infinite part of any $\omega^*$-sequence within the interval. So as before we compute $K$ by performing tests relative to the relevant $(b_n, c_n)$ pairs in the interval to determine if $k \in K$ for any $k \in \mathbb{N}$.

Alternatively the Level 2 active interval may be redefined infinitely many times, or
in other words there may be infinitely many stages at which there is a 2-computed pair. In this case, if \( R_e \) is a linearisation of \( P_e \), then the sequence \( \{a'_n\}_{n \in \mathbb{N}} \) is a computable \( \omega^* \)-sequence in \( R_e \) and we still have that any \( \omega^* \)-sequence in \( P_e \) is not computable, because we can again use the location of (any element of) the sequence relative to the \( b'_i, c'_i \) pairs to compute \( K \).

To show that \( \{a'_n\}_{n \in \mathbb{N}} \) is a computable sequence, notice that the pairs \( (b'_n, c'_n) \) which are 2-computed, can be found by a uniformly computable search, and hence there is an algorithm which computes the sequence of such pairs in order, and therefore also computes \( a'_0, a'_1, a'_2, \ldots \).

To show that any \( \omega^* \)-sequence in \( P_e \) computes \( K \), let \( \{x_n\}_{n \in \mathbb{N}} \) be such a sequence, then in the same way as we computed \( K \) in the earlier case, we can compute \( K \) in this case by using the \( b'_n \) and \( c'_n \), which are computable as discussed above, and simulating the construction to perform tests in the same manner as before but at the higher level.

Since \( e \) was arbitrary, we see that \( P \) has no computably well-founded 2-c.e. linear extension, and any infinite descending sequence in \( P \) computes \( K \) and hence \( P \) is computably well-founded. But we have also seen that the construction of \( P \) is entirely computable. This concludes the proof of Theorem 2.14.

This construction is significantly more complex than the Kleene Tree counterexample of Theorem 2.5. The Kleene Tree is a nice natural example, but cannot be used in this case because it is possible to construct a 2-c.e. linearisation of the Kleene Tree which has no computable infinite descending sequence. This can be done using the method of diagonalising against the c.e. sets and plummeting elements, similarly to the proof of Theorem 2.8.
We now show how to generalise this proof to give a construction of computably well-founded computable partial order which has no computably well-founded \( n \)-c.e. linear extension, for any given \( n \in \mathbb{N} \).

**Corollary 2.19.**

For every \( n \geq 1 \), there exists a computably well-founded computable partial order \( P \) which has no computably well-founded \( n \)-c.e. linear extension. That is, \( \omega^* \) is not computably \( n \)-c.e.-extendible, for any \( n \in \mathbb{N} \).

**Proof.** A computably enumerable 1-c.e. linearisation of \( P \) will be computable. We proved the \( n = 1 \) case as the first part of the proof for \( n = 2 \), and of course the Kleene Tree constructed by Rosenstein and Statman as a witness to Theorem 2.5 also proves this case. We now show how to generalise the proof of the \( n = 2 \) case above to any \( n \in \mathbb{N} \).

Consider the \( n = 3 \) case. We perform the construction as with \( n = 2 \), but with an extra layer. The construction can now define Level 3 active intervals. When a Level 3 active interval is defined or redefined all Level 2 and Level 1 construction is restarted from scratch in the new Level 3 interval. At Level 3 we will be building a computable sequence \( a''_n \), descending in \( R_e \), when pairs \( b''_n, c''_n \) are 3-computed.

We will assume a computable listing of 3-c.e. sets, \( \{ R_e \}_{e \in \mathbb{N}} \), where \( R_{(i,j,k)} = (W_i \setminus W_j) \cup W_k \), and an associated computable approximation \( R_{(i,j,k)}[s] = (W_i[s] \setminus W_j[s]) \cup W_k[s] \).

We define 3-computation in the obvious way. We say that \( m R_e n \) is 3-computed at stage \( s \) if it is computed at stage \( s \) and there is an earlier good stage at which \( n R_e m \) is 2-computed. And in general, if \( R_e \) is a \( k \)-c.e. set, then we say that \( m R_e n \) is \( r \)-computed at stage \( s \), for some \( r \leq k \), if it is computed at stage \( s \) and \( r = 1 \) or \( r > 1 \) and there is an earlier good stage at which \( n R_e m \) is \( r - 1 \)-computed and there is no good stage \( \leq s \) at which \( n R_e m \) is \( r + 1 \)-computed.
The construction in the case where there is 3-computation works in the same way relative to 2-computation as 2-computation works relative to 1-computation. For example when the first pair is 3-computed this is because a 2-computed pair \((b'_n, c'_n)\) has been found such that \(b'_n, c'_n\) is 3-computed at this stage \(s\) (and \(n\) is the least index of such a pair). So we enumerate new elements \(a''_0, d''_0, x, y\) and redefine \(b''_0 = c'_n\) and \(c''_0 = b'_n\), as shown in Figure 2.7.

![Figure 2.7: A diagram showing the construction in the case of 3-computation](image)

Then the construction at Level 2 and Level 1 restarts from scratch in the new Level 3, 2 and 1 active interval \((b''_0, a''_0)\). It behaves as a nested version of the construction in the Theorem above, but with the difference that the Level 3 active interval can be redefined either finitely or infinitely often.

We take the greatest \(t \in \{1, 2, 3\}\) such that the Level \(t\) active interval is redefined infinitely often. If the Level 3 active interval is redefined infinitely often then if \(R_e\) linearises \(P_e\) then the \(a''_n\) form a computable infinite descending chain in \(R_e\), and if \(\{x_n\}_{n \in \mathbb{N}}\) is an \(\omega^*\)-sequence in \(P_e\) then it computes \(K\). If it is redefined only finitely often, then after some finite stage it will be fixed, and the remainder of the construction will be at Level 2 and Level 1. So the outcomes are the same as in the \(n = 2\) case.
The case for \( n \)-c.e. is then a straightforward generalisation of this; take a computable listing of \( n \)-c.e. sets \( R_e \) and associated approximations \( R_e[s] \), and define \( n \)-computation as above. Work through the construction as in the \( n-1 \) case, but when something is \( n \)-computed we define the points \( a_i^{(n-1)}, b_i^{(n-1)}, c_i^{(n-1)}, d_i^{(n-1)} \) and redefine the Level \( n \) active interval. If \( R_e \) linearises \( \mathcal{P} \) then will then be some \( 1 \leq t \leq n \) such that the Level \( t \) active interval is redefined infinitely often. Then is \( t \) is the greatest of those, if \( R_e \) linearises \( \mathcal{P}_e \) then the \( \{a_n^{(t-1)}\}_{n \in \mathbb{N}} \) will be a computable \( \omega^* \)-sequence in \( R_e \).

However, any \( \omega^* \)-sequence in \( \mathcal{P}_e \) will compute \( K \) in the same way as discussed above, since the construction at the \( t \) level is computable and so we can use the position of the sequence relative to the \( \{a_n^{(t-1)}\}_{n \in \mathbb{N}} \) to determine membership of elements of \( K \).

Then for any \( n \) we can construct a witness \( \mathcal{P} \) which is computable and computably well-founded, but has no computably well-founded \( n \)-c.e. linearisation.

It is interesting to note the reasons why this construction fails for \( \omega \)-c.e., which is because if there is no fixed bound on the number of times a pair can be changed in the linearisation, then there will not necessarily be a greatest \( t \) such that the Level \( t \) active interval is redefined infinitely often. In fact we could have no such \( t \) and just keep redefining intervals of greater and greater Level, but only finitely often. Thus we cannot build the computable \( \omega^* \)-sequence necessary in \( R_e \) and the argument fails.

Whilst in the proof of this theorem, we construct a separate counterexample for each \( n \geq 1 \), it is in fact possible to build one witness which proves the theorem for all \( n \geq 1 \). To do this, we need a listing of all \( n \)-c.e. sets, for all \( n \geq 1 \), which we define \( \{R_{(n,e)}\}_{n,e \in \mathbb{N}} \) with uniform approximations \( \{R_{(n,e)}[s]\}_{n,e,s \in \mathbb{N}} \).
Let \( g : \mathbb{N}^3 \to \mathbb{N} \) be a computable function such that

\[
W_{g(n,e,k)} = \begin{cases} 
\bigcap_{0 \leq i \leq k} W_{e_i} & \text{if } 0 \leq k \leq n \ (\text{where } e = \langle e_0, e_1, \ldots, e_{n-1} \rangle), \\
\emptyset & \text{if } k > n.
\end{cases}
\]

Note that \( n \) determines the length of the tuple coded as \( e \), and if \( n = 0 \) then we use the convention that \( e = \langle e \rangle \), so \( e_0 = e \).

Then

\[
R\langle n,e \rangle = \bigcup_{i=0}^{\left\lfloor \frac{n}{2} \right\rfloor} W_{g(n,e,2i)} \setminus W_{g(n,e,2i+1)},
\]

and \( R\langle n,e \rangle[s] \) is defined using the stage \( s \) approximations to all the of c.e. sets involved.

The construction then proceeds in the usual way, with \( P_{\langle e,n \rangle} \) being the component of the witness which shows that if \( R_{\langle e,n \rangle} \) is a linearisation of \( P \) then there exists a computable \( \omega^* \)-sequence in \( R_{\langle e,n \rangle} \). Notice that because \( n \) is encoded in the index of \( P_{\langle e,n \rangle} \), we can computably determine which algorithms from the above proofs to use in the construction, and hence the construction remains computable. In effect we simultaneously perform the constructions for the proof for each \( n \geq 1 \).
Chapter 3

The order type $\zeta$

The order type of the integers $\zeta = \omega^* + \omega$ is another key order type, which we will investigate the extendibility of in this chapter. Unlike the order types considered in Chapters 2 and 4, there have not been any bounds previously established on the complexity of linearisations which do not computably embed $\zeta$ of computable partial orders which do not computably embed $\zeta$. So in this chapter we will establish such bounds from scratch.

3.1 Background

We do have the starting point that $\zeta$ is classically extendible, by a theorem of Jullien [21] and independently of Galvin and McKenzie (unpublished).

Theorem 3.1 (Jullien 1969 [21]).
The order type $\zeta$ is extendible.

Proof. Let $\mathcal{P}$ be a partial order that does not embed $\zeta$. Define $I = \{ x \in P \mid \omega^* \text{ does not embed in } \mathcal{P}(x) \}$ and $F = \{ x \in P \mid \omega \text{ does not embed in } \mathcal{P}^*(x) \}$, where
\[ \mathcal{P}(x) = \{ y \in P \mid y \leq_P x \} \text{ and } \mathcal{P}^*(x) = \{ y \in P \mid y \geq_P x \}. \] Then clearly \( I \cup F = P \).

Obviously \( \omega^* \) does not embed in \( I \) and \( \omega \) does not embed in \( F \), and so there are linear extensions \( \mathcal{L}(I) \) and \( \mathcal{L}(F) \) such that \( \omega^* \) does not embed in \( \mathcal{L}(I) \) and \( \omega \) does not embed in \( \mathcal{L}(F) \), since \( \omega^* \) is extendible by Theorem 2.2, and the dual of an extendible type obviously also being extendible. Then the lexicographic sum \( \mathcal{L}(I) + \mathcal{L}(F)|_{P \setminus I} \) is a linear extension of \( \mathcal{P} \) which does not embed \( \omega^* + \omega \).

The effective content of this theorem has not been discussed in the literature. Given the other results in this Chapter, whilst parallels can in some cases be drawn from the well-founded case, we see that this is not necessarily always so. Hence we do not conjecture about the result for computable orderings, but note that it is an open question.

**Open Question 3.2.**

Does every computable partial order which does not embed \( \zeta \) have a computable linearisation which does not embed \( \zeta \)?

In Theorem 2.5, Rosenstein and Statman used a Kleene Tree as a witness of a computably well-founded computable partial order with no computably well-founded computably linearisation. Or in other words, that \( \omega^* \) is not computably extendible. We can use a similar witness here as a counterexample to the computable extendibility of \( \zeta \).

**Theorem 3.3.**

The order type \( \zeta \) is not computably extendible.

**Proof.** We build a partial order \( \mathcal{P} \) as a copy of the Kleene Tree \( KT \leq_P \)-below the dual of the Kleene Tree \( KT^* \) to witness the theorem. Then if this partial order contains a computable suborder of the type \( \omega^* + \omega \), we must have that \( KT \) contains a computable \( \omega^* \)-sequence and \( KT^* \) contains a computable \( \omega \)-sequence. But by construction that cannot happen, so we know that \( \mathcal{P} \) cannot computably embed \( \omega^* + \omega \).
If $L$ is a computable linearisation of $P$ then it consists of the order sum of a computable linearisation of $KT$ and a computable linearisation of $KT^*$. We know that any computable linearisation of $KT$ must contain a computable $\omega^*$-sequence, and similarly any computable linearisation of $KT^*$ must contain a computable $\omega$-sequence, so $L$ contains a computable $\omega^* + \omega$-sequence as required.

As before, this result immediately gives the same theorem for computably enumerable linearisations, because a computably enumerable linear order with a computable domain is computable. Thus this corollary follows directly from Theorem 3.3.

**Corollary 3.4.**
The order type $\zeta$ is not computably c.e.-extendible.

### 3.2 The lower bound

We can improve the lower bound from c.e. (Corollary 3.4) to $n$-c.e. for any $n \geq 1$, by combining the structure of the proof of Theorem 2.14 with the idea of the counterexample in the proof of Theorem 3.3.

**Theorem 3.5.**
The order type $\zeta$ is not $n$-c.e. extendible for any $n \geq 1$.

**Proof.** In the same way as before, we will demonstrate that $\zeta$ is not computably 2-c.e. extendible, and then generalise to all $n$.

For the $n = 2$ case, we construct as a witness a computable partial order $P = (\mathbb{N}, \leq_P)$ as the disjoint union of sub-partial orderings $P_e$ such that each $P_e$ forms a connected component of $P$, and in particular every element of $P_e$ is incomparable to every
element of every other $P_f$ where $e \neq f$. We will assume a computable listing of 2-c.e. sets, $\{R_e\}_{e \in \mathbb{N}}$, where $R_{(i,j)} = W_i \setminus W_j$, and an associated computable approximation $R_{(i,j)}[s] = W_i[s] \setminus W_j[s]$. Then the purpose of $P_e$ is to show that if $R_e$ is a linearisation of $\leq P$, then there exists a computable $\zeta$-sequence in $R_e$. By construction, such a sequence will be made up of elements of $P_e$.

We will construct $P_e$ in such a way that any $\zeta$-sequence in $P_e$ computes the halting set $K$, and is therefore not computable. Since there are no $\leq_P$-comparabilities between the components and each component has no computable $\zeta$-sequence, we have that $P$ as a whole has no computable $\zeta$-sequence.

At each stage $s$ of the construction we define an approximation $P[s]$ with domain some finite initial segment of $\mathbb{N}$, we also compute the approximations $R_e[s]$ for $e < s$ and an approximation $K[s]$ of the halting set, to be used in the construction. At stage 0, $P[0] = \emptyset$, then at stage $s + 1$ we add up to six new elements to each $P_e[s]$ with $e < s$ and eight new elements to $P_s[s]$.

When a number $n$ enters the construction at a stage $s$, all of the $\leq_P$-comparabilities of $n$ with $k < n$ are set at stage $s$ and do not change at any later stage. So each $P_e$ is computable because if we wish to compute $\leq_P$ relative to a pair of numbers $(n, m)$ then it suffices to run the construction until stage $s = \max\{n, m\} + 1$ and the approximation to $\leq_P$ relative to $(n, m)$ at stage $s$ will be the true value, and hence $P$ is computable.

We now fix some $e \in \mathbb{N}$ and consider the construction of $P_e$. At all stages $s \leq e$, $P_e[s] = \emptyset$, then at stage $s = e + 1$, eight new elements are added to define $P_e[s + 1]$. $P_e$ in this construction will consist of a copy of the construction from Theorem 2.14 and a copy of the dual of that construction. We call these copies $D_e$ and $U_e$ respectively, and everything in $U_e$ is $\leq_P$-greater than everything in $D_e$. 
\( \mathcal{D}_e \) has four elements with the comparabilities \( b_{-1}^{D_1} <_P y^D <_P a_{-1}^D \) and \( b_{-1}^{D_1} <_P x^D <_P a_{-1}^D \), and \( \mathcal{U}_e \) has four elements with the comparabilities \( a_{-1}^{U_1} <_P y^U <_P b_{-1}^U \) and \( a_{-1}^{U_1} <_P x^U <_P b_{-1}^U \) as shown in Figure 3.1.

![Figure 3.1: The first eight elements of \( \mathcal{P}_e \) added at stage \( e + 1 \)](image)

We then proceed to simultaneously construct both parts of the partial order as in the proof of Theorem 2.14 and dually as appropriate. We then get the outcomes as follows. If there are only finitely many good stages, then \( \mathcal{P}_e \) will be finite, and the necessary conditions for the theorem trivially hold, as it contains no \( \zeta \)-sequence, and \( \mathcal{R}_e \) is not a linearisation of \( \mathcal{P}_e \). Indeed if \( \mathcal{R}_e \) is a linearisation of \( \mathcal{P}_e \) then there will be infinitely many good stages for \( e \), at each of which either two or four elements were added to \( \mathcal{P}_e \), and hence \( \mathcal{P}_e \) would be infinite.

If \( \mathcal{P}_e \) is infinite, and \( \mathcal{R}_e \) linearises \( \mathcal{P} \), then we will have by construction a computable \( \zeta \)-sequence in \( \mathcal{R}_e \), just by combining the sequences that we construct in each part of \( \mathcal{P}_e \). The construction builds a computable \( \omega^* \)-sequence \( \sigma \) in \( \mathcal{L}(\mathcal{D}_e) \) and a computable \( \omega \)-sequence \( \tau \) in \( \mathcal{L}(\mathcal{U}_e) \). Thus \( \sigma + \tau \) is a computable \( \zeta \)-sequence in \( \mathcal{L}(\mathcal{P}_e) \) and so in \( \mathcal{L}(\mathcal{P}) \). Of course the sequences \( \sigma \) and \( \tau \) may be extracted at different levels - i.e. \( \sigma \) may be computed at Level 1 and \( \tau \) computed at Level 2, or visa versa. But this does not matter
for the proof.

In $\mathcal{U}_e$ we can show that that below any given any element there are only finitely many elements in $\mathcal{U}_e$ itself, and similarly above any element in $\mathcal{D}_e$. So any $\zeta$-sequence in $\mathcal{P}_e$ must lead to an $\omega^*$-sequence in $\mathcal{D}_e$ and an $\omega$-sequence in $\mathcal{U}_e$, each of which computes $K$ by construction.

Then as before, we can see that since $e$ was arbitrary, we can see every 2-c.e. linear extension of $\mathcal{P}$ computably embeds $\zeta$, and any suborder of $\mathcal{P}$ of order type $\zeta$ must compute $K$. And we have also seen that the construction of $\mathcal{P}$ is entirely computable. This concludes the proof of the $n = 2$ case.

The generalisation to $n$-c.e. then occurs in exactly the same way as before, take a computable listing of $n$-c.e. sets $R_e$ and associated approximations $R_e[s]$, and define $n$-computation as above. Work through the construction in each part as in the $n - 1$ case, but when something is $n$-computed we define the points $a_i^{D(n-1)}$, $b_i^{D(n-1)}$, $c_i^{D(n-1)}$, $d_i^{D(n-1)}$ or $a_i^{U(n-1)}$, $b_i^{U(n-1)}$, $c_i^{U(n-1)}$, $d_i^{U(n-1)}$ and redefine the Level $n$ active interval as appropriate.

If $R_e$ linearises $\mathcal{P}$ then will then be some $1 \leq t \leq n$ such that the Level $t$ active interval is redefined infinitely often. Then is $t$ is the greatest of those, if $R_e$ linearises $\mathcal{P}_e$ then the $\{a_n^{(t-1)}\}_{n \in \mathbb{N}}$ will be a computable $\zeta$-sequence in $R_e$.

However, any $\zeta$-sequence in $\mathcal{P}_e$ will compute $K$ in the same way as discussed above, since the construction at the $t$ level is computable and so we can use the position of the sequence relative to the $\{a_n^t\}_{n \in \mathbb{N}}$ to determine membership of elements of $K$.

Then for any $n$ we can construct a witness $\mathcal{P}$ which is computable and does not computably embed $\zeta$, but has no $n$-c.e. linearisation which does not computably embed $\zeta$. □
This construction fails for $\omega$-c.e. for the same reasons as before. If there is no fixed bound on the number of times a pair can be changed in the linearisation, then there will not necessarily be a greatest $t$ such that the Level $t$ active interval is redefined infinitely often. In fact we could have no such $t$ and just keep redefining intervals of greater and greater Level, but only finitely often. Thus we cannot build the computable $\zeta$-sequence necessary in $R_e$ and the argument fails.

Again, in the proof of this theorem, we construct a separate counterexample for each $n \geq 1$ for reasons of clarity. But it is in fact possible to build one witness which proves the theorem for all $n \geq 1$. To do this, as before, we need a listing of all $n$-c.e. sets, for all $n \geq 1$, which we define $\{R_{(n,e)}\}_{n,e \in \mathbb{N}}$ with uniform approximations $\{R_{(n,e)}[s]\}_{n,e,s \in \mathbb{N}}$.

Let $g : \mathbb{N}^3 \rightarrow \mathbb{N}$ be a computable function such that

$$W_{g(n,e,k)} = \begin{cases} \bigcap_{0 \leq i \leq k} W_{e_i} & \text{if } 0 \leq k \leq n \text{ (where } e = \langle e_0, e_1, \ldots, e_{n-1} \rangle), \\ \emptyset & \text{if } k > n. \end{cases}$$

Note that $n$ determines the length of the tuple coded as $e$, and if $n = 0$ then we use the convention that $e = \langle e \rangle$, so $e_0 = e$.

Then

$$R_{(n,e)} = \lfloor \frac{n}{2} \rfloor \bigcup_{i=0}^{\lfloor \frac{n}{2} \rfloor} W_{g(n,e,2i)} \setminus W_{g(n,e,2i+1)};$$

and $R_{(n,e)}[s]$ is defined using the stage $s$ approximations to all the of c.e. sets involved.

The construction then proceeds in the same way as before, with $P_{(e,n)}$ being the component of the witness which shows that if $R_{(e,n)}$ is a linearisation of $P$ then there exists a computable $\zeta$-sequence in $R_{(e,n)}$. Notice that because $n$ is encoded in the index
Chapter 3. The order type $\zeta$

of $\mathcal{P}_{(e,n)}$, we can computably determine which algorithms from the above proofs to use in the construction, and hence the construction remains computable. In effect we simultaneously perform the constructions for the proof for each $n \geq 1$.

We can improve this result, pushing the bound further up using a similar type of proof. In order to do so, we need a way of $\Delta_2$-approximating the sets of a wider class than $n$-c.e. and so we take the following definition and lemma from Badillo and Harris 2014 [2].

**Definition 3.6.**
A class of sets $C \subseteq \mathcal{P}(\mathbb{N})$ is uniform $\Delta_2$ if there is a binary function $f \leq_T \emptyset'$ such that the class $F$ of characteristic functions of $C$ satisfies $F = \{f_e | e \in \mathbb{N}\}$. Hence $C$ is uniform $\Delta_2$ if and only if there exists a uniform $\Delta_2$ approximation $\{A_{e,s} | e,s \in \mathbb{N}\}$ such that $C = \{A_e | e \in \mathbb{N}\}$.

Note that this definition corresponds to the notion of $\emptyset'$-uniform in the notation used by Jockush [20].

**Lemma 3.7 (Ershov 1968 [13]).**
For any computable ordinal $\alpha$, the class of $\alpha$-c.e. sets is uniform $\Delta_2$.

**Corollary 3.8.**
The class of $\omega$-c.e. sets is uniform $\Delta_2$.

Given the existence of a uniform $\Delta_2$ approximation of the class of $\omega$-c.e. sets, we can now improve the bound to $\omega$-c.e.

**Theorem 3.9.**
The order type $\zeta$ is not $\omega$-c.e.-extendible.

**Proof.** We construct as a witness a computable partial order $\mathcal{P} = (\mathbb{N}, \leq_P)$ as the disjoint union of sub-partial orderings $\mathcal{P}_e$ such that each $\mathcal{P}_e$ forms a connected component of $\mathcal{P}$,
and in particular every element of $P_e$ is incomparable to every element of every other $P_f$ where $e \neq f$. We will assume a listing of $\omega$-c.e. sets, $\{R_e\}_{e \in \mathbb{N}}$, and an associated $\Delta_2$ approximation $\{R_e[s]\}_{e,s \in \mathbb{N}}$, since we know that the class of $\omega$-c.e. sets is uniform $\Delta_2$ by Corollary 3.8. Then the purpose of $P_e$ is to show that if $R_e$ is a linearisation of $\leq_P$, then there exists a computable $\zeta$-sequence in $R_e$. By construction, such a sequence will be made up of elements of $P_e$.

At each stage $s$ of the construction we define an approximation $P[s]$ with domain some finite initial segment of $\mathbb{N}$ and we also compute the approximations $R_e[s]$ for $e < s$. At stage 0, $P[0] = \emptyset$, then at stage $s + 1$ we two new elements to each $P_e[s]$ with $e \leq s$.

When a number $n$ enters the construction at a stage $s$, all of the $\leq_P$-comparabilities of $n$ with $k < n$ are set at stage $s$ and do not change at any later stage. So each $P_e$ is computable because if we wish to compute $\leq_P$ relative to a pair of numbers $(n, m)$ then it suffices to run the construction until stage $s = \max\{n, m\} + 1$ and the approximation to $\leq_P$ relative to $(n, m)$ at stage $s$ will be the true value, and hence $P$ is computable.

We now fix some $e \in \mathbb{N}$ and consider the construction of $P_e$. At all stages $s \leq e$, $P_e[s] = \emptyset$, then at stage $s = e + 1$, to incomparable elements $a_e, b_e$ are added to define $P_e[s + 1]$.

Then at future stages $s$ where $R_e[s]$ decides the order of $a_e, b_e$ (i.e. exactly one of $a_e R_e[s] b_e$ and $b_e R_e[s] a_e$ holds), add two new points. Assuming without loss of generality that $a_e R_e[s] b_e$, we add a new point which is $\leq_P$-above $b_e$ and all points which are $\leq_P$-above $b_e$, and another point which is $\leq_P$-below $a_e$ and all points which are $\leq_P$-below $a_e$. I.e. extending a chain above $b_e$ and a chain below $a_e$.

$\{R_e[s]\}_{s \in \mathbb{N}}$ is a $\Delta_2$ approximation, so it eventually settles on the order of $a_e, b_e$. 


and if $R_e$ is a linearisation then there will be infinitely many points in $P_e$. So if $P_e$ is infinite then it looks like an $\omega$ chain and an $\omega^*$ chain which are mutually incomparable, but form a computable $\zeta$ chain in $R_e$.

Then as before, we can see that since $e$ was arbitrary, we can see the $P$ does not embed $\zeta$ but every $\omega$-c.e. linear extension computably embeds $\zeta$. And we have also seen that the construction of $P$ is entirely computable, and this concludes the proof. \qed

In fact this result can be further improved to $\Delta_2$, although the class of $\Delta_2$ sets is not uniform $\Delta_2$, so the same proof will not work. We have to modify it slightly.

**Lemma 3.10** (Lee, Harris and Cooper 2011 [29]).

For any $\Sigma_2$ set $A$ of codings of computable ordinals\(^1\), the class of $\alpha$-c.e. sets $\Sigma^{-1}_A = \bigcup_{a \in A} \Sigma^{-1}_a$ is uniform $\Delta_2$.

**Theorem 3.11.**

The order type $\zeta$ is not $\Delta_2$-extendible.

**Proof.** Whilst the class of $\alpha$-c.e. sets is uniformly $\Delta_2$ approximable for any computable ordinal $\alpha$, the class of $\Delta_2$ sets is not. However, we can construct a $\Sigma_2$ approximation to the class of $\Sigma_2$ sets, noting that a $\Sigma_2$ linear order with computably presented domain is in fact $\Delta_2$ (just as $\Sigma_1$ linear orders with computably presented domain are computable).

Specifically we construct a uniform $\Sigma_2$ approximation $\{R_e[s]\}_{e,s \in \mathbb{N}}$ of the sets $R_e = \{n \mid \exists s \forall t \geq s R_e(n)[t] = 1\}$ such that $R$ is an infinite $\Delta_2$ set if and only if $R = R_e$ for some $e$. In the context of orderings, the approximation will eventually decide that $aR_e b$, but may also switch between $bR_e a$ and $b \not R_e a$ infinitely often, though if it does converge, then it will be on $b R_e a$.

\(^1\)The coding is performed using Kleene’s $O$, but we suppress the details as they are not needed here.
Chapter 3. The order type $\zeta$

The construction is much the same as above, but has some vital differences. We construct as a witness a computable partial order $\mathcal{P} = (\mathbb{N}, \leq)$ as the disjoint union of sub-partial orderings $\mathcal{P}_e$ such that each $\mathcal{P}_e$ forms a connected component of $\mathcal{P}$, and in particular every element of $\mathcal{P}_e$ is incomparable to every element of every other $\mathcal{P}_f$ where $e \neq f$. We will use our $\Sigma_2$ approximations $\{R_e[s]\}_{e,s \in \mathbb{N}}$. Then the purpose of $\mathcal{P}_e$ is to show that if $R_e$ is a linearisation of $\leq$, then there exists a computable $\zeta$-sequence in $R_e$. By construction, such a sequence will be made up of elements of $\mathcal{P}_e$.

At each stage $s$ of the construction we define an approximation $\mathcal{P}[s]$ with domain some finite initial segment of $\mathbb{N}$ and we also compute the approximations $R_e[s]$ for $e < s$. At stage 0, $\mathcal{P}[0] = \emptyset$, then at stage $s + 1$ we two new elements to each $\mathcal{P}_e[s]$ with $e \leq s$.

When a number $n$ enters the construction at a stage $s$, all of the $\leq$-comparabilities of $n$ with $k < n$ are set at stage $s$ and do not change at any later stage. So each $\mathcal{P}_e$ is computable because if we wish to compute $\leq$ relative to a pair of numbers $(n, m)$ then it suffices to run the construction until stage $s = \max\{n, m\} + 1$ and the approximation to $\leq$ relative to $(n, m)$ at stage $s$ will be the true value, and hence $\mathcal{P}$ is computable.

We now fix some $e \in \mathbb{N}$ and consider the construction of $\mathcal{P}_e$. At all stages $s \leq e$, $\mathcal{P}_e[s] = \emptyset$, then at stage $s = e + 1$, to incomparable elements $a_e, b_e$ are added to define $\mathcal{P}_e[s + 1]$.

This is the point at which the proof diverges from the proof of Theorem 3.9. At stage $s$, if $a_e R_e[s] b_e$ and $a_e R_e[s - 1] b_e$ then two new points are added, one directly above $b_e$ and not above any other point above $b_e$, and one directly below $a_e$ and not below any other point below $a_e$. However, if $a_e R_e[s] b_e$ and $a_e R_e[s - 1] b_e$ then two new points are added, one above the point added above $b_e$ at stage $s - 1$, and one below the point added below $a_e$ at stage $s - 1$. 
Similarly, at stage $s$, if $b_e R_e[s] a_e$ and $b_e / R_e[s-1] a_e$ then two new points are added, one directly above $a_e$ and not above any other point above $a_e$, and one directly below $b_e$ and not below any other point below $b_e$. However, if $b_e R_e[s] a_e$ and $b_e R_e[s-1] a_e$ then two new points are added, one above the point added above $a_e$ at stage $s-1$, and one below the point added below $b_e$ at stage $s-1$.

The reason for this construction is that $R_e$ may change its mind on the order of $a_e, b_e$ infinitely often, and if we used the construction in Theorem 3.9, we would have $\zeta$ embedding in $P_e$. But if we start a new chain every time $R_e$ changes its mind, we avoid this possibility, since by the nature of the approximation, at most one pair of chains can be infinitely long, and this occurs exactly when $R_e$ is a linear extension of $P_e$.

Hence, we cannot embed $\zeta$ in $P_e$ and hence we cannot embed $\zeta$ in $P$, but if $R_e$ linearises $P$ then it contains a computable embedding of $\zeta$, constructed in the same way as before. Recall that a $\Sigma_2$ linear order is in fact $\Delta_2$, and we have that any $\Delta_2$ linearisation of $P$ computably embeds $\zeta$, and $P$ is a computable partial order which does not embed $\zeta$, as required. \hfill $\square$
Chapter 4

Scattered orderings

In this chapter we consider orderings which are scattered, that is have no dense suborder. There is less previously known in this area, but we show that in fact the same bounds apply as to well-founded orderings. Every computably scattered computable partial order has a computably scattered $\omega$-c.e. linear extension, but for any $n \in \mathbb{N}$, there is a computably scattered computable partial order which has no computably scattered $n$-c.e. linear extension.

4.1 Background

Definition 4.1.

We call a partial order $\mathcal{P}$ scattered if $\mathcal{P}$ does not embed $\eta$, the order type of the rational numbers. We say $\mathcal{P}$ is computably scattered if there is no computable embedding of $\eta$ in $\mathcal{P}$.

The classical result was proved by Bonnet and Pouzet, and independently by Galvin and McKenzie (unpublished), in the late 1960’s as part of a larger program. The proof we
give is not from the original paper, but from Bonnet and Pouzet’s 1981 survey of linear extension theory [5].

**Definition 4.2.**
If $P$ is a partial order, and $I$ is a suborder of $P$, we say that $I$ is an *initial segment* of $P$ if for any $x, y \in P$, if $y \in I$ and $x \leq_P y$ then $x \in I$.

**Definition 4.3** (Corominas).
A non-empty partial order $P$ is called an $\eta$-kernel if no non-empty initial or final segment of $P$ (with the induced order) has a scattered linear extension.

**Lemma 4.4.**
If $P$ has no scattered linear extension, then it contains an $\eta$-kernel.

*Proof.* Let $G(P)$ be the set of initial segments $I$ of $P$ which have scattered linear extensions $L(I)$. Then if $A \subseteq I \in G(P)$, we must have that $A$ has a scattered linear extension, although it may not be an initial segment and hence may not be in $G(P)$. Also, if $I, J \in G(P)$, then $I \cup J$ is an initial segment of $P$, and it has a scattered linear extension, which we construct by the order-theoretic sum of the scattered linear extension of $I$ and the scattered linear extension of $J \setminus I$. Hence, $G(P)$ is closed under finite unions.

Also if $(I_\alpha)_{\alpha < \delta}$ is a well-ordered increasing sequence of elements of $G(P)$ then the union $I = \bigcup_{\alpha < \delta} I_\alpha$ is in $G(P)$, since it has scattered linear extension built by the order-theoretic sum of the scattered linear extensions of the partial unions. Hence, $G(P)$ is closed under arbitrary unions, since they will be finite unions of unions of chains of initial segments, and so has a largest element $\hat{I} = \bigcup G(P)$.

Similarly, there is a largest final segment of $P$ which has a scattered linear extension, call it $\hat{F}$. Let $N = P \setminus (\hat{I} \cup \hat{F})$ with the induced order, and then $N$ is an $\eta$-kernel. Note that $N$ is not empty, because otherwise $P = \hat{I} \cup \hat{F}$ and then $P$ would have a scattered linear extension.  
\qed
**Lemma 4.5.**

If $\mathcal{P}$ contains an $\eta$-kernel then it is not scattered.

**Proof.** Let $\mathcal{P}_{\frac{1}{2}}$ be a partial order containing an $\eta$-kernel $N_{\frac{1}{2}}$. Choose an element $x_{\frac{1}{2}} \in N_{\frac{1}{2}}$ and let $\mathcal{P}_{\frac{1}{4}} = \{ x \in N_{\frac{1}{2}} \mid x \leq_{\mathcal{P}_{\frac{1}{2}}} x_{\frac{1}{2}} \}$ and $\mathcal{P}_{\frac{3}{4}} = \{ x \in N_{\frac{1}{2}} \mid x \geq_{\mathcal{P}_{\frac{1}{2}}} x_{\frac{1}{2}} \}$, with the induced orderings.

Since $N_{\frac{1}{2}}$ is an $\eta$-kernel, $\mathcal{P}_{\frac{1}{4}}$ and $\mathcal{P}_{\frac{3}{4}}$ have no scattered linear extensions, and hence contain non-empty $\eta$-kernels $N_{\frac{1}{4}}$ and $N_{\frac{3}{4}}$ respectively, from which we choose elements $x_{\frac{1}{4}}$ and $x_{\frac{3}{4}}$ and continue the process to define a sequence $(x_d)_{d \in D}$ of elements of $\mathcal{P}_{\frac{1}{2}}$ where the indexing set $D$ is the set of dyadic numbers\(^1\). These elements form a subset of $\mathcal{P}_{\frac{1}{2}}$ with order type $\eta$, and hence $\mathcal{P}_{\frac{1}{2}}$ is not scattered. \(\square\)

**Theorem 4.6** (Bonnet and Pouzet 1969 [4]).

Any scattered partial order has a scattered linear extension. I.e. $\eta$ is extendible.

**Proof.** The proof of this theorem then immediately follows by a contrapositive argument on the conjunction of the two lemmas. \(\square\)

This forms part of the general classification by Bonnet [3] of countable order types $\tau$ such that a partial order which does not embed $\tau$ can always be extended to a linear order which does not embed $\tau$. The effective content of this theorem differs slightly from the well-founded case, as Corollaries 4.9 and 4.8 to the following theorem of Downey, Hirschfeldt, Lempp and Solomon demonstrates.

**Theorem 4.7** (Downey, Hirschfeldt, Lempp and Solomon 2003 [12]).

There is a scattered computable partial ordering such that every computable linear extension has a computable densely ordered subchain.

---

\(^1\)The dyadic numbers are those rational numbers with a denominator which is a power of 2, so $D = \{\frac{1}{2}, \frac{1}{4}, \frac{3}{4}, \frac{5}{4}, \ldots\}$. 
Proof. We show that, given a sequence $X_0 \leq_T X_1 \leq_T \cdots$ of uniformly low, uniformly $\Delta^0_2$-sets, there is a scattered computable partial ordering $P = (P, \leq_P)$ such that for any $i \in \omega$, any $X_i$-computable linear extension $\leq_L$ of $\leq_P$ contains a densely ordered $\leq_L$-subchain, computable in $\leq_L$.

We construct $P$ using a finite-injury priority argument, as the disjoint union of $P_{i,e}$ for $i, e \in \omega$. Each $P_{i,e}$ will be a connected component of $P$, when $P$ is viewed as a directed graph, and has the purpose of showing that if the $e$-th binary $X_i$-computable relation $L_{i,e} = L^{X_i}_e$ is a linear extension $\leq_L$ of $\leq_P$ then there is a set of elements of $P_{i,e}$ which form a densely ordered subchain in $\leq_L$ which is computable in $L_{i,e}$. Since we are assuming that $X_i$ is low, uniformly in $i$, we may assume that $L^{X_i}_e$ is either total or finite, and thus $\leq_L$ can be effectively approximated.

Note that any dense $\leq_P$-subchain in $P$ must be contained in one $P_{i,e}$ and so we show that this in fact cannot happen, and therefore $P$ is scattered. We do this by constructing $P_{i,e}$ such that for any $x \in P_{i,e}$, there are either only finitely many elements $\geq_P x$ or only finitely many elements $\leq_P x$.

Fixing $i$ and $e$, we construct $P_{i,e}$ in stages, the full construction will computably interleave the constructions of each part. At stage 0 take three $\leq_P$-incomparable elements $a_0, a_1, a_2$, call them 0-critical and wait for $L_{i,e}$ to decide the $\leq_L$-ordering on them. Assume without loss of generality (relabelling as necessary) that $a_0 \leq_L a_1 \leq_L a_2$ and then place $a_1$ into a set $C$, and call $(-\infty, a_0)$ and $(a_2, \infty)$ the 0-active intervals.

At stage $s + 1$ we enumerate three new $\leq_P$-incomparable elements within each $s$-active interval, look for the $\leq_L$-ordering on them, put the middle one into $C$ and define the $s + 1$-active intervals as between the outer two of the $s + 1$-critical elements and the ends of the $s$-active interval. Note that if at any stage $L_{i,e}$ does not converge on the
Chapter 4. Scattered orderings

≤_L-ordering of the s-critical elements then \( P_{i,e} \) will just be finite.

The approximation of \( L_{i,e} \) may make finitely many mistakes before converging. Each
time the approximation \( L_e^{X_i}[s] \) changes its mind on a critical interval, the construction
throws away the elements that depended on \( a_0 \leq_L a_1 \leq_L a_2 \) and starts the construction
again from that point.

If \( P_{i,e} \) is infinite then this process builds \( C \) to be a \( \leq_L \)-computable densely \( \leq_L \)-
ordered chain, and hence \( L \) is not scattered, but for any element \( x \in P \), there are either
only finitely many elements \( \geq_P x \) or finitely many elements \( \leq_P x \), and so \( P \) is scattered.

Because \( \leq_L \) can be effectively approximated (because it is computable in a low
set), the construction of \( P_{i,e} \), and hence the construction of \( P \), will be computable, as if
\( L_{i,e} \) is a linearisation of \( P \) then it will converge on the order of the elements we add at
each stage, and otherwise \( P_{i,e} \) will just be finite.

Since \( i, e \) were arbitrary, we see that \( P \) has no dense suborder, and any \( X_i \)-computable
linear extension \( \leq_L \) of \( \leq_P \) contains a densely ordered \( \leq_L \)-subchain, computable in
\( \leq_L \). □

We can simplify this, since we don’t actually need to use the sequence of uniformly low,
uniformly \( \Delta_2 \) sets, as we’re not interested in what is implied by \( WKL_0 \). Instead just let
\( P \) be the disjoint union of \( P_e \), where each \( P_e \) ensures that the \( e \)-th computable binary
relation either is not a linear extension of \( \leq_P \) or contains a computable dense subchain.

Theorem 4.7 immediately implies the following two corollaries, just by weakening
the conditions in the statement in one way or another.
**Corollary 4.8** (Downey, Hirschfeldt, Lempp and Solomon 2003 [12]).
There is a scattered computable partial ordering that has no scattered computable linear extension.

**Corollary 4.9** (Downey, Hirschfeldt, Lempp and Solomon 2003 [12]).
There is a computably scattered computable partial ordering that has no computably scattered computable linear extension.

Corollary 4.8 shows a key difference relative to the situation for well-founded orderings, for which we have that any well-founded computable partial ordering has a well-founded computable linear extension, by Theorem 2.5 of Rosenstein and Kierstead [43]. Corollary 4.9, on the other hand, is analogous to the well-founded case proved by Rosenstein and Statman [43].

The difference between Corollary 4.8 and Theorem 2.3 raises the question of at what level of complexity there is always a scattered linear extension of a scattered computable partial order. A bound must exist here and we conjecture that a bound of $\Delta_2$ should be provable, although it is more difficult to construct a scattered linearisation than a computably scattered linearisation. This is because we cannot diagonalise against a list of c.e. sets, so different machinery would be needed.

**Conjecture 4.10.**
Every scattered computable partial order has a scattered $\Delta_2$ linear extension.

Regarding scattered orderings, Rosenstein [43] claims the following Theorem 4.13 for the computable case, suggesting that the proof is by a similar argument to his theorem for computably well-founded orderings, but not providing it. Downey mentions the theorem in [11], citing Rosenstein [43] but also does not give the proof. We provide a proof below, but it has subtleties compared to the well-founded case which Rosenstein did not mention.
Chapter 4. Scattered orderings

It seems Rosenstein intends to use the c.e. sets $\mathbb{W}_e$ and for each, if it is infinite, find a $\leq_P$-incomparable pair and force them to be a $\leq_L$-successive pair, with nothing in the linearisation in between them. However this runs up against the problem of recognising when a c.e. set will in the limit be dense, since naively at any finite stage, any ordering looks like it may be eventually dense. In order to get around this problem, which Rosenstein did not address, we introduce the concept of $\eta$-sequences.

**Definition 4.11.**

Let $\mathcal{P} = (\mathbb{N}, \leq_P)$ be a partial (possibly linear) ordering. If $\mathbb{W}_e$ is infinite, and $x_0, x_1, x_2, \ldots$ is a canonical enumeration of $\mathbb{W}_e$, then say $\mathbb{W}_e$ defines a $\eta$-sequence in $\leq_P$ if the $x_i$ form a chain in $\mathcal{P}$, and they are enumerated into $\mathcal{P}$ in the following way.

We use the standard binary tree coding of $\{0, 1\}^*$, as shown in Figure 4.1, and define a bijection $b : \mathbb{N} \to \{0, 1\}^*$ by setting $b(n) = \sigma \iff [1 \updownarrow \sigma]_2 = n + 1$. That is, $b(n) = \sigma$ if and only if appending the bit 1 to the beginning of $\sigma$ gives the binary number equal to $n + 1$. Figure 4.2 shows the tree after the map from finite binary sequences to natural numbers.

![Figure 4.1: Coding of finite binary strings on a binary tree](image-url)
Now, we say that $W_e$ defines a $\eta$-sequence in $\leq_P$ if the indices of the elements of $W_e$ in the order in which they are enumerated, fit the pattern on the tree. That is odd-indexed elements $x_{2n+1}$ are added to the chain as immediate $\leq_P$-predecessors of the element indexed by the number immediately above $2n + 1$ on the tree, and even-indexed elements $x_{2n}$ are added to the chain as immediate $\leq_P$-successors of the element indexed by the number immediately above $2n$ on the tree.

So for example, the first 10 elements of a $W_e$ which defines an $\eta$-sequence will be enumerated into $P$ with the ordering

$$x_7 <_P x_3 <_P x_9 <_P x_1 <_P x_8 <_P x_5 <_P x_0 <_P x_4 <_P x_6 <_P x_2 <_P x_5.$$ 

**Lemma 4.12.**

A computable partial ordering $P = (\mathbb{N}, \leq_P)$ is computably scattered if and only if for all $e \in \mathbb{N}$, $W_e$ does not define a $\eta$-sequence in $\leq_P$.

**Proof.** If $D$ is a computable dense subordering of $P$ without endpoints, then a computable subset of $D$ can be enumerated as a $\eta$-sequence. Conversely a c.e. set which enumerates a $\eta$-sequence in $P$ contains a dense computable set.  

This idea essentially gives us a “canonical” enumeration of $\eta$, in the sense that we only
need look for one particular computable enumeration of \( \eta \) to ensure there is no computable copy of \( \eta \) present.

**Theorem 4.13** (Rosenstein and Statman 1984 [43]).
Any computably scattered computable partial order has a computably scattered \( \Delta_2 \) linear extension.

**Proof.** Given a computably scattered, computable partial order \( \mathcal{P} = (P, \leq_P) \) (and assuming for simplicity that \( P = \mathbb{N} \)) we construct a linearisation \( \mathcal{L} \) to be computably scattered, and consider how complicated it need be.

We construct \( \mathcal{L} \) by enumerating numbers in turn as high as possible into \( \mathcal{L} \) whilst respecting \( \mathcal{P} \). To ensure that \( \mathcal{L} \) is computably scattered it suffices to guarantee that no \( W_e \) enumerates a \( \eta \)-sequence in \( \mathcal{L} \). In order to ensure that the subordering induced on \( W_e \) is not a \( \eta \)-sequence, we find some \( x^e_i, x^e_{i+1} \) which are either \( \leq_P \)-comparable and witness that \( W_e \) does not define an \( \eta \)-sequence or that are \( \leq_P \)-incomparable. If \( W_e \) is infinite then we will always be able to eventually find some \( x^e_i, x^e_{i+1} \), and place them into \( \mathcal{L} \) to prevent \( W_e \) from defining an \( \eta \)-sequence, as otherwise \( W_e \) will be a dense subordering of \( \mathcal{P} \).

We can use the first \( x^e_i, x^e_{i+1} \) enumerated that are \( \leq_P \)-incomparable for this purpose, if the enumeration of \( W_e \) up to that point looks like a \( \eta \)-sequence. For any infinite \( W_e \) there will be some finite stage in its enumeration at which there will either be a \( \leq_P \)-incomparable pair, or a pair which stops \( W_e \) from looking like a \( \eta \)-sequence, due to the fact that \( \mathcal{P} \) is computably scattered.

So, as in the proof of Theorem 2.7, the oracle we need is one to tell us whether \( W_e \) is infinite. So we again get the construction of \( \mathcal{L} \) being computable in a \( \Sigma_1 \)-set, and the result follows. \( \square \)
4.2 The upper bound

We can use the concept of \( \eta \)-sequences and adapt the argument of Theorem 2.8 to the scattered case in a mostly straightforward way, to lower the upper bound from \( \Delta_2 \) to \( \omega \)-c.e. analogously to the well-founded case in the previous chapter.

**Theorem 4.14.**

The order type \( \eta \) is computably \( \omega \)-c.e.-extendible. That is, every computably scattered computable partial order has a computably scattered \( \omega \)-c.e. linear extension.

**Proof.** Let \( P = (\mathbb{N}, \leq_P) \) be a computable partial order that is computably scattered. We build a linearisation \( L \) of \( P \) as the limit of a uniformly computable sequence of finite linear orderings \( (L_s = (\mathbb{N}|s, \leq_{L,s}))_{s \in \omega} \), such that the limit \( \lim_{s \to \infty} \leq_{L,s} = \leq_L \) exists and is a linear extension of \( \leq_P \).

By examining the construction, we shall show there is a computable bound on the number of changes to each element of \( \leq_{L,s} \), that is the set of \( s \) such that \( \leq_{L,s} \) does not agree with \( \leq_{L,s+1} \) on given \( a, b \in \mathbb{N} \) is bounded by a computable function in \( a \) and \( b \) (and so in \( (a, b) \)). Hence, the limit \( \leq_L = \lim_{s \to \infty} \leq_{L,s} \) is \( \omega \)-c.e.

Let \( W_e \) be the \( e \)-th computably enumerable set, under the standard listing and let \( x^e_0, x^e_1, x^e_2, \ldots \) be the elements of \( W_e \) in the order they are enumerated. Clearly, if \( (y_i)_{i \in \omega} \) is a computable sequence, then there is some \( e \) such that \( y_i = x^e_i \) for all \( i \).

By Lemma 4.12, to ensure that \( L \) is computably scattered, it suffices to ensure that no sequence \( (x^e_i)_{i \in \omega} \) defines a \( \eta \)-sequence in \( L \). As in the proof of Theorem 4.13 we achieve this for each infinite \( W_e \) by finding some \( x^e_i, x^e_{i+1} \in W_e \) such that either they are \( \leq_P \)-comparable and are not part of an \( \eta \)-sequence, or \( x^e_i |_P x^e_{i+1} \) and swapping them around to ensure that \( W_e \) cannot define a \( \eta \)-sequence.
If \( W_e \) would define an \( \eta \)-sequence in \( \mathcal{L} \) if we did not act to prevent it, then such \( x_i^e, x_{i+1}^e \) must exist as otherwise \( W_e \) would define an \( \eta \)-sequence in \( \mathcal{P} \), contradicting the assumption that \( \mathcal{P} \) is computably scattered.

### 4.2.1 Requirements

The construction will satisfy the following requirements, for \( e \in \omega \).

- \( N \): \( \mathcal{L} \) is a linear extension of \( \mathcal{P} \).
- \( R_e \): \( W_e \) does not define a \( \eta \)-sequence in \( \leq_L \).

The requirements are ordered in the priority ordering \( N \succ R_0 \succ R_1 \succ R_2 \succ \cdots \).

### 4.2.2 Strategy

We ensure that \( \mathcal{L} \) is a linear extension of \( \mathcal{P} \) by defining at every stage \( \mathcal{L}_s \) to be a linear extension of \( \mathcal{P} \upharpoonright s \). Then to ensure that the limit \( \lim_{s \to \infty} \mathcal{L}_s \) exists, we restrain the requirement \( R_e \) from acting on \( \leq_{L_s}^e \). So only finitely many requirements can modify any part of \( \leq_{L_s}^e \), and since we will show each requirement can only act finitely often, the limit exists. This also gives a computable bound on the number of changes, and hence that the ordering \( \leq_L \) is \( \omega \)-c.e.

For each requirement \( R_e \), we set a restraint threshold \( t_e[s] \), which is the \( \mathbb{N} \)-greatest number which \( R_e \) needs preserved from future alteration in order to ensure it remains satisfied, at the beginning of stage \( s \). We also define the restraint \( T_e[s] = \{ n \in \mathbb{N} \mid n \leq \mathbb{N} \max\{ t_{e'}[s] \mid e' < e \} \} \), which is the portion of \( \leq_{L,s} \) which \( R_e \) is not permitted to alter, in order to avoid injuring higher priority requirements. In fact we restrict \( R_e \) from changing the order of any element of the linearisation with any element.
of the restraint.

To satisfy $R_e$, we look for elements $x^e_i, x^e_j \geq \mathbb{N} e$, such that either they are $\leq_P$-comparable and prevent $W_e$ from defining a $\eta$-sequence, or they are $\leq_P$-incomparable and such that we can swap them around without affecting $<_{L,s+1} \cap T_e[s]$, in particular that there is no element of the restraint between them, and therefore without injuring the restraint of any higher priority requirement. We will argue that if $W_e$ is infinite, then we will eventually find some suitable pair of elements.

In the first case, if we find a pair of elements which prevent $W_e$ from defining a $\eta$-sequence, then we say that $R_e$ has been discharged, and move on.

In the second case, we plummet the element $x^e_i$, called the plummet witness, $\leq_{L,s-1}$-below $x^e_j$ as follows. Suppose that $x^e_j = c_k \leq_{L,s-1} \cdots \leq_{L,s-1} c_1$ is a list in $L_{s-1}$ of the finite set

$$\{c \mid c \in L_{s-1} \text{ and } x^e_j \leq_{L,s-1} c <_{L,s-1} x^e_i \text{ and } c \not\leq_P x^e_i\}$$

and notice that clearly $c_n \not\leq_P x^e_i$ for all $c_n$ (and therefore $c_n$ is also $\leq_P$-incomparable with any element $\leq_P$-comparable to $x^e_i$), since $\leq_{L,s-1}$ extends $\leq_P$ on the elements already enumerated.

Now, for any $c_1 \leq_{L,s-1} d \leq_{L,s-1} x^e_i$, we must have that $d \leq_P x^e_i$ but by definition $c_1 \not\leq_P x^e_i$ and therefore $c_1 \not\leq_P d$. Denote by $x^e_i + 1$ the immediate successor of $x^e_i$ in $L_{s-1}$ if it exists and move $c_1$ until $x^e_i \leq_{L,s} c_1 \leq_{L,s} x^e_i + 1$. Then by the same argument, we can repeat the process for all $c_i$ to get $x^e_i \leq_{L,s} c_k \leq_{L,s} \cdots \leq_{L,s} c_1 \leq_{L,s} x^e_i + 1$. If the successor $x^e_i + 1$ does not exist then just move $c_1$ to immediately above $x^e_i$ and then proceed as if $c_1 = x^e_i + 1$.

Notice that since $x^e_j = c_k$, at the end of stage $s$ we have $x^e_i \leq_{L,s} x^e_j$, and hence
$W_{e,s}$ is not an $\eta$-sequence. Now set $t_e[s + 1] = \max\{x_i^e, x_j^e\}$. After we have completed the plummeting, we say that $Re$ is discharged.

### 4.2.3 Construction

At stage 0, set $t_e[0] = 0$ for all $e$.

At stage $2s + 1$ we enumerate the number $s$ into $L_s$, as high as possible\(^2\) whilst ensuring that $L_s$ is a linear extension of $\mathcal{P}_{s}$, and whilst not placing $s$ inbetween any pair of elements which is witnessing some $R_e$.

We say that $R_e$ requires attention at stage $s$ if the elements of $W_{e,s}$ describe the start of a $\eta$-sequence in $\leq_L$, and $R_e$ can be satisfied at stage $s$, i.e. there are two elements $x_i^e, x_j^e >_{\mathcal{P}} T_e[s]$ which are $\leq_{\mathcal{P}}$-incomparable, greater than $T_e[s]$ and do not have an element of the restraint for $e$ between them in $L_{s-1}$, or if there is a pair of elements $x_i^e <_{\mathcal{P}} x_j^e$ which prevent $W_e$ from defining an $\eta$-sequence and $W_e$ has not been already discharged.

At stage $2s + 2$ we find the least $e$ such that $R_e$ requires attention (if no such $e$ exists then end the stage and go on to stage $2s + 3$). If $W_e$ requires attention because there is a pair of elements $x_i^e <_{\mathcal{P}} x_j^e$ which prevent $W_e$ from defining an $\eta$-sequence, then say $R_e$ is discharged and move on to stage $2s + 3$.

Otherwise take the elements $x_i^e, x_j^e$ as described above and if, without loss of generality, $x_i^e \leq_{L,s} x_j^e$, then plummet $x_j^e$ as described above in order to get $x_j^e \leq_{L,s+1} x_i^e$. This ensures that $W_e$ does not define a $\eta$-sequence, but $L_{s+1}$ is still a linear extension of $\mathcal{P}_{s+1}$.

---

\(^2\)Enumerating elements as high as possible is an arbitrary choice, it would be equally valid to enumerate as low as possible or in some other way as long as it is compatible with the partial order.
Then say $R_e$ has been discharged and move on to stage $2s + 3$. Note that it is possible that a higher priority requirement will act later to undo this switch, and in that case $R_e$ stops being discharged and may again require attention.

### 4.2.4 Verification

**Lemma 4.15.**

$L$ is a linear extension of $P$.

*Proof.* At every stage $L_s$ is defined to be a linear extension of $P \mid_s$, and any finite initial segment of the linearisation is fixed after a finite number of stages, so the limit $\lim_{s \to \infty} L_s = L$ exists, and is a linear extension of $P$. □

**Lemma 4.16.**

$L$ is computably scattered.

*Proof.* If $L$ were not computably scattered then there would be some computable subchain of $L$ which had order type $\eta$. And so there would be some computably enumerable set $W_e$ which enumerated this subchain as a $\eta$-sequence, and so the requirement $R_e$ would not be satisfied.

Let $s_0$ be a stage such that $T_e[s]$ is fixed for all $s \geq s_0$, which exists because the requirements with higher priority can only act a finite number of times. We can show this by induction on $e$, obviously $R_0$ only acts at most once, then assuming that $R_{e'}$, $e' < e$ act finitely often, we see that $R_e$ must act at most once more than the combined number of actions of all $R_{e'}$, which will also be finite. Hence, $s_0$ exists. Then we argue that $R_e$ is satisfied after $s_0$.

We suppose that $W_e$ is infinite, because if $W_e$ is finite then $R_e$ is trivially satisfied.
If the finite part of $W_e$ which has been enumerated by stage $s_0$ does not define the start of a $\eta$-sequence, then $R_e$ will be satisfied, so suppose it does. Then if $R_e$ is never satisfied, the enumeration of $W_e$ after stage $s_0$ must go on to define a $\eta$-sequence, and never require attention (because if it did then $R_e$ would act to satisfy itself with highest priority), and so it must be the case that there is no pair $x_i^e, x_j^e \geq \omega \ T_e[s]$ which is incomparable in $\mathcal{P}$ and without any number less than $T_e[s]$ between them in $\mathcal{L}$. But then since $T_e[s]$ is finite, there will be a dense subset of $W_e$ which is entirely between two successive points in the threshold $T_e[s]$. This subset will define a computable $\eta$-sequence in $\mathcal{P}$, which contradicts the fact that $\mathcal{P}$ is computably scattered.

Hence, all requirements $R_e$ are satisfied and $\mathcal{L}$ is computably scattered. 

**Lemma 4.17.**

$\mathcal{L}$ is $\omega$-c.e.

**Proof.** If $R_e$ acts at stage $s$, it will remain satisfied and will not act again at a subsequent stage, unless it is injured by a requirement with higher priority. Hence, any requirement will act at most $2^e$ times. Since $R_e$ cannot make changes to $\leq_{L_e}$ for numbers $\leq e, \leq_{B,s}|e$ can change at most $2^e - 1$ times, and so changes in $\leq_{L_e}$ are computably bounded, and $\mathcal{L}$ is $\omega$-c.e. 

This completes the proof of Theorem 4.14.
4.3 The lower bound

The \( n \)-c.e. lower bound result also carries over from well-founded to scattered orderings, but the proof simplifies considerably, because rather than needing any dense suborders to compute the Halting set, we can just construct the witness to have no dense suborders. We couldn’t, in the well-founded case, create the order without any infinite descending chains because of König’s Lemma, but here we can construct a partial order such that any element has either only finitely many predecessors or only finitely many successors, using a similar technique to that of Downey, Hirschfeldt, Lempp and Solomon [12] to prove Theorem 4.7.

**Theorem 4.18.**

The order type \( \eta \) is not computably 2-c.e.-extendible. That is, there exists a computably scattered\(^3\) computable partial order \( \mathcal{P} \) which has no computably scattered 2-c.e. linear extension.

**Proof.** We construct as a witness a computable partial order \( \mathcal{P} = (\mathbb{N}, \leq_{\mathcal{P}}) \) as the disjoint union of sub-partial orderings \( \mathcal{P}_e \) such that each \( \mathcal{P}_e \) forms a connected component of \( \mathcal{P} \), and in particular every element of \( \mathcal{P}_e \) is incomparable to every element of every other \( \mathcal{P}_f \) where \( e \neq f \).

We will assume a computable listing of 2-c.e. sets, \( \{ R_e \}_{e \in \mathbb{N}} \), where \( R_{(i,j)} = W_i \setminus W_j \), and an associated computable approximation \( R_{(i,j)}[s] = W_i[s] \setminus W_j[s] \). Then the purpose of \( \mathcal{P}_e \) is to show that if \( R_e \) is a linearisation of \( \leq_{\mathcal{P}} \), then there exists a computable \( \eta \)-sequence in \( R_e \). By construction, such a sequence will be made up of elements of \( P_e \).

We will construct \( \mathcal{P}_e \) in such a way that every element has either only finitely many predecessors or only finitely many successors, and thus \( \mathcal{P}_e \) cannot contain any

---

\(^3\)In fact classically scattered, as noted above.
$\eta$-sequences. Since there are no $\leq_P$-comparabilities between the components and each component is computably scattered, we have that $\mathcal{P}$ as a whole is computably scattered. There is a key difference here from the proofs of Theorems 2.14 and 3.5, in that we do not need to construct $\mathcal{P}$ such that any $\eta$-sequence in $\mathcal{P}$ computes $K$, i.e. such that it is computably scattered but not necessarily scattered. We will in fact construct $\mathcal{P}$ so that we have the stronger condition that $\mathcal{P}$ is scattered, which simplifies the proof considerably.

At each stage $s$ of the construction we define an approximation $\mathcal{P}[s]$ with domain some finite initial segment of $\mathbb{N}$, we also compute the approximations $R_e[s]$ for $e < s$. At stage 0, $\mathcal{P}[0] = \emptyset$, then at stage $s + 1$ we add up to two new elements to each $\mathcal{P}_e[s]$ with $e < s$ and four new elements to $\mathcal{P}_s[s]$.

When a number $n$ enters the construction at a stage $s$, all of the $\leq_P$-comparabilities of $n$ with $k < n$ are set at stage $s$ and do not change at any later stage. So each $\mathcal{P}_e$ is computable because if we wish to computer $\leq_P$ relative to a pair of numbers $(n, m)$ then it suffices to run the construction until stage $s = \max\{n, m\} + 1$ and the approximation to $\leq_P$ relative to $(n, m)$ at stage $s$ will be the true value, and hence $\mathcal{P}$ is computable.

We now fix some $e \in \mathbb{N}$ and consider the construction of $\mathcal{P}_e$. At all stages $s \leq e$, $\mathcal{P}_e[s] = \emptyset$, then at stage $s = e + 1$, four new elements are added to define $\mathcal{P}_e[s+1]$, with the comparabilities $b_{-1} <_P y <_P a_{-1}$ and $b_{-1} <_P x <_P a_{-1}$ shown in Figure 4.3. We define $(b_{-1}, a_{-1})$ to be the Level 1 active interval and also to be the Level 2 active interval.

The labels $y$ and $x$ are temporary and will be reused many times during the construction for different elements before they are given a permanent label.
Definition 4.19.
We call a stage \( s > e + 1 \) a good stage for \( e \) if \( R_e[s] \) linearises \( P_e[s] \). The construction of \( P_e \) will only proceed at good stages for \( e \), note that the set of good stages for \( e \) is computable.

Definition 4.20.
If \( s \) is a good stage for \( e \), then for \( m, n \in P_e[s] \) we say that \( m R_e n \) is computed at stage \( s \) if \( \langle m, n \rangle \in R_e[s] \). Since \( s \) is good for \( e \) and \( R_e[s] \) is therefore a linear order, this obviously means that \( \langle n, m \rangle \notin R_e[s] \).

Definition 4.21.
We say that \( m R_e n \) is 1-computed at stage \( s \) if it is computed at stage \( s \) and there is no earlier good stage at which \( n R_e m \) is computed.

We say that \( m R_e n \) is 2-computed at stage \( s \) if it is computed at stage \( s \) and there is an earlier good stage at which \( n R_e m \) is 1-computed.

Note that for a pair \( m, n \in P_e \), \( n R_e m \) can only be computed at a good stage for \( e \), so from now on when we say that \( m R_e n \) is computed (or 1-computed, 2-computed, etc) at stage \( s \) we mean by definition that \( s \) is a good stage for \( e \).
For some $R_e$ it may be the case that $m R_e n$ is 1-computed at stage $s$ and there has been an earlier stage $t < s$ such that $(n, m) \in R_e[t]$, and it may even be the case that $(m, n) \in R_e [t]$ as well. However, this just means that by definition $t$ is not a good stage for $e$, and hence neither $m R_e n$ or $n R_e m$ was computed at stage $t$.

We now consider the simple case in which there is no stage $s$ and pair $k, l \in P_e$ such that $k R_e l$ is 2-computed at stage $s$. In effect we are initially proving the case for $n = 1$, which is a reproof of Corollary 4.9 (since as in the well-founded case we have that the result for computably enumerable linearisations immediately follows from the result for computably linearisations). We will then consider the case where there are 2-computed pairs, and show how the construction nests.

Now, suppose that $s$ is the first good stage for $e$, and without loss of generality suppose that $y R_e x$ is 1-computed at stage $s$. Then relabel $x$ as $a_0$ and $y$ as $b_0$, and we add two new $\leq_P$-incomparable elements $y, x$ into $P_e[s + 1]$ such that $b_{-1} <_P y <_P b_0$ and $b_{-1} <_P x <_P b_0$, as shown in Figure 4.4, and we define the Level 1 active interval to be $(b_{-1}, b_0)$.

![Figure 4.4: After the first good stage for $e$](image)

At each subsequent good stage $s$ for $e$, we will define $a_i$ and $b_i$, redefine the Level 1 active
interval and add two new elements $y, x$. When we want to know where to put the pair $y, x$ and the new Level 1 active interval, we use a coding similar to that in Definition 4.11. Again we create an tree of indices and an isomorphism to the standard tree of finite binary strings, as shown in Figures 4.5 and 4.6.

![Figure 4.5: Coding of finite binary strings on a binary tree](image)

![Figure 4.6: Coding of indices on a binary tree](image)

We add 1 to the index of the last $a, b$ pair, and find the corresponding position on the tree. Then place the pair $y, x <_p$-above $b_j$ and $<_p$-below $a_k$, where $j$ is the first index above and to the left on the tree, and $k$ is the first index above and to the right on the tree. Using either $j$ or $k$ is $−1$ if they don’t exist.

In Figure 4.7 we see what $\mathcal{P}_e$ looks like after several good stages, notice the how we can see the $\eta$ pattern of pairs building up. The next Level 1 active interval $(b_{−1}, b_7)$ is marked by a dotted line, where the next $y, x$ pair will be added.
Figure 4.7: After the first few stages

If there are only finitely many good stages, then $\mathcal{P}_e$ will be finite, and the necessary conditions for the theorem trivially hold, as it contains no $\eta$-sequence, and $R_e$ is not a linearisation of $\mathcal{P}_e$. Indeed if $R_e$ is a linearisation of $\mathcal{P}_e$ then there will be infinitely many good stages for $e$, at which elements were added to $\mathcal{P}_e$ and $\mathcal{P}_e$ would be infinite.

If there are infinitely many good stages for $e$ and $\mathcal{P}_e$ is therefore infinite, still under the supposition that there is no pair which is ever 2-computed, then the sequence \( \{a_n\}_{n \in \mathbb{N}} \) forms a computable dense suborder of $R_e$, but not of $\mathcal{P}_e$. This is because any given $a_n$
has finitely many elements $\leq P$ below it, since once $a_n$ is defined, at no stage will any new element be enumerated into $P_e$ directly below $a_n$.

We note that similarly for the $b_n$, once they are defined there will not be any new elements placed directly above $b_n$, and so every element of $P_e$ has either finitely many elements above it or below it, and therefore $P_e$ is scattered.

If $R_e$ linearises $P_e$, then we can find a computable $\eta$-sequence in $R_e$. In fact by construction, $\{a_n\}_{n \in \mathbb{N}}$ is a computable $\eta$-sequence in $R_e$, which will be infinite as we know that $R_e$ linearising $P_e$ means that $P_e$ will be infinite, and therefore $R_e$ is not computably scattered. This concludes the proof of the simpler case in which there is no pair which is ever 2-computed.

Now, suppose that in fact there is at least one pair $n, m \in P_e$ such that $n R_e m$ is 2-computed at some stage $s$. Note that without loss of generality we can in fact assume that the 2-computed pair is $a_n, b_n$ for some $n$, because it is clear by inspection of the construction that if any pair is 2-computed then there will be some $a_n, b_n$ that is 2-computed. Thus during the construction below we only need to search for the least index $k$ such that $b_k R_e c_k$ is 2-computed at stage $s + 1$.

So, suppose we are at the first stage at which any $a_n, b_n$ is 2-computed, and suppose that $n$ is the least index of a 2-computed pair. Then we relabel $a_n$ as $b'_0$, $b_n$ as $a'_0$ and add two new numbers $y', x'$ to $P_e[s + 1]$ such that $b_{-1} < P y' < P b'_0$ and $b_{-1} < P x' < P b'_0$. Then we define $(b_{-1}, b'_0)$ to be the Level 1 active interval and also the Level 2 active interval. We then restart the Level 1 construction inside the Level 2 active interval.

The Level 1 construction then restarts from scratch in the new Level 1 interval, building a new $a_n$ sequence until such a time as there is another stage $t$ at which some
pair is 2-computed. As further pairs are 2-computed, we do the same thing, labelling
them as $a'_i, b'_i$ and then restarting the Level 1 construction in the new Level 2 interval,
which is defined in the correct place using the same coding as in the Level 1 construction
to so that the 2-computed pairs as they appear, form an $\eta$-sequence.

The construction continues as above. We see that if there are only finitely many
stages at which the Level 2 active interval is redefined, then from some stage $s$ all activity
in the construction will be at Level 1, and we get the same outcome as in the first case.
Namely that if $Re$ is a linearisation of $Pe$ then there is a computable dense suborder in
$Re$, and by construction the $a_n$ form such a suborder. And $Pe$ is still constructed to be
scattered.

Alternatively the Level 2 active interval may be redefined infinitely many times, or
in other words there may be infinitely many stages at which there is a 2-computed pair.
But then the $\{a'_n\}_{n \in \mathbb{N}}$ form a computable dense suborder of $Re$.

To show that $\{a'_n\}_{n \in \mathbb{N}}$ is a computable sequence, notice that the pairs $(a'_n, b'_n)$ which
are 2-computed, can be found by a uniformly computable search, and hence there is
an algorithm which computes the sequence of such pairs in order, and therefore also
computes $a'_0, a'_1, a'_2, \ldots$.

We see that $P$ is still scattered because any element either has only a finite number of
elements above it or below it in $P$.

Since $e$ was arbitrary, we see that $P$ has no computably scattered 2-c.e. linear
extension, and $P$ is scattered. But we have also seen that the construction of $P$ is entirely
computable. This concludes the proof of Theorem 4.18.
Corollary 4.22.
The order type $\eta$ is not computably $n$-c.e.-extendible for any $n \geq 1$. That is, for every $n \geq 1$, there exists a scattered computable partial order $\mathcal{P}$ which has no computably scattered $n$-c.e. linear extension.

Proof. A computably enumerable (1-c.e.) linearisation of $\mathcal{P}$ will be computable. We proved the $n = 1$ case as the first part of the proof for $n = 2$, and of course the partial order constructed by Downey et. al. as a witness to Theorem 4.7 also proves this case. We now show how to generalise the proof of the $n = 2$ case above to any $n \in \mathbb{N}$.

Consider the $n = 3$ case. We perform the construction as with $n = 2$, but with an extra layer, the construction can now define Level 3 active intervals. When a Level 3 active interval is defined or redefined all Level 2 and Level 1 construction is restarted from scratch in the new Level 3 interval. At Level 3 we will be building a computable $\eta$-sequence $a''_n$ in $R_e$, when pairs $b''_n, c''_n$ are 3-computed.

We will assume a computable listing of 3-c.e. sets, $\{R_e\}_{e \in \mathbb{N}}$, where $R_{(i,j,k)} = (W_i \setminus W_j) \cup W_k$, and an associated computable approximation $R_{(i,j,k)}[s] = (W_i[s] \setminus W_j[s]) \cup W_k[s]$. We define 3-computation in the obvious way, we say that $m R_e n$ is 3-computed at stage $s$ if it is computed at stage $s$ and there is an earlier good stage at which $n R_e m$ is 2-computed. And in general, if $R_e$ is a $k$-c.e. set, then we say that $m R_e n$ is $r$-computed at stage $s$, for some $r \leq k$, if it is computed at stage $s$ and there is an earlier good stage at which $n R_e m$ is $r - 1$-computed and there is no earlier good stage at which $n R_e m$ is $r + 1$-computed.

Then the construction in the case where there is 3-computation works in the same way relative to 2-computation as 2-computation works relative to 1-computation. For example when the first pair is 3-computed this is because a 2-computed pair $(b'_n, a'_n)$ has
been found such that $b'_n R_e a'_n$ is 3-computed at this stage $s$ (and $n$ is the least index of such a pair). Then we enumerate new elements $x, y$ and redefine $b''_0 = a'_n$ and $a''_0 = b'_n$, as shown in Figure 4.8.

![Figure 4.8: After the first pair is 3-computed](image)

Then the construction at Level 2 and Level 1 restarts from scratch in the new Level 3, 2 and 1 active interval $(b''_0, a''_0)$. It behaves as a nested version of the construction in the Theorem above, but with the difference that the Level 3 active interval can be redefined either finitely or infinitely often.

We take the greatest $t \in \{1, 2, 3\}$ such that the Level $t$ active interval is redefined infinitely often. If the Level 3 active interval is redefined infinitely often then if $R_e$ linearises $\mathcal{P}_e$ then we define a sequence of $a''_n$ which will be a computable $\eta$-sequence in $R_e$. If it is redefined only finitely often, then after some finite stage it will be fixed, and the remainder of the construction will be at Level 2 and Level 1. So the outcomes are the same as in the $n = 2$ case, either the $a'_n$ defined form a computable $\eta$-sequence in $R_e$ (if $R_e$ linearises $\mathcal{P}_e$), or after a finite time there is only Level 1 activity, and the $a_n$ form a computable $\eta$-sequence in $R_e$ (if $R_e$ linearises $\mathcal{P}_e$).

The case for $n$-c.e. is then a straightforward generalisation of this, take a computable listing of $n$-c.e. sets $R_e$ and associated approximations $R_e[s]$, and define $n$-computation
as above. Work through the construction as in the \( n - 1 \) case, but when something is \( n \)-computed we define the points \( a_i^{(n-1)}, b_i^{(n-1)} \) and redefine the Level \( n \) active interval. There will then be some \( 1 \leq t \leq n \) such that the Level \( t \) active interval is redefined infinitely often. Then if \( t \) is the greatest of those, if \( R_e \) linearises \( P \) then the \( \{a_i^{t-1}\}_{n \in \mathbb{N}} \) will be a computable \( \eta \)-sequence in \( R_e \).

Then for any \( n \) we can construct a witness \( P \) which is computable and computably scattered, but has no computably scattered \( n \)-c.e. linearisation.

Again, we note that this construction fails for \( \omega \)-c.e., because if there is no fixed bound on the number of times a pair can be changed in the linearisation, then there will not necessarily be a greatest \( t \) such that the Level \( t \) active interval is redefined infinitely often. In fact we could have no such \( t \) and just keep redefining intervals of greater and greater Level, but only finitely often. Thus we cannot build the computable \( \eta \)-sequence necessary in \( R_e \) and the argument fails.

As in the previous chapter, we could have proved this by building one witness which proves the theorem for all \( n \geq 1 \), rather than constructing a separate counterexample for each \( n \geq 1 \). To do this, we need a listing of all \( n \)-c.e. sets, for all \( n \geq 1 \), which we define \( \{R_{(n,e)}\}_{n,e \in \mathbb{N}} \) with uniform approximations \( \{R_{(n,e)[s]}\}_{n,e,s \in \mathbb{N}} \).

Let \( g : \mathbb{N}^3 \rightarrow \mathbb{N} \) be a computable function such that

\[
W_{g(n,e,k)} = \begin{cases} 
\bigcap_{0 \leq i \leq k} W_{e_i} & \text{if } 0 \leq k \leq n \text{ (where } e = \langle e_0, e_1, \ldots, e_{n-1} \rangle), \\
\emptyset & \text{if } k > n.
\end{cases}
\]

Note that \( n \) determines the length of the tuple coded as \( e \), and if \( n = 0 \) then we use the convention that \( e = \langle e \rangle \), so \( e_0 = e \).
Then
\[ R_{(n,e)} = \left\lfloor \frac{n}{2} \right\rfloor \bigcup_{i=0}^{s} W_{g(n,e,2i)} \setminus W_{g(n,e,2i+1)}, \]

and \( R_{(n,e)[s]} \) is defined using the stage \( s \) approximations to all the of c.e. sets involved.

The construction then proceeds in the usual way, with \( \mathcal{P}_{(e,n)} \) being the component of the witness which shows that if \( R_{(e,n)} \) is a linearisation of \( \mathcal{P} \) then there exists a computable \( \eta \)-sequence in \( R_{(e,n)} \). Notice that because \( n \) is encoded in the index of \( \mathcal{P}_{(e,n)} \), we can computably determine which algorithms from the above proofs to use in the construction, and hence the construction remains computable. In effect we simultaneously perform the constructions for the proof for each \( n \geq 1 \).
Bibliography


