A proof theoretic study of abstract termination principles

Thomas Powell

Abstract

We carry out a proof theoretic analysis of the wellfoundedness of recursive path orders in an abstract setting. We outline a general termination principle and extract from its wellfoundedness proof subrecursive bounds on the size of derivation trees that can be defined in Gödel's system T plus bar recursion. We then carry out a complexity analysis of these terms, and demonstrate how this can be applied to bound the derivational height of term rewrite systems.

Keywords. Program extraction, term rewriting, recursive path orders, termination analysis, complexity, bar recursion

1 Introduction

The ability to deduce whether or not a program terminates is crucial in computer science. Informally, if we envisage a *run* of some a program to be a sequence of transitions between states

$$s_0 \rightsquigarrow s_1 \rightsquigarrow \ldots \rightsquigarrow s_k,$$

where each transition represents some elementary operation, termination corresponds to the notion that there are no infinite runs, or alternatively, every possible run ends in a *normal form*: a state that cannot be evaluated any further.

Though termination is not a decidable property, a number of powerful *proof rules*, or *termination principles*, have been developed, which set out general conditions under which programs can be shown to terminate. Examples of these include path orders for rewrite systems [7], the size-change principle [17] and more recently methods based on Ramsey's theorem [20].

Any termination principle P gives rise to the following question: Given that a program can be proven to terminate using P, can we infer an upper bound on its *runtime complexity*, namely the maximum length of runs starting from some given state s_0? This is in turn an instance of a much more general problem in proof theory, captured by Kreisel in his famous quote from [15]:

> *"What more do we know if we have proved a theorem by restricted means than if we merely know the theorem is true?"*

In this article we focus on the termination of rewrite systems via path orders. This area already contains a number of well known complexity results of the above kind. For example, termination via the multiset path ordering implies primitive recursive runtime complexity (where here transitions correspond to single rewrite steps), while the lexicographic path ordering guarantees at worst multiple recursive complexity. These bounds were initially established via direct calculations in [12] and [28] respectively.

Here, we take an approach to complexity closer to the spirit of Kreisel by addressing the following question: Given a proof that some abstract order is wellfounded, can we extract from this proof a subrecursive program which computes rewrite sequences of terms and thus provides a bound on their length?

Broadly speaking, there are two ways of accomplishing this. We could choose to concentrate on the *proof*, showing that it can be formalised in some weak theory and then appealing to an appropriate logical metatheorem which would guarantee that a function for bounding rewrite sequences can be defined in some corresponding calculus. Alternatively, we could directly extract such a function by hand and then show that it can be defined in a suitable restricted system. The latter approach is chosen here: Exhibiting an explicit bounding function is not only more illuminating, but we can appeal to mathematical properties of that function to obtain more refined complexity results.

The starting point of this work is the elegant paper of Buchholz [6], who was one of the first to apply proof theoretic techniques to termination principles. More specifically, Buchholz rederived the aforementioned bounds on the multiset and lexicographic orders by showing that wellfoundedness of these orders could be formalised in weak fragments of Peano arithmetic, and then applying a program extraction theorem to obtain the corresponding bounds on the length of reduction sequences. Key to this method is to consider only *finitely branching* variants of the usual path orders - an approach which will be essential to us as well. We work in a more abstract framework than Buchholz, but demonstrate in Section 4 how his bounded path orders can be viewed as an instance of ours.

A second source of inspiration is the recent collection of papers (including [3, 4, 10]), which study both size-change termination and techniques based on Ramsey's theorem from the perspective of proof theory. In particular, in [3] an upper bound on the length of transition sequences is given as a term of System T extended with bar recursion. It turns out that bar recursion - a form of recursion over wellfounded trees - is naturally suited to computing normalization trees for programs. Moreover, one can directly appeal to *closure properties* of bar recursion ([19, 23]) to establish upper bounds on the size of these trees.

In this paper, we study an abstract termination principle which subsumes the majority of path orders encountered in the literature, including both multiset and lexicographic orderings and e.g. the large class of standard orderings encompassed by the unifying work of Fereirra and Zantema [9]. In fact, our termination principle is closely related to the very general first termination theorem considered by Goubault-Larrecq in [11], though here it is based on re-

lations which are assumed to be finitely branching. We give a classical proof of our termination principle, which we then analyse using proof-theoretic methods: More specifically, we show that given moduli which form computational analogues of the theorem's main conditions, a function bounding the size of rewrite sequences can be defined using bar recursion of lowest type. A number of initial complexity results can already be given by appealing to [19] and related works.

We then consider a variant of the theorem in which a computationally stronger realizer to the premise is given. In this case, more refined complexity results are possible, which are set out in Corollary 3.18. We conclude by showing how the well known upper bounds for the complexity of simplification orders follow from this result, giving a concrete sketch in the case of the multiset path order.

Our hope is that the results of this paper form a framework for complexity which can be developed further in the future, with potential for both more general and more refined results. In addition, in the process of our proof theoretic analysis we explore a number of deep mathematical concepts which underlie path orders, including minimal-bad-sequence style constructions, realizability and bar recursion, connections between the latter having been explored from a more general perspective in e.g. [22, 24]. We aim to demonstrate how these concepts all come together to form a particularly elegant illustration of the correspondence between proofs and programs.

1.1 Prerequisites and notation

In this article, we work over an informal type theory, where types are defined by the following grammar:

$$\rho, \tau ::= \mathbb{N} \mid \rho \to \tau \mid \rho \times \tau \mid \rho^*$$

We write either $a \in \rho$ or $a : \rho$ to denote that an object a is of type ρ (typically we use $a \in \rho$ when referring to basic objects such as numbers or sequences, and $a : \rho$ when talking of functions or functionals, but this convention is not necessarily rigorously adhered to). Most of the proofs that follow can be formalized in some standard higher type extension of Peano arithmetic (such as the E-PA$^\omega$ of [26] enriched with product and sequence types), although occasionally we make use of stronger principles such as wellfounded induction or countable dependent choice.

We assume that the reader is familiar with Gödel's System $\mathtt{T}$ of primitive recursive functionals in all finite types, which we will use as our base programming language, though we recall some key facts here (a detailed summary of System T can be found in e.g. [2, 26] or [14, Chapter 3]).

Terms of System $\mathtt{T}$ include variables of each type, together with the usual zero and successor constants, and allow for the construction of new terms via function application $ts : \tau$ for $t : \rho \to \tau$ and $s : \rho$ and lambda application $\lambda x.t : \rho \to \tau$ for $x : \rho$ and $t : \tau$. In addition, System $\mathtt{T}$ possesses *recursors* $\mathtt{Rec}^\rho$ of

each type ρ, which satisfy the following axiom schema

$$\mathtt{Rec}^{\rho}_{a,f}(0) = a \qquad \mathtt{Rec}^{\rho}_{a,f}(n+1) = fn(\mathtt{Rec}^{\rho}_{a,f}(n)),$$

where $\mathtt{Rec}^{\rho}_{a,f}(n) : \rho$. In addition to all this, our version of System T also contains projection and pairing constants for dealing with cartesian products, together with some basic operations for working with sequences. We collect below some important notational conventions which will be used throughout:

- We denote by 0_ρ a canonical zero element of type ρ, defined in the obvious way ($0_{\mathbb{N}} = 0$, $0_{\rho\to\tau} = \lambda x^\rho.0_\tau$, $0_{\rho\times\tau} = (0_\rho, 0_\tau)$ and $0_{\rho^*} = []$),
- $|a|$ is the length of the sequence $a \in \rho^*$,
- if $a = [a_0, \dots, a_{k-1}] \in \rho^*$ and $x \in \rho$ then $a * x := [a_0, \dots, a_{k-1}, x]$ denotes the concatenation of a with the element x, while similarly $x :: a := [x, a_0, \dots, a_{k-1}]$. For two lists a and b we write $a^\frown b := [a_0, \dots, a_{k-1}, b_0, \dots, b_{l-1}]$, and more generally we write $a_1{}^\frown \dots {}^\frown a_k$ to denote the successive concatenation of a finite sequence of lists,
- $\bar{a} := a_{k-1}$ denotes the last element of a (we just set $\bar{a} = 0_\rho$ if $a = []$),
- we write $x \in a$ if $x = a_i$ for some $i < |a|$,
- for $a \in \rho^*$ we define $\hat{a} : \mathbb{N} \to \rho$ by $\hat{a}_n := a_n$ if $n < |a|$ and $\hat{a}_n := 0_X$ otherwise.

Remark 1.1. We often denote *infinite sequences* of objects of type ρ using the type notion $\rho^{\mathbb{N}}$. Though technically there is absolutely no difference between $\rho^{\mathbb{N}}$ and the function type $\mathbb{N} \to \rho$, throughout the article we make an informal distinction between infinite sequences and functions, which tend to play quite different roles, and this is also reflected in our notation α_n for the former and $\alpha(n)$ for the latter.

Our main complexity results involve *fragments* of System T in which the type complexity of the recursor is restricted. To be more precise, each type is defined a *level* is the following standard way: $\mathrm{level}(\mathbb{N}) = 0$, $\mathrm{level}(\rho \to \tau) = \max\{\mathrm{level}(\rho) + 1, \mathrm{level}(\tau)\}$, $\mathrm{level}(\rho \times \tau) = \max\{\mathrm{level}(\rho), \mathrm{level}(\tau)\}$ and $\mathrm{level}(\rho^*) = \mathrm{level}(\rho)$. We then denote by $\mathtt{T}_i$ the subsystem of System T which only permits recursors $\mathtt{Rec}^\rho$ for types ρ with $\mathrm{level}(\rho) \le i$. In particular, it is well known (cf. [30]) that the closed terms of type $\mathbb{N} \to \mathbb{N}$ definable in $\mathtt{T}_0$ are the usual Kleene primitive recursive functions, while those definable in $\mathtt{T}_1$ correspond the multiple recursive functions (alternatively the fast growing hierarchy below ω^ω).

Finally, at several points we will need to extend T with constants $\mathtt{Rec}^{\rhd,\rho}$ for wellfounded recursion of output type ρ over some decidable wellfounded binary relation $\rhd$ on $\mathbb{N}$, which will satisfy the defining axiom

$$\mathtt{Rec}^{\rhd,\rho}_f(x) =_\rho fx(\lambda y \lhd x\ .\ \mathtt{Rec}^{\rhd,\rho}_f(y))$$

where $\lambda y \lhd x\ .\ g(y)$ is shorthand for 'if $y \lhd x$ then $g(y)$ else 0_ρ'.

Remark 1.2. When defining recursive functionals we typically use the convention, as above, of writing parameters which don't change in the defining equation as a subscript.

2 Finitely branching binary relations

We start off in this section by covering some basic facts and definitions concerning finitely branching binary relations in general, and introduce the concept of bar recursion.

2.1 Basic notions

Our basic object of study will be a binary relation $\succ$ on some set X. In the context of termination analysis, X is typically a set of terms in some language. A program p is then considered to be reducing with respect to $\succ$ if whenever

$$t_0 \rightsquigarrow t_1 \rightsquigarrow \ldots \rightsquigarrow t_k$$

is a run on p, then $t_i \succ t_{i+1}$ for all $i < k$. Thus wellfoundedness of $\succ$ implies that the program terminates.

However, up until Section 4, everything will be carried out in an abstract setting. For now, the only assumption we make about X is that it can be *arithmetized* i.e. comes equipped with some encoding $\ulcorner \cdot \urcorner : X \to \mathbb{N}$, and similarly $\succ$ is a primitive recursive relation, i.e. there is some term $r : \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ definable in T_0 such that $x \succ y$ iff $r(\ulcorner x \urcorner, \ulcorner y \urcorner) = 0$. For the sake of clarity, we continue to refer to the set as X rather than $\mathbb{N}$, but it should be remembered that for practical purposes $\succ$ is a relation on natural numbers, and this will indeed be crucial when we come to our complexity results later.

In this paper we will primarily be concerned with relations which are finitely branching.

Definition 2.1. We say that $\succ$ is *finitely branching* if

$$(\forall x \in X)(\exists a \in X^*)(\forall y)(x \succ y \leftrightarrow y \in a).$$

where we recall that $y \in a$ if $y = a_i$ for some $i < |a|$. In particular, the number of distinct elements y with $x \succ y$ is bounded above by $|a|$.

We now need to give a precise definition of what we mean by *wellfoundedness*. We will primarily be interested in the following formulation, which in [3] is referred to as *classical wellfoundedness*:

Definition 2.2. We call a sequence $\alpha \in X^{\mathbb{N}}$ (classically) wellfounded (w.r.t. $\succ$) and write $W_\succ(\alpha)$ if

$$\exists n(\alpha_n \not\succ \alpha_{n+1})$$

where $x \not\succ y$ denotes $\neg(x \succ y)$. Similarly, we say that an element $x \in X$ is wellfounded, and also write $W_\succ(x)$, if

$$\forall \alpha(x = \alpha_0 \to W_\succ(\alpha)).$$

The relation $\succ$ is wellfounded if $(\forall x)W_\succ(x)$.

In Section 3 we will consider an equivalent formulation of wellfoundedness which is classically equivalent to the above but computationally stronger. We now make precise how we measure the 'runtime complexity' of some object in X.

Definition 2.3 (Finite derivation). We call a finite sequence $a \in X^*$ a $\succ$-derivation, and write $C_\succ(a)$, if $a_i \succ a_{i+1}$ for all $i < |a| - 1$.

Definition 2.4 (Derivational height). Let $x \in X$ and suppose that there exists some k such that

$$C_\succ(x * a) \to |a| \leq k.$$

We call the minimal such k the *derivational height* of x and denote it by $dh(x)$. We say that the derivational height of some wellfounded $\succ$ is bounded by some function $f : X \to \mathbb{N}$ if $dh(x) \leq f(x)$ for all $x \in X$.

Remark 2.5. In this paper, the derivational height will form our main notion of runtime complexity. It is closely related to the concept of *derivational complexity*, which involves in addition a size measure $s : X \to \mathbb{N}$ on terms, and is defined to be a function $g : \mathbb{N} \to \mathbb{N}$ such that $s(x) \leq n$ implies $dh(x) \leq g(n)$ for all $x \in X$ and $n \in \mathbb{N}$. For most sensible size measures (which when X is a term structure is invariably the syntactic size of a term), the derivational complexity can be defined in terms of a function bounding the derivation height, assuming our underlying language is finite.

We now give a syntactic formulation of wellfoundedness which will be crucial to us later, and which is adapted from Buchholz's notion of a derivation [6]. Here we work with a structure which encodes in a slightly more precise way the derivation tree generated by some wellfounded x, where informally, the derivation tree for x is that whose root is x and whose branches are derivations starting at x.

Definition 2.6 (Derivation tree). The predicate $T_\succ(x, d)$ on $X \times X^*$ is defined by induction on the length of d as follows: If $[y_0, \ldots, y_{k-1}]$ is the unique sequence consisting exactly of those elements y with $x \succ y$, ordered so that $y_i < y_j$ (as number encodings) iff $i < j$, then $T(x, d)$ holds precisely when $d = x :: d_0 \frown \ldots \frown d_{k-1}$ and $T_\succ(y_i, d_i)$ holds for all $i < k$.

Intuitively, $T_\succ(x, d)$ holds iff d represents the flattening of the tree of finite derivations starting from x which would be obtained by a depth first search and ordering each child node by its encoding. Take for example the relation on $\{1, 2, \ldots, 7\}$ defined by

$$2 \succ 4, 7 \qquad 4 \succ 1, 3, 6 \qquad 3 \succ 5 \tag{1}$$

Then we would have $T_\succ(2, d)$ iff $d = [2, 4, 1, 3, 5, 6, 7]$. Note that $T_\succ(x, d)$ makes sense even when x is not wellfounded: in that case $T_\succ(x, d)$ would simply be false for all d. However, when it holds for some d then this must be unique.

Lemma 2.7. *If $T_\succ(x, d)$ and $T_\succ(x, e)$ then $d = e$.*

Lemma 2.8. *If $T_\succ(x, d)$ then $dh(x) \leq |d|$.*

Proof. Induction on the length of d. Suppose that $x \succ x_1 \succ \ldots \succ x_k$. Then we have $T_\succ(x_1, e)$ for some e contained in d, and assuming inductively that $k - 1 \leq dh(x_1) \leq |e| < |d|$ we obtain $k \leq |d|$ and thus $dh(x) \leq |d|$. □

Theorem 2.9. *If $\succ$ is finitely branching then $W_\succ(x)$ holds iff $(\exists d)T_\succ(x, d)$.*

Proof. One direction follows immediately from Lemma 2.8. For the other, $(\forall d)\neg T_\succ(x, d)$ would imply that the derivation tree of x is infinite, and so by König's lemma this tree must have an infinite branch. But that would contradict $W_\succ(x)$. □

2.2 Computing derivation trees

The focus of this article will be on the construction of explicit derivation functions for wellfounded binary relations:

Definition 2.10. A function $\Phi : X \to X^*$ is a *derivation function* for the wellfounded relation $\succ$ if $(\forall x)T_\succ(x, \Phi(x))$ holds.

Whenever Φ is a derivation function for $\succ$, by Lemma 2.8 in particular it follows that the map $\lambda x.|\Phi(x)|$ bounds the derivational height of $\succ$. Therefore, whenever we can guarantee that Φ can be defined in some restricted class of functions, we can produce a subrecursive bound for the derivational height of $\succ$.

Note that for any finitely branching $\succ$, provided we know in advance that x is wellfounded, its derivation tree can be computed by a simple brute force search. However, a much stronger result would be to show that the computation of a derivation tree can be defined in some *subrecursive* calculus, which takes into account the strength of the system in which $W_\succ(x)$ can be proved.

In this section we give a short preliminary result of this kind, where we analyse the statement

$$\text{if } \succ \text{ is finitely branching and wellfounded then } (\forall x)(\exists d)T_\succ(x, d). \tag{2}$$

The statement follows as in Theorem 2.9 from an application of König's lemma. We are interested in giving (2) a *computational interpretation*, namely the construction of a derivation function for $\succ$ which takes as parameters some functionals which give a computational interpretation to the premise of (2). This leads us to the following key definitions:

Definition 2.11. (a) A *branching modulus* for $\succ$ is a function $c : X \to X^*$ satisfying

$$x \succ y \leftrightarrow y \in c(x)$$

for all $x, y \in X$. We assume w.l.o.g. that $c(x)$ is ordered with respect to our encoding and contains no repetitions.

(b) A *modulus of wellfoundedness* for $\succ$ is a function $\omega : X^{\mathbb{N}} \to \mathbb{N}$ satisfying

$$(\exists i < \omega(\alpha))(\alpha_i \not\succ \alpha_{i+1})$$

for all $\alpha \in X^{\mathbb{N}}$.

Moduli of wellfoundedness have also been studied in [3] in the context of the Podelski-Rybalchenko termination theorem, and we take our terminology from them. Given a branching modulus and $x \in X$, one can easily compute the derivation tree d for x by implementing a depth first search. We now show that given, in addition, a modulus of wellfoundedness, we can give a subrecursive definition of the derivation function in System $\mathtt{T}$ plus bar recursion, where the latter is a recursion scheme over wellfounded trees. There are numerous different variants of bar recursion (see [21]), but here we will be primarily concerned with the original version due to Spector [25].

Definition 2.12 (Bar recursion). The constant $\mathtt{BR}^{\rho,\tau}$ of bar recursion of type ρ, τ is characterised by the following defining equation (cf. Section 1.1 for notation, and note again our convention of writing parameters which don't vary in defining equations in the subscript):

$$\mathtt{BR}^{\rho,\tau}_{\omega,g,h}(a) :=_\tau \begin{cases} g(a) & \text{if } \omega(\hat{a}) < |a| \\ ha(\lambda x.\mathtt{BR}^{\rho,\tau}_{\omega,g,h}(a * x)) & \text{otherwise} \end{cases}$$

Here $a \in \rho^*$ and the other parameters have types $\omega : \rho^{\mathbb{N}} \to \mathbb{N}$, $g : \rho^* \to \tau$ and $h : \rho^* \to (\rho \to \tau) \to \tau$.

Definition 2.13. We denote by $\mathtt{T} + \mathtt{BR}^{\rho,\tau}$ the theory of System $\mathtt{T}$ extended with constants for bar recursion of type ρ, τ, and similarly for e.g. $\mathtt{T} + \mathtt{Rec}^{\triangleright,\rho}$ and $\mathtt{T} + \mathtt{Rec}^{\triangleright,\rho'} + \mathtt{BR}^{\rho,\tau}$.

Before we give our construction, we need some notation for some simple recursive operations on sequences. We use the symbol $\bigcirc$ to denote iterated list concatenation. So for $a = [a_0, \ldots, a_{k-1}] \in (\rho^*)^*$ we have

$$\bigcirc_{x \in a} x := a_0 {}^\frown \ldots {}^\frown a_{k-1} \in \rho^*.$$

We abuse this symbol just like a summation symbol, so for example if $a \in \rho^*$ and $p : \rho \to \rho^*$ then

$$\bigcirc_{x \in a} p(x) := p(a_0) {}^\frown \ldots {}^\frown p(a_{k-1}) \in \rho^*$$

and so on. Note that this operation is definable in $\mathtt{T}_0$ using recursion over the length of $|a|$.

Lemma 2.14. *Let $x \in X$ and suppose that $p : X \to X^*$ is a function satisfying $T_\succ(y, p(y))$ for all $y \prec x$. Then*

$$d := x :: \bigcirc_{y \in c(x)} p(y)$$

satisfies $T_\succ(x, d)$ whenever c is a branching modulus for $\succ$.

Proof. Directly from the Definitions 2.6 and 2.11 (a). □

Theorem 2.15. *In* $\mathtt{T}_0 + \mathtt{BR}^{X,X^*}$ *we can define a function* $\Psi_{c,\omega} : X^* \to X^*$ *which takes parameters* $c : X \to X^*$ *and* $\omega : X^{\mathbb{N}} \to \mathbb{N}$ *and satisfies*

$$\Psi_{c,\omega}(a) := \begin{cases} [] & \text{if } |a| = 0 \text{ or } \omega(\hat{a}) < |a| \\ \bar{a} :: \bigcirc_{y \in c(\bar{a})} \Psi_{c,\omega}(a * y) & \text{otherwise.} \end{cases}$$

Moreover, if c *resp.* ω *is a branching modulus resp. modulus of wellfoundedness for* $\succ$*, then the function* $\lambda x.\Psi_{c,\omega}([x])$ *is a derivation function for* $\succ$*.*

For the proof of Theorem 2.15 we appeal to the principle of *countable dependent choice* in addition to standard reasoning within arithmetic in finite types. This refers to a weak variant of the axiom of choice given by the following scheme:

$$\forall n \in \mathbb{N}, x \in \rho\ \exists y \in \rho\ A(n, x, y) \to \exists f : \mathbb{N} \to \rho\ \forall n \in \mathbb{N}\ A(n, f(n), f(n+1)).$$

Proof of Theorem 2.15. That Ψ is definable in $\mathtt{T}_0 + \mathtt{BR}^{X,X^*}$ is a simple exercise, and we omit it here (though definability results in later sections are included in full). For the verification proof, we first show that for any sequence satisfying $|a| > 0$ and $C_\succ(a)$ we have:

$$\neg T_\succ(\bar{a}, \Psi_{c,\omega}(a)) \to (\exists y \prec a) \neg T_\succ(y, \Psi_{c,\omega}(a * y)) \tag{3}$$

To see this, note that $C_\succ(a)$ implies that $\omega(\hat{a}) \geq |a|$, else there would be some $i, i+1 < |a|$ with $a_i \not\succ a_{i+1}$. Therefore by the contrapositive of Lemma 2.14 we obtain (3).

Now, suppose that there exists some x such that $\neg T_\succ(x, \Psi_{c,\omega}([x]))$. Then by dependent choice together with (3) there exists some infinite descending sequence $\alpha_0 \succ \alpha_1 \succ \ldots$, contradicting the fact that $\alpha_i \not\succ \alpha_{i+1}$ for some $i < \omega(\alpha)$. Thus $T_\succ(x, \Psi_{c,\omega}([x]))$ holds for all x, and we're done. □

The above theorem is not deep in itself, but is included as a simple illustration of the results which will follow. Note that though the proof uses classical logic together with dependent choice, we could convert this into an intuitionistic proof which instead uses some variant of bar induction, as is typically the case for program extraction theorems. However, in this paper we have no broader foundational goals which would require the verification of our extracted terms to be formalisable in a weak intuitionistic theory, so we stick to classical logic as it is usually more intuitive.

3 Abstract path orders

Path orders form one of the earliest proof rules for termination, and are a central concept in the theory of term rewriting. Today, a huge variety of different path orders have been developed, ranging from the general - such as the unified ordering of [31] - which focus on the common structure shared by termination orders, to the highly specialised - such as the polynomial path ordering of [1]

- which aim to capture a very precise class of terminating programs. We talk about path orders in more detail in Section 4, but for now we give a simple explanation which helps motivate the abstract principle studied here.

3.1 Path orders and termination

Very roughly, path orders capture 'termination via minimal sequences'. Consider the Ackermann-Péter function, which is recursively defined by the rules

$$\begin{aligned} A(0,n) &\succ n+1 \\ A(m,0) &\succ A(m-1,1) \\ A(m+1,n+1) &\succ A(m,A(m+1,n)) \end{aligned}$$

Suppose for contradiction that $A(m,n)$ is not wellfounded for some $m,n \in \mathbb{N}$ i.e. it triggers an infinite computation. Then either $A(m,n-1)$ is not wellfounded, or there is some well defined $k := A(m,n-1)$ such that $A(m-1,k)$ is not wellfounded. In other words, there is some (m',n') lexicographically less than (m,n) such that $A(m',n')$ is not wellfounded. By repeating this reasoning, non-wellfoundedness of $A(m,n)$ gives rise to an infinite sequence

$$A(m_0,n_0) \gg A(m_1,n_2) \gg A(m_2,n_2) \gg \dots \tag{4}$$

where $A(m,n) \gg A(m',n')$ denotes that (m',n') is lexicographically smaller than (m,n). Thus $A(m,n)$ must be wellfounded by wellfoundedness of the lexicographic ordering. Informally speaking, (4) plays the role of a minimal sequence, in the sense that it represents instances of A whose arguments i.e. subterms are wellfounded.

3.2 The abstract termination principle

On an abstract level, path orders are a proof rule which implement the idea that termination of a program can be inferred from wellfoundedness of minimal sequences. To make this idea formal, we first need to introduce an auxiliary binary relation on X. Recall that a binary relation on X is primitive recursive if on the level of encodings it can be represented as a primitive recursive function of type $\mathbb{N} \times \mathbb{N} \to \mathbb{N}$.

Definition 3.1. Let $\triangleright$ be a primitive recursive binary relation on X which is *inductively wellfounded*, by which we mean that $\triangleright$ has the usual induction property and as a result we have access to the recursor $\mathtt{Rec}^{\triangleright,\rho}$ as defined in Section 1.1.

Inductive wellfoundedness is equivalent to classical wellfoundedness as defined in Section 2. However, from a computational point of view the two differ: Classical wellfoundedness is realized by some modulus of wellfoundedness of type $X^{\mathbb{N}} \to \mathbb{N}$, while the computational analogue of inductive wellfoundedness will be the recursor $\mathtt{Rec}^{\triangleright}$. Note that a modulus of wellfoundedness for $\triangleright$ is easily computable in $\mathtt{Rec}^{\triangleright}$, but defining $\mathtt{Rec}^{\triangleright}$ in some modulus of wellfoundedness for $\triangleright$ would seem to require bar recursion in addition.

The reason that we choose $\rhd$ to be inductively wellfounded is that when X is some set of terms, $\rhd$ usually represents the subterm relation, recursion over which is trivially definable in $\mathtt{T}_0$. A key concept in our abstract termination principle is the formal notion of a *minimal sequence*. The precise definition is as follows:

Definition 3.2 (Minimal sequence). An infinite sequence $\alpha \in X^{\mathbb{N}}$ is minimal (with respect to $\succ$ and $\rhd$) if $W_{\succ}(y)$ for all $y \lhd \alpha_n$ and $n \in \mathbb{N}$.

In addition to $\succ$ and $\rhd$ we consider a third binary relation $\gg$, which interacts with the other relations in a specific way as outlined in Definition 3.3 below. This is an abstract formulation of the notion of $>$ being *decomposable* as presented be Ferreira and Zantema [9] (cf. their Definition 3), and essentially coincides with 'Property 1' of Goubault-Larrecq's relations $>$, $\rhd$ and $\gg$ in [11].

Definition 3.3 (Decomposition). A primitive recursive binary relation $\gg$ on X is called a decomposition of $\succ$ with respect to $\rhd$ if it satisfies

(i) whenever $x \succ y$ then either $x \gg y$ or there exists some $z \lhd x$ such that $z \succeq y$ (where $\succeq$ denotes the reflexive closure of $\succ$),

(ii) whenever $x \gg y$ and $y \rhd z$ then $x \succ z$.

We are now ready to state and prove our main abstract termination principle, which is in turn closely related to Theorem 1 of [11], though our proof uses dependent choice instead of the more intuitionistically flavoured argument via bar induction in [11].

For the reader familiar with path orders from term rewriting, it might be helpful in what follows to keep in mind the following informal associations:

abstract object	*possible interpretation*
X	set of terms over some signature
main relation $\succ$	path order itself
wellfounded relation $\rhd$	immediate subterm relation
auxiliary relation $\gg$	term lifting (cf. [9, Section 3] or [11, Section 3])

Theorem 3.4 (Abstract termination principle)**.** *Let $\succ$, $\rhd$ and $\gg$ be binary relations on X, such that*

1. *$\rhd$ is inductively wellfounded,*
2. *$\gg$ is a decomposition of $\succ$ with respect to $\rhd$,*
3. *$\gg$ is classically wellfounded on minimal sequences, by which we mean that for any sequence $\alpha \in X^{\mathbb{N}}$ which is minimal with respect to $\succ$ and $\rhd$, we cannot have $\alpha_n \gg \alpha_{n+1}$ for all $n \in \mathbb{N}$.*

Then $\succ$ is wellfounded on X.

Proof. Defining

$$A := \{x \in X \ : \ (\forall y \lhd x) W_{\succ}(y)\},$$

we claim that for any nonempty sequence $a \in A^*$ satisfying $C_{\gg}(a)$ we have:

$$\neg W_{\succ}(\bar{a}) \to (\exists y \ll \bar{a})(\neg W_{\succ}(y) \wedge y \in A). \tag{5}$$

To see this, observe that $\neg W_{\succ}(\bar{a})$ implies that the set

$$S_{\bar{a}} := \{x \in X \mid x \prec a \wedge \neg W_{\succ}(x)\}$$

is nonempty. Thus by classical logic together with induction on $\rhd$ (which is possible since $\rhd$ is inductively wellfounded), $S_{\bar{a}}$ has some minimal element y i.e. such that $z \lhd y$ implies $z \notin S_{\bar{a}}$. Now, it follows that $\bar{a} \gg y$, otherwise, by decomposition property (i) we would have $\bar{a} \rhd z \succeq y$ for some z, and since $\bar{a} \in A$ which means that $W_{\succ}(z)$ holds, this would contradict $\neg W_{\succ}(y)$. But using property (ii) we can therefore also show that $y \in A$: since $\bar{a} \gg y$ then for any $z \lhd y$ we have $\bar{a} \succ z$, and therefore $W_{\succ}(z)$ since $\neg W_{\succ}(z)$ would imply that $z \in S_{\bar{a}}$, contradicting minimality of y. This proves the claim.

For the main result, suppose that $\neg W_{\succ}(x)$ holds for some x, and define α_0 to be the minimal such x with respect to $\rhd$, so that $\neg W_{\succ}(\alpha_0)$ and $\alpha_0 \in A$. Then $C_{\gg}([\alpha_0])$ trivially holds, and by applying dependent choice together with (5) we obtain an infinite sequence $\alpha_0 \gg \alpha_1 \gg \alpha_2 \gg \ldots$ with $\alpha_i \in A$ for all i. But the latter means that α is minimal, which contradicts the assumption that $\gg$ is wellfounded on minimal sequences. □

Remark 3.5. One of the anonymous referees made an observation regarding Definition 3.3 and Theorem 3.4, which we include here as a remark. Suppose we define the relation $\gg_{\rhd}$ by

$$\gg_{\rhd} := \{\rhd^{n+1} \cup (\rhd^n \circ \ \gg) \mid n \in \mathbb{N}\}$$

or alternatively $x \gg_{\rhd} y$ iff $x \rhd z_0 \rhd \ldots \rhd z_{n-1} \rhd y$ or $x \rhd z_0 \rhd \ldots \rhd z_{n-1} \gg y$ for some $z_0, \ldots, z_{n-1}$. Then assuming that $\rhd$ is wellfounded, we have that $\gg$ is a decomposition of $\succ$ with respect to $\rhd$ if

$$(\gg \circ \rhd) \subseteq \succ \subseteq \gg_{\rhd} . \tag{6}$$

In this sense, Theorem 3.4 could be replaced by the slightly more compact statement that any $\succ$ satisfying (6) is wellfounded, provided that $\gg$ is wellfounded on minimal sequences w.r.t. $\succ$ and $\rhd$. However, our formulation is much closer to related results in the term rewriting literature, including [9] and [11], and as such we prefer it here.

3.3 A computational interpretation of the termination principle

We now give a computational interpretation of Theorem 3.4, in the case where both $\succ$ and $\rhd$ are finitely branching. Similarly to before this assumption will be represented by a pair of branching moduli $c_{\succ}$ and $c_{\rhd}$. The computational

analogue of inductive wellfoundedness of $\triangleright$ will be access to wellfounded recursion over $\triangleright$, so it remains to formulate our main assumption that $\gg$ is classically wellfounded on the set of all minimal sequences.

Definition 3.6. The predicate $M_{\succ,\triangleright}(x,u)$ on $X \times X^{**}$ is defined to be true whenever for $[y_0,\ldots,y_{k-1}] = c_\triangleright(x)$ where $c_\triangleright$ is a branching modulus for the relation $\triangleright$ (cf. Definition 2.11 (a)) we have

$$|u| = k \wedge (\forall i < k) T_\succ(y_i, u_i).$$

Continuing with our earlier example (1), suppose that $x \triangleright y$ only when y is a proper divisor of x. Then $M_{\succ,\triangleright}(6,[[1],[3,5]])$, since 1 and 3 are the only proper subdivisors of 6 and both $T_\succ(1,[1])$ and $T_\succ(3,[3,5])$.

Lemma 3.7. *Assuming that both $\succ$ and $\triangleright$ are finitely branching, a sequence $\alpha \in X^{\mathbb{N}}$ is minimal iff there exists a sequence $\beta \in (X^{**})^{\mathbb{N}}$ such that $M_{\succ,\triangleright}(\alpha_n,\beta_n)$ holds for all $n \in \mathbb{N}$.*

Proof. Directly from Theorem 2.9. □

This syntactic characterisation of minimal sequences for finitely branching relations informs the following adaptation of the modulus of wellfoundedness, now restricted to minimal sequences:

Definition 3.8. A modulus of *minimal wellfoundedness* for $\gg$ (with respect to $\succ$ and $\triangleright$) is a function $\omega : (X \times X^{**})^{\mathbb{N}} \to \mathbb{N}$ satisfying

$$(\forall n) M_{\succ,\triangleright}(\alpha_n,\beta_n) \to (\exists i < \omega(\alpha,\beta))(\alpha_i \not\gg \alpha_{i+1})$$

where by for clarity we represent the two components of $(X \times X^{**})^{\mathbb{N}}$ separately as $\alpha \in X^{\mathbb{N}}$ and $\beta \in (X^{**})^{\mathbb{N}}$, and write e.g. $\omega(\alpha,\beta)$ instead of $\omega(\lambda i.(\alpha_i,\beta_i))$.

In the construction that follows we denote by $\bigodot$ the usual *map* function i.e. given $a \in X^*$ and $p : X \to \rho$ we have

$$\bigodot_{x \in a} p(x) := [p(a_0),\ldots,p(a_{k-1})] \in \rho^*$$

where $a = [a_0,\ldots,a_{k-1}]$. The following lemma follows directly from the definitions:

Lemma 3.9. *Let $x \in X$ and suppose that $q : X \to X^*$ is a function satisfying $T_\succ(y,q(y))$ for all $y \lhd x$. Then*

$$u := \bigodot_{y \in c_\triangleright(x)} q(y)$$

satisfies $M_{\succ,\triangleright}(x,u)$ whenever $c_\triangleright$ is a branching modulus for $\triangleright$.

The following two results form an analogue of Theorem 2.15 in the more complex setting of abstract path orders. They comprise a technical lemma below, which establishes that the analogue of our function Ψ in this case can be defined in the theory $T_0 + \mathtt{Rec}^{\triangleright,X^*} + \mathtt{BR}^{Y,X^*}$ (the analogue to this part was omitted from the proof of Theorem 2.15, but here it is given in full), which is then followed by the main result as Theorem 3.12.

Lemma 3.10. *Define $Y := X \times X^{**}$, and suppose that $c_\succ, c_\rhd : X \to X^*$ are some fixed terms of $\mathtt{T}_0$ which form branching moduli for $\succ$ and $\rhd$. Then there is a functional $\Psi : (Y^{\mathbb{N}} \to \mathbb{N}) \to Y^* \to X^*$ definable in $T_0 + \mathtt{Rec}^{\rhd, X^*} + \mathtt{BR}^{Y, X^*}$ which satisfies*

$$\Psi_\omega(a,b) = \begin{cases} [] & \text{if } |a| = 0 \text{ or } \omega(\widehat{a,b}) < |a| \\ \bar{a} :: \bigcirc_{y \in c_\succ(\bar{a})} R_{a,b}(y) & \text{otherwise} \end{cases}$$

where the term $R_{a,b} : X \to X^*$ satisfies

$$R_{a,b}(y) := \begin{cases} \bar{b}_i[y] & \text{if } y \preceq c_\rhd(\bar{a})_i \text{ for some } i < |c_\rhd(\bar{a})| \\ \Psi_\omega(a * y, b * \odot_{z \in c_\rhd(y)} R_{a,b}(z)) & \text{otherwise} \end{cases}$$

where in the first line, for $d \in X^*$ and $y \in X$, $d[y] \subset d$ denotes some sequence contained in d and satisfying $T_\succ(y, d[y])$ whenever it exists (and just $[]$ otherwise).

Remark 3.11. The intuition behind Ψ_ω is that it is designed to play the following role in the proof of Theorem 3.12 below: Given a sequence $a \in X^*$ together with some $b \in (X^{**})^*$ which witnesses that a is minimal in the sense that $M_{\succ,\rhd}(a_i, b_i)$ for all $i < |a|$, then $\Psi_\omega(a,b)$ is a derivation tree for $\bar{a}$ (the last element of a). The reader who is happy to believe that such a functional can be defined via bar recursion is encouraged to skip ahead to Theorem 3.12, as the routine but somewhat technical proof of Lemma 3.10 is somewhat orthogonal to the proof of the main theorem.

Proof of Lemma 3.10. We define functions $g : Y^* \to X^*$ and $h : Y^* \to (Y \to X^*) \to X^*$ by

$$g(a,b) := []$$

$$h(a,b)(p) := \begin{cases} [] & \text{if } |a| = 0 \\ \bar{a} :: \bigcirc_{y \in c_\succ(\bar{a})} \mathtt{Rec}^{\rhd}_{f_{\bar{a},b,p}}(y) & \text{otherwise} \end{cases}$$

where $f : X \to X^{**} \to (Y \to X^*) \to X \to (X \to X^*) \to X^*$ is defined by

$$f_{x,u,p}(y)(q) := \begin{cases} u_i[y] & \text{if } y \preceq c_\rhd(x)_i \text{ for some } i < |c_\rhd(x)| \\ p(y, \odot_{z \in c_\rhd(y)} q(z)) & \text{otherwise} \end{cases}$$

and $d[y]$ is defined as in the statement of the lemma. Now, it is not difficult to see that since $\succ$ is primitive recursive and $c_\succ$ a branching modulus then $T_\succ(y,d)$ is a primitive recursive predicate, and thus so is $d[y]$ since this can be computed via a bounded search. Moreover, the case distinction in the definition of f is primitive recursively decidable, and so the functional as a whole is clearly definable in $\mathtt{T}_0$. It is obvious then that h is definable in $\mathtt{T}_0 + \mathtt{Rec}^{\rhd, X^*}$ and so

$$\Psi_\omega := \mathtt{BR}^{Y, X^*}_{\omega, g, h}$$

is definable in $\mathtt{T}_0 + \mathtt{Rec}^{\rhd, X^*} + \mathtt{BR}^{Y, X^*}$. To see that it satisfies the relevant equations is just a matter of unwinding definitions: We have $\Psi_\omega(a,b) = h(a,b)(\ldots) = []$ if

$|a| = 0$ and $\Psi_\omega(a,b) = g(a,b) = []$ if $\omega(a\hat{}b) < |a|$, and otherwise

$$\Psi_\omega(a,b) = \bar{a} :: \bigcirc_{y \in c_\succ(\bar{a})} R_{\omega,a,b}(y)$$

for $R_{\omega,a,b} := \mathtt{Rec}^{\triangleright}_{f_{\bar{a},\bar{b},p}}$ and $p := \lambda x, u.\Psi_\omega(a * x, b * u)$. But then

$$\begin{aligned} R_{\omega,a,b}(y) &= f_{\bar{a},\bar{b},p}(y)(\lambda z \lhd y \,.\, R_{\omega,a,b}(z)) \\ &= \begin{cases} \bar{b}_i[y] & \text{if } y \preceq c_\triangleright(\bar{a})_i \text{ for some } i < |c_\triangleright(x)| \\ p(y, \bigodot_{z \in c_\triangleright(y)} R_{\omega,a,b}(z)) & \text{otherwise} \end{cases} \end{aligned}$$

and in the second line

$$p(y, \bigodot_{z \in c_\triangleright(y)} R_{\omega,a,b}(z)) = \Psi_\omega(a * y, b * \bigodot_{z \in c_\triangleright(y)} R_{\omega,a,b}(z))$$

which completes the proof. □

For the purposes of our main proof, we now make a small assumption: That 0 encodes some object of X which is minimal w.r.t. $\lhd$, in other words $\neg(x \lhd 0)$ for all $x \in X$. In particular, this would imply that $M_{\succ,\triangleright}(0, 0_{X^*})$ since 0_{X^*} is assumed to be the empty sequence. While not strictly necessary, this assumption allows us to use the usual variant of bar recursion as above, which would otherwise need to be modified slightly.

Theorem 3.12. *Let $\succ$, $\triangleright$ and $\gg$ be primitive recursive binary relations, such that branching moduli for $\succ$ and $\triangleright$ are definable in $\mathtt{T}_0$, and suppose in addition that*

1. *$\triangleright$ is inductively wellfounded,*
2. *$\gg$ is a decomposition of $\succ$ with respect to $\triangleright$,*
3. *ω is a modulus of minimal wellfoundedness for $\gg$.*

Let Ψ be defined as in Lemma 3.10, and writing $Y := X \times X^{**}$ define $\Phi : (Y^{\mathbb{N}} \to \mathbb{N}) \to X \to X^*$ in $T_0 + \mathtt{Rec}^{\triangleright,X^*} + \mathtt{BR}^{Y,X^*}$ as

$$\Phi_\omega(x) = \Psi_\omega([x], [\bigodot_{y \in c_\triangleright(x)} \Phi_\omega(y)]).$$

Then Φ_ω is a derivation function for $\succ$.

Remark 3.13. It is instructive to compare the statement of Theorem 3.12 with that of the original abstract result Theorem 3.4. Each of the main assumption 1-3 of 3.4 are present in Theorem 3.12, with inductive wellfoundedness of $\triangleright$ also implicit in our use of the wellfounded recursor $\mathtt{Rec}^{\triangleright,X^*}$, and classical wellfoundedness of $\gg$ replaced by the existence of a modulus ω of minimal wellfoundedness here. Moreover, wellfoundedness of $\succ$ is represented here by providing a concrete derivation function. The reader is encouraged to consult the proof of Theorem 3.4 when reading the proof of Theorem 3.12 below, as the two are closely related.

Proof. We first claim that for any nonempty $(a, b) \in Y^*$ such that $M_{\succ,\rhd}(a_i, b_i)$ for all $i < |a|$ and $C_{\gg}(a)$, then

$$\neg T_{\succ}(\bar{a}, \Psi_\omega(a, b)) \to (\exists y \ll \bar{a}\ , \exists u \in X^{**})(\neg T_{\succ}(y, \Psi_\omega(a * y, b * u)) \wedge M_{\succ,\rhd}(y, u)). \quad (7)$$

To prove the claim, we begin by observing that $\omega(\widehat{a, b}) \geq |a|$. To see this, observe that $\hat{a}$ is a minimal sequence relative to $\hat{b}$, by our assumption that $M_{\succ,\rhd}(0, [])$ holds. Thus $\omega(\widehat{a, b}) < |a|$ would imply that there exist $i, i + 1 < |a|$ such that $a_i \gg a_{i+1}$, contradicting $C_{\gg}(a)$.

Therefore $\Psi_\omega(a, b) = \bar{a} :: \bigcirc_{y \in c_{\succ}(\bar{a})} R_{a,b}(y)$ and by Lemma 2.14 together with $\neg T_{\succ}(\bar{a}, \Psi_\omega(a, b))$ there exists some $y \prec \bar{a}$ such that $\neg T_{\succ}(y, R_{a,b}(y))$. Therefore the set

$$S_{a,b} := \{x \in X \ : \ x \prec \bar{a} \wedge \neg T_{\succ}(x, R_{a,b}(x))\}$$

is nonempty, and thus contains some y which is minimal with respect to $\rhd$.

Let $[z_0, \ldots, z_{k-1}] := c_{\rhd}(\bar{a})$. If $y \preceq z_i \lhd \bar{a}$ for some $i < k$, then since $M_{\succ,\rhd}(\bar{a}, \bar{b})$ and thus $T_{\succ}(z_i, \bar{b}_i)$ we would have $T_{\succ}(y, \bar{b}_i[y])$ (for $\bar{b}_i[y]$ defined as in the statement of Lemma 3.10) and thus $T_{\succ}(y, R_{a,b}(y))$, since in this case $R_{a,b}(y) = \bar{b}_i[y]$. This is a contradiction.

Therefore as before $y \ll \bar{a}$ by decomposition property (i). Now for $z \in c_{\rhd}(y)$, by property (ii) we have $z \prec \bar{a}$ and thus $T_{\succ}(z, R_{a,b}(z))$ by minimality of y. Therefore by Lemma 3.9, $u := \bigodot_{z \in c_{\rhd}(y)} R_{a,b}(z)$ satisfies $M_{\succ,\rhd}(y, u)$ and from $\neg T_{\succ}(y, R_{a,b}(y))$ we obtain $\neg T_{\succ}(y, \Psi_\omega(a * y, b * u))$ since in this case $R_{a,b}(y) = \Psi_\omega(a * y, b * u)$. This proves the claim.

Now suppose the theorem is false and take some minimal x such that $\neg T_{\succ}(x, \Phi_\omega(x))$. Then $M_{\succ,\rhd}(x, v)$ and $\neg T_{\succ}(x, \Psi_\omega([x], [v]))$ hold for $v := \bigodot_{y \in c_{\rhd}(x)} \Phi_\omega(y)$, and by dependent choice in conjunction with (7) we obtain a pair of sequences α, β such that $(\forall n) M_{\succ,\rhd}(\alpha_n, \beta_n)$ but $\alpha_n \gg \alpha_{n+1}$ for all n, contradicting the assumption that $\alpha_i \not\gg \alpha_{i+1}$ for some $i < \omega(\alpha, \beta)$. □

3.4 Primitive recursive bounds via closure results for bar recursion

Having extracted a bar recursive term which computes derivation trees for $\succ$, we can already apply a variety of closure results from the literature to obtain coarse upper bounds on the derivational height of $\succ$.

The term Ψ_ω in Lemma 3.10 is formally definable not just from bar recursion but from a *single instance* of $\mathtt{BR}^{X \times X^{**}, X^*}_{\omega, g, h}$ where g and h are definable in $\mathtt{T}_0 + \mathtt{wRec}^{\rhd, X^*}$. As a consequence, we can show that the derivational height of $\succ$ is bounded by some Gödel primitive recursive function whenever the modulus of minimal wellfoundedness is definable in System T. This follows directly from Schwichtenberg's classic result [23] that System T is closed under the rule of bar recursion, whenever bar recursion has sequence type level 0 or 1. A more fine-grained analysis is the following:

Corollary 3.14. *Suppose that $\succ$, $\rhd$ and $\gg$ satisfy the assumptions of Theorem 3.12, and that $\mathtt{Rec}^{\rhd, X^*}$ is definable in $\mathtt{T}_0$. Then*

(a) *whenever $\gg$ has a modulus of minimal wellfoundedness ω which is definable in $\mathtt{T}_i$, the derivational height of $\succ$ is bounded by some function in $\mathtt{T}_{i+3}$,*

(b) *in the special case where ω is definable in $\mathtt{T}_0$, the derivational height is bounded by some function in $\mathtt{T}_1$.*

Proof. Since X is coded in the natural numbers, both the sequence type $X \times X^{**}$ and the output type X^* can also be encoded in $\mathbb{N}$, and so the functional Ψ_ω is definable from a single instance $B_{\omega,g,h}$ of bar recursion of lowest type. By the recent analysis of Oliva and Steila [19], whenever the parameters g, h are in $\mathtt{T}_0$ and ω is in $\mathtt{T}_i$, the bar recursor $B_{\omega,g,h}$ can be defined in $\mathtt{T}_{i+3}$ (see [19, Corollary 3.5]). But then Ψ_ω and hence also Φ_ω are definable in $\mathtt{T}_{i+3}$, and since a derivational height function for $\succ$ is given by $\lambda x.|\Phi_\omega(x)|$, this gives us (a). Part (b) follows analogously using Howard's more refined result for lower types [13]. □

Corollary 3.14 is by no means exhaustive. For example, Howard's closure theorem [13] is extended to fragments of the Grzegorzyk hierarchy by Kreuzer [16], though it is unclear whether this would be applicable here, since these fragments do not have access to the full recursor of lowest type. Note that it could also be that a more carefully analysis of the particular form of bar recursion would likely lead to a significantly improved version of Corollary 3.14, although we leave this open for now.

All of this demonstrates how our approach of extracting concrete programs and then appealing to computability theory of those programs leads to extremely general complexity results. In the next section, we show that the situation improves further if we strengthen our hypothesis by replacing the modulus of minimal wellfoundedness by some explicit recursor.

3.5 Derivation functions for inductively wellfounded binary relations

We now demonstrate how Theorem 3.12 and the associated complexity bounds can be further improved if we take as a stronger premise that minimal sequences are inductively wellfounded with respect to some concrete binary relation $\blacktriangleright$ on $X \times X^{**}$. From a computational point of view, the idea here is that under certain assumptions, we are able to replace a modulus ω of minimal wellfoundedness (corresponding to classical wellfoundedness of $\gg$) with a wellfounded binary relation $\blacktriangleright$, thereby replacing the instance of bar recursion in Theorem 3.12 with the wellfounded recursor $\mathtt{Rec}^{\blacktriangleright}$.

Lemma 3.15. *Suppose that $\succ$, $\rhd$ and $\gg$ satisfy the assumptions of Theorem 3.12, and that $\blacktriangleright$ is an inductively wellfounded binary relation on $X \times X^{**}$ satisfying*

$$M_{\succ,\rhd}(x,u) \wedge M_{\succ,\rhd}(y,v) \wedge x \gg y \to (x,u) \blacktriangleright (y,v) \tag{8}$$

for all $(x,u),(y,v)$. Then there is a functional $\Gamma : X \times X^{**} \to X^*$ definable in $T_0 + \mathtt{Rec}^{\rhd,X^*} + \mathtt{Rec}^{\blacktriangleright,X^*}$ which satisfies

$$\Gamma(x,u) = x :: \bigcirc_{y \in c_\succ(x)} \tilde{R}_{x,u}(y)$$

where $\tilde{R}_{x,u} : X \to X^*$ *is given by*

$$\tilde{R}_{x,u}(y) := \begin{cases} u_i[y] & \text{if } y \preceq c_\triangleright(x)_i \text{ for some } i < |c_\triangleright(x)| \\ \Gamma(y,v) & \text{if } (x,u) \blacktriangleright (y,v) \\ [] & \text{otherwise} \end{cases}$$

for $v := \bigodot_{z \in c_\triangleright(y)} \tilde{R}_{x,u}(z)$*. Moreover, if* ω *is a modulus of minimal wellfoundedness for* $\gg$*, then for any* x, u *satisfying* $M_{\succ,\triangleright}(x,u)$ *we have*

$$(\forall a,b)(\forall i < |a|\ M_{\succ,\triangleright}(a_i,b_i) \wedge C_{\gg}(a * x) \to \Psi_\omega(a * x, b * u) = \Gamma(x,u)) \tag{9}$$

where Ψ_ω *is defined as in Lemma 3.10.*

Remark 3.16. That Γ is definable in T_0+$\mathtt{Rec}^{\triangleright,X^*}$+$\mathtt{Rec}^{\blacktriangleright,X^*}$ is just a simple adaptation of the proof of Lemma 3.10. To see informally that Γ is well-defined here using recursion over $\triangleright$ and $\blacktriangleright$, we argue that $\Gamma(x,u)$ is well-defined whenever $\Gamma(y,v)$ is well-defined for all $(x,u) \blacktriangleright (y,v)$. To this end, it suffices to show that $\tilde{R}_{x,u}(y)$ is well-defined for all $y \in c_\succ(x)$, which follows using an auxiliary induction over $\triangleright$: If $\tilde{R}_{x,y}(x)$ is well-defined for all $y \triangleright z$ then in particular we have $v := \bigodot_{z \in c_\triangleright(y)} \tilde{R}_{x,u}(z)$ is well-defined, and thus so is $\tilde{R}_{x,u}(y)$, which only calls $\Gamma(y,v)$ for $(x,u) \blacktriangleright (y,v)$.

Proof. We prove (9) by induction on $\blacktriangleright$, to which end we fix a, b and assume that $M_{\succ,\triangleright}(a_i,b_i)$ for all $i < |a|$ together with $C_{\gg}(a*x)$ and $M_{\succ,\triangleright}(x,u)$. Note that $C_{\gg}(a*x)$ implies that $\omega(a * \widehat{x, b} * u) \geq |a|+1$ and thus $\Psi_\omega(a*x, b*u) = x :: \bigcap_{y \in c_\succ(x)} R_{a*x,b*u}(y)$. So we're done if we can show that $R_{a*x,b*u}(y) = \tilde{R}_{x,u}(y)$ for all $y \prec x$, assuming that (9) holds for with (x,u) replaced by (x',u') for $(x,u) \blacktriangleright (x',u')$.

We do this by a side induction on $\triangleright$, so fix some y and assume that $R_{a*x,b*u}(z) = \tilde{R}_{x,u}(z)$ for all $z \lhd y$ with $z \prec x$. We only need to check the case $x \gg y$, where we aim to show that $(x,u) \blacktriangleright (y,v)$ for $v := \bigodot_{z \in c_\triangleright(y)} \tilde{R}_{x,u}(z) = \bigodot_{z \in c_\triangleright(y)} R_{a*x,b*u}(z)$. By the side induction hypothesis, if $y \triangleright z$ then also $x \succ z$ (since $x \gg y$), and thus $R_{a*x,b*u}(z) = \tilde{R}_{x,u}(z)$ and hence $M_{\succ,\triangleright}(y,v)$. Therefore by (8) together with $M_{\succ,\triangleright}(x,u)$ and $x \gg y$ it follows that $(x,u) \blacktriangleright (y,v)$, and since $C_{\gg}(a * x * y)$ we can apply the main induction hypothesis on (y,v) with $(a*x, b*y)$ for (a,b) to obtain

$$\tilde{R}_{x,u}(y) = \Gamma(y,v) = \Psi_\omega(a * x * y, b * u * v) = R_{a*x,b*u}(y)$$

by the main induction hypothesis. Thus eliminating the side induction hypothesis yields $R_{a*x,b*u}(y) = \tilde{R}_{x,u}(y)$ for all $y \prec x$ and thus $\Psi_\omega(a*x, b*u) = \Gamma(x,u)$, and so eliminating the main induction hypothesis we're done. □

Corollary 3.17. *Under the conditions of Lemma 3.15, the functional* Φ_ω *in Theorem 3.12 is definable from* Γ *and thus in* $\mathtt{T}_0 + \mathtt{Rec}^{\triangleright,X^*} + \mathtt{Rec}^{\blacktriangleright,X^*}$ *for any modulus of minimal wellfoundedness* ω*.*

Proof. More specifically, we define $\phi : X \to X^*$ using Γ and recursion over $\triangleright$ (and hence in $\mathtt{T}_0 + \mathtt{Rec}^{\triangleright,X^*} + \mathtt{Rec}^{\blacktriangleright,X^*}$) by

$$\phi(x) = \Gamma(x, \bigodot_{y \in c_\triangleright(x)} \phi(y))$$

Then we can prove that $\Phi_\omega(x) = \phi(x)$ using induction over $\triangleright$: For the induction step, if $\Phi_\omega(y) = \phi(y)$ for all $y \lhd x$ then in particular by Theorem 3.12 we have $T_\succ(y, \phi(y))$ for all $y \lhd x$ and thus $M_{\succ,\triangleright}(y, u)$ for

$$u = \bigodot_{y \in c_\triangleright(x)} \phi(y) = \bigodot_{y \in c_\triangleright(x)} \Phi_\omega(y)$$

and therefore by Lemma 3.15, applying (9) for $|a| = 0$ we have

$$\phi(x) = \Gamma(x, u) = \Psi_\omega([x], [\bigodot_{y \in c_\triangleright(x)} \Phi_\omega(y)]) = \Phi_\omega(x)$$

and we're done. □

We can now give a corresponding formulation of Corollary 3.14:

Corollary 3.18. *Suppose that $\succ$, $\triangleright$ and $\gg$ satisfy the assumptions of Lemma 3.15, and that $\mathtt{Rec}^{\triangleright,X^*}$ is definable in $\mathtt{T}_0$. Then whenever $\mathtt{Rec}^{\blacktriangleright,X^*}$ is definable in $\mathtt{T}_i$, the derivational height of $\succ$ is bounded by some function also definable in $\mathtt{T}_i$. In particular:*

(a) For $i = 0$, the derivational height is bounded by a primitive recursive function;

(b) For $i = 1$, the derivational height is bounded by a multiple recursive function.

Proof. First of all, from (8) and wellfoundedness of $\blacktriangleright$, one can infer that $\gg$ is wellfounded on minimal sequences. If not then there would be some α, β such that $M_{\succ,\triangleright}(\alpha_n, \beta_n)$ and $\alpha_n \gg \alpha_{n+1}$ for all $n \in \mathbb{N}$, which would imply that $(\alpha_n, \beta_n) \blacktriangleright (\alpha_{n+1}, \beta_{n+1})$ for all $n \in \mathbb{N}$, a contradiction. Therefore there exists some modulus of wellfoundedness ω, but by Corollary 3.17 the resulting derivation function Φ_ω for $\succ$ is definable from Γ and hence in $\mathtt{T}_0 + \mathtt{Rec}^{\triangleright,X^*} + \mathtt{Rec}^{\blacktriangleright,X^*}$, which under the assumptions on the definability of $\mathtt{Rec}^{\triangleright,X^*}$ and $\mathtt{Rec}^{\blacktriangleright,X^*}$ is ultimately definable in $\mathtt{T}_i$. □

4 Application: Path orders and term rewriting

We conclude by sketching how our abstract results, particularly Corollary 3.18, can be applied in the special case where X denotes a set of terms in some programming language, which we take here to be a simple term rewrite system. More specifically we show how the formalization of Buchholtz [6] can be incorporated into our framework. The difference here is that we work in a more abstract setting, and that we directly construct derivation functions in fragments of System T, rather than formalizing wellfoundedness proofs in fragments of Peano arithmetic.

4.1 $(X, \rhd)$ as a term structure

Let X now be instantiated as the set of terms ranging over some countable set of variables and some finite signature $\{f_1, \ldots, f_k\}$, where we assume for simplicity that each f_i has a fixed arity (note that this latter restriction is not essential: see [6, Section 3]). Clearly X can be arithmetized, and following [6] we can assign each term a size as follows:

(i) $|x_i| := i$;

(ii) $|f_j(t_1, \ldots, t_n)| := \max\{n, |t_1|, \ldots, |t_n|\} + 1$.

Note that this definition of size ensures that there exists some monotone primitive recursive function h such that $|t| \leq t < h(|t|)$ for all t. Let $\rhd$ denote the immediate subterm relation: in other words, $f(t_1, \ldots, t_n) \rhd t_i$ for all $i = 1, \ldots, n$. Then by (ii) above we have $s \lhd t$ implies $|s| < |t|$, and so recursion over $\rhd$ is definable from the usual Gödel recursor over $>$. In particular, $\mathtt{Rec}^{\rhd, X^*}$ is definable in $\mathtt{T}_0$.

4.2 Approximations to recursive path orders

In general, recursive path orders on terms, such as the multiset or lexicographic path orders, can be characterized in the abstract as follows: We set $t = f(t_1, \ldots, t_n) \succ s$ if either

(a) $t_i \succeq s$ for some $i = 1, \ldots, n$;

(b) $t \succ_0 s$ and $t \succ s_i$ for all subterms s_i of s,

where typically $\succ_0$ is recursively defined in terms of $\succ$ itself (we will see a concrete example of all this in Section 4.3). Note that by (a), $\succ$ contains the subterm relation, which means that it is a *simplification order*. Condition (b) is closely related to the notion of a *lifting* as studied in e.g. [9]. In any case, such an order is a decomposition in the sense of our Definition 3.3 relative to $\gg$, where $t \gg s$ denotes the second case (b) above. This is because if $t \gg s$ then in particular $t \succ s_i$ for all $s_i \lhd s$.

Recursive path orders of this kind are fundamental tools in the theory of term rewriting, as they provide us with a criterion for checking if finitely defined term rewrite system $\mathcal{R}$ is terminating. Here we would work with orders $\succ$ which are closed under contexts and substitutions, and then whenever the rules $l \to r$ of $\mathcal{R}$ satisfy $l \succ r$, then $\mathcal{R}$ is guaranteed to be terminating. The main challenge is always to show that $\succ$ itself is wellfounded.

When it comes to computing complexity bounds, the first issue is that in general, recursive path orders are not finitely branching, and as a result proofs of wellfoundedness tend to use rather heavy proof theoretic machinery such as Kruskal's theorem. However, this is overcome by Buchholz in [6] by considering finitary variants of the usual path orders, whose wellfoundedness can be proven in low fragments of arithmetic.

One can describe Buchholz' idea in a slightly more general form as follows: for some primitive recursive function $b : \mathbb{N} \to \mathbb{N}$ define the bounded b-approximation $\succ_b$ of $\succ$ by $t = f(t_1, \ldots, t_n) \succ_b s$ if $b(|t|) \geq |s|$ and either of the following hold:

(a) $t_i \succeq_b s$ for some $i = 1, \ldots, n$;

(b) $t \gg_b s$ and $t \succ_b s_i$ for all subterms s_i of s,

where now $\gg_b$ is recursively defined in terms of $\succ_b$. Not only is $\succ_b$ now by definition finitely branching, but assuming that $\succ_b$ is a primitive recursive relation, as it invariably is, then $\succ_b$ is computably finitely branching: Because $t \succ_b s$ only if $b(|t|) \geq |s|$ and hence $h(b(|t|)) \geq h(|s|) > s$, and we can therefore take the branching function $c_\succ(t) \in X^*$ to be the primitive recursively definable sequence consisting of exactly those terms $s \leq h(b(|t|))$ satisfying $s \prec_b t$.

The crux of the idea is the following: For any finite rewrite system $\mathcal{R}$ reducing under some $\succ$, we can typically compute some b such that $\mathcal{R}$ is reducing under $\succ_b$. In other words, for any *fixed* $\mathcal{R}$ we can find a finitely branching approximation $\succ_b$ of $\succ$ sufficient for proving wellfoundedness of $\mathcal{R}$. Then the derivational height of $\mathcal{R}$ is bounded by the derivational height of $\succ_b$.

In this case, our complexity results in Corollaries 3.14 and 3.18 provide us with a means of bounding the derivational height of rewrite systems, and the generality of our results suggest that they are indeed applicable to a wide range of different path orders. We finish by sketching a simple example.

4.3 Example: the multiset path order

We now show how the well known primitive recursive bound on the complexity of rewrite systems terminating under the multiset path order can be reobtained in our setting. Actually, for simplicity we take a restricted form of the product path order: Though we can deal with the full multiset path order, it is slightly more technical and here our priority is to simply provide an illustration of our abstract result.

Our simple variant of the multiset path order is obtained by instantiating $\gg$ as $t = f(t_1, \ldots, t_n) \gg s$ if

- $s = g(s_1, \ldots, s_m)$ with $f >_F g$ and $t \succ s_i$ for all i, or
- $s = f(s_1, \ldots, s_n)$ and $t \succ s_i$ for all i and $t_i \succ s_i$ for some $i = 1, \ldots, n$ and $s_j = t_j$ for all $j \neq i$,

where $>_F$ is some wellfounded binary relation on function symbols. It turns out that any rewrite system reducing under the multiset path order is also reducing under the approximate order $\succ_k$ in which the bounding function b is simply $b(n) = n + k$, and k is some sufficiently large number which can be effectively computed from the rules of the rewrite system $\mathcal{R}$ (more specifically, following [6] we can take $k = \max\{ |r| \mid l \mapsto r \in \mathcal{R}\}$).

Given in full, then, the approximate multiset path order $\succ_k$ is defined as follows: $t = f(t_1, \ldots, t_n) \succ_k s$ iff $k + |t| \geq |s|$ and either

(a) $t_i \succeq_k s$ for some $i = 1, \ldots, n$;

(b) $f(t_1, \ldots, t_n) \gg_k s$

where $f(t_1, \ldots, t_n) \gg_k s$ iff[1]

(i) $s = g(s_1, \ldots, s_m)$ with $f >_F g$ and $t \succ_k s_i$ for all i;

(ii) $s = f(s_1, \ldots, s_n)$ and $t \succ_k s_i$ for all i and $t_i \succ_k s_i$ for some i and $s_j = t_j$ for all $j \neq i$.

Now, define the relation $\blacktriangleright_k$ on $X \times X^{**}$ as follows: $(f(t_1, \ldots, t_n), u) \blacktriangleright_k (g(s_1, \ldots, s_m), v)$ iff

$$f >_F g \quad \text{or} \quad f = g \wedge (\exists i)(u_i \supset v_i \wedge (\forall j \neq i)(u_j \supseteq v_j)).$$

where $(\supset)\ \supseteq$ denotes the (strict) inclusion order on lists. It is easy to see that if $M_{\succ,\rhd}(t, u) \wedge M_{\succ,\rhd}(s, v) \wedge t \gg_k s$ then $(t, u) \blacktriangleright_k (s, v)$: In the case that (i) holds then $f >_T g$ so this is clearly true, while if (ii) holds there is some i such that $t_i \succ_k s_i$ but $t_j = s_j$ otherwise. But $M_{\succ,\rhd}(t, u)$ implies that $T(t_i, u_i)$, and analogously $M_{\succ,\rhd}(s, v)$ implies $T(s_i, v_i)$, and so $t_i \succ_k s_i$ implies that v_i is a subsequence of u_i. Similarly, we must have $u_j = v_j$ otherwise.

Not only is $\blacktriangleright_k$ primitive recursive, but it is not difficult to show that $\mathtt{Rec}^{\blacktriangleright_k, X^*}$ is definable in $\mathtt{T}_0$: This is just a bounded recursion in the first component, while in the second component we can find an encoding of T^{**} into $\mathbb{N}$ such that $(\exists i)(u_i \supset v_i \wedge (\forall j \neq i)(u_j \supseteq v_j))$ implies that $u > v$.

Therefore by Corollary 3.18, the derivational height of $\succ_k$ is bounded by a primitive recursive function, and therefore the same is true for any rewrite system $\mathcal{R}$ reducing under $\succ$. Note that this primitive recursive bound on the derivation height can be easily converted to one in terms of a bound on $|t|$: if $|t| \leq n$ then $t < h(|t|) \leq h(n)$ by monotonicity of h, and since the latter is also primitive recursive, we can primitive recursively compute the maximum of the derivation height for all $t < h(n)$.

Both the multiset and lexicographic path orders are studied in more detail by the author together with Georg Moser in [18], where a more detailed construction of the derivational height functions is given than our brief sketch here. However, the main results of this paper constitute a considerable generalisation of [18].

5 Conclusion

The main result of this paper was a computational analysis of the wellfoundedness of abstract path orders, which in particular subsumes the usual recursive path orders encountered in the term rewriting literature. A such, the paper is a contribution to the proof theoretic analysis of termination, which has seen a

[1] Note that in (ii) the condition $t \succ_k s_i$ is actually redundant since $t_i \succeq_k s_i$ and thus we can infer $t \succ_k s_i$ from clause (a).

number of recent developments [3, 4, 10, 18]. As a consequence of our theoretical work, we provide a series of metatheorems which allow us to relate the complexity of a wellfounded order to some subrecurive system of functionals. As far as applications are concerned, while we only sketched an illustration of this in Section 4, we believe that the formal extraction of programs from termination proofs has a great deal of potential in providing upper bounds on the complexity of programs, and in this article hope to have provided a promising first step in this direction.

An obvious direction for future research is to use the techniques presented here to obtain new bounds and metatheorems for the complexity of concrete termination orders. While we mentioned the well-known recursive path orders as a simple example of where Corollary 3.18 could be applied, of particular interest would be the analysis of termination orders for which an upper bound on the induced derivational height is not known. These might include, for example, general orderings arising from the recent unifying work of Yamada et al. [31], or extensions of the multiset path order considered in [5][2].

Most termination orders in the literature work on sets of first order terms. However, up to the very final section we do not assume anything about the structure of X, and it would be interesting to find out whether our termination arguments can be applied to more interesting structures. In particular, Goubault-Larrecq [11] considers wellfounded orders on graphs, automata and higher-order functionals, and it would be intriguing to see whether any meaningful complexity results could be obtained in this context.

In our approach, we establish complexity bounds by extracting higher-order recursive programs in some subrecursive calculus of functionals, and looking at the type 1 functions definable in these calculi. A number of similar proof theoretic investigations of path orders and abstract notions of termination exist in the literature, notably those due to Weiermann [27, 29] which are based on an intricate ordinal analysis. It would be instructive to make more precise how our framework based on variants of bar recursion compares to his.

Finally, as briefly mentioned in Section 3.4, it would be interesting to formally establish a set of closure properties along the lines of [8, 19, 23] for *finitely branching* bar recursion, which would give a direct correspondence between the subrecursive strength of bar recursors and the derivational height of abstract orders.

Acknowledgements. First of all, I am very grateful to the two anonymous referees for their many detailed comments and corrections, which much improved the paper. I am indebted to Georg Moser for suggesting to me a proof theoretic study of termination principles in the first place, and in particular for pointing out that the results of [6] can be viewed in a more abstract way. Sam Sanders read an earlier draft of this paper, and I thank him for his suggestions. This work was partially supported by the Austrian Science Fund (FWF) project

[2] the path orders in [5] were suggested to me by one of the referees as a potential application of the complexity results here

P 25781-N15.

References

[1] M. Avanzini and G. Moser. Polynomial path orders. *Logical Methods in Computer Science*, 9(4), 2013.

[2] J. Avigad and S. Feferman. Gödel's functional ("Dialectica") interpretation. In S. R. Buss, editor, *Handbook of Proof Theory*, volume 137 of *Studies in Logic and the Foundations of Mathematics*, pages 337–405. North Holland, Amsterdam, 1998.

[3] S. Berardi, P. Oliva, and S. Steila. An analysis of the Podelski-Rybalchenko termination theorem via bar recursion. *Journal of Logic and Computation*, 29(4):555–575, 2019.

[4] S. Berardi and S. Steila. An intuitionistic version of Ramsey's Theorem and its use in program termination. *Annals of Pure and Applied Logic*, 166(12):1382–1406, 2015.

[5] G. Bonfante, J.-Y. Marion, and J.-Y. Moyen. Quasi-interpretations: A way to control resources. *Theoretical Computer Science*, 412(25):2776–2796, 2011.

[6] W. Buchholz. Proof-theoretic analysis of termination proofs. *Annals of Pure and Applied Logic*, 75:57–65, 1995.

[7] N. Dershowitz. Orderings for term rewriting systems. *Theoretical Computer Science*, 17(3):279–301, 1982.

[8] M. Escardó, P. Oliva, and T. Powell. System T and the product of selection functions. In *Proceedings of Computer Science Logic (CSL '11)*, volume 12 of *LIPIcs*, pages 233–247, 2011.

[9] M. C. F. Ferreira and H. Zantema. Well-foundedness of term orderings. In N. Dershowitz, editor, *Conditional Term Rewriting Systems (CTRS '94)*, volume 968 of *LNCS*, pages 106–123, 1995.

[10] E. Frittaion, S. Steila, and K. Yokoyama. The strength fo the SCT criterion. In *Proceedings of TAMC '17*, volume 10185 of *LNCS*, pages 260–273. Springer, 2017.

[11] J. Goubault-Larrecq. Well-founded recursive relations. In *Computer Science Logic (CSL'01)*, volume 2142 of *LNCS*, pages 484–498, 2001.

[12] D. Hofbauer. Termination proofs by multiset path orderings imply primitive recursive derivation lengths. *Theoretical Computer Science*, 105(1):129–140, 1992.

[13] W. A. Howard. Ordinal analysis of bar recursion of type zero. *Compositio Mathematica*, 42:105–119, 1981.

[14] U. Kohlenbach. *Applied Proof Theory: Proof Interpretations and their Use in Mathematics*. Monographs in Mathematics. Springer, 2008.

[15] G. Kreisel. A Survey of Proof Theory II. *Studies in Logic and the Foundations of Mathematics*, 63:109–170, 1971.

[16] A. Kreuzer. Primitive recursion and the chain antichain principle. *Notre Dame Journal of Formal Logic*, 53(2):245–265, 2012.

[17] C. S. Lee, N. D. Jones, and A. M. Ben-Amran. The size-change principle for program termination. In *Proceedings of POPL'01*, volume 36 of *ACM SIGPLAN Notices*, pages 81–92, 2001.

[18] G. Moser and T. Powell. On the computational content of termination proofs. In *Proceedings of Computability in Europe (CiE 2015)*, volume 9136 of *LNCS*, pages 276–285, 2015.

[19] P. Oliva and S. Steila. A direct proof of Schwichtenberg's bar recursion closure theorem. *Journal of Symbolic Logic*, 83(1):70–83, 2018.

[20] A. Podelski and A. Rybalchenko. Transition invariants. In *Proceedings of Logic in Computer Science (LICS 2004)*, pages 32–41. IEEE Press, 2004.

[21] T. Powell. The equivalence of bar recursion and open recursion. *Annals of Pure and Applied Logic*, 165(11):1727–1754, 2014.

[22] T. Powell. Well quasi-orders and the functional interpretation. In P. Schuster, M. Seisenberger, and A. Weiermann, editors, *Well-Quasi Orders in Computation, Logic, Language and Reasoning*, volume 53 of *Trends in Logic*, pages 221–269. Springer, 2020.

[23] H. Schwichtenberg. On bar recursion of types 0 and 1. *The Journal of Symbolic Logic*, 44:325–329, 1979.

[24] M. Seisenberger. *On the Constructive Content of Proofs*. PhD thesis, Ludwig Maximilians Universität München, 2003.

[25] C. Spector. Provably recursive functionals of analysis: a consistency proof of analysis by an extension of principles in current intuitionistic mathematics. In F. D. E. Dekker, editor, *Recursive Function Theory: Proc. Symposia in Pure Mathematics*, volume 5, pages 1–27. American Mathematical Society, Providence, Rhode Island, 1962.

[26] A. S. Troelstra. *Metamathematical Investigation of Intuitionistic Arithmetic and Analysis*, volume 344 of *Lecture Notes in Mathematics*. Springer, Berlin, 1973.

[27] A. Weiermann. Complexity bounds for some finite forms of Kruskal's theorem. *Journal of Symbolic Computation*, 18:463–488, 1994.

[28] A. Weiermann. Termination proofs with lexicographic path orderings imply multiply recursive derivation. *Theoretical Computer Science*, 139:355–362, 1995.

[29] A. Weiermann. Bounding derivation lengths with functions from the slow growing hierarchy. *Archive for Mathematical Logic*, 37:427–441, 1998.

[30] A. Weiermann. How is it that infinitary methods can be applied to finitary mathematics? Gödel's T: A case study. *Journal of Symbolic Logic*, 63(4):1348–1370, 1998.

[31] A. Yamada, K. Keiichirou, and T. Sakabe. A unified orderings for termination proving. *Science of Computer Programming*, 111:110–134, 2015.