An Abstract Framework for the Analysis of Cyclic Derivations

MSc Thesis *(Afstudeerscriptie)*

written by

Dominik Wehr
(born 9th of February 1998 in Edinburgh, Scotland)

under the supervision of **Bahareh Afshari**, and submitted to the Examinations Board in partial fulfillment of the requirements for the degree of

MSc in Logic

at the *Universiteit van Amsterdam.*

Date of the public defense:	**Members of the Thesis Committee:**
26th of August 2021	Maria Aloni (Chair)
	Bahareh Afshari (Supervisor)
	Benno van den Berg
	Benedikt Löwe
	Alex Simpson

Institute for Logic, Language and Computation

Abstract

Cyclic derivations are finite graphs in which adjacent nodes are labeled by sequents according to locally sound derivation rules. These graphs serve as a finite representation of the infinite derivation trees obtained by unraveling them. To be considered proofs, such derivations need to satisfy an additional soundness condition called the global trace condition. Intuitively, any given cyclic derivation system can be separated into two different aspects: (i) its logical content, captured by the derivation rules and by how the global trace condition relates to the sequents being derived, and (ii) its cyclic content, which is concerned with the combinatorial aspects of the global trace condition and which kinds of graphs can serve as derivations. When comparing cyclic derivation systems, even ones with wildly different logical content, one often observes that their cyclic content is very similar. A general theory of the cyclic content of cyclic derivation systems could prove useful when designing, studying and comparing concrete cyclic derivation systems.

In this thesis, we formalize the cyclic content described above in an abstract manner easily applicable to most cyclic derivation systems. We do this by abstracting infinite branches through derivations to infinite sequences of morphisms through trace categories, categories which are equipped with a condition on such sequences of morphisms which is invariant under certain transformations. Cyclic derivations can then be represented as functors mapping cyclic graphs into trace categories. Notably, this representation discards all of the logical content of a cyclic derivation. Indeed, a handful of different trace categories are sufficient to capture the cyclic content of all cyclic derivation systems in the literature we have examined so far.

We demonstrate the viability of these abstract notions of cyclic content in two ways: First, we use the framework to derive uniform proofs of properties of cyclic derivation systems. Among them the regularization theorem for infinite proofs in finite derivation systems and a well-known application of Ramsey's theorem which reduces proof checking to the examination of certain periodic paths. Second, we demonstrate that various cyclic derivation systems from the literature fit within our framework, allowing the uniform proofs to be applied to them.

Acknowledgements

First and foremost, I want to thank Bahareh Afshari for agreeing to supervise this thesis and for giving me so much control over its topic and direction. Working with her is truly inspiring, not the least because of her deep knowledge of cyclic proof theory and her ability to always pick out the right questions and references to drive the research forward. I specifically want to thank her for all of her encouragement and infectious enthusiasm in times when I felt stuck. Doing research together with her has always been a pleasure and I am looking forward to our continued collaboration.

Furthermore, I owe an unfathomable amount of gratitude to my family, who have always supported and encouraged me in my academic endeavors. Specifically, I want to thank my parents for both graciously funding my studies in Amsterdam and later allowing me to stay with them while writing my thesis during the lockdown. I am thankful for my father's willingness to listen to me ramble about my research and his valuable feedback and pointers concerning the category theory in this thesis. I thank my mother for all of her encouragement and for always letting me tag along when I was looking for diversion. I thank my sister for never failing to making me laugh, especially for all of her efforts to lift my spirits during the final weeks before the deadline.

My thanks also go out to everyone I met during my time at the ILLC. Specifically, I want to thank Bahareh Afshari, Benno van den Berg and Robert Passman for each supervising a research project I undertook as part of the Master of Logic. I also want to thank Benno van den Berg, Nick Bezhanishvili, Benedikt Löwe and Yde Venema for all of the beautiful lectures they so meticulously prepared.

I also want to, once again, thank the people at the Programming Systems Lab at Saarland University, where I spent much time while obtaining my undergraduate degree. Leaving for another university has made me appreciate once more not only the wealth of technical knowledge but also the lessons of academic life they have imparted on me. Among them, I want to thank Dominik Kirst specifically for his insightful feedback on an earlier draft of this thesis and for supervising a Master of Logic research project.

Lastly, I want to thank Maria Aloni, Benno van den Berg, Benedikt Löwe and Alex Simpson for agreeing to be on my thesis committee and for their valuable feedback on this thesis.

Contents

Chapter 1

Introduction

Cyclic derivations are rooted, cyclic graphs whose nodes are labeled with sequents according to locally sound derivation rules. While a cyclic derivation is a finite object, unfolding its cycles yields an infinite derivation tree, if at least one such cycle is present in the graph. The fact that each derivation rule is locally sound in such an infinite derivation, and by extension in a cyclic derivation, is not sufficient to guarantee that it embodies a valid argument. This is because the soundness of its infinite branches cannot be reduced to the validity of axioms. For a cyclic derivation to be a proof, it thus needs to satisfy an additional soundness condition called the global trace condition (GTC). While the details vary from system to system, the GTC generally requires every infinite branch through the derivation to satisfy some desirable property. Usually, this "desirable property" is some event taking place infinitely often in a trace along the branch, such as some object decreasing according to a well-order or some fixed-point being unfolded.

Cyclic derivation systems exhibit useful properties which set them apart from more traditional derivation systems based on finite trees. For example, if there already exists an infinite game semantics for a logic, it can often be "transformed" into a cyclic proof system quite naturally because the winning conditions of infinite games are usually structurally similar to GTCs. Examples of systems derived in this manner are the cyclic tableaux for the modal μ-calculus [28] and the cyclic proof system for the higher-order fixed-point logic with natural numbers [20]. When comparing the cyclic and finite tree-based proof systems in logics for which both are known, such as Peano arithmetic [33] or the modal μ-calculus [21, 28], the cyclic derivation systems usually require fewer axioms, often omitting some of the more complex axioms, such as induction schemata or fixed-point rules. This difference makes cyclic proof systems very attractive for use as underlying derivation systems of automated theorem provers (ATPs), programs designed to find proofs of logical statements without further help or interaction from the user. When designing an ATP, determining how to "guess" the right place and instantiation of complex axioms during proof search often is a major point of difficulty. By using a cyclic derivation system without such axioms, this issue can often be sidestepped completely, although at the cost of having to perform an often computationally difficult GTC satisfaction check on any candidate derivation. Indeed, there exist multiple automated theorem provers using cyclic derivation systems [5, 6, 27, 38] and current investigations such as that of Stratulat [37] promise to further reduce the cost currently imposed by the GTC checks during proof searches, making cyclic derivation systems a promising tool for designing future ATPs.

Cyclic proofs have also proven useful in carrying out proof-theoretic investigations. As a recent example, cut-free cyclic proof systems have been applied to derive various interpolation properties in a proof-theoretic manner, such as in [2, 24, 31]. Their suitability to this task again stems from the absence of some of the more complex rules whose application can disturb the proof-theoretic approach to interpolation (see e.g. [7]).

While cyclic derivations lend themselves to a broad range of different logics, they nonetheless tend to have some features in common, such as reasoning about inductively defined structures [4, 20, 33], the presence of fixed-point quantifiers [10, 20, 28, 34] and proofs generating (co-)recursively defined functions [9, 12, 30]. Even when comparing wildly different logics, their cyclic derivation systems themselves often have a common structure. This similarity extends to logical properties desired of cyclic derivation systems: Apart from soundness (and possibly completeness) the most important property of a cyclic derivation system is the decidability of whether a sequent-annotated cyclic graph constitutes a proof. While deciding whether it is a cyclic derivation is quite easy (simply check whether each rule is applied correctly), deciding whether it satisfies the GTC tends to be more complex. Most papers on cyclic derivation systems employ a variant of the GTC decision procedure based on Büchi automata inclusion first put forward by Sprenger and Dam [34] (for examples of this, see e.g. [9, 20, 33]). While targeting very different logics, the automata constructed during this decision procedure tend to be very similar from system to system. These observations suggest that any given cyclic derivation system can be separated into two different aspects: (i) its logical content, captured by the derivation rules and by how the global trace condition relates to the sequents being derived, and (ii) its cyclic content, which is concerned with the combinatorial properties of the global trace condition and which kinds of graphs can serve as derivations. Two systems of wildly different logical content may thus be very similar in their trace content.

1.1 Contributions

In this thesis, we develop an abstract, categorical framework capturing the trace content of cyclic derivation systems. We represent infinite paths through derivations by infinite sequences of morphisms through trace categories, categories which come equipped with a trace condition on such paths which is invariant under certain transformations. Cyclic derivations are then abstracted into the functorial images of finite, rooted, cyclic graphs into a trace category. Notably, this approach discards virtually all logical content of cyclic derivation systems, allowing cyclic proofs for many different logics to be represented in terms of the same trace category.

We demonstrate the viability of this framework in two ways:

1. We state theorems about cyclic derivation systems in terms of our categorical framework and prove them in a uniform manner;
2. We demonstrate that the cyclic content of various cyclic derivation systems from the literature is captured by our framework.

The abstract theorems of point 1 can then be specialized to the systems considered under point 2 to obtain the theorems specific to those systems.

We believe that such a general notion of trace condition and cyclic proof can be applied to multiple lines of investigation:

(a) *Unifying* the proofs of commonly reproven theorems about cyclic derivation systems by carrying them out within the categorical framework. Besides making the arguments clearer by divorcing them from specific logical details, these general theorems can then directly be applied to newly designed cyclic derivation systems, saving their designers the trouble of having to reprove the theorems once again.

(b) *Comparing* the cyclic content of cyclic derivation systems, highlighting the similarities and differences between their global trace conditions by investigating in which trace categories they can be expressed.

(c) *Asking and Answering* new questions about the general properties of cyclic derivation systems in a manner which applies to many such systems at once.

(d) *Generalizing* results which are currently tied to specific cyclic derivation systems by viewing them through the lens of the categorical framework, allowing them to be easily transferred to many other cyclic derivation systems.

Within this thesis, we present results belonging to points (a) through (c). Notably, we derive a novel GTC decision procedure which does not make use of automata theory and which arises quite naturally from the notions of our abstract framework. Examples of some problems suitable to inquiry along the line of (d) are given as future work in Section 7.2.

Lastly, we note that ours is not the first proposed abstract notion of trace condition. A system with similar underlying ideas and goals has previously been put forward by Brotherston in his dissertation [4]. Brotherston's approach is purely combinatorial while we employ a category theoretic framework. Nonetheless, there are both significant overlaps and differences between our approaches. For example, while his abstract notion of trace and our $\mathcal{A}$-activated trace categories are of equivalent expressivity, our more general notion of trace categories has no analogue in his work. We postpone a detailed comparison between his framework and ours to Section 7.1 since it requires referencing of a lot of the more technical definitions and results laid out within this thesis. However, it should be noted that all definitions and results concerning our framework have been derived independently of those of Brotherston since we only became aware of his work during the late writing stage of this thesis.

1.2 Outline

We close this chapter in Section 1.3 by laying out the cyclic derivation system for Peano arithmetic to serve as a running example for later chapters. After this, the thesis is structured as follows:

In Chapter 2 we introduce the concepts underpinning the rest of the thesis. Section 2.1 defines *trace categories* (Definition 2.7) and *abstract cyclic derivations* (ACDs) (Definition 2.12), which separate the "trace content" of cyclic derivations from the specific logic they embody. Section 2.2 gives two procedures for transforming abstract cyclic derivations into simpler, equivalent ACDs.

Chapter 3 gives a categorical mechanism for specifying the transformation procedure which converts the cyclic derivations of a given logic into ACDs. For this, Section 3.1 provides an account of infinite trees constructed according to certain rule sets, a slightly

more general rendering of infinite derivation trees. We then introduce *categories of derivations* (Definition 3.12) which are a categorical reformulation of the ideas form Section 3.1. Based on categories of derivations, we define *trace-interpretations* (Definition 3.24), functors which annotate derivations with trace information. These allow us to formalize the distinction between ∞-derivations and ∞-proofs categorically. The chapter closes with Section 3.4 in which we demonstrate how a trace-interpretation induces a transformation from cyclic derivations of a logic to ACDs.

In Chapter 4 we prove general theorems about cyclic derivations using the abstract notions from Chapters 2 and 3. The results in Section 4.1 are obtained via automata-theoretic methods. We prove that the global trace condition of abstract cyclic derivations over certain trace conditions is decidable (Theorem 4.4) and that in certain finite derivation systems, all ∞-proofs can be regularized (Theorem 4.6). The results in Section 4.2 are based on Ramsey's theorem. We identify an alternative soundness condition for abstract cyclic derivations (Lemma 4.13) which is equivalent to the global trace condition and derive from it a novel GTC decision algorithm which does not use automata (Theorem 4.14).

Chapter 5 is concerned with giving and comparing concrete examples of trace categories. In Section 5.1 we introduce *activation algebras* (Definition 5.1) and their induced trace categories (Definition 5.2), which yield a diverse range of trace categories. We then prove for which kinds of activation algebras the induced trace categories exhibit the properties required for results from Chapters 3 and 4. In Section 5.2 we analyze and compare activation algebras and their associated trace categories via *activation-respecting homomorphisms* (Definition 5.18). We close the chapter in Section 5.3 by investigating which properties of trace categories are inherited by their subcategories, applying these results to the subcategory of injective trace maps of a given $\mathcal{A}$-activated trace category.

In Chapter 6 we demonstrate how a selection of cyclic derivation systems from the literature fits within the abstract framework outlined in the previous chapters and deduce some of their properties as corollaries of results from Chapter 4.

We close the thesis in Chapter 7. We discuss related work in Section 7.1, including a detailed comparison between our framework and that put forward in Brotherston's dissertation [4]. In Section 7.2 we lay out some questions raised by our work and point out a number of problems suitable to investigations using our formalism.

Category Theory Prerequisites Most definitions and theorems of this thesis are presented using ideas from category theory. We assume that the reader is already familiar with the basic concepts of category theory, including categories, functors and natural transformations. Readers unfamiliar with such matters are recommended to first consult introductory works on category theory, such as [23]. All of the less common concepts of category theory we employ, such as semi-categories and monoidal categories, are defined in Appendix A.

1.3 Running Example: Cyclic Arithmetic

This thesis deals mostly in abstract definitions and theorems. It is thus useful to have at hand a concrete cyclic derivation system to refer back to in examples. The system we have chosen for this purpose is the cyclic derivation system for Peano arithmetic

first put forward by Simpson [33], commonly referred to as "cyclic arithmetic" (CA) in the literature. We begin by recalling the syntax and semantics of first-order arithmetic.

Definition 1.1. The *terms and formulas of first-order arithmetic* are specified by the grammar given below:

$$s, t \in \textsc{Term} ::= x \mid 0 \mid S\,s \mid s + t \mid s \cdot t$$
$$\varphi, \psi \in \textsc{Form} ::= s = t \mid s < t \mid \bot \mid \varphi \wedge \psi \mid \varphi \vee \psi \mid \varphi \to \psi \mid \forall x.\varphi \mid \exists x.\varphi$$

Definition 1.2. For assignments $\rho : \textsc{Var} \to \omega$ we define a *term interpretation* by

$$0^\rho := 0 \qquad x^\rho := \rho(x) \qquad (S\,t)^\rho := t^\rho + 1 \qquad (s+t)^\rho := s^\rho + t^\rho \qquad (s \cdot t)^\rho := s^\rho \cdot t^\rho$$

and *satisfaction in the standard model* $\rho \vDash \varphi$

$\rho \vDash s = t$ holds iff $s^\rho = t^\rho$	$\rho \vDash s < t$ holds iff $s^\rho < t^\rho$
$\rho \vDash \varphi \wedge \psi$ holds iff $\rho \vDash \varphi$ and $\rho \vDash \psi$	$\rho \vDash \varphi \vee \psi$ holds iff $\rho \vDash \varphi$ or $\rho \vDash \psi$
$\rho \vDash \forall x.\varphi$ holds iff $\rho[x \mapsto n] \vDash \varphi$ for all $n \in \omega$	$\rho \vDash \varphi \to \psi$ holds iff $\rho \vDash \varphi$ entails $\rho \vDash \psi$
$\rho \vDash \exists x.\varphi$ holds iff $\rho[x \mapsto n] \vDash \varphi$ for some $n \in \omega$	$\rho \vDash \bot$ holds never

We write $\rho \vDash (\Gamma \Rightarrow \Delta)$ if there is $\varphi \in \Gamma$ such that $\rho \nvDash \varphi$ or there is $\varphi \in \Delta$ such that $\rho \vDash \varphi$. We write $\vDash (\Gamma \Rightarrow \Delta)$ if $\rho \vDash (\Gamma \Rightarrow \Delta)$ for all assignments $\rho : \textsc{Var} \to \omega$.

The derivation system of cyclic arithmetic (Definition 1.3) is mostly standard, consisting of the usual sequent rules for classical first-order logic (including the Cut-rule) and the axioms of Robinson arithmetic, all arithmetical axioms of Peano arithmetic save for the scheme of induction. The absence of the induction principle is a noteworthy example of the pattern of complex axiom schemata being omitted from cyclic variants of proof systems. The system furthermore contains two rather unusual rules: The Sub-rule allows for some occurrences of a term t to be replaced by a fresh variable x, an operation admissible in most derivation systems. The second rule, called Repeat, is more curious: If it is applied at a node labeled with $\Gamma \Rightarrow \Delta$, any occurrence of $\Gamma \Rightarrow \Delta$ above the point of application in the derivation may be treated as an axiom. The purpose of the Repeat-rule is to create cycles in the derivation: When, while reading a CA proof bottom-up, one reaches a leaf discharged because of the Repeat-rule, one circles back to the Repeat-application and continues following the branches upward. The Repeat-rule thus allows for the subderivation above it to be "pasted" onto suitable nodes above it, thus creating cycles which induce an infinite derivation. We have chosen to present CA in terms of such a Repeat-rule, inspired by that from [1], instead of general cyclic derivations graphs, the variant more common in the literature, due to the similarity of its presentation to traditional finitary derivation systems.

Definition 1.3. The sequents we consider are $\Gamma \Rightarrow \Delta$ where Γ, Δ are sets of formulas. The expressions Γ, φ and Γ, Γ' are shorthands for $\Gamma \cup \{\varphi\}$ and $\Gamma \cup \Gamma'$, respectively.

The *derivation rules of cyclic arithmetic* are the following standard rules for first-order logic, denoting by $[t/x]$ the operation of substituting a term t into all free occurrences of

the variable x,

$$\textsc{Ax}\frac{}{\Gamma, \varphi \Rightarrow \varphi, \Delta} \qquad \rightarrow\!\textsc{L}\frac{\Gamma, \varphi \Rightarrow \Delta \quad \Gamma \Rightarrow \psi, \Delta}{\Gamma, \varphi \to \psi \Rightarrow \Delta} \qquad \rightarrow\!\textsc{R}\frac{\Gamma, \varphi \Rightarrow \psi, \Delta}{\Gamma \Rightarrow \varphi \to \psi, \Delta}$$

$$\wedge\textsc{L}\frac{\Gamma, \varphi, \psi \Rightarrow \Delta}{\Gamma, \varphi \wedge \psi \Rightarrow \Delta} \qquad \wedge\textsc{R}\frac{\Gamma \Rightarrow \varphi, \Delta \quad \Gamma \Rightarrow \psi, \Delta}{\Gamma \Rightarrow \varphi \wedge \psi, \Delta} \qquad \vee\textsc{L}\frac{\Gamma, \varphi \Rightarrow \Delta \quad \Gamma, \psi \Rightarrow \Delta}{\Gamma, \varphi \vee \psi \Rightarrow \Delta}$$

$$\vee\textsc{R}\frac{\Gamma \Rightarrow \varphi, \psi, \Delta}{\Gamma \Rightarrow \varphi \vee \psi, \Delta} \qquad \forall\textsc{L}\frac{\Gamma, \varphi[t/x] \Rightarrow \Delta}{\Gamma, \forall x.\varphi \Rightarrow \Delta} \qquad \forall\textsc{R}\frac{\Gamma \Rightarrow \varphi, \Delta \quad x \notin \mathrm{FV}(\Gamma, \Delta)}{\Gamma \Rightarrow \forall x.\varphi, \Delta}$$

$$\exists\textsc{L}\frac{\Gamma, \varphi \Rightarrow \Delta \quad x \notin \mathrm{FV}(\Gamma, \Delta)}{\Gamma, \exists x.\varphi \Rightarrow \Delta} \qquad \exists\textsc{R}\frac{\Gamma \Rightarrow \varphi[t/x], \Delta}{\Gamma \Rightarrow \exists x.\varphi, \Delta} \qquad \bot\textsc{L}\frac{}{\Gamma, \bot \Rightarrow \Delta}$$

$$=\!\textsc{L}\frac{\Gamma[t/x, s/y] \Rightarrow \Delta[t/x, s/y]}{\Gamma[s/x, t/y], s = t \Rightarrow \Delta[s/x, t/y]} \qquad =\!\textsc{R}\frac{}{\Gamma \Rightarrow t = t, \Delta}$$

together with the following structural rules

$$\textsc{Wk}\frac{\Gamma \Rightarrow \Delta}{\Gamma, \Gamma' \Rightarrow \Delta, \Delta} \qquad \textsc{Cut}\frac{\Gamma, \varphi \Rightarrow \Delta \quad \Gamma \Rightarrow \varphi, \Delta}{\Gamma \Rightarrow \Delta}$$

$$\textsc{Sub}\frac{\Gamma \Rightarrow \Delta}{\Gamma[s/x] \Rightarrow \Delta[s/x]} \qquad \textsc{Repeat}\frac{\begin{array}{c}[\Gamma \Rightarrow \Delta] \\ \vdots \\ \Gamma \Rightarrow \Delta\end{array}}{\Gamma \Rightarrow \Delta}$$

and the following arithmetic-specific axioms and rules

$$\begin{array}{rll} s < t, t < u \Rightarrow s < u & s < t \Rightarrow S\,s < S\,t & \Rightarrow s + S\,t = S\,(s + t) \\ s < t, t < s \Rightarrow & \Rightarrow s < t, s = t, t < s & \Rightarrow t \cdot 0 = 0 \\ s < t, t < S\,s \Rightarrow & \Rightarrow t < S\,t & \Rightarrow s \cdot S\,t = (s \cdot t) + s \\ t < 0 \Rightarrow & \Rightarrow t + 0 = t & \end{array}$$

and the arithmetic-specific derivation rule

$$\frac{\Gamma, t = S\,x \Rightarrow \Delta \quad x \text{ fresh}}{\Gamma, 0 < t \Rightarrow \Delta}$$

A *cyclic derivation of* $\Gamma \Rightarrow \Delta$ is a finite derivation, derived according to the rules above such that all of its leaves $\Gamma' \Rightarrow \Delta'$ are either axioms or discharged via an application of the Repeat-rule. In the latter case, the leaf-occurrence of $\Gamma' \Rightarrow \Delta'$ is called a *bud* and the application of the Repeat-rule is called *its companion*.

An ∞*-derivation* of $\Gamma \Rightarrow \Delta$ is a possibly infinite derivation of $\Gamma \Rightarrow \Delta$ constructed according to the rules above, excluding the Repeat-rule, such that all of its leaves are axioms.

As the scheme of induction is an axiom of Peano arithmetic and cyclic arithmetic is a proof system for Peano arithmetic, the scheme of induction should be provable in cyclic arithmetic. This is demonstrated by the derivation below which proves the scheme of induction for a given formula φ. We employ the shorthand $\mathrm{prg}(\varphi) \coloneqq \varphi \to \varphi[S\,x/x]$ to

denote the fact that φ is progressing. Note that, for the sake of readability, we do not restrict ourselves to only use rules from Definition 1.3 but also employ other rules which are easily seen to be admissible in cyclic arithmetic. Observe specifically the necessity of the Sub-rule in the right-hand branch to obtain a sequent which may be discharged by the Repeat-rule.

$$
\dfrac{\mathrm{W_K}\dfrac{\varphi[0/x] \Rightarrow \varphi[0/x]}{\varphi[0/x], \forall x.\mathrm{prg}(\varphi) \Rightarrow \varphi[0/x]} \qquad \dfrac{\dfrac{\mathrm{W_K}\dfrac{\varphi[y/x] \Rightarrow \varphi[y/x]}{\ldots, \varphi[y/x] \Rightarrow \varphi[y/x]} \qquad \dfrac{\dfrac{[\varphi[0/x], \forall x.\mathrm{prg}(\varphi) \Rightarrow \varphi]\ (*)}{\varphi[0/x], \forall x.\mathrm{prg}(\varphi) \Rightarrow \varphi[y/x]}\,\textsc{Sub}}{y < x, \varphi[0/x], \forall x.\mathrm{prg}(\varphi) \Rightarrow \varphi[y/x]}}{y < x, \varphi[0/x], \forall x.\mathrm{prg}(\varphi), \mathrm{prg}(\varphi)[y/x] \Rightarrow \varphi[S\,y/x]} \qquad x = S\,y, \varphi[0/x], \forall x.\mathrm{prg}(\varphi) \Rightarrow \varphi[S\,y/x]}{\varphi[0/x], \forall x.\mathrm{prg}(\varphi) \Rightarrow \varphi}\,{\to}\mathrm{L}}{(*)\ \text{Repeat}\ \dfrac{\varphi[0/x], \forall x.\mathrm{prg}(\varphi) \Rightarrow \varphi}{\varphi[0/x], \forall x.\mathrm{prg}(\varphi) \Rightarrow \varphi}}
$$

The attentive reader may have noticed that we have not yet introduced the global trace condition. Indeed, based on our explanation in the introduction of this thesis, one would expect that simply taking cyclic CA derivations as a notion of proof should be unsound. As an example of this, consider the derivation of $\Rightarrow \bot$ given below:

$$
(*)\ \text{Repeat}\ \dfrac{[\Rightarrow \bot]\ (*)}{\Rightarrow \bot}
$$

Specifying the global trace condition of cyclic arithmetic requires further definitions. We begin by first introducing the GTC for ∞-derivations and derive the GTC for cyclic derivations from it later in the chapter.

Definition 1.4. A term t *occurs* in a sequent $\Gamma \Rightarrow \Delta$ if it appears, possibly as a subterm of another term, in a formula in Γ or Δ.

Let $\Gamma' \Rightarrow \Delta'$ be a premise of derivation rule R concluding $\Gamma \Rightarrow \Delta$ as below

$$
\mathrm{R}\dfrac{\ldots \quad \Gamma' \Rightarrow \Delta' \quad \ldots}{\Gamma \Rightarrow \Delta}
$$

Let t and t' be terms occurring in $\Gamma \Rightarrow \Delta$ and $\Gamma' \Rightarrow \Delta'$, respectively. The term t' is called a *precursor* of t if one of the following three conditions holds:

- $t = t'$;
- if $R = \textsc{Sub}$ and $\Gamma = \Gamma'[s/x], \Delta = \Delta'[s/x]$ and $t = t'[s/x]$;
- if $R = (=\mathrm{L})$ and $\Gamma = \Gamma''[s/x, t/y], \Gamma' = \Gamma''[t/x, s/y]$ and analogously for the Δ and there exists a term t'' such that $t = t''[s/x, t/y]$ and $t' = t''[t/x, s/y]$.

When $\Gamma \Rightarrow \Delta$ and $\Gamma' \Rightarrow \Delta'$ are clear from the context, we write $t' \prec_R t$ to denote the fact that t' is a precursor of t. Similarly, if the derivation rule can also be inferred, we often omit the R, writing $t' \prec t$.

Definition 1.5. Let $(\Gamma_i \Rightarrow \Delta_i)_{i\in\omega}$ be an infinite branch through an ∞-derivation and $(R_i)_{i\in\omega}$ be a sequence of rules such that $\Gamma_i \Rightarrow \Delta_i$ is derived via R_i and $\Gamma_{i+1} \Rightarrow \Delta_{i+1}$ is one

of the premises. A *trace* through $(\Gamma_i \Rightarrow \Delta_i)_{i\in\omega}$ is a sequence of terms $(t_i)_{i\in\omega}$ such that t_i occurs in $\Gamma_i \Rightarrow \Delta_i$ and for every $i \in \omega$

$$t_{i+1} \prec_{R_i} t_i \qquad \text{or} \qquad \text{there exists } s \prec_{R_i} t_i \text{ s.t. } (t_{i+1} < s) \in \Gamma_{i+1}$$

If the latter condition holds for t_i, t_{i+im} the index $i \in \omega$ is called a *progress point*. We call such a trace *progressing* if it has infinitely many progress points.

Remark. Intuitively, $s \prec t$ means that s and t denote the same natural number. This observation extends to the fact that at any progress point $i \in \omega$ the natural number denoted by t_{i+1} is strictly smaller than that denoted by t_i. A progressing trace thus denotes a sequence of never increasing, infinitely often decreasing natural numbers.

Definition 1.6. An ∞-derivation is a *∞-proof* if it satisfies the *global trace condition*: every branch $(\Gamma_i \Rightarrow \Delta_i)_{i\in\omega}$ through the proof has a suffix through which there exists a progressing trace.

Example 1.7. It is quite clear that unfolding the cyclic derivation of $\Rightarrow \bot$ cannot yield an ∞-proof: It does not even contain any terms which could serve as the basis of a progressing trace. On the other hand, the cyclic derivation of the induction scheme we have given above does unfold into an ∞-derivation which satisfies the global trace condition. To see why, observe that there is only one branch which runs through it ("always go right"). The progressing trace through it starts with $t_0 = x$ and tracks it until switching to the term y above the $\to$L-application, focusing back on x at the Sub application. This trace is progressing as during the switch from x to y, the formula $y < x$ is an assumption. As that means all branches through the ∞-derivation have a progressing trace, it is an ∞-proof.

Having formally separated ∞-derivations and ∞-proofs, we now check that the global trace condition indeed constitutes a soundness condition of ∞-derivations.

Theorem 1.8. *If there is an ∞-proof of $\Gamma \Rightarrow \Delta$ then $\vDash (\Gamma \Rightarrow \Delta)$.*

Proof. For this suppose that there was some $\rho \nvDash (\Gamma \Rightarrow \Delta)$. From this, we can obtain a branch annotated with sequents $(\Gamma_i \Rightarrow \Delta_i)_{i\in\omega}$ through the ∞-proof, together with a sequence of assignments $(\rho_i)_{i\in\omega}$ such that $\rho_i \nvDash (\Gamma_i \Rightarrow \Delta_i)$ and furthermore if s and t are terms occurring in $\Gamma_i \Rightarrow \Delta_i$ and $\Gamma_{i+1} \Rightarrow \Delta_{i+1}$, respectively, then $t \prec s$ entails $t^{\rho_{i+1}} = s^{\rho_i}$. We start by taking $\Gamma_0 := \Gamma, \Delta_0 := \Delta$ and $\rho_0 := \rho$. Now suppose this sequence had already been derived up to some index $i \in \omega$. Then we perform a case distinction on the rule R_i used to derive $\Gamma_i \Rightarrow \Delta_i$. We only spell out three illustrative cases, the other cases are all analogous to one of them.

- $\Gamma_i \Rightarrow \Delta_i$ *is an axiom:* This is not possible as any axiom is valid under every assignment and $\rho_i \nvDash (\Gamma_i \Rightarrow \Delta_i)$.
- $R =$ Sub: Then $\Gamma_i = \Gamma'[s/x], \Delta_i = \Delta'[s/x]$. Pick $\Gamma_{i+1} := \Gamma'$ and $\Delta_{i+1} := \Delta'$. Furthermore, pick $\rho_{i+1} := \rho_i[x \mapsto s^{\rho_i}]$. Then $\rho_{i+1} \nvDash (\Gamma_{i+1} \Rightarrow \Delta_{i+1})$ as we know that $\rho_i \nvDash (\Gamma_{i+1}[s/x] \Rightarrow \Delta_{i+1}[s/x])$. For the condition on precursors, observe that if $t_{i+1} \prec_{\text{Sub}} t_i$ then either $x \in \mathrm{FV}(t_{i+1})$ meaning $t_i = t_{i+1}[s/x]$ and thus $t_i^{\rho_i} = t_{i+1}^{\rho_{i+1}}$, or $x \notin \mathrm{FV}(t_{i+1})$ and thus $t_i = t_{i+1}$ and $t_i^{\rho_i} = t_{i+1}^{\rho_{i+1}}$ as $\rho_i \restriction \mathrm{Var} \setminus \{x\} = \rho_{i+1} \restriction \mathrm{Var} \setminus \{x\}$.

– $R = \wedge\mathrm{R}$: Then $\Delta_i = \Delta', \varphi_1 \wedge \varphi_2$. Because $\rho_i \nVdash (\Gamma_i \Rightarrow \Delta_i)$ we know that $\rho_i \nVdash \varphi_1 \wedge \varphi_2$ and thus that there has to be $j \in \{1, 2\}$ such that $\rho_i \nVdash \varphi_j$. Then pick $\Gamma_{i+1} \coloneqq \Gamma_i$ and $\Delta_{i+1} \coloneqq \Delta', \varphi_j$ with $\rho_{i+1} \coloneqq \rho_i$.

Because the ∞-proof satisfies the global trace condition, there exists a good trace $(t_i)_{i\in\omega}$ along $(\Gamma_i \Rightarrow \Delta_i)_{i\in\omega}$. Consider the sequence of natural numbers given by $(t_i^{\rho_i})_{i\in\omega}$: At every index $i \in \omega$, either $t_{i+1} \prec_{R_i} t_i$, and thus $t_{i+1}^{\rho_{i+1}} = t_i^{\rho_i}$, or there is some term s such that $s \prec_{R_i} t_i$ and $t_{i+1} < s \in \Gamma_{i+1}$. In the latter case, we know by the same reasoning that $s^{\rho_{i+1}} = t_i^{\rho_i}$ and furthermore that $\rho_{i+1} \nVdash (\Gamma_{i+1} \Rightarrow \Delta_{i+1})$ means that $\rho_{i+1} \vDash t_{i+1} < s$ and thus $t_{i+1}^{\rho_{i+1}} < s^{\rho_{i+1}} = t_i^{\rho_i}$. As the latter case occurs infinitely often along $(t_i^{\rho_i})_{i\in\omega}$, it is a non-increasing sequence of natural numbers which decreases infinitely often, contradicting the well-orderedness of the natural numbers. □

Remark. Intuitively, the global trace condition of CA enforces that ∞-proofs constitute arguments by infinite descent: Along every infinite branch there is an infinitely decreasing sequence of terms. This guarantees that each "semantic instance" of the derivation, the derivation in which variables are interpreted by some assignment ρ, does not have any infinite paths as these would give rise to a contradictory sequence of naturals which decreases infinitely often. This means that each "semantic instance" of an ∞-derivation is well-founded and its validity can thus be reduced to the validity of the axioms.

The global trace condition for ∞-derivations of CA extends quite naturally to cyclic derivations of CA.

Definition 1.9. An *infinite branch through a cyclic derivation* of $\Gamma \Rightarrow \Delta$ is a sequence $(\Gamma_i \Rightarrow \Delta_i)_{i\in\omega}$ such that $\Gamma_0 = \Gamma, \Delta_0 = \Delta$ and furthermore for every $i \in \omega$

– $\Gamma_i \Rightarrow \Delta_i$ is an inner node of the derivation and $\Gamma_{i+1} \Rightarrow \Delta_{i+1}$ is one of its premises

– $\Gamma_i \Rightarrow \Delta_i$ is a bud and $\Gamma_{i+1} \Rightarrow \Delta_{i+1}$ is the premise of its companion

A cyclic derivation is a *cyclic proof* if it satisfies the *global trace condition*: infinite every branch through it has a suffix which has a progressing trace.

Proposition 1.10. *If there is a cyclic proof of* $\Gamma \Rightarrow \Delta$ *then* $\vDash (\Gamma \Rightarrow \Delta)$.

Proof. Observe that any cyclic proof can be "unfolded" into an ∞-proof whose infinite branches are precisely the infinite branches through the cyclic proof. Then the claim follows by Theorem 1.8. □

Showing that cyclic arithmetic proves the same theorems as Peano arithmetic is quite complex, which is why we choose to omit this result from our account of cyclic arithmetic. We encourage the interested reader to consult [33] for the original proof and [8] for a more detailed analysis of the derivations produced by the proof method.

Chapter 2

Abstract Cyclic Derivations

This chapter covers most ideas at the heart of this thesis: cyclic trees, trace categories and abstract cyclic derivations. These are introduced in order to obtain an abstract representation of cyclic derivations that captures all of their cyclic content (which infinite branches run through the derivation and whether these branches satisfy the trace condition) while discarding all of their logical content (the sequents and the specific derivation rules used to manipulate them as well as how the trace condition relates to said sequents). Section 2.2 defines algorithms to simplify these abstract representations in different ways.

2.1 Definitions and Motivation

The notions we introduce in this section are easier understood in terms of examples, hence we fix the following derivation of $\Rightarrow x = x$ in cyclic arithmetic as a running example. To fit the page, the derivation has been split into two trees at the node marked with "Cont.".

$$
\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{x < 0 \Rightarrow}{x < 0 \Rightarrow x = 0, 0 < x}\,\textsc{Wk} \qquad \Rightarrow x < 0, x = 0, 0 < x}{\Rightarrow x = 0, 0 < x}\,\textsc{Cut}}{\Rightarrow x = 0 \lor 0 < x}\,\lor\mathrm{L}}{\Rightarrow x = x, x = 0 \lor 0 < x}\,\textsc{Wk} \qquad \dfrac{\text{Cont.}}{x = 0 \lor 0 < x \Rightarrow x = x}}{\Rightarrow x = x}\,\textsc{Cut}}{\Rightarrow x = x}\,(*)\ \text{Repeat}
$$

$$
\dfrac{\dfrac{\dfrac{}{x = 0 \Rightarrow x = x}\,{=}\mathrm{R} \qquad \dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\Rightarrow y < S\,y}{\Rightarrow S\,y = S\,y, y < S\,y}\,\textsc{Wk} \qquad \dfrac{\dfrac{\dfrac{}{y = y \Rightarrow S\,y = S\,y}\,{=}\mathrm{R} \qquad \dfrac{\dfrac{[\Rightarrow x = x](*)}{\Rightarrow y = y}\,\textsc{Sub}}{\Rightarrow S\,y = S\,y, y = y}\,\textsc{Wk}}{\Rightarrow S\,y = S\,y}\,\textsc{Cut}}{y < S\,y \Rightarrow S\,y = S\,y}\,\textsc{Wk}}{\Rightarrow S\,y = S\,y}\,\textsc{Cut}}{x = S\,y \Rightarrow x = x}\,{=}\mathrm{L}}{0 < x \Rightarrow x = x}}{\dfrac{x = 0 \lor 0 < x \Rightarrow x = x}{\text{Cont.}}}\,\lor\mathrm{L}
$$

The arithmetical content of the proof is extremely trivial as the same statement could be proven with just one application of the (=R) rule. However, it serves as a simple example

of a CA proof with a cycle. Its size is owed to the fact that we did not omit any details in the proof. To verify that this is indeed a "proof" and not just a cyclic derivation, note that there is only one infinite branch through the derivation ("always go right"). The progressing trace along that branch tracks x until encountering the application of (=L), briefly tracking $S\,y$ and then switching the tracking to y at the (Cut) which is maintained until the (Sub) application, when x is tracked again. Said trace is progressing as $y < S\,y$ is an assumption when the focus is switched from $S\,y$ to y, making it a progress point which occurs infinitely often along the trace.

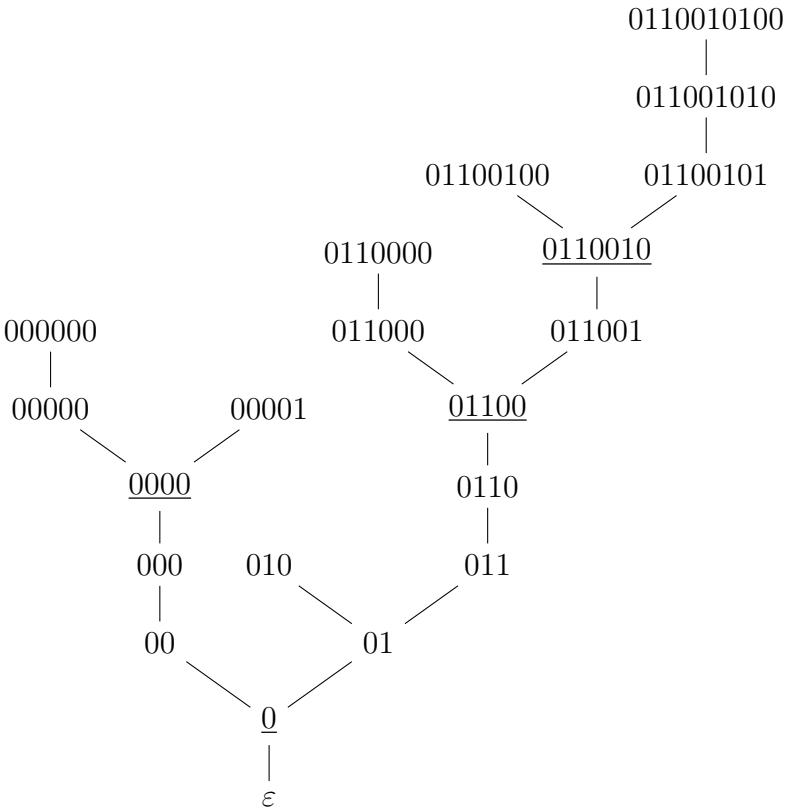
An important cyclic aspect of any cyclic derivation is its "shape" which dictates the branches which can run through it. We capture the tree shape of a derivation as trees over sets.

Definition 2.1. Given a set X, the *prefix-ordering* on X^* is defined as follows: For $s, t \in X^*$ is s a *prefix* of t, writing $s < t$, if there is $u \in X^+$ such that $su = t$. If $|u| = 1$ then one also writes $s <_+ t$ and calls t a *direct extension* of s.

Definition 2.2. Any non-empty set $T \subseteq X^*$ is called a *tree* if it is prefix-closed, that is, if $t \in T$ and $s < t$ then $s \in T$. Every $s \in T$ is called a *node of* T and $\varepsilon \in T$ its *root*.

The *children* of a node $s \in T$ are given by $\mathrm{Chld}(s) := \{t \in T \mid s <_+ t\}$ and its *breadth* is $\mathrm{bd}(s) = |\mathrm{Chld}(s)|$. A tree T is *finitely branching* if $\mathrm{bd}(s) \in \omega$ for all $s \in T$. A sequence $s \in T$ is called a *leaf* if $\mathrm{Chld}(s) = \emptyset$ and an *inner node* otherwise.

Example 2.3. The tree shape of the derivation of $\Rightarrow x = x$ can be formalized as the binary tree $T \subseteq 2^*$ depicted below. Each node is annotated with the element $s \in T$ representing it.



To highlight the correspondence between tree with the derivation above, we have underlined all nodes at which the (Cut) rule is applied. Note also that 01 is where the derivation given above is split.

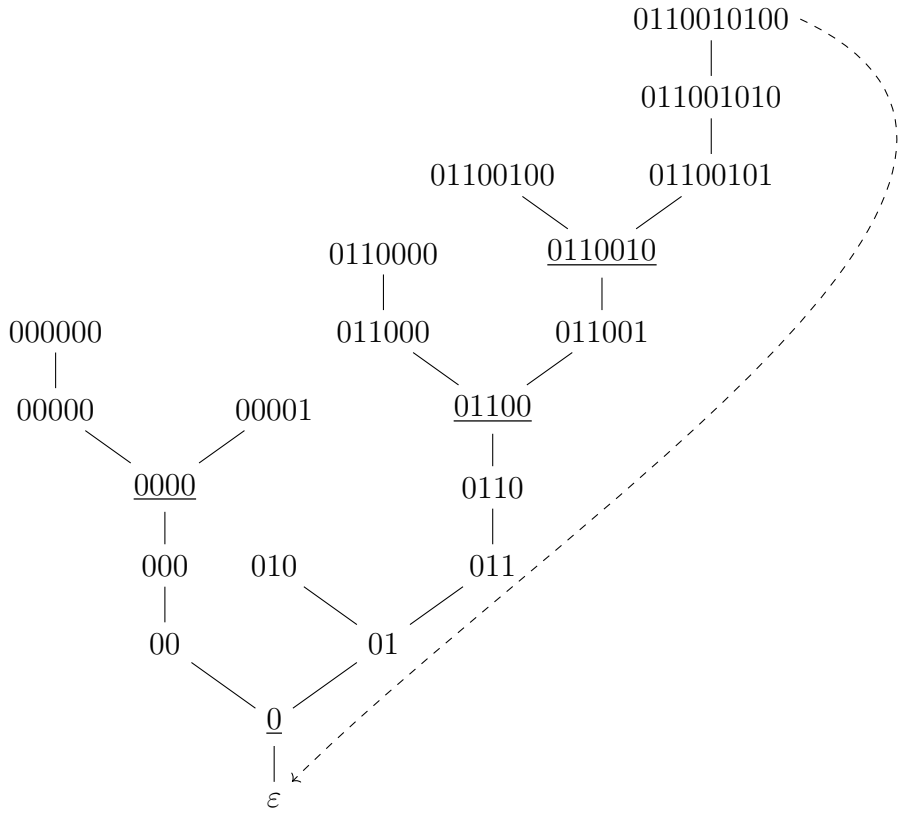
If one was so inclined, one could also specify the tree explicitly as the set

$$T := \{\varepsilon, 0, 00, 000, 0000, 00000, 0000000, 00001, 01, 010, 011, 0110, 01100, 011000, 0110000 \\ 011001, 0110010, 01100100, 01100101, 011001010, 0110010100\}$$

While such trees arguably capture the "tree shape" of cyclic derivations, they do not include the most important structural feature of cyclic derivations: the repeats. To fully capture the "shape" or "structure" of cyclic derivations, we introduce cyclic trees. In addition to the usual tree structure, they also include a means of connecting buds with their companions.

Definition 2.4. A *cyclic tree on* X is a pair $C = (T, \beta)$ consisting of a finite tree $T \subseteq X^*$ and a partial map $\beta : \mathrm{Leaf}(T) \rightharpoonup T$ with $\beta(s) \neq s$ for all $s \in \mathrm{dom}(\beta)$. Each $s \in \mathrm{dom}(\beta)$ is called a *bud* and $\beta(s)$ *its companion.*

Example 2.5. The full structure of the derivation of $\Rightarrow x = x$ can be formalized as the cyclic tree $C := (T, \beta)$ where T is the same as in Example 2.3 and $\beta := \{(0110010100, \varepsilon)\}$. The cyclic tree C is pictured below:



Apart from the structure of cyclic derivations, we also wish for our abstract cyclic derivations to capture the global trace condition and its satisfaction. As noted in the introduction, the GTC of most cyclic derivation systems is of the form "all infinite branches

through the derivation exhibit a desired property". It suffices to capture this "desired property", which we call the trace condition, to define the notion of global trace condition in an abstract manner. We begin by defining an abstract notion of path. We ask readers unfamiliar with semi-categories and semi-functors to refer to Appendix A for their definitions.

Definition 2.6. Fix a category $\mathcal{T}$ and denote by ω the preorder category on $(\omega, <)$.

1. A *path* through $\mathcal{T}$ is a functor $P : \omega \to \mathcal{T}$.
2. For two paths $P, P' : \omega \to \mathcal{T}$ we write $P \subseteq P'$ and say P *is a subpath of* P' if there is a semi-functor $S : \omega \to \omega$ such that $P = P' \circ S$

Remark. When used as the domain or codomain of a semi-functor, the symbol ω always refers to the irreflexive order semi-category $(\omega, <)$, analogous to the usual preorder category on ω but excluding the identity morphisms.

Remark. We require $S : \omega \to \omega$ to be a semi-functor in point 2. of Definition 2.6 to exclude the choice of $S := \Delta_n$ where $\Delta_n(i) := n$ is the constant functor. $P \circ \Delta_n \subseteq P$ would break both the intuition what a subpath is and some of the theorems in this thesis.

We formalize trace conditions as predicates on such paths through $\mathcal{T}$ which are invariant under subpaths. Our definition associates each such trace condition with a trace category. Such trace categories capture the "essence" of the global trace conditions of cyclic derivation systems. In practice, the global trace conditions of many cyclic derivation systems which seem similar on an intuitive level are characterized by the same trace category.

Definition 2.7. A *trace category* is a category $\mathcal{T}$ which comes equipped with a trace condition. A *trace condition* is a predicate on paths $P : \omega \to \mathcal{T}$ with the property that for any $P \subseteq P'$, the predicate holds for P if and only if it holds for P'.

Remark. The subpath invariance of trace conditions guarantees two properties:

1. Finite prefixes of paths never matter for trace condition satisfaction: The first n "steps" of a path P can be dropped by taking $S(m) := m + n$ and observing that $P \circ S \subseteq P$.
2. The trace condition is invariant under composition. That is, if there is a path P of the shape

$$X_0^0 \xrightarrow{\tau_0^0} X_0^1 \xrightarrow{\tau_0^1} X_0^2 \dots X_0^{n_0} \xrightarrow{\tau_0^{n_0}} X_1^0 \xrightarrow{\tau_1^0} \dots \xrightarrow{\tau_1^{n_1}} X_2^0 \xrightarrow{\tau_2^0} \dots$$

   then it suffices to consider the path P' below

$$X_0^0 \xrightarrow{\tau_0^{n_0} \circ \dots \circ \tau_0^0} X_1^0 \xrightarrow{\tau_1^{n_1} \circ \dots \circ \tau_1^{n_1}} X_2^0 \xrightarrow{\tau_2^{n_2} \circ \dots \circ \tau_2^{n_2}} \dots$$

   since $P' = P \circ S$ with $S(i) := \sum_{j<i} n_j$. Vice versa, one may also decompose paths such as P' into paths of the shape of P.

The second property is quite natural in a categorical setting: No "information" about trace condition satisfaction should be lost when composing morphisms. The first property arises directly from Definition 2.7. It is not unreasonable as the trace conditions of all cyclic

derivation systems from the literature we know of also exhibit this property. Nevertheless, it could be eliminated by requiring that $S(0) = 0$ in the definition of subpath.

We continue by demonstrating how to represent the unique infinite branch through our example derivation of $\Rightarrow x = x$ with the formalism of paths through trace categories. We begin by defining a trace category. It can be seen as a variant of the abstract notion of trace given by Brotherston et. al in [6] with some slight adjustments to better fit our categorical framework. It also is a special case of a more general family of trace categories we define in Section 5.1. There, we also give a generic proof that the trace condition of such trace categories is indeed invariant under subpaths. As the proof of this property is very involved, we do not reprove the special case for this trace category here but instead rely on the reader's intuition that the trace condition is indeed subpath invariant.

Definition 2.8. The *Brotherston category* $\mathcal{B}$ has as its objects the finite sets. Given two such sets X, Y, we define $\textsc{Hom}_{\mathcal{B}}(X, Y) \coloneqq \mathcal{P}(X \times Y \times \{0, 1\})$. The identity morphism of a set X is given by $1_X \coloneqq \{(x, x, 0) \mid x \in X\}$. For $\tau : X \to Y$, $\tau' : Y \to Z$ we define:

$$\tau' \circ \tau \coloneqq \{(x, z, \max\{a, b\}) \mid \exists y \in Y, a, b \in \{0, 1\}.\ (x, y, a) \in \tau, (y, z, b) \in \tau'\}$$

Given a path $P : \omega \to \mathcal{B}$, we say that P *satisfies the trace condition* if there exists $s \in \omega$ and sequences $\sigma : \Pi i \in \omega.\ P(s + i)$ and $a : \{0, 1\}^\omega$ such that for each $i \in \omega$ we have $(\sigma_i, \sigma_{i+1}, a_i) \in P(s + i < s + i + 1)$ and furthermore $a_i = 1$ for infinitely many $i \in \omega$.

Remark. The purpose of the $s \in \omega$ in the definition of the trace condition is to allow for prefixes of paths to be discarded.

Example 2.9. With this in mind, we can formalize the branch through our example derivation in our framework and verify that it indeed satisfies the trace condition. This also serves as a demonstration of the idea behind the Brotherston category.

First recall that the part of the derivation which generates the infinite branch looks as follows:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{[\Rightarrow x = x](*)}{\Rightarrow y = y}\,\textsc{Sub}}{\Rightarrow S\,y = S\,y, y = y}\,\textsc{Wk}}{\Rightarrow S\,y = S\,y}\,\textsc{Cut}}{y < S\,y \Rightarrow S\,y = S\,y}\,\textsc{Wk}}{\Rightarrow S\,y = S\,y}\,\textsc{Cut}}{x = S\,y \Rightarrow x = x}\,\textsc{=L}}{0 < x \Rightarrow x = x}}{x = 0 \vee 0 < x \Rightarrow x = x}\,\textsc{$\vee$L}}{\Rightarrow x = x}\,\textsc{Cut}}{\Rightarrow x = x}\,(*)\ \textsc{Repeat}$$

To represent such an infinite branch through a derivation as a path $P : \omega \to \mathcal{B}$ we need to assign to each sequent on the branch a finite set and then find appropriate $\mathcal{B}$-morphisms between these sets such that the trace condition is modeled accurately. In the case of cyclic arithmetic, we assign to each sequent $\Gamma \Rightarrow \Delta$ the set $\textsc{Term}(\Gamma, \Delta)$ of terms occurring in Γ and Δ. Then, if along a branch the sequent $\Gamma \Rightarrow \Delta$ is followed by $\Gamma' \Rightarrow \Delta'$ (meaning $\Gamma \Rightarrow \Delta$ is the conclusion of a rule one of whose premises is $\Gamma' \Rightarrow \Delta'$), we connect

$\textsc{Term}(\Gamma, \Delta)$ with $\textsc{Term}(\Gamma', \Delta')$ via the morphism $\tau : \textsc{Term}(\Gamma, \Delta) \to \textsc{Term}(\Gamma', \Delta')$ given below

$$\tau \coloneqq \{(t, t', 0) \mid t' \prec t\} \cup \{(t, t', 1) \mid \exists s.\ s \prec t \wedge (t' < s) \in \Gamma'\}$$

where $s \prec t$ is the precursor relation defined in Definition 1.4. In other words, $(t, t', a) \in \tau$ if t' can be the successor of t along a trace as defined in Definition 1.5 with $a = 1$ if progress has been made. For a concrete example, consider the application of =L in the branch above. In the Brotherston category, we would represent it as follows:

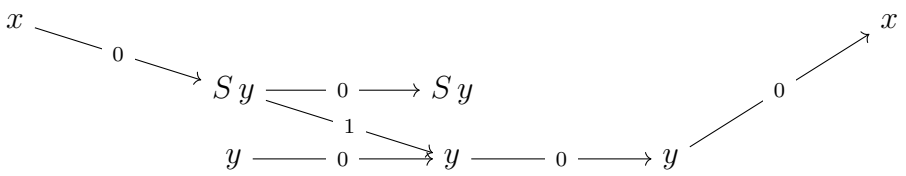
$$\begin{aligned} X &\coloneqq \textsc{Term}(x = S\,y, x = x) = \{x, S\,y, y\} \\ Y &\coloneqq \textsc{Term}(S\,y = S\,y) = \{S\,y, y\} \\ \tau : X \to Y &\coloneqq \{(x, S\,y, 0), (S\,y, S\,y, 0), (y, y, 0)\} \end{aligned}$$

The full branch through our example derivation could be represented in this manner as $P : \omega \to \mathcal{B}$. However, defining it fully would be quite tedious, requiring us to spell out 10 different morphisms. As we only need this $\mathcal{B}$-representation of the branch to check whether it satisfies the trace condition, there is one simplification we can apply: We know that the trace condition of $\mathcal{B}$ is invariant under subpaths, thus is suffices to define a path $Q \subseteq P$. Consider the following $Q : \omega \to \mathcal{B}$, where $n \in \omega$:

$$\begin{aligned} Q(4n) &\coloneqq \textsc{Term}(x = x) \coloneqq \{x\} \\ Q(4n+1) &\coloneqq \textsc{Term}(S\,y = S\,y) \coloneqq \{S\,y, y\} \\ Q(4n+2) &\coloneqq \textsc{Term}(y < S\,y, S\,y = S\,y) \coloneqq \{S\,y, y\} \\ Q(4n+3) &\coloneqq \textsc{Term}(y = y) \coloneqq \{y\} \\ Q(4n < 4n+1) &\coloneqq \{(x, S\,y, 0)\} \\ Q(4n+1 < 4n+2) &\coloneqq \{(S\,y, S\,y, 0), (y, y, 0), (S\,y, y, 1)\} \\ Q(4n+2 < 4n+3) &\coloneqq \{(y, y, 0)\} \\ Q(4n+3 < 4(n+1)) &\coloneqq \{(y, x, 0)\} \end{aligned}$$

This is a subpath of the $\mathcal{B}$-representation of the branch through the derivation. Here, the segment $Q(4n < 4n+1)$ combines the rule applications from the Repeat to above =L, $Q(4n+1 < 4n+2)$ is the Cut-step, $Q(4n+2 < 4n+3)$ the steps until below the Sub-application and $Q(4n+3 < 4(n+1))$ the Sub-application, cycling back to above the Repeat. The path $Q : \omega \to \mathcal{B}$ can be visualized as follows:

$$\cdots \quad Q(4n) \qquad Q(4n+1) \qquad Q(4n+2) \qquad Q(4n+3) \qquad Q(4(n+1)) \quad \cdots$$



The picture above already suggests that there is only one possible choice for the sequences $\sigma : \Pi i \in \omega.Q(i)$ and $a : \{0,1\}^\omega$ to demonstrate trace condition satisfaction: The trace connecting the $x \in Q(4n)$ with the $x \in Q(4(n+1))$ for all $n \in \omega$. As such, we pick

$$\sigma_{4n} \coloneqq x \quad \sigma_{4n+1} \coloneqq S\,y \quad \sigma_{4n+2} \coloneqq y \quad \sigma_{4n+3} \coloneqq y \qquad a_i \coloneqq \begin{cases} 1 & \text{if } i = 4n+1 \\ 0 & \text{otherwise} \end{cases}$$

Since these two sequences exhibit the properties required by the trace condition of the Brotherston category, including infinitely many $a_i = 1$ (in this case whenever $i = 4n+1$), the path $Q : \omega \to \mathcal{B}$ satisfies the trace condition.

Concluding this example, observe that if arbitrary infinite branches through CA derivations are represented as some $P : \omega \to \mathcal{B}$ in this manner, the sequence $\sigma : \Pi i \in \omega.P(i)$ witnessing the trace condition is precisely a good trace $(\sigma_i)_{i\in\omega}$ as defined in Definition 1.5, that is, a sequence of terms which never increases but decreases infinitely often. The trace condition of CA can thus be faithfully captured by the Brotherston category.

We have now defined abstract renditions of (a) the structure of cyclic derivations, capturing which infinite branches run through them, and (b) the notion of branches exhibiting a desired property. By combining both, an abstract representation of the trace content of a cyclic derivation can be obtained. Formally, we define them as functors mapping the paths through a cyclic tree C into a trace category $\mathcal{T}$.

Definition 2.10. For a cyclic tree $C = (T, \beta)$ we inductively define the *finite paths* $Path_C(s,t) \subseteq T^+$ *through* C, starting at s and ending at t, as follows

- For any $s \in T$ we have $s \in \mathrm{Path}_C(s,s)$
- For any $s,t,u \in T$ such that $t <_+ u$ we have $pu \in \mathrm{Path}_C(s,u)$ if $p \in \mathrm{Path}_C(s,t)$
- For any $s \in T$ and $t \in \mathrm{dom}(\beta)$ we have $p\beta(t) \in \mathrm{Path}_C(s,\beta(t))$ if $p \in \mathrm{Path}_C(s,t)$

Definition 2.11. For some cyclic tree $C = (T, \beta)$ the *category* $\mathcal{P}_C$ *of paths through* C has as its objects the nodes of T and fixes $\textsc{Hom}_{\mathcal{P}_C}(s,t) = \mathrm{Path}(s,t)$ with the identity $1_s = s : s \to s$. Given $p : s \to t$ and $q : t \to u$ they compose to pq' where $q = tq'$ for $q' \in T^*$.

We sometimes consider the *semi-category* $\mathcal{P}_C^S$ *of progressing paths through* C which is the same as $\mathcal{P}_C$ except that $\textsc{Hom}_{\mathcal{P}_C^S}(s,t) \coloneqq \{p \in \mathrm{Path}(s,t) \mid |p| > 1\}$.

Definition 2.12. An *abstract cyclic derivation* (ACD) is a pair (C, Tr) of a cyclic tree $C = (T, \beta)$ and a functor $\mathrm{Tr} : \mathcal{P}_C \to \mathcal{T}$ from the category of paths through C into a trace category $\mathcal{T}$ such that for any $s \in \mathrm{dom}(\beta)$, $\mathrm{Tr}(s) = \mathrm{Tr}(\beta_s)$ and $\mathrm{Tr}(s\beta_s) = 1_{\mathrm{Tr}(s)}$.

Example 2.13. The cyclic derivation at the beginning of this chapter can be represented as an ACD by defining a functor from $\mathcal{P}_C$, the path category of the cyclic tree given in Example 2.5, into the Brotherston category $\mathcal{B}$. Analogous to Example 2.9, each node $x \in T$ is mapped to $\textsc{Term}(\Gamma, \Delta)$ if x corresponds to the node of the derivation tree annotated with $\Gamma \Rightarrow \Delta$. The morphisms are also defined in terms of the precursor relation as outlined in Example 2.9. We forgo writing down this abstract cyclic derivation explicitly due to its size.

Remark. At first glance, the preorder category generated by the edges of a cyclic tree C, or equivalently the preorder category of the node reachability relation of C, might seem like a simpler candidate for the domain of the functors Tr of abstract cyclic derivations. However, this choice would, in general, interfere with the functoriality of Tr in the following manner: Suppose x was a node of C and there was a cycle on x, represented by a morphism $c : x \to x$ in the preorder category. Functoriality of Tr would require that

$\mathrm{Tr}(c) \circ \mathrm{Tr}(c) = \mathrm{Tr}(c)$ because $c \circ c = c$ holds in the preorder category. As this equation does not hold for many legitimate trace morphisms which should be possible images of $\mathrm{Tr}(c)$, taking these preorder categories as the domains of abstract cyclic derivations is thus problematic.

We close the section by defining what it means for an abstract cyclic derivation to constitute an abstract cyclic proof in a manner very similar to the distinction made in cyclic arithmetic.

Definition 2.14. For an abstract cyclic derivation $D = (C, \mathrm{Tr})$ we call a path $P : \omega \to \mathcal{T}$ a *path through* D if there exists a semi-functor $P' : \omega \to \mathcal{P}_C^S$ such that $P = \mathrm{Tr} \circ P'$.

We say an abstract cyclic derivation D *satisfies the global trace condition* (GTC) if every path $P : \omega \to \mathcal{T}$ through D satisfies the trace condition of $\mathcal{T}$. We an ACD satisfying the GTC an *abstract cyclic proof*.

Remark. This definition differs somewhat from that given for cyclic proofs of CA in Definition 1.9: Instead of only considering infinite branches, which always start at the root, the GTC of ACDs is put in terms of arbitrary infinite paths. However, these two presentations are equivalent as the trace condition is subpath invariant and every infinite path is the suffix of an infinite branch. We have chosen the variant of the definition above as it is slightly simpler.

Example 2.15. If a CA derivation has been faithfully transformed into an abstract cyclic derivation D as described in Example 2.13, then D satisfies the global trace condition if and only if the original cyclic derivation is a proof. In Example 2.9 we have already observed that the trace condition on the paths through such an ACD coincides with the notion of a good trace through an infinite branch in CA. Thus, both notions of global trace conditions coincide as well as they are defined analogously.

2.2 Transforming Abstract Cyclic Derivations

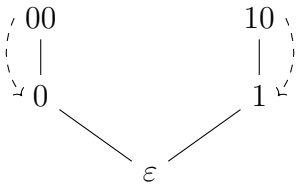
This section is concerned with outlining two algorithms which transform abstract cyclic derivations into ones which are simpler, in two different senses, and equivalent in terms of GTC satisfaction, meaning the simpler ACD satisfies the GTC if and only if the original one does. Such simplification procedures are of interest as they, for example, can be used to improve the efficiency of algorithms for checking GTC satisfaction.

The first notion of "simplicity" for ACDs is "efficiency". An efficient cyclic tree is composed essentially exclusively of nodes which are either buds or companions. Such a cyclic tree could intuitively be considered "efficient" as this means that all nodes of the tree "contribute" towards its most interesting aspect: Which cycles exist within it and how they can be combined into infinite paths. In practice, transforming an ACD induced by a derivation of some cyclic derivation system into an efficient ACD can lead to a drastic reduction in the number of its nodes (see Example 2.18 for an example). Such a size reduction is of interest as all algorithms for deciding whether an ACD satisfies the GTC, including those we give in Chapter 4, have time complexities dependent on the number of nodes in the cyclic trees underlying the ACD in question. Reducing this number via some

transformation thus holds the promise of great performance improvements in concrete implementations of such decision procedures.

Definition 2.16. We call a cyclic tree $C = (T, \beta)$ *efficient* if for every $s \in T$ either $s \in \mathrm{im}(\beta) \cup \mathrm{dom}(\beta)$ or $s = \varepsilon$ with $\mathrm{bd}(\varepsilon) > 1$. An ACD $D = (C, \mathrm{Tr})$ is *efficient* if C is.

Remark. The edge-case for $\varepsilon \notin \mathrm{dom}(\beta)$ stems from the fact that (abstract) cyclic derivations can contain multiple cycles which are not inter-connected. For example, the cyclic tree pictured below should also be considered efficient as there is, in general, no way to eliminate the root ε without inherently changing which infinite paths run through the cyclic tree. However, this is only necessary if ε has at least two children.



Proposition 2.17. *Let $D = (C, Tr)$ be an abstract cyclic derivation with $dom(\beta) \neq \emptyset$. Then it can be transformed into an efficient $D' = (C', Tr')$ with $Tr' : \mathcal{P}_{C'} \to \mathcal{T}$ such that D satisfies the global trace condition if and only if D' does.*

Proof. To obtain C', take $G \coloneqq \mathrm{im}(\beta) \cup \mathrm{dom}(\beta)$. Now we iteratively construct a function $f : G \to \omega^*$ which will yield the tree for C' as follows. To begin, let $\{s_0, \ldots, s_n\} \subseteq G$ be the $<$-minimal elements of G. If there is only one such $s \in G$ then fix $f_0(s) \coloneqq \varepsilon$. Otherwise, take $f_0(s_i) \coloneqq i$. Now, iteratively extend f_i to f_{i+1} as follows: For each $s \in f_i^{-1}(\mathrm{Leaf}(\mathrm{im}(f_i))) \setminus \mathrm{dom}(\beta)$, in other words any companion node added in the previous construction step, let $\{t_0, \ldots, t_n\} \subseteq G$ be those $t \in G$ such that there is $spt \in \mathrm{Path}(s, t)$ for which $p \in (T \setminus G)^*$. Then set $f_{i+1}(t_j) \coloneqq f_i(s)j$ and keep $f_{i+1}(s) = f_i(s)$ for any $s \in \mathrm{dom}(f_i)$. Take $f \coloneqq \bigcup_{i \leq |G|} f_i$ and observe that indeed $f : G \to \omega^*$ as each iterative construction step needs to extend $\mathrm{dom}(f_i)$ by at least one element from G. If $\varepsilon \notin \mathrm{dom}(f)$, take $f(\varepsilon) \coloneqq \varepsilon$. Observe that now $T' \coloneqq \mathrm{im}(f) \subseteq \omega^*$ is a finite tree. Then we take $C' \coloneqq (T', \beta')$ where $\beta' \coloneqq f \circ \beta \circ f^{-1}$ is well-defined as f is a bijection on $\mathrm{im}(f)$.

To extend this cyclic tree to an abstract cyclic derivation, take $\mathrm{Tr}'(s) \coloneqq \mathrm{Tr}(f^{-1}(s))$ for $s \in \mathrm{im}(f)$. For any $s <_+ t \in T'$, observe that per construction of T', there exists a unique path $f^{-1}(s)p_{st}f^{-1}(t) \in \mathrm{Path}_C(f^{-1}(s), f^{-1}(t))$ with the property that $p_{st} \in (T - G)^*$. Then pick $\mathrm{Tr}'(st) \coloneqq \mathrm{Tr}(f^{-1}(s)p_{st}f^{-1}(t))$ and observe that this is both well-defined and fully specifies $\mathrm{Tr}' : \mathcal{P}_{C'} \to \mathcal{T}$.

It remains to prove that D satisfies the GTC if and only if D' does. For the forwards direction, we show that any path $P : \omega \to \mathcal{T}$ through D' is a path through D as well and thus satisfies the trace condition. Indeed, if $Q : \omega \to \mathcal{P}_{C'}$ is such that $P = \mathrm{Tr}' \circ Q$ then each $Q(i < i + 1) = s_0 \ldots s_n$ and thus

$$\begin{aligned} P(i < i+1) &= \mathrm{Tr}(f^{-1}(s_{n-1})p_{s_{n-1}s_n}f^{-1}(s_n)) \circ \ldots \circ \mathrm{Tr}(f^{-1}(s_0)p_{s_0s_1}f^{-1}(s_1)) \\ &= \mathrm{Tr}(f^{-1}(s_0)p_{s_0s_1}f^{-1}(s_1) \ldots f^{-1}(s_{n-1})p_{s_{n-1}s_n}f^{-1}(s_n)) \end{aligned}$$

meaning $P = \mathrm{Tr} \circ Q'$, and thus P is a path through D, where $Q'(j < j + 1) \coloneqq f^{-1}(s_0)p_{s_0s_1} \ldots f^{-1}(s_m)$ whenever $Q(j < j + 1) = s_0 \ldots s_m$.

Now conversely, pick some path $P : \omega \to \mathcal{T}$ through D with $P = \mathrm{Tr} \circ Q$. We may, without loss of generality, assume that $Q(i < i+1) = s_i s_{i+1}$ where either $s_i <_+ s_{i+1}$ or $\beta_{s_i} = s_{i+1}$. Now it is easy to observe that for any such infinite path through a cyclic tree the set $\{i \in \omega \mid Q(i) \in \mathrm{im}(\beta) \cup \mathrm{dom}(\beta)\}$ must be infinite. Then let $S : \omega \to \omega$ be the strictly monotone enumeration of said set. Now simply observe that $Q(S(i < i+1)) = sp_{f(s)f(t)}t$ for $Q(S(i)) = s, Q(S(i+1)) = t$ and $s < t$ and $Q(S(i < i+1)) = st$ with $\beta_s = t$ if $s \in \mathrm{dom}(\beta)$. In either case, $\mathrm{Tr}(Q(S(i < i+1))) = \mathrm{Tr}'(f(Q(S(i)))f(Q(S(i+1))))$, meaning that $P \circ S \subseteq P$ is a sequence through D' as witnessed by $Q' : \omega \to \mathcal{P}_{C'}$ where $Q'(i) = Q(S(i))$ and $Q'(i < i+1) = Q(S(i))Q(S(i+1))$. As $P \circ S$ is a path through D', it satisfies the trace condition of $\mathcal{T}$ meaning P does so as well. □

Remark. A tight characterization of the time complexity of the algorithm described by the proof above is likely very dependent on the choice representation of ACDs. However, it is quite clear that for any such choice which is somewhat reasonable, the complexity obtained would be at most polynomial in the number of nodes in the ACD. Given that the common, automata-based GTC decision procedure is of a complexity super-exponential in the number of nodes (see [37] for more details), it is to be expected that a GTC decision procedure which operates on ACDs would see noticeable performance improvements when combined with the algorithm given above.

Remark. The requirement that $\mathrm{dom}(\beta) \neq \emptyset$ stems from the fact that otherwise $\mathrm{im}(\beta) \cup \mathrm{dom}(\beta) = \emptyset$ which makes it impossible to construct a (cyclic) tree on that basis as we have defined trees to be non-empty. However, as any cyclic tree with $\mathrm{dom}(\beta) = \emptyset$ satisfies the global trace condition vacuously, there is no need to improve the performance of the GTC decision procedure for such cases in the first place.

Example 2.18. In Example 2.13 we describe how to obtain an ACD representing the cyclic derivation of $\Rightarrow x = x$. In that example, we chose not to spell out the full definition of said ACD due to its size. When applying the procedure of Proposition 2.17 to the ACD described in Example 2.13, we obtain the equivalent, efficient ACD (C, Tr) with

$$\mathrm{Tr}(\varepsilon) = \{x\} \qquad \mathrm{Tr}(0) = \{x\} \qquad \mathrm{Tr}(\varepsilon 0)\{(x, x, 1)\}$$

and C as pictured below:

This ACD is not only remarkably small when compared to the original derivation, it is also trivial to check that it satisfies the global trace condition.

The other notion of simplicity we consider are cyclic trees having only backward edges, meaning that every bud's companion is on the path between it and the root. In the derivation system for cyclic arithmetic in Section 1.3, this property is guaranteed for all derivations as we chose to represent cycles via a Repeat-rule. For cyclic derivation systems in general, and thus their corresponding ACDs, this need not always be the case. Backward edges have previously been studied in the literature under many different names (for example, Brotherston [4] calls derivations with this property "cyclic normal forms"). The simplicity afforded by backward edges is somewhat more subtle, generally making

some definitions easier as certain edge cases are eliminated. One notable application are Brotherston's trace manifolds [4], an alternative soundness condition which only applies to cyclic derivations with backward edges.

Definition 2.19. A cyclic tree C has *backward edges* if for every $s \in \mathrm{dom}(\beta)$, $\beta_s < s$.

Proposition 2.20. *Let D be an abstract cyclic derivation. Then it can be transformed into an abstract cyclic derivation D' which has backwards edges and satisfies the global trace condition if and only if D does.*

Proof. This proof works by iteratively eliminating all non-backwards edges, which we call *sideways edges*, from D. We begin by laying out the iteration step. Pick some companion $t \in \mathrm{im}(\beta)$ such that $S \coloneqq \{s \in \mathrm{dom}(\beta) \mid \beta(s) = t, t \not< s\}$ is not empty. Then consider the cyclic tree $C' \coloneqq (T', \beta')$ defined as follows

$$T' \coloneqq T \cup \bigcup_{s \in S} \{su \mid u \in X^*, tu \in T\}$$

$$\beta'(x) \coloneqq \begin{cases} \beta(x) & x \in T \setminus S \\ sv & x = su \text{ for } s \in S \text{ and } \beta(tu) = tv \\ \beta(tu) & x = su \text{ for } s \in S \text{ and } t \not< \beta(tu) \end{cases}$$

We extend this to a cyclic abstract derivation $D' \coloneqq (T', \mathrm{Tr}')$ with

$$\mathrm{Tr}'(x) \coloneqq \begin{cases} \mathrm{Tr}(x) & x \in T \setminus S \\ \mathrm{Tr}(tu) & x = su, s \in S \end{cases} \qquad \mathrm{Tr}'(xy) \coloneqq \begin{cases} \mathrm{Tr}((tu)(tv)) & x = su, y = sv, s \in X \\ \mathrm{Tr}(xy) & \text{otherwise} \end{cases}$$

Per construction, it is easy to see that D' is a well-defined cyclic abstract derivation and that the paths through D' are precisely the paths through D. Furthermore, there exist no more sideways edges whose companion is t as all $s \in S$ are no longer sideways edges and no new sideways edges with t as their companion have been introduced in D'.

To fully eliminate all sideways edges from D, we need to apply this sideways edge elimination procedure iteratively as follows: At every step, choose a companion $t \in \mathrm{dom}(\beta)$ eligible for the elimination procedure at the greatest possible height, that is with $|t|$ maximal among the eligible companions, and perform the elimination procedure on it. Continue doing this until there are no more sideways edges left.

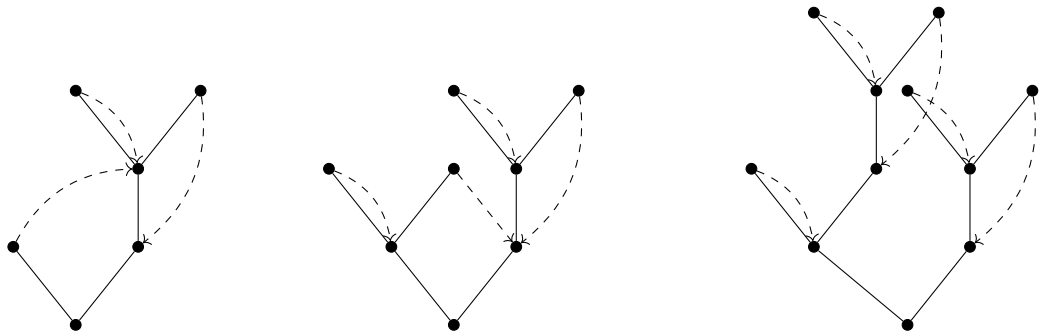
Now it remains to argue that this process always terminates. Calling the companion being eliminated the *target*, observe that one elimination step of the process can only introduce new sideways edges into the resulting abstract cyclic derivation in two ways:

1. If the subtree above the target already contains a sideways edge, this may remain a sideways edge after the grafting process. Observe that while this may introduce new sideways edges, it does not introduce any more companions of sideways edges as the subtree above the target already contains a bud which has a sideways edge to any such companion.
2. A backwards edge whose bud sits above the target but whose companion is below it turn into a sideways edge above the grafting points (see the first step Example 2.21 for an example of this). Observe that the companions of such new sideways edges are always more shallow than the target currently being eliminated.

In summary, each transformation step completely removes one sideways edge companion at the greatest depth while only introducing new sideways edge companions which are more shallow. As there can only ever be finitely many companions at each depth level, the process will always eventually move on to eliminating sideways edges with companions at lower depths until finally reaching depth 0, meaning all sideways edges have been eliminated. □

Remark. Similar transformation procedures have already been given in the literature. Notably, Brotherston's dissertation [4] describes essentially the same algorithm, both for concrete cyclic derivations and his abstract representation of the same. We chose to also present it in this thesis as it is a procedure of common interest in the field of cyclic proof theory.

Example 2.21. For an example of the algorithm described in Proposition 2.20, consider the sequence of cyclic trees below which depicts the state of the cyclic tree after each grafting step. For the sake of legibility, we have opted not to label the nodes. The first step eliminates all sideways edges (in this case, the unique one) with the highest companion. During this elimination step, a new sideways edge is introduced at the left side of the grafted fork. This sideways edge is then eliminated in the second step.



Remark. The procedure yielding efficient ACDs described in Proposition 2.17 is unique to ACDs or similar abstract representations of cyclic derivations as it relies on the fact that the morphisms of trace categories can be composed. The algorithm of Proposition 2.20, on the other hand, can, in principle, also be used to transform sequent-labeled cyclic derivations into ones with backward edges.

Chapter 3

Infinite Derivation Trees

The overarching goal of this chapter is to fully formalize the transformation procedure from cyclic derivations into ACDs over a trace category which we sketched in Section 2.1. For this, we first formalize infinite derivation trees (Sections 3.1 and 3.2) and their trace conditions (Section 3.3) in a categorical manner. By viewing cyclic derivations as regular infinite derivation trees, these categorical notions can be used to naturally specify the desired transformation procedure as we demonstrate in Section 3.4.

3.1 Infinite Labeled Trees and Construction Rule-Sets

We begin by defining a slightly generalized notion of infinite derivations: Infinite labeled trees which are derived according to certain "rule-sets".

Definition 3.1. An *infinite labeled tree* is a triple (Σ, S, λ) consisting of some labeling alphabet Σ, a finitely branching tree $S \subseteq \omega^*$, which we call the tree's *skeleton*, and a labeling function $\lambda : S \to \Sigma$ which assigns to each node of S a label from Σ. For convenience, we require the children of a node to be in sequence, that is, for $s \in \omega^*$ with $si \in S$ we also have $sj \in S$ for any $j < i$.

Remark. Although we usually speak of "infinite labeled trees", nothing in Definition 3.1 prevents the skeleton S from being finite. A more accurate, although much more awkward, description would thus be "possibly infinite labeled trees".

Remark. We consider such labeled trees instead of simply working with trees $T \subseteq \Sigma^*$ because these (i) allow for nodes with multiple children labeled with the same $a \in \Sigma$ and (ii) impose a fixed order on the children of a node. Both of these properties give fine-grained control over which kinds of branches run through a labeled tree which is crucial for representing derivation trees with trace conditions.

Example 3.2. The triple $T_{ab} \coloneqq (\{a, b\}, S_{ab}, \lambda_{ab})$ with

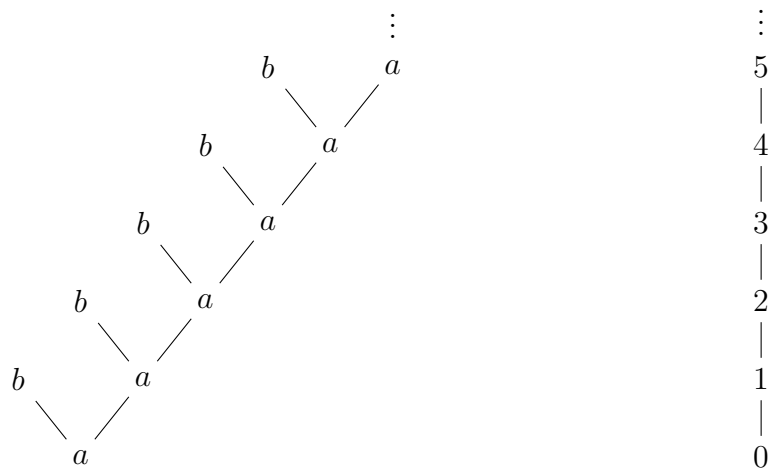
$$S_{ab} \coloneqq \{1^n \mid n \in \omega\} \cup \{1^n 0 \mid n \in \omega\} \qquad \lambda_{ab}(s) \coloneqq \begin{cases} a & s \text{ matches } 0^* \\ b & s \text{ matches } 0^*1 \end{cases}$$

describes the infinite, right-leaning tree depicted on the left below.

The triple $T_\omega \coloneqq (\omega, S_\omega, \lambda_\omega)$ with

$$S_\omega \coloneqq \{0^n \mid n \in \omega\} \qquad \lambda_\omega(0^n) \coloneqq n$$

describes the line-shaped tree on the right.



Definition 3.3. A *construction rule-set* is given by a triple $(\Sigma, \mathcal{R}, r)$ where Σ is an alphabet, $\mathcal{R}$ is a set of rules and $r : \mathcal{R} \to \Sigma \times \Sigma^*$ assigns to each a *rule interpretation.* We call a construction rule-set *deterministic* if for each $\sigma \in \Sigma$ there is at most one $R \in \mathcal{R}$ with $\sigma = \pi_1(r(R))$.

Definition 3.4. A tree $T = (\Sigma, S, \lambda)$ is *constructed according to a rule-set* $(\Sigma, \mathcal{R}, r)$ if there exists a tree $J = (\mathcal{R}, S, \rho)$ such that for each $s \in S$, if $r(\rho(s)) = (\sigma, \sigma_0\sigma_1 \dots \sigma_n)$ then $\mathrm{bd}(s) = n$, $\lambda(s) = \sigma$ and $\lambda(si) = \sigma_i$ for $i \leq n$. In this case, we call J the *construction justification.*

Example 3.5. The tree T_{ab} from Example 3.2 is constructed according to the finite, deterministic rule-set $(\{a, b\}, \{a, b\}, r_{ab})$ with

$$r_{ab}(a) \coloneqq (a, ba) \qquad r_{ab}(b) \coloneqq (b, \varepsilon)$$

The tree T_ω is constructed according to the deterministic rule-set $(\omega, \omega, r_\omega)$

$$r_\omega(n) \coloneqq (n, n + 1)$$

In both cases, the construction justification is the tree itself.

Example 3.6. The derivations of any sequent calculus are given by a construction rule-set, namely the sequent calculus' derivation rules. If SEQ is the set of legal sequents then the corresponding rule set is $(\mathrm{SEQ}, \mathcal{R}, r)$ where for every instance of a derivation rule

$$\mathrm{R}\frac{\Gamma_0 \quad \dots \quad \Gamma_n}{\Delta}$$

we add $R(\Delta, \Gamma_0 \dots \Gamma_n)$ to $\mathcal{R}$ and take

$$r(R(\Delta, \Gamma_0 \dots \Gamma_n)) \coloneqq (\Delta, \Gamma_0 \dots \Gamma_n)$$

Remark. The idea of Example 3.6 also applies in the converse direction. We may view any construction rule-set $(\Sigma, \mathcal{R}, r)$ as an — often admittedly quite strange — sequent calculus on the "sequents" in Σ. Motivated by this intuition, we can represent the fact that $r(R) = (\sigma, \sigma_0 \dots \sigma_n)$ for some $R \in \mathcal{R}$ as

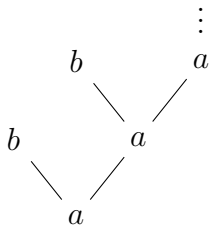
$$\mathrm{R}\frac{\sigma_0 \quad \dots \quad \sigma_n}{\sigma}$$

The regular trees, a subclass of labeled infinite trees, are of special interest in this thesis as they correspond naturally to regular ∞-derivations, which in turn correspond to cyclic derivations.

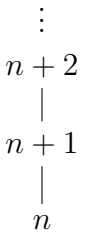
Definition 3.7. A tree $T = (\Sigma, S, \lambda)$ is a *subtree* of another tree $T' = (\Sigma, S', \lambda')$ if there exists some $s \in \omega^*$ such that $S = S'[s] \coloneqq \{t \in \omega^* \mid st \in S'\}$ and $\lambda'(t) = \lambda(st)$.

Definition 3.8. A tree T is *regular* if it has only finitely many distinct subtrees.

Example 3.9. The tree T_{ab} is regular, as it only has two distinct subtrees: the singular node b and itself, namely,



On the other hand, T_ω is not regular, as for each $n \in \omega$, the tree



is a subtree of T_ω, giving rise to countably many distinct subtrees.

The property of a tree being regular can be expressed in terms of construction rule-sets. This property is crucial for our categorical rendering of regular ∞-derivations in Section 3.2.

Definition 3.10. A tree $T = (\Sigma, S, \lambda)$ is called a *relabeling* of a tree $T' = (\Sigma', S, \lambda')$ if there exists a function $f : \Sigma' \to \Sigma$ such that $\lambda = f \circ \lambda'$.

Proposition 3.11. *A tree is regular if and only if it is a relabeling of a tree constructed according to a finite, deterministic rule-set.*

Proof.

→ Let $T = (\Sigma, S, \lambda)$ be such that $\mathrm{SubTree}(T)$ is finite. Consider the tree $T' = (\mathrm{SubTree}(T), S, \lambda')$ where $\mathrm{SubTree}(T)$ is the set of subtrees of T and where

$$\lambda'(s) \coloneqq T[s] \coloneqq (\Sigma, S[s], t \mapsto \lambda(st))$$

i.e. the labeling which labels each node $s \in S$ with the subtree of T starting at s. It is easy to see that T is a relabeling of T' via $f : \mathrm{SubTree}(T) \to \Sigma$ defined as

$$f(\Sigma, S'', \lambda'') \coloneqq \lambda''(\varepsilon)$$

It thus remains tho show that T' is constructed according to a finite, deterministic rule-set. For this, pick $(\mathrm{SubTree}(T), \mathrm{SubTree}(T), r)$ where

$$r(T'') \coloneqq (T'', T''[0] \dots T''[n]) \quad \text{if } \mathrm{bd}_{T''}(\varepsilon) = n + 1$$

As in the examples above, T' itself serves as the construction justification. The rule-set is finite as $\mathrm{SubTree}(T)$ is and deterministic as $\pi_i \circ r$ is the identity function.

← Now suppose $T = (\Sigma, S, \lambda)$ was a relabeling of $T' = (\Sigma', S, \lambda')$, which in turn was constructed according to a finite, deterministic rule-set. As the rule-set is deterministic, which rule to apply at a given $s \in S$ is fully determined by $\lambda(s)$. This means that $T'[s] = T'[s']$ for $s, s' \in S$ whenever $\lambda'(s) = \lambda'(s')$. Furthermore, $\mathrm{im}(\lambda')$ has to be finite as there are only finitely many rules in the rule-set deriving T'. Taken together, this means that $\mathrm{SubTree}(T')$ is finite. Now, as T is a relabeling of T', every subtree of T must be a relabeling of a subtree of T'. But this means that $\mathrm{SubTree}(T)$ is finite as well. □

Remark. Motivated by this result, we often call finite deterministic rule-sets *regular*.

Remark. There is another equivalent condition for regularity: An infinite labeled tree is regular if and only if it is generated by unfolding a finite, labeled graph. This equivalence bridges the gap between ∞-derivations and cyclic derivations: Any ∞-derivation which can be described by a cyclic derivation has to be regular. For this reason, the formalism we develop in this chapter uses regular ∞-derivations as a "stand-in" for cyclic derivations.

We forgo proving this equivalence at this point of the chapter as we also prove it, somewhat implicitly, in Section 3.4 where we give conversion procedures between ∞-derivations and abstract cyclic derivations.

3.2 Categories of Derivations

This section gives a categorical account of construction rule-sets and accordingly derived trees. This allows us to give a completely formal description of the process for transforming the concrete derivations of a cyclic system into abstract cyclic derivations, which we lay out in Section 3.4.

We begin by defining the category of tree derivations induced by a construction rule-set. Throughout this section, we denote the rules of construction rule-sets in the familiar style of derivation system rules and speak of sequents $\Gamma, \Delta \in \textsc{Seq}$, rather than labels $\sigma, \sigma' \in \Sigma$, to aid legibility and make clear the ultimate purpose of these definitions: formalizing ∞-proofs of sequent calculi.

Definition 3.12. Given a construction rule-set $\mathcal{R} = (\textsc{Seq}, \mathcal{R}, r)$, its associated *category of derivations* $\mathcal{D}_\mathcal{R}$ has as its objects the finite lists of sequents $\Gamma \in \textsc{Seq}$. The morphisms between two such lists, say of type $[\Gamma_0, \ldots, \Gamma_n] \to [\Delta_0, \ldots, \Delta_m]$, correspond to lists of derivations $[\Pi_0, \ldots, \Pi_m]$ where each Π_i derives Δ_i and which, taken together, have the open leaves $\Gamma_0, \ldots, \Gamma_n$ as sketched below.

$$\dfrac{\overset{\Gamma_0}{\vdots} \quad \ldots \quad \overset{\Gamma_{i_0}}{\vdots}}{\Pi_0 \quad \Delta_0} \qquad \dfrac{\overset{\Gamma_{i_0+1}}{\vdots} \quad \ldots \quad \overset{\Gamma_{i_1}}{\vdots}}{\Pi_1 \quad \Delta_1} \qquad \ldots \qquad \dfrac{\overset{\Gamma_{i_{m-1}+1}}{\vdots} \quad \ldots \quad \overset{\Gamma_n}{\vdots}}{\Pi_m \quad \Delta_m}$$

Importantly, the assumptions Γ_i are used linearly and in-order. That is, each Γ_i corresponds to precisely one open leaf of one of the derivations Π_j and Γ_i is used to the left of Γ_{i+1}. For the sketch above, this means $0 \leq i_1 \leq i_2 \leq \ldots \leq i_m = n$.

The identity morphisms on any $[\Gamma_0, \ldots, \Gamma_n]$ are given by the derivations

$$\Gamma_0 \qquad \Gamma_1 \qquad \ldots \qquad \Gamma_n$$

that is, those in which no rule is applied and each conclusion is "passed on" as an assumption. The composition operation is given by simply "plugging in" each derivation into the corresponding open leaf. For example, if we were to compose the morphism $[\Pi_0, \ldots, \Pi_m] : [\Gamma_0, \ldots, \Gamma_n] \to [\Delta_0, \ldots, \Delta_m]$ with, for sake of simplicity, a morphism $[\Pi'] : [\Delta_0, \ldots, \Delta_m] \to [\Theta]$ this would result in the morphism $[\Gamma_0, \ldots, \Gamma_n] \to [\Theta]$ corresponding to the derivation sketched below.

$$\dfrac{\dfrac{\overset{\Gamma_0}{\vdots} \quad \ldots \quad \overset{\Gamma_{i_0}}{\vdots}}{\Pi_0 \quad \Delta_0} \qquad \dfrac{\overset{\Gamma_{i_0+1}}{\vdots} \quad \ldots \quad \overset{\Gamma_{i_1}}{\vdots}}{\Pi_1 \quad \Delta_1} \qquad \ldots \qquad \dfrac{\overset{\Gamma_{i_{m-1}+1}}{\vdots} \quad \ldots \quad \overset{\Gamma_n}{\vdots}}{\Pi_m \quad \Delta_m}}{\Pi' \quad \Theta}$$

Remark. In particular, this means that each rule $R \in \mathcal{R}$ with $r(R) = (\Delta, \Gamma_0 \ldots \Gamma_n)$, corresponding to a derivation rule

$$\dfrac{\Gamma_0 \quad \ldots \quad \Gamma_n}{\Delta}$$

in turn induces a morphism $R : [\Gamma_0, \ldots, \Gamma_n] \to [\Delta]$ in $\mathcal{D}_\mathcal{R}$. The axioms of a sequent calculus then correspond to morphisms in $\textsc{Hom}_{\mathcal{D}_\mathcal{R}}([], [\Delta])$.

Remark. The general idea behind this categorical rendering of derivation trees, representing finite derivations as morphisms, is not new. For example, similar representations of proofs, or equivalently terms in type theories, are explored in [15, 16, 22] to name only a few examples.

The finite lists over a set form a monoid with regard to the list appending operation $A +\!\!+ B$. This monoidal structure extends to any category of derivations $\mathcal{D}_\mathcal{R}$.

Proposition 3.13. *Any category of derivations induced by a construction rule-set* $\mathcal{R} = (\textsc{Seq}, \mathcal{R}, r)$ *is a monoidal category for the functor* $- \otimes - : \mathcal{D}_\mathcal{R} \times \mathcal{D}_\mathcal{R} \to \mathcal{D}_\mathcal{R}$ *given by*

$$A \otimes B \coloneqq A +\!\!+ B \qquad (D : A \to B) \otimes (D' : A' \to B') \coloneqq D +\!\!+ D' : A \otimes A' \to B \otimes B'$$

and the neutral object given by the empty list $[]$.

Proof. For functoriality of $-\otimes-$, observe that, given $A = [\Gamma_0, \ldots, \Gamma_n]$ and $B = [\Gamma'_0, \ldots, \Gamma'_m]$ appending the two lists "no progress" proofs $1_A : A \to A$ and $1_B : B \to B$ into $1_A \otimes 1_B$ yields precisely the list of "no progress" proofs of $A \otimes B = A +\!\!+ B$, meaning identities are preserved. Similarly, since the derivations are always "plugged in" in-order during composition, list-appending and derivation composition distribute over each other, leading the desired fact of $(D_3 \circ D_1) \otimes (D_4 \circ D_2) = (D_3 \otimes D_4) \circ (D_1 \otimes D_2)$ for morphisms $D_1 : A \to B, D_2 : A' \to B', D_3 : B \to C, D_4 : B' \to C'$.

As the lists on Seq themselves form a monoid with regards to list-appending, all of the law-witnessing natural transformations can simply be taken as the identity morphisms. For example, the identity morphism of some $A \in \mathrm{Ob}(\mathcal{D}_\mathcal{R})$ can also be presented as $1_A : A \otimes [] \to A$ because $A \otimes [] = A +\!\!+ [] = A$. □

This monoidal structure gives rise to natural notions of derivation and an associated normal form for morphisms in $\mathcal{D}_\mathcal{R}$.

Definition 3.14. We call a morphism $D : A \to B$ a *derivation* if B is a singleton $[\Delta]$.

For any given $D : A \to B$ if there are derivations $D_0 : A_0 \to B_0, \ldots, D_n : A_n \to B_n$ such that $D = D_0 \otimes \ldots \otimes D_n$ we call $D_0 \otimes \ldots \otimes D_n$ the *derivation normal-form of* D.

Proposition 3.15. *Every morphism has a unique derivation normal-form.*

Proof. Follows from the fact that we defined morphisms as lists of derivations. □

The aim of our definition of $\mathcal{D}_\mathcal{R}$ is to capture the notion of trees constructed according to the rule-set $\mathcal{R}$ in a categorical manner. For finite trees, this is quite simple: For any given sequent $\Delta \in \mathrm{Seq}$, the Hom-set $\mathrm{Hom}_{\mathcal{D}_\mathcal{R}}([], [\Delta])$ contains precisely all of the finite trees which are labeled by Δ at their root and are constructed according to $\mathcal{R}$. For infinite labeled trees, the definition is a bit more subtle.

Definition 3.16. Each rule set $\mathcal{R}$ induces a *semi-category of progressing derivations* $\mathcal{D}^\mathcal{P}_\mathcal{R}$. Its objects are the same as those of $\mathcal{D}_\mathcal{R}$ and any morphism $D : A \to B$ of $\mathcal{D}_\mathcal{R}$ with derivation normal form $D = D_0 \otimes \ldots \otimes D_n$ is a morphism of $\mathcal{D}^\mathcal{P}_\mathcal{R}$ if and only if there is no $i \leq n$ and $\Delta \in \mathrm{Seq}$ such that $D_i = 1_{[\Delta]}$.

Remark. Intuitively, the semi-category of progressing derivations restricts $\mathcal{D}_\mathcal{R}$ to only those morphisms $D : [\Gamma_0, \ldots, \Gamma_n] \to [\Delta_0, \ldots, \Delta_m]$ which "progress" the derivation process of each Δ_i by at least one rule-application. Observe also that, as $1_{[]} = []$ contains no derivations, all of its derivations vacuously progress and thus $1_{[]} \in \mathrm{Hom}_{\mathcal{D}^\mathcal{P}_\mathcal{R}}([], [])$.

Definition 3.17. An *infinite tree constructed according to* $\mathcal{R}$ is a contra-variant semi-functor $T : \omega^{\mathrm{op}} \to \mathcal{D}^\mathcal{P}_\mathcal{R}$ with $T(0) = [\Delta]$ for some $\Delta \in \mathrm{Seq}$.

Remark. If the construction rule-set $\mathcal{R}$ describes a sequent calculus, we also call such semi-functors ∞-*derivations*.

Example 3.18. For an example, recall the tree T_{ab} from Example 3.2 and its constructing rule-set $\mathcal{R}_{ab} \coloneqq (\{a, b\}, \mathcal{R}_{ab}, r_{ab})$. We can represent it as $T : \omega \to \mathcal{D}^{\mathcal{P}}_{\mathcal{R}_{ab}}$ with

$$T(0) \coloneqq [a] \qquad\qquad T(i+1) \coloneqq [b, a]$$

and

$$T(0 < 1) \coloneqq \left[R_a \frac{b \quad a}{a}\right] \qquad\qquad T(i+1 < i+2) \coloneqq \left[R_b \frac{}{b}, R_a \frac{b \quad a}{a}\right]$$

where $R_a : [b, a] \to [a]$ and $R_b : [] \to [b]$. Note that this already specifies $T : \omega \to \mathcal{D}^{\mathcal{P}}_{\mathcal{R}_{ab}}$ fully. For example, by the definition of composition in $\mathcal{D}^{\mathcal{P}}_{\mathcal{R}_{ab}}$ we know that

$$T(0 < 2) \coloneqq \left[\frac{\dfrac{}{b} \quad \dfrac{b \quad a}{a}}{a}\right]$$

Indeed $T(0 < n)$ will always yield an initial segment of depth n of the infinite derivation tree which ultimately derives T_{ab} from $\mathcal{R}_{ab}$.

An important point to observe is that this is not the only possible way of representing T_{ab} as such a semi-functor. For example, consider the semi-functor

$$T'(0) \coloneqq [a] \qquad\qquad T'(i+1) \coloneqq [b, a]$$

with

$$T'(0 < 1) \coloneqq \left[\frac{\dfrac{}{b} \quad \dfrac{b \quad a}{a}}{a}\right] \qquad T'(i+1 < i+2) \coloneqq \left[R_b \frac{}{b}, R_a \frac{b \quad a}{a}\right]$$

which also describes a partial approximation of the infinite derivation tree demonstrating that T_{ab} can be derived from $\mathcal{R}_{ab}$. While T and T' describe the same derivation tree, T' can in some sense be seen as doing so "faster" than T as $T(0 < n)$ is of depth n while $T'(0 < n)$ is of depth $n + 1$. Indeed, there are many different semi-functors $T : \omega \to \mathcal{D}^{\mathcal{P}}_{\mathcal{R}}$ which all describe the same infinite derivation tree.

Remark. The contra-variance of these semi-functors $T : \omega^{\mathrm{op}} \to \mathcal{D}^{\mathcal{P}}_{\mathcal{R}}$ stems naturally from the fact that in the semi-category of progressing derivations $\mathcal{D}^{\mathcal{P}}_{\mathcal{R}}$, a derivation rule

$$\frac{\Gamma_0 \quad \dots \quad \Gamma_n}{\Delta}$$

corresponds to a morphism $D : [\Gamma_0, \dots, \Gamma_n] \to [\Delta]$, i.e. a "mapping" of assumptions to conclusions. This means that derivation trees in $\mathcal{D}^{\mathcal{P}}_{\mathcal{R}}$ naturally "grow backwards", that is, to extend a derivation $D : [\Gamma_0, \dots, \Gamma_n] \to [\Delta]$, one pre-composes it with another morphism $D' : [\Gamma'_0, \dots, \Gamma'_m] \to [\Gamma_0, \dots, \Gamma_n]$ to obtain a "deeper derivation" $D \circ D' : [\Gamma'_0, \dots, \Gamma'_m] \to [\Delta]$. The semi-category on ω grows precisely in the opposite direction: To increase a map $0 < n$ to the map $0 < n + 1$, one post-composes $(n < n + 1) \circ (0 < n)$. Thus the arrows need to be reversed by these semi-functors to match the intuition that the derivation described by $T : \omega^{\mathrm{op}} \to \mathcal{D}^{\mathcal{P}}_{\mathcal{R}}$ grows alongside the numbers of ω. While there is no technical obstacle to defining it this way, the current definition seems more aesthetically pleasing and natural, at least in cases in which the construction rule-set is a sequent calculus, as one deduces *from* assumption *to* conclusions.

Remark. The requirement that the semi-functors have codomain $\mathcal{D}^{\mathcal{P}}_{\mathcal{R}}$ guarantees that for any $T : \omega^{\mathrm{op}} \to \mathcal{D}^{\mathcal{P}}_{\mathcal{R}}$ which describes an infinite derivation (i.e. one of unbounded depth), the depth of $T(0 < n+1)$ is always at least of depth $n+1$ and any open leaf of $T(0 < n)$ has been extended by at least one rule application to obtain $T(0 < n+1)$. This rules out nonsensical "derivation trees" such as $T : \omega^{\mathrm{op}} \to \mathcal{D}_{\mathcal{R}}$ with

$$T(i) = [\Delta] \qquad T(i < j) = 1_{[\Delta]}$$

However, observe that the above guarantee only applies if $T : \omega^{\mathrm{op}} \to \mathcal{D}^{\mathcal{P}}_{\mathcal{R}}$ describes an infinite derivation. Any finite derivation tree $D : [] \to [\Delta]$ can be described by a functor

$$T(0) \coloneqq [\Delta] \qquad T(i+1) \coloneqq [] \qquad T(0 < 1) \coloneqq D \qquad T(1 < n) \coloneqq 1_{[]}$$

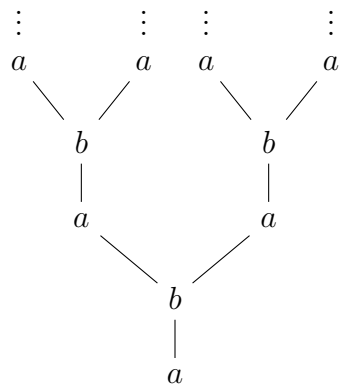
which stops progressing the derivation after $T(0 < 1)$. This nicely parallels Definition 3.1 which also allows for trees of finite depth.

Inspired by Proposition 3.11 we give a categorical characterization of regularity.

Definition 3.19. An infinite tree $T : \omega^{\mathrm{op}} \to \mathcal{D}^{\mathcal{P}}_{\mathcal{R}}$ is called *regular* if there exists a tree $R : \omega^{\mathrm{op}} \to \mathcal{D}^{\mathcal{P}}_{\mathcal{R}'}$ derived according to a regular rule-set $\mathcal{R}'$ and a functor $F : \mathcal{D}^{\mathcal{P}}_{\mathcal{R}'} \to \mathcal{D}^{\mathcal{P}}_{\mathcal{R}}$ such that $T = F \circ R$.

Example 3.20. This definition is a little more complex than a simple rephrasing of Proposition 3.11. The functor $F : \mathcal{D}^{\mathcal{P}}_{\mathcal{R}'} \to \mathcal{D}^{\mathcal{P}}_{\mathcal{R}}$ assigns to each derivation rule $R' : [\Gamma_1, \ldots, \Gamma_n] \to [\Delta]$ of $\mathcal{R}'$ a morphism $F(R')$ from $\mathcal{D}^{\mathcal{P}}_{\mathcal{R}}$. Observe, however, that $F(R)$ need not be a derivation as nothing in the definition above prevents $F([\Delta]) = [\Delta'_0, \ldots, \Delta'_m]$ for $0 < m$. Furthermore, the derivations in $F(R')$ could be of greater depth than 1. Intuitively, this could be described as the functor F being able to "encode more information" than the relabeling $f : \Sigma' \to \Sigma$ of Proposition 3.11.

For an example, consider the following regular tree



Clearly, it is constructed according to the rule-set $\mathcal{R} \coloneqq (\{a, b\}, \{a, b\}, r)$ with $r(a) = (a, b)$ and $r(b) = (b, aa)$. Furthermore, it cannot be the relabeling of any tree constructed according to a regular rule-set with fewer rules.

However, this changes when moving to our categorical representation of infinite labeled trees. Consider the rule-set $\mathcal{R}' \coloneqq (\{a\}, \{R'\}, r')$ with $r'(R') \coloneqq (a, aa)$ which describes an infinite binary tree. The regularity of the tree above can be justified with the functor

$F : \mathcal{D}^{\mathcal{P}}_{\mathcal{R}'} \to \mathcal{D}^{\mathcal{P}}_{\mathcal{R}}$ given by

$$F([]) \coloneqq [] \qquad F([a]) \coloneqq [a] \qquad F(R') \coloneqq \left[\frac{\mathrm{a} \quad \mathrm{a}}{\frac{\mathrm{b}}{\mathrm{a}}}\right]$$

even though $\mathcal{R}'$ only has one rule.

This difference in "expressive power" does not change the notion of regularity exhibited by these two different definitions as the functor representation can only ever "compress" finitely many rule-applications into the image of a rule from $\mathcal{R}'$.

We close this section with two technical definitions that make it easier to work in the setting of ∞-derivation where any given infinite tree can be represented by many different functors $T : \omega^{\mathrm{op}} \to \mathcal{D}^{\mathcal{P}}_{\mathcal{R}}$.

Definition 3.21. We may represent any derivation $D : [\Gamma_0, \ldots, \Gamma_n] \to [\Delta]$ in $\mathcal{D}^{\mathcal{P}}_{\mathcal{R}}$ as a labeled tree $T_D = (\mathcal{R}, S, \lambda)$ such that $\pi_1(r(\lambda(\varepsilon))) = \Delta$ and for every $s \in S \setminus \mathrm{Leaf}(S)$ we have $r(\lambda(s)) = (\Delta', \Gamma'_0 \ldots \Gamma'_m)$ where $\mathrm{bd}(s) = m + 1$ and for each $0 \leq i \leq m$ we know $\pi_1(r(\lambda(si))) = \Gamma'_i$. Furthermore, if $\mathrm{Leaf}(S) = \{s_0, \ldots, s_m\}$ is ordered lexicographically then $\pi_2(r(\lambda(s_i))) = \Gamma_{l_{i-1}+1} \ldots \Gamma_{l_i}$ for $-1 = l_{-1} \leq l_0 \leq \ldots \leq l_n = n$, i.e. the "open assumptions" of the leafs of S are precisely $[\Gamma_0, \ldots, \Gamma_n]$.

For a $D : \omega^{\mathrm{op}} \to \mathcal{D}^{\mathcal{P}}_{\mathcal{R}}$ we define $\lim D \coloneqq (\mathcal{R}, \bigcup S_i, \bigcup \lambda_i)$ where $(\mathcal{R}, S_i, \lambda_i) \coloneqq T_{D(0<i)}$.

Proposition 3.22. *For $D : \omega^{op} \to \mathcal{D}^{\mathcal{P}}_{\mathcal{R}}$, $\lim D$ is a well-formed construction justification.*

Remark. We can now formally express the intuitive notion that two given derivations $D, D' : \omega^{\mathrm{op}} \to \mathcal{D}^{\mathcal{P}}_{\mathcal{R}}$ *represent the same infinite tree* by the statement $\lim D = \lim D'$.

Definition 3.23. We call $D : \omega^{\mathrm{op}} \to \mathcal{D}^{\mathcal{P}}_{\mathcal{R}}$ a *step-wise* ∞-derivation if, in derivation normal form, $D(i < i+1) = D_0 \otimes \ldots \otimes D_n$ with $D_0, \ldots, D_n \in \mathcal{R}$ or $D(i < i+1) = 1_{[]}$ for every $i \in \omega$.

Remark. Observe that if $D, D' : \omega^{\mathrm{op}} \to D^{\mathcal{P}}_{\mathcal{R}}$ are step-wise ∞-derivations, we have $D = D'$ if and only if $\lim D = \lim D'$. The step-wise derivation can thus serve as the *canonical categorical representation* of any given infinite labeled tree derived according to the construction rule-set $\mathcal{R}$.

3.3 Trace-Interpretations and Branch-Tracking Categories

This section introduces trace-interpretations and path tracking categories. Trace-interpretations are a formal means for associating the branches through derivations with paths through trace categories. Branch-tracking categories allow for branches through ∞-derivations to be defined. When combined, the two concepts induce a notion of ∞-proofs. Throughout this section, we use $\sigma, \sigma_1, \sigma_2, \ldots$ to denote various natural isomorphisms between product functors.

Definition 3.24. We call a contra-variant functor $I : \mathcal{D}^{\mathrm{op}}_{\mathcal{R}} \to \mathcal{T}$ a *trace-interpretation* if $\mathcal{T}$ is a trace category with finite products and I is a strong monoidal functor between the

monoids $- \otimes -$ and $- \times -$ of respectively $\mathcal{D}_{\mathcal{R}}$ and $\mathcal{T}$. We denote the main witnessing natural isomorphism by $\eta_{A,B} : I(A \otimes B) \simeq I(A) \times I(B)$.

Remark. The contra-variance again stems from the fact that we took the morphisms describing derivations to lead from the assumptions to the conclusion, whereas the paths through a derivation lead from the conclusion to the assumptions.

Remark. The natural isomorphism entails that $I([\Gamma_0, \ldots, \Gamma_n]) \simeq I([\Gamma_0]) \times \ldots \times I([\Gamma_n])$, as we prove in Corollary 3.27. This means that $I([\Gamma_i])$ can be viewed as "the I-image of Γ_i". Based on this intuition, we from now on write $I(\Gamma_i) \coloneqq I([\Gamma_i])$ for brevity.

The following result states that intuitive accounts of "how to represent branches in derivations as trace maps", such as the one we gave for cyclic arithmetic in Example 2.9, already contain enough information to fully specify a trace-interpretation. It also allows us to specify trace-interpretations for the cyclic systems in Chapter 6 without having to always consider the technicalities of monoidal functors.

Lemma 3.25. *Let the construction rule-set $\mathcal{R}$ specify a sequent calculus on sequents* Seq *and $\mathcal{T}$ be a trace category with finite products. Fix a mapping $i : \textsc{Seq} \to Ob(\mathcal{T})$ together with, for each $R \in \mathcal{R}$ of shape*

$$R\frac{\Gamma_0 \quad \ldots \quad \Gamma_n}{\Delta}$$

a choice of morphisms $\tau_j^R : i(\Delta) \to i(\Gamma_j)$ in $\mathcal{T}$ for $j \leq n$. This induces a trace-interpretation $I : \mathcal{D}_{\mathcal{R}}^{op} \to \mathcal{T}$ with $I(\Gamma) = i(\Gamma)$ for $\Gamma \in \textsc{Seq}$.

Proof. We begin by specifying what the functor does on objects. Observe that any object $A \in \mathrm{Ob}(\mathcal{D}_{\mathcal{R}})$ is of the form $[\Gamma_0] +\!\!+ \ldots +\!\!+ [\Gamma_n] +\!\!+ []$. Thus we can recursively specify

$$I([\Gamma] +\!\!+ A) \coloneqq i(\Gamma) \times I(A) \qquad\qquad I([]) \coloneqq 1$$

Fix for each $A, B \in \mathrm{Ob}(\mathcal{D}_{\mathcal{R}})$ the isomorphism $\eta_{A,B} : I(A \otimes B) \to I(A) \times I(B)$ which can be constructed from the natural isomorphisms witnessing the monoidality of $- \times -$ in $\mathcal{T}$.

For the functor actions on morphisms, notice similarly that all morphisms of $\mathcal{D}$ can be obtained via the $\otimes$-functor and compositions from the rules $R \in \mathcal{R}$. We may thus specify the functor actions on morphism recursively via their "construction history". As such, we take, denoting the unique morphism from 1 to $F(\Delta)$ by $!_{F(\Delta)}$,

$$I(D' \circ D) \coloneqq I(D) \circ I(D') \qquad I\left(\mathrm{R}\frac{\Gamma_i \quad \ldots \quad \Gamma_n}{\Delta}\right) \coloneqq \langle \tau_0^R, \langle \ldots \langle \tau_n^R, !_{F(\Delta)} \rangle \rangle \rangle$$

and for $D : A \to B$, $D' : A' \to B'$ we take

$$I(D \otimes D') \coloneqq \eta^{-1}_{A \otimes A'} \circ (I(D) \circ \pi_1 \times I(D') \circ \pi_2) \circ \eta_{B \otimes B'}$$

As most morphisms in $\mathcal{D}_{\mathcal{R}}$ have more than one "construction history", we need to check that the result of $I(D)$ is independent of which "construction history" of D was used to compute it. But this follows readily from the fact that both $- \otimes -$ and $- \times -$ are monoids and that the $\eta_{A \otimes B}$ are isomorphisms.

It remains to argue that $I : \mathcal{D}_{\mathcal{R}}^{\mathrm{op}} \to \mathcal{T}$ is monoidal. This follows from the fact that both $\eta_{A,B} : I(A \otimes B) \simeq I(A) \times I(B)$ and $\epsilon : I([]) \simeq 1$ are obtained via the natural isomorphisms between product functors and thus make all of the required diagrams commute. □

The fact that a trace-interpretation is a monoidal functor gives rise to some nice structural properties and guarantees which can be expressed via certain natural isomorphisms.

Proposition 3.26. *If $I : \mathcal{D}_{\mathcal{R}}^{op} \to \mathcal{T}$ is a trace-interpretation then there are isomorphisms*

$$\eta_{A_0,\ldots,A_n} : I(A_0 \otimes \ldots \otimes A_n) \simeq I(A_0) \times \ldots \times I(A_n)$$

naturally in $A_0, \ldots, A_n \in Ob(\mathcal{D}_{\mathcal{R}})$.

Proof. We can define $\eta_{A_0,\ldots,A_n} : I(A_0 \otimes \ldots \otimes A_n) \simeq I(A_0) \times \ldots \times I(A_n)$ inductively

$$\eta_A \coloneqq 1_{I(A)} \qquad \eta_{A_0,\ldots,A_n,A_{n+1}} \coloneqq \sigma \circ (1^{n-1} \times \eta_{A_n,A_{n+1}}) \circ \eta_{A_0,\ldots,A_n \otimes A_{n+1}}$$

where we employ short-hands $1^n \times f \coloneqq 1_{I(A_0)} \times \ldots \times 1_{I(A_n)} \times f$ and take the unique isomorphism $\sigma : A_0 \times \ldots \times A_{n-1} \times (A_n \times A_{n+1}) \simeq A_0 \times \ldots \times A_{n+1}$. Naturality follows once again from the definition of $I(D \otimes D')$ and the fact that these isomorphisms are induced by the monoid-witnessing natural transformations of products in $\mathcal{T}$. □

Corollary 3.27. *If $I : \mathcal{D}_{\mathcal{R}}^{op} \to \mathcal{T}$ is a trace-interpretation then*

$$\eta_{\Gamma_0,\ldots,\Gamma_n} : I([\Gamma_0, \ldots, \Gamma_n]) \simeq I([\Gamma_0]) \times \ldots \times I([\Gamma_n])$$

for any $\Gamma_0, \ldots, \Gamma_n \in$ Seq.

This leads us to introduce a projection operation on the images of a trace-interpretation which "projects out" the I-image of one of the elements of a list.

Definition 3.28. Let $I : \mathcal{D}_{\mathcal{R}}^{\mathrm{op}} \to \mathcal{T}$ be a trace-interpretation and $\Gamma_0, \ldots, \Gamma_n \in$ Seq. We define the *i-th trace projection map*

$$\tau_i \coloneqq \pi_i \circ \eta_{\Gamma_0,\ldots,\Gamma_n} : I([\Gamma_0, \ldots, \Gamma_n]) \to I(\Gamma_i)$$

Remark. If $I : \mathcal{D}_{\mathcal{R}}^{\mathrm{op}} \to \mathcal{T}$ was obtained via Lemma 3.25 then $\tau_i \circ I(R) = \tau_i^R$ for any $R \in \mathcal{R}$.

The following proposition essentially states that all trace-interpretations can be viewed as being defined "trace-wise" in the style of Lemma 3.25.

Proposition 3.29. *If $I : \mathcal{D}_{\mathcal{R}}^{op} \to \mathcal{T}$ is a trace-interpretation and $D : [\Gamma_0, \ldots, \Gamma_n] \to [\Delta]$ is a derivation then*

$$I(D) = \eta^{-1}_{\Gamma_0,\ldots,\Gamma_n} \circ \langle \tau_0 \circ I(D), \ldots, \tau_n \circ I(D) \rangle$$

Proof. For this, simply observe that

$$\begin{aligned} I(D) &= \eta^{-1}_{\Gamma_0,\ldots,\Gamma_n} \circ \eta_{\Gamma_0,\ldots,\Gamma_n} \circ I(D) \\ &= \eta^{-1}_{\Gamma_0,\ldots,\Gamma_n} \circ \langle \pi_0, \ldots, \pi_n \rangle \circ \eta_{\Gamma_0,\ldots,\Gamma_n} \circ I(D) \\ &= \eta^{-1}_{\Gamma_0,\ldots,\Gamma_n} \circ \langle \tau_0 \circ I(D), \ldots, \tau_n \circ I(D) \rangle \end{aligned}$$

□

The following lemma, while very technical in nature, can be understood as asserting that trace projection maps are well-behaved with regards composition in $\mathcal{D}_{\mathcal{R}}$.

Lemma 3.30. *If $I : \mathcal{D}_{\mathcal{R}}^{op} \to \mathcal{T}$ is a trace-interpretation and*

$$D' : [\Gamma_0, \ldots, \Gamma_n] \to [\Delta] \qquad D_i : [\Gamma_i] \to [\Theta_{l_{i-1}+1}, \ldots, \Theta_{l_i}]$$

are morphisms in $\mathcal{D}_{\mathcal{R}}$ where $-1 = l_{-1} \leq l_0 \leq \ldots \leq l_n$ then

$$\tau_{l_{i-1}+k} \circ I(D' \circ (D_0 \otimes \ldots \otimes D_n)) = \tau_k \circ I(D_i) \circ \tau_i \circ I(D')$$

where $i \leq n$ and $k \leq l_i - l_{i-1}$.

Proof. The claim can be proven as follows

$$\begin{aligned}
&\tau_{l_{i-1}+k} \circ I(D' \circ (D_0 \otimes \ldots \otimes D_n)) \\
\coloneqq\ &\pi_{l_{i-1}+k} \circ \eta_{\Theta} \circ I(D_0 \otimes \ldots \otimes D_n) \circ I(D') \\
=\ &\pi_{l_{i-1}+k} \circ \eta_{\Theta} \circ I(D_0 \otimes \ldots \otimes D_n) \circ \eta_{\Gamma}^{-1} \circ \eta_{\Gamma} \circ I(D') \\
=\ &\pi_{l_{i-1}+k} \circ \eta_{\Theta} \circ \eta_{\Theta'_{\otimes}}^{-1} \circ (I(D_0) \times \ldots \times I(D_n)) \circ \eta_{\Gamma} \circ I(D') && (1) \\
=\ &\pi_{l_{i-1}+k} \circ \sigma \circ (\eta_{\Theta^0} \times \ldots \times \eta_{\Theta^n}) \circ (I(D_0) \times \ldots \times I(D_n)) \circ \eta_{\Gamma} \circ I(D') && (2) \\
=\ &\pi_k \circ \pi_i \circ (\eta_{\Theta^0} \circ I(D_0) \times \ldots \times \eta_{\Theta^n} \circ I(D_n)) \circ \eta_{\Gamma} \circ I(D') && (3) \\
=\ &\pi_k \circ \eta_{\Theta^i} \circ I(D_i) \circ \pi_i \circ \eta_{\Gamma} \circ I(D') && (() \\
\coloneqq\ &\tau_k \circ I(D_i) \circ \tau_i \circ I(D')
\end{aligned}$$

where we use short-hands $\eta_{\Theta} \coloneqq \eta_{\Theta_0,\ldots,\Theta_{l_n}}$, $\eta_{\Gamma} \coloneqq \eta_{\Gamma_0,\ldots,\Gamma_1}$, $\eta_{\Theta'_{\otimes}} \coloneqq \eta_{(\Theta_0 \otimes \ldots \otimes \Theta_{l_0}),\ldots,(\Theta_{l_{n-1}+1} \otimes \ldots \otimes \Theta_{l_n})}$ and $\eta_{\Theta^i} \coloneqq \eta_{\Theta_{l_{i-1}+1},\ldots,\Theta_{l_i}}$. Further, we take

$$\sigma : (I(\Theta_0) \times \ldots \times I(\Theta_{l_0})) \times \ldots \times (I(\Theta_{l_{n-1}+1} \times \ldots I(\Theta_{l_n}))) \simeq I(\Theta_0) \times \ldots \times I(\Theta_{l_n})$$

Step (1) follows by the naturality of η_{Γ}, step (2) by Lemma A.8 — whose proof is quite technical and thus relegated to Appendix A — and steps (3) and (4) by the relationships of projections, product functors and the natural isomorphisms between product functors. □

We close this section by formally defining the global trace condition on ∞-derivations and thus the notion of ∞-proofs. To this end, we first define the auxiliary branch-tracking categories.

Definition 3.31. Let $\mathcal{T}$ be a trace category. The *branch-tracking category of* $\mathcal{T}$, denoted by $\mathcal{B}_{\mathcal{T}}$, has as its objects pairs (X, φ) where X is a finite set and $\varphi : X \to \mathrm{Ob}(\mathcal{T})$ assigns to each element of X an object of $\mathcal{T}$. Given two such objects $(X, \varphi), (Y, \psi)$ a morphism $R : (X, \varphi) \to (Y, \psi)$ is a relation $R \subseteq \Sigma x \in X.\Sigma y \in Y.\textsc{Hom}_{\mathcal{T}}(\varphi(x), \psi(y))$. The identity is given by

$$1_{(X,\varphi)} \coloneqq \{(x, x, 1_{\varphi(x)}) \mid x \in X\}$$

and two such relations $R : (X, \varphi) \to (Y, \psi), R' : (Y, \psi) \to (Z, \theta)$ compose as

$$R' \circ R \coloneqq \{(x, z, \tau' \circ \tau) \mid (x, y, \tau) \in R, (y, z, \tau') \in R'\}$$

i.e. by composing the relation and the transition maps.

Definition 3.32. Fix a path $P : \omega \to \mathcal{B}_{\mathcal{T}}$. A path $Q : \omega \to \mathcal{T}$ is called a *branch through* P if there exists a sequence $x_i : \Pi i \in \omega.X_i$ where $P(i) = (X_i, \varphi_i)$ and for each $i \in \omega$ we have $Q(i) = \varphi_i(x_i)$ and $(x_i, x_{i+1}, Q(i < i + 1))$.

Path-tracking categories allow us to give a reasonable definition of branches through ∞-derivations, and thus of ∞-proofs.

Definition 3.33. Fix a category of derivations $\mathcal{D}_\mathcal{R}$ and an associated trace-interpretation $I : \mathcal{D}_\mathcal{R}^{\mathrm{op}} \to \mathcal{T}$. We define the *branch-tracking functor* $B : \mathcal{D}_\mathcal{R}^{\mathrm{op}} \to \mathcal{B}_\mathcal{T}$ as follows. We take

$$B([\Gamma_1, \dots, \Gamma_n]) \coloneqq (\{1, \dots, n\}, i \mapsto I(\Gamma_i))$$

For a $D : [\Gamma_1, \dots, \Gamma_n] \to [\Delta_1, \dots, \Delta_m]$, consider its derivation normal form $D = D_1 \otimes \dots \otimes D_m$ with $D_i : [\Gamma_{k_{i-1}+1}, \dots, \Gamma_{k_i}] \to [\Delta_i]$ for $-1 = k_{-1} \leq k_0 \leq \dots \leq k_m = n$ and take

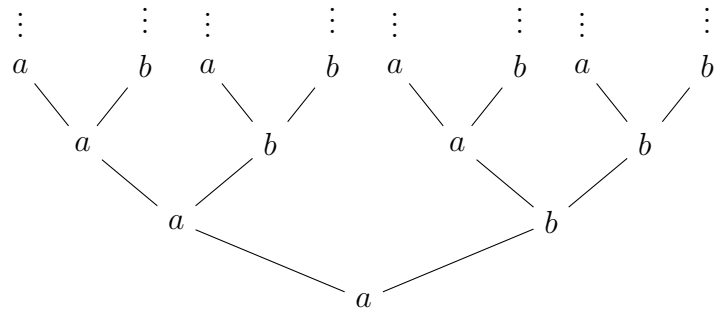
$$B(D) \coloneqq \{(i, k_{i-1} + 1 + l, \tau_l \circ I(D_i)) \mid 1 \leq i \leq n, 0 \leq l \leq k_i - k_{i-1} - 1\}$$

i.e. precisely the traces from the conclusion of some D_i to one of its premises.

Remark. Let us elaborate on the relationship between the concepts of "branch" and "path". Formally, every branch is a path $P : \omega \to \mathcal{T}$. However, the branches through a derivation D are "special" paths: Those which start at the root of the derivation and in which $P(i)$ is $I(\Delta)$ for some sequent occurring in $D(i)$. Notably, every proper subpath of P (i.e. not P itself) will most likely not be a branch through D anymore.

Definition 3.34. Let $D : \omega^{\mathrm{op}} \to \mathcal{D}_\mathcal{R}^\mathcal{P}$ be an ∞-derivation and $I : \mathcal{D}_\mathcal{R}^{\mathrm{op}} \to \mathcal{T}$ a trace-interpretation. $P : \omega \to \mathcal{T}$ is a *branch through* D if it is a branch through $B \circ D : \omega \to \mathcal{B}_\mathcal{T}$. We call D an *∞-proof* if it satisfies the *global trace condition*: Every branch $P : \omega \to \mathcal{T}$ through D satisfies the trace condition of $\mathcal{T}$.

Example 3.35. For an example of how the branch-tracking functor works for a concrete derivation, consider the regular tree depicted below



which can be derived via the following regular construction rule-set $\mathcal{R}$

$$\mathrm{A}\frac{a \quad b}{a} \qquad\qquad \mathrm{B}\frac{a \quad b}{b}$$

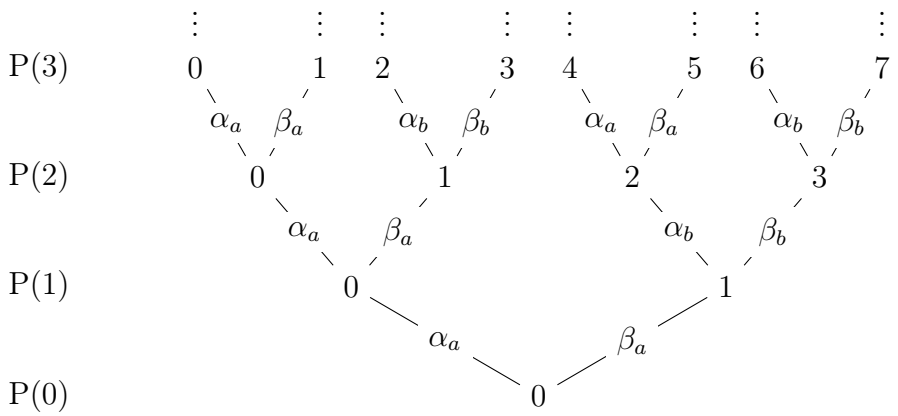
as described by the step-wise derivation $D : \omega^{\mathrm{op}} \to \mathcal{D}_\mathcal{R}^\mathcal{P}$ defined via

$$D(0) \coloneqq [a] \quad D(n+1) \coloneqq [a, b]^n \qquad \text{where } [a, b]^0 \coloneqq [a, b] \quad [a, b]^{n+1} \coloneqq [a, b]^n \otimes [a, b]^n$$

$$D(0 < 1) \coloneqq \left[\mathrm{A}\frac{a \quad b}{a}\right] \qquad D(1 < 2) \coloneqq \left[\mathrm{A}\frac{a \quad b}{a}, \mathrm{B}\frac{a \quad b}{b}\right]$$

$$D(n+2 < n+3) \coloneqq D(n+1 < n+2) \otimes D(n+1 < n+2)$$

Now fix some trace category $\mathcal{T}$ and a trace-interpretation $I : \mathcal{D}_\mathcal{R} \to \mathcal{T}$. When denoting $\alpha_a \coloneqq \tau_0 \circ I(A), \beta_a \coloneqq \tau_1 \circ I(A)$ and $\alpha_b \coloneqq \tau_0 \circ I(B), \beta_b \coloneqq \tau_1 \circ I(B)$, then the path $P \coloneqq B \circ D : \omega \to \mathcal{B}_\mathcal{T}$ can be drawn as follows:



where $P(i) = (X, \varphi)$ with $\varphi(2n) := I(a)$ and $\varphi(2n+1) := I(b)$ for $n \in \omega$. Given this path through the branch-tracking category $\mathcal{B}_{\mathcal{T}}$, it is quite easy to discern that from among the $Q_i : \omega \to \mathcal{T}$ given below, Q_1 and Q_2 are branches through D while Q_3 is not.

$$Q_1(i) := I(a) \qquad Q_1(i < i+1) := \alpha_a$$

$$Q_2(i) := \begin{cases} I(a) & \text{if } i = 2n \\ I(b) & \text{otherwise} \end{cases} \qquad Q_2(i < i+1) := \begin{cases} \beta_a & \text{if } i = 2n \\ \alpha_b & \text{otherwise} \end{cases}$$

$$Q_3(i) := I(b) \qquad Q_3(i < i+1) := \beta_b$$

To close the section, we prove that the global trace condition from Definition 3.34, and thus the induced notion of ∞-provability, is well defined with regards to different representations of the same infinite derivation.

Proposition 3.36. *Let $D, D' : \omega^{op} \to \mathcal{D}^{\mathcal{P}}_{\mathcal{R}}$ with $\lim D = \lim D'$. Then D is an ∞-proof if and only if D' is.*

Proof. Observe that the step-wise derivation $D_S : \omega^{\mathrm{op}} \to \mathcal{D}^{\mathcal{P}}_{\mathcal{R}}$ of $\lim D = \lim D'$ is unique. It thus suffices to prove that for every $D : \omega^{\mathrm{op}} \to \mathcal{D}^{\mathcal{P}}_{\mathcal{R}}$, D is an ∞-proof if and only if the step-wise ∞-derivation $D_S : \omega^{\mathrm{op}} \to \mathcal{D}^{\mathcal{P}}_{\mathcal{R}}$ of $\lim D$ is an ∞-proof.

For one direction, we prove that any branch $P : \omega \to \mathcal{T}$ through D is the subpath of some branch $P \subseteq P'$ through D_S. If P is a branch through D, that means there is a sequence $(k_i)_{i\in\omega}$ such that, if $D(i) = [\Gamma^i_0, \ldots, \Gamma^i_n]$, $P(i) = I(\Gamma^i_{k_i})$ and k_{i+1} is the index of the j_ith premise of $D^i_{k_i}$ where $D(i < i+1) = D^i_0 \otimes \ldots \otimes D^i_n$ in derivation normal form and furthermore $P(i < i+1) = \tau_{j_i} \circ I(D^i_{k_i}) : I(\Gamma^i_{k_i}) \to I(\Gamma^{i+1}_{k_{i+1}})$. Taking $\lim D = (\mathcal{R}, S, \lambda)$ it is easy to see that this essentially describes a sequence $(s_i \in S)_{i\in\omega}$ through $\lim D$ such that $\pi_1(r(\lambda(s_i))) = \Gamma^i_{k_i}$ and $s_i < s_{i+1}$. We can then find a similar sequence of indices $(l_i)_{i\in\omega}$ inducing a path $P' : \omega^{\mathrm{op}} \to \mathcal{D}^{\mathcal{P}}_{\mathcal{R}}$ through D_S "tracing the s_i", with the same properties. That is, if $D_S(i) = [\Delta^i_0, \ldots, \Delta^i_n]$ then $P'(i) = I(\Delta_{l_i})$ and l_{i+1} is the index of the p_ith premise of $R^i_{l_i}$ where $D(i < i+1) = R^i_0 \otimes \ldots \otimes R^i_n$ in DNF and $P'(i < i+1) = \tau_{p_i} \circ I(R^i_{l_i})$. However, only a subsequence of the l_i matches the path s_i through $\lim D$, that is, there exists a subsequence q_i of l_i such that $\pi_1(r(\lambda(s_i))) = I(\Delta^{q_i}_{l_{q_i}})$. Notably, this means that, taking $Q : \omega \to \omega$ to be the semi-functor $i \mapsto q_i$, we already know that $P' = P \circ Q$ on objects. It then remains to prove the same on morphisms. For this, observe that we have

$$D(i < i+1) = (X_n \otimes R^{q_i+n}_{l_{q_i+n}} \otimes Y_n) \circ \ldots \circ (X_1 \otimes R^{q_i+1}_{l_{q_i+1}} \otimes Y_1) \circ (X_0 \otimes R^{q_i}_{l_{q_i}} \otimes Y_0)$$

where $n = q_i - q_{i+1}$ and the X_i and Y_i are morphisms in $\mathcal{D}^{\mathcal{P}}_{\mathcal{R}}$ "filling in" the assumptions not touched by the sequence l_i. By repeatedly applying Lemma 3.30, we can then conclude

$$P(i < i+1) = \tau_{j_i} \circ I(D(i < i+1)) = \tau_{p_{q_i+n}} \circ I(R^{q_i+n}_{lq_i+n}) \circ \ldots \circ \tau_{p_{q_i}} \circ I(R^{q_i}_{l_{q_i}}) = P'(Q(i < i+1))$$

which means $P' = P \circ Q$ and thus $P \subseteq P'$ as desired.

The converse direction, i.e. finding for each branch $P : \omega^{\mathrm{op}} \to \mathcal{T}$ through D_S a subpath $P' \subseteq P$ branching through D is analogous. □

3.4 Converting between Regular ∞-Derivations and Abstract Cyclic Derivations

We close this chapter by combining all of the definitions from the previous sections to give formal transformation procedures between regular ∞-derivations and abstract cyclic derivations.

Proposition 3.37. *Fix a trace-interpretation $I : \mathcal{D}^{op}_{\mathcal{R}} \to \mathcal{T}$ and let $D : \omega^{op} \to \mathcal{D}^{\mathcal{P}}_{\mathcal{R}}$ be a regular ∞-derivation. Then there exists an abstract cyclic derivation $D_C = (C, Tr)$ such that*

(i) any branch P through D is a path through D_C

(ii) any path P through D_C is the subpath of some branch P' through D

Proof. As D is regular, there exists a regular rule-set $\mathcal{R}'$, a translation $F : \mathcal{D}_{\mathcal{R}'} \to \mathcal{D}_{\mathcal{R}}$ and an ∞-derivation $R : \omega^{\mathrm{op}} \to \mathcal{D}^{\mathcal{P}}_{\mathcal{R}'}$ with $D = F \circ R$. As $\mathcal{R}'$ is finite, we may assume that $\mathcal{R}' = (n, \mathcal{R}', r')$, meaning that its underlying alphabet are the natural numbers $i < n$ for some $n \in \omega$. For simplicity, we also assume that for any $R' \in \mathcal{R}'$ with $r'(R') = (i, m_0 \ldots m_j)$ all the m_i are distinct, a state of affairs that can always be brought about by renaming some premises and adding "renamed copies" for their derivation rules.

We begin by constructing the cyclic tree $C = (T, \beta)$ by inductively defining

$$T_0 := \{R(0)\}$$
$$T_{i+1} := \bigcup_{uk \in T_i} \left\{ ukl_j \mid \frac{l_1 \quad \ldots \quad l_n}{k} \text{ rule in } \mathcal{R}', 0 \leq j < n, \text{no letter appears in } uk \text{ twice} \right\}$$

and set $T := \{u \mid R(0)u \in T_i, i \leq n\}$, thus ensuring that $\varepsilon \in T$. It is easily observed that T describes a tree following the rules of $\mathcal{R}'$. There are two kinds of $u \in \mathrm{Leaf}(T)$: If $u = vi$ such that $A : [] \to [i]$ is a derivation rule of $\mathcal{R}'$ then we call u an axiomatic leaf. Otherwise, there exist $v, v' \in n^*$ with $u = viv'i$ for a unique i which is repeated twice in u, in which case we call u a bud. For each bud $u \in \mathrm{Leaf}$ we pick $\beta(u) := vi$, i.e. jumping back to the first occurrence of i in u. Then $C := (T, \beta)$ is the cyclic tree underlying the abstract cyclic derivation D_C.

For the functor $\mathrm{Tr} : \mathcal{P}_C \to \mathcal{T}$, we take $\mathrm{Tr}(ui) := I(F(R([i])))$. Observe that this means that $\mathrm{Tr}(u) = \mathrm{Tr}(\beta(u))$ for any bud u. For the morphism-part of Tr, it suffices to specify what to map direct successor edges to. Thus if $ui, uim_j \in S$ then $\mathcal{R}'$ has the rule

$$\mathrm{A}\frac{m_1 \quad \ldots \quad m_k}{i}$$

meaning $A : [m_1, \dots, m_k] \to [i]$ is a morphism of $\mathcal{R}'$. Then take

$$\mathrm{Tr}(ui < uim_j) \coloneqq \tau_j \circ I(F(R(A))) : I(F(R([i]))) \to I(F(R([m_j])))$$

as $\mathrm{Tr}(ui) = I(F(R([i])))$ and $\mathrm{Tr}(uim_j) = I(F(R([m_j])))$.

Observe that any path $P : \omega \to \mathcal{T}$ through D is also a path through R w.r.t. the trace-interpretation $I \circ F : \mathcal{D}^{\mathrm{op}}_{\mathcal{R}'} \to \mathcal{T}$. Using this observation, it is an easy, albeit somewhat involved, exercise to prove the properties (i) and (ii). □

Remark. Importantly, if an ACD D_C was obtained from $D : \omega^{\mathrm{op}} \to \mathcal{D}^{\mathcal{P}}_{\mathcal{R}}$ via this procedure, D_C is an abstract cyclic proof iff D is an ∞-proof because of the conditions (i) and (ii).

Remark. As noted in Example 2.13, the procedure above can be used to obtain, given a trace-interpretation, an ACD for any given cyclic derivation: Simply unfold the cyclic proof into an ∞-derivation and recall that any ∞-derivation obtained in this manner is regular. Then apply the procedure to said regular ∞-derivation.

Proposition 3.38. *If $D = (C, Tr : \mathcal{P}_C \to \mathcal{T})$ is a abstract cyclic derivation, then there exists a regular rule-set $\mathcal{R}$, a trace-interpretation $I : \mathcal{D}^{\mathcal{P}}_{\mathcal{R}} \to \mathcal{T}$ and an ∞-derivation $D' : \omega^{op} \to \mathcal{D}^{\mathcal{P}}_{\mathcal{R}}$ such that*

(i) any path P through D is the subpath of some branch P' through D'

(ii) any branch P through D' is a path through D

Proof. For $C = (T, \beta)$, denote by $\sim$ the reflexive, symmetric, transitive closure of β and by $T_\sim$ the quotient of T by $\sim$. For $t \in T$, let $[\![t]\!] \in T_\sim$ be the class in $T_\sim$ containing t. Furthermore, observe that for each $A \in T_\sim$, there exists precisely one $t \in A$ with $t \notin (A \setminus \mathrm{Leaf}(T))$ and call it $q(A)$.

We define $\mathcal{R} \coloneqq (T_\sim, T_\sim, r)$ with

$$r(A) \coloneqq (A, [\![s_0]\!] \dots [\![s_m]\!]) \text{ where } \mathrm{Chld}(q(A)) = \{s_0, \dots, s_m\}$$

It is easy to observe that $\mathcal{R}$ is indeed regular. Then, take $D' : \omega^{\mathrm{op}} \to \mathcal{D}^{\mathcal{P}}_{\mathcal{R}}$ to be the unique step-wise derivation with $D'(0) = [[\![\varepsilon]\!]]$. To obtain the interpretation $I : \mathcal{D}^{\mathrm{op}}_{\mathcal{R}} \to \mathcal{T}$, apply Lemma 3.25 choosing

$$i(A) \coloneqq \mathrm{Tr}(q(A)) \qquad \text{and} \qquad \tau^A_j \coloneqq \mathrm{Tr}(q(A)s_i) \quad \text{where } \mathrm{Chld}(q(A)) = \{s_0, \dots, s_m\}.$$

Again, it is easy to verify that the conditions (i) and (ii) hold. □

Remark. The proofs of the two previous propositions together also demonstrate, somewhat implicitly, that every regular infinite tree (equivalently, ∞-derivation) is generated by unfolding a finite graph (equivalently, ACD): Proposition 3.38 describes the unfolding procedure while Proposition 3.37 describes how the finite graph can be derived from a regular tree.

Chapter 4

Abstract Theorems of Cyclic Proof Theory

This chapter employs the abstract framework laid out in Chapters 2 and 3 to state and subsequently prove theorems of cyclic proof theory in such a way that they can be applied uniformly to any cyclic derivation system whose cyclic content is captured by said abstractions. The theorems in Section 4.1 are generalized variants of theorems which are wide-spread in the literature and whose proofs make use of automata theory. In Section 4.2 we give an alternative soundness condition whose equivalence to the GTC follows from Ramsey's theorem and outline a novel GTC decision procedure based on it.

4.1 Proofs via Automata Theory

We begin by recalling the definitions of infinite word automata.

Definition 4.1. An *infinite word automaton* is a quintuple $\mathfrak{A} = (Q, \Sigma, \Delta, q_0, C)$ where Q is a finite set of states, Σ is a finite alphabet. We call $q_0 \in Q$ the *starting state*, the relation $\Delta \subseteq Q \times \Sigma \times Q$ the *transition relation* and the set $C \subseteq Q^\omega$ is the *acceptance condition*. If for each $q \in Q, s \in \Sigma$ there is at most one $q' \in Q$ such that $(q, s, q') \in \Delta$, we call Δ and the automaton $\mathfrak{A}$ overall *deterministic*.

Given a word $\sigma \in \Sigma^\omega$, the sequence $\rho \in Q^\omega$ is called a *run of* $\mathfrak{A}$ *on* σ if $\rho_0 = q_0$ and for each $i \in \omega$ we have $(\rho_i, \sigma_i, \rho_{i+1}) \in \Delta$. A run ρ is *accepting* if $\rho \in C$. A word σ *is accepted by* $\mathfrak{A}$ if there exists an accepting run of $\mathfrak{A}$ on σ. The set $L(\mathfrak{A}) \coloneqq \{\sigma \in \Sigma^\omega \mid \sigma \text{ is accepted by } \mathfrak{A}\}$ is the *language of* $\mathfrak{A}$.

Definition 4.2.

1. For a subset $F \subseteq Q$ the *Büchi condition* is defined by

$$\textsc{Büchi}(F) \coloneqq \{\rho \in Q^\omega \mid \mathrm{Inf}(\rho) \cap F \neq \emptyset\}$$

2. For a coloring $c : Q \to \omega$ the *parity condition* is defined by

$$\textsc{Parity}(c) \coloneqq \{\rho \in Q^\omega \mid \max\{c(q) \mid q \in \mathrm{Inf}(\rho)\} \text{ is even}\}$$

Remark. If $\mathfrak{A} = (Q, \Sigma, \Delta, q_0, C)$ with $C = \textsc{Büchi}(F)$ or $C = \textsc{Parity}(c)$, we call $\mathfrak{A}$ a Büchi or parity automaton, respectively. If the type of automaton is clear from the context, we sometimes omit the construction function, for example simply denoting a Büchi automaton by $(Q, \Sigma, \Delta, q_0, F)$.

The usefulness of automata theory in the field of cyclic proof theory stems from the fact that the trace condition of many cyclic derivation systems can be recognized by infinite word automata. Transferring to our framework, all proofs in this section rely on the automata recognizability of the trace condition of a trace category.

Definition 4.3. Pick a trace category $\mathcal{T}$. We say that its trace condition is *deterministically parity-recognizable* if for any finite set T of morphisms $\tau : A \to B$ of $\mathcal{T}$ and starting object $S \in \mathrm{Ob}(\mathcal{T})$, there exists a deterministic parity automaton with T as its alphabet which accepts a sequence

$$\tau_0\tau_1\tau_2\tau_3\tau_4 \ldots \in T^\omega \qquad \text{with } \tau_i : A_i \to B_i$$

if and only if the following

$$P(i) \coloneqq A_i \qquad P(i < i+1) \coloneqq \tau_i$$

induces a path $P : \omega \to \mathcal{T}$ with $P(0) = S$ and P satisfies the trace condition of $\mathcal{T}$.

Similar concepts, such as a trace category being *non-deterministically Büchi-recognizable* are defined analogously.

The first theorem we prove is the fact that, for cyclic derivation system's whose trace condition is recognizable by some infinite word automaton, it is decidable whether a cyclic derivation constitutes a proof. Such an automata-based GTC decision algorithm was first put forward by Sprenger and Dam [34] and has since become a staple of cyclic proof theory (for further instance of it, see e.g. [9, 20, 33]).

Theorem 4.4. *Let $D = (C, Tr : \mathcal{P}_C \to \mathcal{T})$ be an abstract cyclic derivation and the trace condition of $\mathcal{T}$ be non-deterministically Büchi-recognizable. It is decidable whether D satisfies the global trace condition.*

Proof. For $C = (T, \beta)$, consider the set of trace maps in $\mathcal{T}$ given by

$$M := \{\mathrm{Tr}(uv) \mid u <_+ v \in T\} \cup \{1_{\mathrm{Tr}(u)} \mid u \in \mathrm{dom}(\beta)\}$$

which is finite as T is. Per assumption, there exists a non-deterministic Büchi-automaton $\mathfrak{A}$ which recognizes valid paths consisting of maps from M satisfying the trace condition of $\mathcal{T}$. We construct a second non-deterministic Büchi-automaton $\mathfrak{B} = (T, M, \varepsilon, \Delta, T)$ with

$$\Delta \coloneqq \{(u, \mathrm{Tr}(uv), v) \mid u <_+ v \in T\} \cup \{(u, 1_{\mathrm{Tr}(u)}, \beta(u)) \mid u \in \mathrm{dom}(\beta)\}$$

Clearly, $\mathfrak{B}$ accepts precisely the single-step paths through D. A path $P : \omega \to \mathcal{T}$ thus is a path through D if and only if it is a subpath of a path recognized by $\mathfrak{B}$. Thus we can decide whether D satisfies the global trace condition by checking $L(\mathfrak{B}) \subseteq L(\mathfrak{A})$. As this is decidable for non-deterministic Büchi-automata [26], we are done. □

Instances of the next two theorems are not quite as common in cyclic proof theory as they come with a strong precondition: they only apply to finite derivation systems. That means, for example, that they do not apply to cyclic arithmetic as its derivation system includes countably infinitely many instances of the ∃R-rule alone. Note that, in Chapter 6, we present two cyclic derivation systems to which this theorem does apply. Notably, the proofs of the next two theorems also make use of the abstract notion of derivation for Chapter 3.

Definition 4.5. An *infinite tree automaton* is a quintuple $\mathfrak{A} = (Q, \Sigma, q_0, \Delta, C)$. Same as for infinite word automata, Q and Σ are a finite set of states and a finite alphabet, $q_0 \in Q$ is the starting state and $C \subseteq Q^\omega$ is an acceptance condition. Differing from infinite word automata, the transition relation is $\Delta \subseteq Q \times \Sigma \times Q^*$ and furthermore required to be finite.

Given a labeled tree $T = (\Sigma, S, \lambda)$, we call $R = (Q, S, \rho)$ a *run of* $\mathfrak{A}$ *on* T if $\rho(\varepsilon) = q_0$ and for each $s \in S$ with $\mathrm{bd}(s) = n$ we have $(\rho(s), \lambda(s), \rho(s0) \dots \rho(s(n-1))) \in \Delta$. If $\sigma \in Q^\omega$ is such that there exists $\tau \in \omega^\omega$ with $\tau \upharpoonright i \in S$ and $\rho(\tau \upharpoonright i) = \sigma_i$ for every $i \in \omega$ then we call σ *a path through* R. We call a run R *accepting* if $\sigma \in C$ for every path σ through R. A tree T is then *accepted by* $\mathfrak{A}$ if there exists an accepting run of $\mathfrak{A}$ on it.

To the best of our knowledge, the first instance of the following result in the literature is given by Niwiński and Walukiewicz for a tableaux system for the modal μ-calculus in [28].

Theorem 4.6. *Let $\mathcal{D}_\mathcal{R}$ be the category of derivations of a finite sequent calculus on* Seq*. Furthermore, let $I : \mathcal{D}_\mathcal{R}^{op} \to \mathcal{T}$ be a trace-interpretation into a trace category $\mathcal{T}$ whose trace condition is deterministically parity recognizable. Then there exists an ∞-proof of a sequent $\Gamma \in$ Seq if and only if there exists a regular ∞-proof of Γ.*

Proof. The right-to-left direction is immediate. We begin to prove the other direction by constructing a parity tree automaton $\mathfrak{B}$ which can recognize the ∞-proofs of Γ.

As the sequent calculus $\mathcal{R} = (\text{Seq}, \mathcal{R}, r)$ is finite, we know that the set $\mathcal{R} = \{R_0, \dots, R_n\}$ of rules generating $\mathcal{D}_\mathcal{R}$ is finite. Each such rule $D_i : [\Theta_i^0, \dots, \Theta_i^{m_i}] \to [\Gamma_i]$ corresponds to a morphism $I(R_i) : I(\Gamma_i) \to I([\Theta_i^0, \dots, \Theta_i^{m_i}])$ via the trace-interpretation I, which in turn have components $\tau_i^j \coloneqq \tau_j \circ I(R_i) : I(\Gamma_i) \to I(\Theta_i^j)$. Now consider the set

$$T \coloneqq \{\tau_i^j : I(\Gamma_i) \to I(\Theta_i^j) \mid 0 \leq i \leq n, 0 \leq j \leq m_i\}$$

By the parity recognizability of the trace condition of $\mathcal{T}$ there exists deterministic parity automaton $\mathfrak{A} \coloneqq (Q, T, \delta, q_0, c)$ which recognizes paths consisting of morphisms from T, starting at $I(\Gamma)$, which satisfy the trace condition of $\mathcal{T}$. With this, we can construct a non-deterministic parity tree automaton $\mathfrak{B} \coloneqq (\text{Seq} \times Q, \mathcal{R}, \Delta_\mathfrak{B}, (\Gamma, q_0), c \circ \pi_2)$ on infinite trees labeled by rules in $\mathcal{R}$ where we take $\Delta_\mathfrak{B} \coloneqq \bigcup_{R_i \in \mathcal{R}} \Delta_{R_i}$. specifically, for each derivation rule

$$R_i \frac{\Theta_i^0 \quad \dots \quad \Theta_i^{m_i}}{\Gamma_i}$$

we take

$$\Delta_{R_i} \coloneqq \left\{ \left((\Gamma_i, q), D_i, \left((\Theta_i^0, \delta(q, \tau_i^1)), \dots, (\Theta_i^{n_i}, \delta(q, \tau_i^{n_i})) \right) \right) \;\middle|\; q \in Q \right\}$$

i.e. the automaton may take steps corresponding to any applicable derivation rule, walking the state from Q along according to $\mathfrak{A}$ for the trace map τ_i^j chosen by the trace-interpretation I.

For the correctness of the automaton, it is easy to see by the choice of $\Delta_{\mathcal{B}}$ that $\mathfrak{B}$ has a run on some tree T at all if and only if T at least constitutes a construction justification of a valid ∞-derivation of Γ, i.e. if each rule is correctly applied. Furthermore, the automaton accepts if and only if the $\mathcal{T}$-path induced by each infinite path of the derivation tree described by the construction justification is accepted by $\mathfrak{A}$. Thus it is easy to see that $\mathfrak{A}$ will only accept construction justifications of ∞-proofs of Γ.

Taken together, the argument is thus as follows: If there is an ∞-proof $D : \omega^{\mathrm{op}} \to \mathcal{D}_{\mathcal{R}}^{\mathcal{P}}$ of Γ there is step-wise $D' : \omega^{\mathrm{op}} \to \mathcal{D}_{\mathcal{R}}^{\mathcal{P}}$ of Γ. Then the construction justification inherent in D' is recognized by $\mathfrak{B}$, meaning $L(\mathfrak{B})$ is not empty. By Corollary 8.20 of [13] this means there is a regular tree $J = (\mathcal{R}, S, \lambda) \in L(\mathfrak{B})$. As $\mathfrak{B}$ only recognizes construction justifications of ∞-proofs of Γ, this J must thus be such. Obtaining the regular ∞-proof $D'' : \omega^{\mathrm{op}} \to \mathcal{D}_{\mathcal{R}}^{\mathcal{P}}$ from this construction justification is straight-forward. □

Remark. An interesting application of the result above is establishing the equivalence between cyclic proofs and Cut-free cyclic proofs in cyclic derivation systems with a Cut-rule. Many Cut-elimination procedures for cyclic derivation systems in the literature (e.g. [3, 12, 31]) are corecursive algorithms which lazily transform cyclic proofs with Cut-applications into ∞-proofs without Cut-applications. If the Cut-free fragment of the derivation system is finite, the result above can then be applied to conclude that there must also exist a regular, Cut-free ∞-proof and thus a Cut-free cyclic proof, as is done in [31].

The construction employed in the proof of Theorem 4.6 also yields a decision procedure for ∞-provability.

Corollary 4.7. *Let $\mathcal{D}_{\mathcal{R}}$ be the category of derivations of a finite sequent calculus on* Seq*. Let furthermore $I : \mathcal{D}^{op} \to \mathcal{T}$ be a trace-interpretation into a trace category $\mathcal{T}$ whose trace condition is deterministically parity recognizable. Then the ∞-provability of any sequent $\Gamma \in$ Seq is decidable.*

Proof. The proof of Theorem 4.6 constructs a parity tree automaton $\mathfrak{B}$ such that $L(\mathfrak{B})$ is precisely the construction justifications of ∞-proofs of Γ. As the emptiness problem for parity tree automata is decidable (see Theorem 8.19 of [13]), we can thus decide if there exists an ∞-proof of Γ by deciding whether $L(\mathfrak{B})$ is empty. □

4.2 Proofs via the Ramsey Theorem

We begin this section by recalling Ramsey's theorem. For this, we employ the notation $[A]^n := \{X \subseteq A \mid |X| = n\}$ for $n \in \omega$.

Theorem 4.8 ([29], Theorem A)**.** *Let A be a countably infinite set and $n, m \leq \omega$. For any coloring $f : [A]^n \to m$ there exists a color $c \leq m$ and countably infinite $C \subseteq A$ such that for any $X \in [C]^s$ we have $f(X) = c$.*

When examining the applications of Ramsey's theorem in cyclic proof theory through our framework, many of them can be identified as instances of the following categorical corollary of Ramsey's theorem.

Definition 4.9. For every endomorphism $\tau : X \to X$ of a category $\mathcal{T}$, the *periodic* τ *path* is the path $\tau^\omega : \omega \to \mathcal{T}$ given by $\tau^\omega(i) \coloneqq X$ and $\tau^\omega(i < i+1) \coloneqq \tau$.

Definition 4.10. We call an endomorphism $\tau : X \to X$ idempotent if $\tau = \tau \circ \tau$.

Lemma 4.11. *Let $P : \omega \to \mathcal{T}$ be a path and $X \in Ob(\mathcal{T})$ such that $P(i) = X$ infinitely often. If $\textsc{Hom}_\mathcal{T}(X, X)$ is finite then $\tau^\omega \subseteq P$ for some idempotent $\tau : X \to X$.*

Proof. $P(i) = X$ holding infinitely often means there exists $Q \subseteq P$ such that $Q(i) = X$ for all $i \in \omega$. Then $Q(\{i, j\}) \coloneqq Q(i < j)$ induces a coloring $Q : [\omega]^2 \to \textsc{Hom}_\mathcal{T}(X, X)$ on ω. By the Ramsey theorem, there exists $\tau \in \textsc{Hom}_\mathcal{T}(X, X)$ and $M \subseteq Q$ such that for $i < j \in M$ we have $Q(i < j) = \tau$. Then $\tau^\omega = Q \circ S$ where $S(i) \coloneqq$ the ith least $m \in M$. The idempotence of τ follows as $\tau = Q(S(0 < 2)) = Q(S(1 < 2)) \circ Q(S(0 < 1)) = \tau \circ \tau$. □

This consequence of Ramsey's theorem allows us to derive an alternative soundness condition on ACDs which is equivalent to the GTC.

Definition 4.12. For any abstract derivation $D = (C, \mathrm{Tr})$ we define its *progressing semi-functor* $\mathrm{Tr}^S : \mathcal{P}^S_C \to \mathcal{T}$ as Tr restricted to the semi-category $\mathcal{P}^S_C$.

Remark. We employ the shorthand $\textsc{Hom}_{\mathrm{Tr}^s}(u, v) \coloneqq \textsc{Hom}_{\mathrm{im}(\mathrm{Tr}^S)}(\mathrm{Tr}^S(u), \mathrm{Tr}^S(v))$ to aid readability from this point forward.

Lemma 4.13. *Let $D = (C, Tr)$ be an abstract cyclic system such that for each $u \in dom(\beta)$ the set $\textsc{Hom}_{Tr^S}(u, u)$ is finite. Then D satisfies the global trace condition if and only if for every $u \in dom(\beta)$ and every idempotent $\tau \in \textsc{Hom}_{Tr^S}(u, u)$, the path $\tau^\omega : \omega \to \mathcal{T}$ satisfies the trace condition.*

Proof. First, suppose D satisfied the GTC. Then pick $u \in \mathrm{dom}(\beta)$ and $\tau : \mathrm{Tr}(u) \to \mathrm{Tr}(u)$ from $\mathrm{im}(\mathrm{Tr}^S)$. As τ is in the image of Tr^S, we know that there exists some $p \in \mathrm{Path}(u, u)$ with $|p| > 1$ and $\mathrm{Tr}(p) = \tau$. Then $\tau^\omega = \mathrm{Tr} \circ P$ for $P : \omega \to \mathcal{P}^S_C$ with $P(i) \coloneqq u$ and $P(i < i+1) \coloneqq p$, meaning τ^ω is a path through D and thus satisfies the trace condition.

Now suppose that all periodic paths on idempotent endomorphisms of buds satisfied the trace condition and pick some path $\mathrm{Tr} \circ P$ for $P : \omega \to \mathcal{P}^S_C$. Clearly, there exists $P \subseteq P'$ with $|P'(i < i+1)| = 2$ for all $i \in \omega$, i.e. a representation of P which does not "skip" any nodes. As P' describes an infinite path through C, there needs to be some $u \in \mathrm{dom}(\beta)$ with $P'(i) = u$ for infinitely many $i \in \omega$ and thus by Lemma 4.11 an idempotent $\tau : \mathrm{Tr}(u) \to \mathrm{Tr}(u)$ such that $\tau^\omega \subseteq \mathrm{Tr} \circ P' \subseteq \mathrm{Tr} \circ P$. Per assumption, τ^ω satisfies the trace condition, meaning $\mathrm{Tr} \circ P$ does so as well. □

Remark. By restricting our attention to $\textsc{Hom}_{\mathrm{Tr}^s}(u, u)$, we guarantee that $1_{\mathrm{Tr}(u)}$ can only occur in $\textsc{Hom}_{\mathrm{Tr}^s}(u, u)$ if there is some $p \in \mathrm{Path}_C(u, u)$ such that $\mathrm{Tr}(p) = 1_{\mathrm{Tr}(u)}$. This is important as in most trace categories — including all trace categories defined in this thesis — the path $1^\omega_{\mathrm{Tr}(u)}$ does not satisfy the trace condition. Naïvely including $1_{\mathrm{Tr}(u)}$ in

the collection of idempotent morphisms to consider when checking for GTC satisfaction would thus invalidate the condition given above.

This alternative soundness condition gives rise to an alternative GTC decision procedure. While this procedure could be formalized in any formal model of computation covered by the Church-Turing thesis, such as Turing machines or the λ-calculus, we forgo such a technical presentation, instead relying on the readers intuitive understanding of computability. To the best of our knowledge, we are the first in the literature to describe this procedure.

Theorem 4.14. *Let $\mathcal{T}$ be such that*

(1) from $\tau : X \to Y$ and $\tau' : Y \to Z$ one can compute $\tau' \circ \tau : X \to Z$

(2) for $\tau, \tau' : X \to Y$ one can decide whether $\tau = \tau'$

(3) for idempotent $\tau : X \to X$ one can decide whether $\tau^\omega : \omega \to \mathcal{T}$ satisfies the trace condition

Let further $D = (C, Tr : \mathcal{P}_C \to \mathcal{T})$ be an abstract cyclic system such that for $u, v \in T$ the set $\textsc{Hom}_{Tr^S}(u, v)$ is finite. Then it is decidable whether D satisfies the global trace condition.

Proof. By Lemma 4.13 we know that it suffices to check that for each $u \in \mathrm{dom}(\beta)$ and for all idempotent $\tau \in \textsc{Hom}_{\mathrm{Tr}^S}(u, u)$, τ^ω satisfies the trace condition. For this, we compute all of the Hom-sets and then check each morphism in question via procedure guaranteed by assumption (3). The Hom-sets are computed by an iterative procedure with base cases

$$H^0(u, v) \coloneqq \{\mathrm{Tr}(uv) \mid \text{if } u <_+ v\} \cup \{1_{\mathrm{Tr}(u)} \mid \text{if } u \in \mathrm{dom}(\beta) \text{ and } \beta(u) = v\}$$

and the iterative steps, using the procedure from assumption (1),

$$H^{i+1}(u, v) \coloneqq H^i(u, v) \cup \{\tau \circ \tau' \mid w \in T, \tau' \in H^i(u, w), \tau \in H^i(w, v)\}$$

We carry out the procedure until a fixed point is reached, i.e. $H^i(u, v) = H^{i+1}(u, v)$ for all $u, v \in T$, which can be detected using the equality decision procedure guaranteed by assumption (2). Observe that for any $i \in \omega$, $H^i(u, v) \subseteq H^{i+1}(u, v) \subseteq \textsc{Hom}_{\mathrm{Tr}^S}(u, v)$. Thus, a fixed point will be reached after $N \coloneqq \sum_{u,v \in T} |\textsc{Hom}_{\mathrm{Tr}^S}(u, v)|$ steps. Any fixed point $H(-, -)$ of this procedure has the property that $H(u, v) = \textsc{Hom}_{\mathrm{Tr}^S}(u, v)$: Simply pick some $\tau \in \textsc{Hom}_{\mathrm{Tr}^S}(u,)$ with $\tau = \mathrm{Tr}(ux_0 \dots x_n v)$, as all of the $H(-, -)$ are closed under composition it follows that $\tau = \mathrm{Tr}(x_n v) \circ \dots \circ \mathrm{Tr}(x_0 x_1) \circ \mathrm{Tr}(ux_0) \in H(u, v)$. Then we can decide whether D satisfies the GTC by using the procedure (3) on all $\tau \in H(u, u)$ for $u \in \mathrm{dom}(\beta)$ which satisfy $\tau = \tau \circ \tau$, which can in turn be decided by using the procedures (1) and (2). □

Remark. Note that, as we establish in Chapter 5, many trace categories satisfy the conditions (1) through (3), including those used to model the cyclic content cyclic arithmetic and the cyclic derivation systems in Chapter 6. The procedure thus seems to be widely applicable.

Remark. Naïvely, the procedure described in Theorem 4.14 seems preferable to the common automata-based procedure outlined in Theorem 4.4 due to its greater simplicity. The

language inclusion check of the latter procedure relies on quite complicated automata transformations, including Büchi-automaton complementation. The procedure from Theorem 4.14, on the other hand, does not require any further theory to be implemented.

Of course, a much better comparison of these two procedures would be one in terms of their computational complexities. Unfortunately, we have not been able to carry out such a comparison due to the time-restrictions inherent to Master's thesis projects.

Chapter 5

Concrete Trace Categories

In this chapter, we define and study concrete examples of trace categories. In Section 5.1 we introduce *activation algebras* and trace categories activated by them, which give rise to a diverse family of trace categories. We then prove for which kinds of activation algebras the induced trace categories exhibit the properties required for results from Chapters 3 and 4. In Section 5.2 we analyze and compare activation algebras and their associated trace categories via *activation-respecting homomorphisms*. We close the chapter in Section 5.3 by investigating which properties of trace categories are inherited by their subcategories, applying these results to the subcategory of injective trace maps of a given $\mathcal{A}$-activated trace category.

5.1 $\mathcal{A}$-activated Trace Categories

Activation algebras induce a family of trace categories which model a broad range of possible trace conditions. They generalize the "some event happens infinitely often along a trace" of the Brotherston category to more complex conditions.

Definition 5.1. An *activation algebra* $\mathcal{A} = (A, \leq, \vee, 0, \alpha)$ is a semilattice $(A, \leq, \vee, 0)$ together with a fixed *activation element* $\alpha \in A$ where $0 \neq \alpha$. That is,

- $\leq$ is a reflexive, transitive, antisymmetric order or A
- for $a, b \in A$, $a \vee b \in A$ is the least upper bound of a and b
- $0 \in A$ is the least element, meaning $0 \leq a$ for any $a \in A$

Definition 5.2. Let $\mathcal{A}$ be an activation algebra. The $\mathcal{A}$-*activated trace category* $\mathcal{T}_\mathcal{A}$ has as its objects the finite sets. The morphisms between sets X, Y are given by relations $R \subseteq X \times \mathcal{A} \times Y$. Given $R : X \to Y, R' : Y \to Z$ they compose as follows

$$(x, a \vee b, z) \in R' \circ R \qquad \text{iff} \qquad \exists y \in Y.\ (x, a, y) \in R \text{ and } (y, b, z) \in R'$$

For any set X, the identity is given by $1_X \coloneqq \{(x, 0, x) \mid x \in X\}$.

For some $R : X \to Y$ we often write $xR^a y$ to mean $(x, a, y) \in R$.

Definition 5.3. Let $\mathcal{A} = (A, \leq, \vee, 0, \alpha)$ be an activation algebra then a path $P : \omega \to \mathcal{T}_{\mathcal{A}}$ *satisfies the trace condition of* $\mathcal{T}_{\mathcal{A}}$ if there exists a subpath $P' \subseteq P$ and along it a sequence $\sigma : \Pi i \in \omega . P'(i)$ such that $\sigma_i P'(i < i+1)^{\alpha} \sigma_{i+1}$ for all $i \in \omega$.

The lemma below yields a different perspective on the trace condition in Definition 5.3: A path satisfying the trace condition can be split into infinitely many finite segments such that on each such segment $\sigma_{i+1}(R_n^{a_n} \circ \ldots \circ R_1^{a_1})\sigma_i$ we have $\bigvee a_j = \alpha$.

Lemma 5.4. *In an $\mathcal{A}$-activated trace category, pick $R_i : X_i \to X_{i+1}$ for $1 \leq i \leq n$ and suppose $x_1(R_n \circ \ldots \circ R_1)^a x_{n+1}$. Then there exist $x_2, \ldots, x_n$ and $a_1, \ldots, a_n$ such that $x_i R_i^{a_i} x_{i+1}$ with $a = \bigvee_{i=1}^n a_i$.*

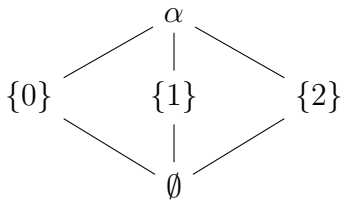
Proof. Proof by induction on n. For $n = 1$, the matter is trivial. For the inductive case, suppose $x_1(R_{n+1} \circ \ldots \circ R_1)^a x_{n+2}$, this means $x_1(R_{n+1} \circ (R_n \circ \ldots \circ R_1))^a x_{n+2}$. Per definition, there then exists $x_{n+1} \in X_{n+1}$ as well as $a_{n+1}, b \in \mathcal{A}$ such that $x_1(R_n \circ \ldots \circ R_1)^b x_{n+1}$, $x_{n+1} R_{n+1}^{a_{n+1}} x_{n+2}$ and $a = b \vee a_{n+1}$. The inductive hypothesis then yields $a_1, \ldots, a_n$ and $x_2, \ldots, x_n$ such that $x_i R_i^{a_i} x_{i+1}$ and $b = \bigvee_{i=1}^n a_i$. These $x_2, \ldots, x_n, x_{n+1}$ are as desired since $a = a_{n+1} \vee b = a_{n+1} \vee \bigvee_{i=1}^n a_i = \bigvee_{i=1}^{n+1} a_i$ □

Before we continue with a technical analysis of $\mathcal{A}$-activated trace categories, we give a few examples of activation algebras and the sort of trace conditions induced by them.

Example 5.5. The simplest example of an activation algebra is given by the simplest semilattice of at least two elements: The binary Boolean algebra $\mathbb{B} = \{\top, \bot\}$. Here the choice $\alpha := \top$ is forced as necessarily $\bot = 0 \neq \alpha$.

With Lemma 5.4, it is easy to notice that the $\mathbb{B}$-activated trace category is essentially the Brotherston trace category: A path satisfying the trace condition can be split into infinitely many segments such that $\sigma_{i+1}(R_n^{a_n} \circ \ldots \circ R_1^{a_1})\sigma_i$ and $\bigvee a_j = \top$. This means there must always exist at least one $a_j = \top$ in such a segment and there thus exists a trace along the path with in which infinitely many activations occur overall.

Example 5.6. A more complex example are the "k out of n" algebras for $0 < k \leq n$ given by $\binom{n}{k} := (A, \leq, \vee, \emptyset, \alpha)$ where $A := \{X \subseteq n \mid |X| < k\} \cup \{\alpha\}$ and the order $\leq$ is such that $X \leq Y$ iff $X \subseteq Y$ for $X, Y \subseteq n$, and $a \leq \alpha$ for all $a \in A$. As a more concrete example, observe the Hasse-diagram of $\binom{3}{2}$:



The idea behind $\binom{n}{k}$ is to view the singleton sets $\{i\}$ as "events" which can occur along a trace. To achieve activation, k distinct "events" need to take place along a segment. This intuitive understanding is made formal by Lemma 5.7.

Lemma 5.7. *Let $P : \omega \to \mathcal{T}_{\binom{n}{k}}$ with $1 < n$ be such that for each $i \in \omega$ and each $(x, a, y) \in P(i < i+1)$, we have that $a = \{j\}$ for some $j \leq n$. Then P satisfies the trace*

condition iff there exists $s \in \omega$ and sequences $\sigma : \Pi i \in \omega. P(s+i)$ and $a : \binom{n}{k}^\omega$ such that $\sigma_i P(s+i < s+i+1)^{a_i} \sigma_{i+1}$ for all $i \in \omega$, and there furthermore exists a set $E \subseteq n$ with $|E| \geq k$ such that for each $e \in E$, $a_i = \{e\}$ for infinitely many $i \in \omega$.

Proof. First, suppose P satisfied the trace condition, meaning there is an $S : \omega \to \omega$ such that for $P' \coloneqq P \circ S : \omega \to \mathcal{T}_{\binom{n}{k}}$ there exists a sequence $\sigma' : \Pi i \in \omega. P'(i)$ with $\sigma'_i P'(i < i+1)^\alpha \sigma'_{i+1}$ for all $i \in \omega$. Then take $s \coloneqq S(0)$ and observe that by Lemma 5.4 this means that for each $i \in \omega$, when taking $l_i \coloneqq S(i+1) - S(i)$, there exist $x^i_0, \ldots, x^i_{l_i}$ and $a^i_0, \ldots, a^i_{l_i - 1}$ such that for $j < l_i$, we have $x^i_j P(S(i)+j < S(i)+j+1)^{a^i_j} x^i_{j+1}$ and $\alpha = \bigvee a^i_j$. This already entails that the sequences $\sigma := x^0_0 x^0_1 \ldots x^0_{l_0-1} x^1_0 \ldots x^1_{l_1-1} x^2_0 \ldots$ and $a := a^0_0 a^0_1 \ldots a^0_{l_0-1} a^1_0 \ldots a^1_{l_1-1} a^2_0 \ldots$ are as desired. It remains to find the set of events E. For this, we apply the Ramsey Theorem to singleton subsets of ω as follows: For any $i \in \omega$, pick the color $c(\{i\}) \coloneqq \{e < n \mid \exists j < l_i.\ a^i_j = \{e\}\}$. This coloring is finite as $c(\{i\}) \subseteq n$ for every $i \in \omega$. Furthermore, observe that $|c(\{i\})| \geq k$ for each $i \in \omega$ as per assumption, only singleton events can occur along σ and at least k distinct singleton events are required for their join to be α. The Ramsey theorem thus yields some coloring $E \subseteq n$ and an infinite subset $I \subseteq \omega$ such that for $i \in I$, $c(\{i\}) = E$. That means each event in E takes place infinitely often along σ.

For the opposite direction, simply observe that $\alpha = \bigvee_{e \in E} \{e\}$. □

Example 5.8. In both examples so far, α was maximal. However, this need not be the case for general activation algebras. Such lattice elements $f \not\leq \alpha$ can act as a "failure state" for the trace condition: Observe that there exists no finite subset $S \subseteq A$ of lattice elements such that $f \vee \bigvee S = \alpha$. This means that once some transition $x R^f y$ has been taken, the activation element α can no longer be reached by taking further transition steps. Such "failure" elements can thus be used to rule out certain kinds of "bad events" taking place along a trace.

One specific example of an activation algebra with such a failure element is the three-value lattice $\mathbb{F} \coloneqq (3, \leq, \vee, 0, 1)$ in which 2 serves as the failure element. As we demonstrate in Section 6.2, the $\mathbb{F}$-activated trace category lends itself well to modeling trace conditions which are most naturally expressed as a parity condition, such as in the case of the modal μ-calculus.

We now proceed by establishing various properties of $\mathcal{A}$-activated trace categories. First, we need to show that the trace condition we gave above is indeed a trace condition.

Proposition 5.9. *The condition in Definition 5.3 is invariant under subpaths.*

Proof. First of all, suppose $P \subseteq Q$ with $P = Q \circ S$ and P satisfies the trace condition, meaning there exists $P' \subseteq P$ with $P' = P \circ S'$ and a validating sequence σ. Then $P' \subseteq Q$ via $P' = Q \circ S \circ S'$, meaning σ is a validating sequence through a subpath of Q as well.

Conversely, let $Q \subseteq P$ with $Q = P \circ S$ and suppose P satisfied the trace condition as witnessed by $P' \subseteq P$ with $P' = P \circ S'$ and a sequence $\sigma : \Pi i \in \omega. P'(i)$. Let $b \coloneqq S'(0)$ then by Lemma 5.4 this induces two sequences $\sigma' : \Pi i \in \omega. P(b+i)$ and $a : \omega \to \mathcal{A}$ such that $\sigma'_i P(b+i < b+i+1)^{a_i} \sigma'_{i+1}$ with $a_i \leq \alpha$, for $i_j \coloneqq S'(j) - b$ we have $\sigma'_{i_j} = \sigma_j$ and

$\bigvee_{i=i_j}^{i\leq i_{j+1}} a_i = \alpha$. Now construct the following sequence

$$\begin{aligned} k_0 &:= \text{the least } i \in \omega \text{ such that } S(i) \geq b \\ k_{n+1} &:= \text{the least } i > k_n \text{ such that there is } j \in \omega \text{ with } S(k_n) \leq S'(j) < S'(j+1) \leq S(i) \end{aligned}$$

and claim that $S''(i) := S(k_i)$ induces the desired subsequence $Q' \subseteq Q$ via $Q' = Q \circ k_-$ as witnessed by $\sigma'' : \Pi i \in \omega . Q'(i)$ given by $\sigma''_i := \sigma'_{S(k_i)}$. For this, we need to check that $\sigma''_{i+1}\, Q'(i < i+1)^\alpha\, \sigma''_{i+1}$. Let $j \in \omega$ be such that $S(k_i) \leq S'(j) < S'(j+1) \leq S(k_{i+1})$ then $Q'(i < i+1) = P(b + i_{j+1} \leq k_{i+1}) \circ P(b + i_j < b + i_{j+1}) \circ P(k_i \leq b + i_j)$ meaning

$$\left(\sigma''_i, \underbrace{\bigvee_{l=b+i_{j+1}}^{l<k_{i+1}} a_{l-b}}_{\leq \alpha} \vee \underbrace{\bigvee_{l=i_j}^{l<i_{j+1}} a_l}_{=\alpha} \vee \underbrace{\bigvee_{l=k_i}^{l<b+i_j} a_{l-b}}_{\leq \alpha}, \underbrace{\sigma'_{k_{i+1}-b}}_{=\sigma''_{i+1}} \right) \in Q'(i < i+1)$$

and thus $(\sigma''_i, \alpha, \sigma''_{i+1}) \in Q'(i < i+1)$ as desired. □

The remainder of this section is concerned with proving that $\mathcal{A}$-activated trace categories exhibit all of the properties required by the definitions and theorems in Chapters 3 and 4. We begin by proving that they have finite products and thus allow for the definition of trace-interpretations.

Proposition 5.10. *For any activation algebra $\mathcal{A}$ the $\mathcal{A}$-activated trace category $\mathcal{T}_\mathcal{A}$ has finite products.*

Proof. For this, it suffices to show that $\mathcal{T}_\mathcal{A}$ has a terminal object and binary products.

- *Terminal object:* We take $1 := \emptyset$. This is a terminal object as we know for any $X \in \mathrm{Ob}(\mathcal{T}_\mathcal{A})$ that $\textsc{Hom}(X, 1) = \{!_X\}$ where $!_X := \emptyset$.
- *Binary products:* For $X, Y \in \mathrm{Ob}(\mathcal{T}_\mathcal{A})$ we take $X \times Y := X + Y$ (here $X + Y$ denotes the sum in Set[1]). The projections $\pi_X : X \times Y \to X$ and $\pi_Y : X \times Y \to Y$ are given by
$$\pi_X := \{(L\,x, 0, x) \mid x \in X\} \qquad \pi_Y := \{(R\,y, 0, y) \mid y \in Y\}$$
For morphisms $R : X \to Y, R' : X \to Z$ we define their product $\langle R, R' \rangle : X \to Y \times Z$ as
$$\langle R, R' \rangle := \{(x, a, L\,y) \mid (x, a, y) \in R\} \cup \{(x, a, R\,z) \mid (x, a, z) \in R'\}$$
It is easy to verify that these definitions together yield a product. □

For the results from Section 4.1, we need to establish that the trace condition of $\mathcal{A}$-activated trace categories is recognizable by various kinds of infinite word automata. In general, this is only the case if $\mathcal{A}$ is finite. However, this restriction should not be an issue as all the global trace conditions from the literature which we know of can be modeled in terms of finite AAs. Indeed, the study of infinite AAs seems quite futile as their trace condition need not be Büchi-recognizable, meaning proof checking for cyclic derivation systems whose trace condition is modeled by them might not be decidable either.

[1] We assume here that $X + Y := \{L\,x \mid x \in X\} \cup \{R\,y \mid y \in Y\}$ where $L\,x$ and $R\,y$ denote distinct "taggings" such as $L\,x := (x, 0)$ and $R\,y := (y, 1)$. Any other representation would work just as well.

Proposition 5.11. *Let $\mathcal{A}$ be a finite activation algebra. The trace condition of $\mathcal{T}_\mathcal{A}$ is non-deterministically Büchi-recognizable.*

Proof. Let $O \subseteq \mathrm{Ob}(\mathcal{T}_\mathcal{A})$ be finite, $M = \{R_i : X_i \to Y_i \mid 1 \leq i \leq n, X_i, Y_i \in O\}$ be the choice of morphisms and $S \in O$ be the starting object. We construct a Büchi automaton $\mathfrak{B} \coloneqq (M, Q, \Delta, S, F)$ as follows: Take $Q \coloneqq O \cup \Sigma X \in O.X \times (\mathcal{A} \cup \{r\})$ for some $r \notin \mathcal{A}$. For the transitions Δ we take

$$\begin{aligned}\Delta \coloneqq &\{(X, R : X \to Y, Y) \mid R : X \to Y \in M\}\\ &\cup \{(X, \varepsilon, (X, x, 0)) \mid X \in O, x \in X\}\\ &\cup \{((X, x, a), R : X \to Y, (Y, y, a \vee b)) \mid xR^b y, R \in M\}\\ &\cup \bigcup \{\{((X, x, \alpha), \varepsilon, (X, x, r)), ((X, x, r), \varepsilon, (X, x, 0))\} \mid X \in O, x \in X\}\end{aligned}$$

and for the acceptance condition $F \subseteq Q$ we pick

$$F \coloneqq \{(X, x, r) \mid X \in O, x \in X\}$$

Towards the correctness of this automaton, first suppose $P : \omega \to \mathcal{T}_\mathcal{A}$ is a path such that $P(i) \in O$ and $P(i < i+1) \in M$ for all $i \in \omega$ which satisfies the trace condition of $\mathcal{T}_\mathcal{A}$ and $P(0) = S$. Per definition, there thus exists a semi-functor $G : \omega \to \omega$ and a sequence $\sigma : \Pi i \in \omega.P(G(i))$ such that $\sigma_i \, P(G(i < i+1))^\alpha \, \sigma_{i+1}$. From this, we construct the following run $\rho : Q^\omega$ of $\mathfrak{B}$:

- Begin by reading the prefix of P "irrelevant" to the trace condition with

  $$S \xrightarrow{P(0<1)} P(2) \xrightarrow{P(1<2)} \ldots \xrightarrow{P(G(0)-1<G(0))} P(G(0))$$

  and start the actual recognition via $P(G(0)) \xrightarrow{\varepsilon} (P(G(0)), \sigma_0, 0)$.
- At a state $(P(G(i)), \sigma_i, 0)$, having already read $P(0 < 1) \ldots P(G(i) - 1 < G(i))$, continue as follows: Take $b \coloneqq G(i)$, $n \coloneqq G(i+1) - G(i)$ and observe that

  $$P(G(i < i+1)) = P(b + n - 1 < b + n) \circ \ldots \circ P(b < b + 1)$$

  By Lemma 5.4 this means there are $x_0, \ldots, x_n$ and $a_0, \ldots, a_{n-1}$ such that for $0 \leq j < n$ we have $x_j P(b + j < b + j + 1)^{a_j} x_{j+1}$ as well as $x_0 = \sigma_i, x_n = \sigma_{i+1}$ and $\alpha = \bigvee_{j=0}^{n-1} a_j$. Now we read $P(G(i < i+1))$ via the sequence:

  $$\begin{aligned}&(P(G(i)), \sigma_i, 0) \xrightarrow{P(G(i<i+1))} (P(G(i+1)), \sigma_{i+1}, \alpha) \xrightarrow{\varepsilon}\\ &(P(G(i+1), \sigma_{i+1}, r)) \xrightarrow{\varepsilon} (P(G(i+1)), \sigma_{i+1}, 0)\end{aligned}$$

  where $(P(G(i)), \sigma_i, 0) \xrightarrow{P(G(i<i+1))} (P(G(i+1)), \sigma_{i+1}, \alpha)$ is a shorthand for the steps

  $$(P(b+0), x_0, \textstyle\bigvee_{j=0}^{0-1} a_j) \xrightarrow{P(b+0<b+1)} \ldots \xrightarrow{P(b+n-1<b+n)} (P(b+n), x_n, \bigvee_{j=0}^{n-1} a_j)$$

  We may then continue in the same manner as this yields a state $(P(G(i+1)), \sigma_{i+1}, 0)$ having already read $P(0 < 1) \ldots P(G(i) - 1 < G(i))$.

Clearly, this is a run ρ of $\mathfrak{B}$. Furthermore, observe that there is a finite prefix of states outside of F along ρ, all occurring along the prefix $S \xrightarrow{P(0<G(0))} G(0)$, which we may thus discard in our consideration of the acceptance of ρ. On the other hand, there are infinitely many states in F along ρ, namely the interim state of

$$(P(G(i+1)), \sigma_{i+1}, a) \xrightarrow{\varepsilon} (P(G(i+1), \sigma_{i+1}, r)) \xrightarrow{\varepsilon} (P(G(i+1)), \sigma_{i+1}, 0)$$

for each $i \in \omega$. Thus ρ is an accepting run of $\mathfrak{B}$, meaning $\mathfrak{B}$ accepts the path P overall.

Now, conversely, suppose $P \in M^\omega$ was a sequence accepted by $\mathfrak{B}$, meaning there is an accepting run $\rho \in Q^\omega$ of $\mathfrak{B}$ on P. It is easy to observe that the fact that M has any run on $\mathfrak{B}$ at all means that the induced path $P : \omega \to \mathcal{T}$ is indeed a functor, as only "well-typed" transition steps may be taken, and furthermore $P(0) = S$, as this is the starting state of $\mathfrak{B}$. Now, as ρ is accepted, it has to have a prefix of the form

$$S \xrightarrow{P(0<G(0))} G(0) \xrightarrow{\varepsilon} (P(G(0)), \sigma_0, 0)$$

for some $G(0) \in \omega$ and $\sigma_0 \in P(G(0))$. Otherwise, all of ρ would be outside of F, meaning ρ is not accepting. Further, there need to be infinitely many occurrences of states in F along ρ. Denote them, in sequence, by $(P(G(i+1)), \sigma_{i+1}, r)$ for $i \in \omega$. Clearly, each such state needs to be directly preceded by $(P(G(i+1)), \sigma_{i+1}, \alpha)$ and directly succeeded by $((P(G(i+1))), \sigma_{i+1}, 0)$. Then these indexes induce a semi-functor $G : \omega \to \omega$ because necessarily $G(i) < G(i+1)$: We know that

$$(P(G(i)), \sigma_i, r) \xrightarrow{\varepsilon} (P(G(i)), \sigma_i, 0) \xrightarrow{s} (P(G(i+1)), \sigma_{i+1}, \alpha) \xrightarrow{\varepsilon} (P(G(i+1)), \sigma_{i+1}, r)$$

for some $s \in M^*$. Now observe that $G(i) \neq G(i+1)$, as that would mean that $s = \varepsilon$ and thus $0 = a$, contradicting our assumption. Thus $G(i) < G(i+1)$, since labeling the $G(i)$ in sequence rules out $G(i) > G(i+1)$. Lastly, the fact that

$$(P(G(i)), \sigma_i, 0) \xrightarrow{P(G(i<i+1))} (P(G(i+1)), \sigma_{i+1}, \alpha)$$

with no states in F in between means that $\sigma_i P(G(i < i+1))^a \sigma_{i+1}$. This allows us to conclude that $P : \omega \to \mathcal{T}$ satisfies the trace condition as witnessed by the subpath $P \circ G \subseteq P$ and the sequence $\sigma : \Pi i \in \omega.\, P(G(i))$. □

Remark. This theorem was the main motivation behind the requirement that $\alpha \neq 0$ in activation algebras. While the trace condition of an "activation algebra" with $\alpha = 0$ is still Büchi-recognizable, the automaton construction is quite different. As the trace condition induced by such an "activation algebra" also diverges from the usual trace conditions quite drastically (intuitively, it is "nothing ever happens along the trace"), we decided to rule out the case in the definition of activation algebras, rather than giving an additional, different construction for it.

Corollary 5.12. *Let $\mathcal{A}$ be a finite activation algebra. The trace condition of $\mathcal{T}_\mathcal{A}$ is deterministically parity-recognizable.*

Proof. This follows because for every non-deterministic Büchi-automaton, one can construct an equivalent deterministic Muller automaton via McNaughton's theorem [26], which can in turn be transformed into a deterministic parity automaton. □

We close the section by proving the properties required by the algorithm for deciding the global trace condition based on Ramsey's Theorem as presented in Section 4.2.

Proposition 5.13. *If $\mathcal{A}$ is finite, then for $X, Y \in Ob(\mathcal{T}_\mathcal{A})$ the set $\text{HOM}_{\mathcal{T}_\mathcal{A}}(X, Y)$ is finite.*

Proof. Observe that $\text{HOM}_{\mathcal{T}_\mathcal{A}}(X, Y) = \mathcal{P}(X \times \mathcal{A} \times Y)$ which is finite as X, $\mathcal{A}$ and Y are. □

Proposition 5.14. *Let $\mathcal{A}$ be finite. Then the following is true for $\mathcal{T}_\mathcal{A}$*

(1) from $R : X \to Y$ and $R' : Y \to Z$ one can compute $R' \circ R : X \to Z$

(2) for $R, R' : X \to Y$ one can decide whether $R = R'$

(3) for idempotent $R : X \to X$ one can decide whether R^ω satisfies the trace condition

if equality on the finite sets X, Y, Z is decidable.

Proof. Observe that any finite AA $\mathcal{A}$ can be suitably encoded such that the

(i) from $a, b \in A$ one can compute $a \vee b$

(ii) for $a, b \in A$ one can decide whether $a = b$

The procedures for (1) and (2) are easily derived by making use of points (i) and (ii), respectively. For (3), we show that for a given $R : X \to X$, R^ω satisfies the trace condition if and only if there exists some $x \in X$ such that $xR^\alpha x$, which is easily decided using (ii). First, if there exists such an $x \in X$ than R^ω itself is the trace condition satisfying subpath of R^ω with the witnessing sequence $\sigma_i := x$. Now suppose, $P \subseteq R^\omega$ and $\sigma : \Pi i \in \omega.P(i)$ such that $\sigma_i P(i < i + 1)^\alpha \sigma_{i+1}$. First observe that any subpath of the periodic path of an idempotent trace map is just that periodic path again, meaning $P = R^\omega$ and thus $\sigma_i R^\alpha \sigma_{i+1}$ for $\sigma : \omega \to X$. Then simply observe that among $\sigma_0, \ldots, \sigma_{|X|}$ there need to be $i < j$ with $\sigma_i = \sigma_j$ by the pigeon hole principle. We may thus conclude $xR^\alpha x$ for $x := \sigma_i$ by the idempotence of R, since we know that $\sigma_i(\underbrace{R \circ \ldots \circ R}_{j-i \text{ times}})^\alpha \sigma_j$. □

5.2 Comparing Activation Algebras

This section studies the relative expressive power of activation algebras by examining what kind of trace conditions they can describe. Intuitively, an AA $\mathcal{B}$ can be considered more powerful than some other AA $\mathcal{A}$ if any path in $\mathcal{T}_\mathcal{A}$ can be transformed homogeneously into a path in $\mathcal{T}_\mathcal{B}$ which satisfies the trace condition iff the original path did. A suitable notion of "homogeneous transformation" is given by functors.

Definition 5.15. For trace categories $\mathcal{T}, \mathcal{T}'$ we call a functor $F : \mathcal{T} \to \mathcal{T}'$ a *trace-simulation* if for any path $P : \omega \to \mathcal{T}$, P satisfies the trace condition of $\mathcal{T}$ if and only if $F \circ P$ satisfies the trace condition of $\mathcal{T}'$.

If there exists a trace-simulation $F : \mathcal{T} \to \mathcal{T}'$ we say $\mathcal{T}$ *is simulated by* $\mathcal{T}'$. If there exists trace-simulations in both direction, we call $\mathcal{T}$ and $\mathcal{T}'$ *inter-simulated*.

As we want to compare the strength of the activation algebras, we restrict our attention to a special class of simulations: Those induced by activation algebra homomorphisms. Intuitively, these only modify the "activation information" of a path while leaving the

underlying sets unchanged, meaning the simulation truly witnesses a fact about AAs.

Definition 5.16. Pick two activation algebras $\mathcal{A} = (A, \leq, \vee, 0, \alpha)$ and $\mathcal{B} = (B, \leq, \vee, 0, \beta)$. We call a function $f : A \to B$ a *activation algebra homomorphism*, if

- f is monotone, that is, if $a \leq b$ then $f(a) \leq f(b)$
- f distributes over joins, that is, $f(a \vee b) = f(a) \vee f(b)$
- f preserves 0 and α, that is, $f(0) = 0$ and $f(\alpha) = \beta$

We denote such a homomorphism by $f : \mathcal{A} \to \mathcal{B}$.

Proposition 5.17. *Any $f : \mathcal{A} \to \mathcal{B}$ induces a functor $F : \mathcal{T}_{\mathcal{A}} \to \mathcal{T}_{\mathcal{B}}$ given by*

$$F(X) \coloneqq X \qquad\qquad F(R : X \to Y) \coloneqq \{(x, f(a), y) \mid (x, a, y) \in R\}$$

such that for any $P : \omega \to \mathcal{T}$ if P satisfies the trace condition of $\mathcal{T}_{\mathcal{A}}$, then $F \circ P$ satisfies the trace condition of $\mathcal{T}_{\mathcal{B}}$.

Proof. First of all, observe that $F : \mathcal{T}_{\mathcal{A}} \to \mathcal{T}_{\mathcal{B}}$ is indeed a functor as

$$F(1_X) = \{(x, f(0), x) \mid x \in X\} = 1_{F(X)}$$

because $f(0) = 0$ and

$$\begin{aligned} F(R_2 \circ R_1) &= \{(x, f(a \vee b), z) \mid x R_1^a y R_2^b z\} \\ &= \{(x, f(a) \vee f(b), z) \mid x F(R_1)^{f(a)} y F(R_2)^{f(b)} z\} = F(R_2) \circ F(R_1) \end{aligned}$$

because $f(a \vee b) = f(a) \vee f(b)$.

Now, if $P : \omega \to \mathcal{T}_{\mathcal{A}}$ satisfies the trace condition then there is $S : \omega \to \omega$ and a sequence $\sigma : \Pi i \in \omega . P(S(i))$ such that $\sigma_i P(S(i))^{\alpha} \sigma_{i+1}$. Observe that, since $F(X) = X$, this means that σ is a sequence of type $\Pi i \in \omega . F(P(S(i)))$ as well with $\sigma_i F(P(S(i)))^{f(\alpha)} \sigma_{i+1}$. Then $F \circ P$ satisfies the trace condition as well, as $f(\alpha) = \beta$. □

However, not every functor induced by a homomorphism is a simulation. We thus further restrict our attention to the so-called activation-respecting homomorphisms, whose induced functors are guaranteed to be simulations.

Definition 5.18. A homomorphism $f : \mathcal{A} \to \mathcal{B}$ is called *activation-respecting* if for $a \in A$, $f(a) = \beta$ implies $a = \alpha$.

Proposition 5.19. *If $f : \mathcal{A} \to \mathcal{B}$ is an activation-respecting homomorphism then its induced functor $F : \mathcal{T}_{\mathcal{A}} \to \mathcal{T}_{\mathcal{B}}$ is a trace-simulation.*

Proof. The forward direction of the condition holds in the general case. Now pick a path $P : \omega \to \mathcal{T}_{\mathcal{A}}$ such that $F \circ P$ satisfies the trace condition of $\mathcal{T}_{\mathcal{B}}$. This means there is an $S : \omega \to \omega$ and a sequence $\sigma : \Pi i \in \omega . F(P(S(i)))$ such that $\sigma_i F(P(S(i)))^{\beta} \sigma_{i+1}$. Observe that then also $\sigma_i P(S(i))^{a_i} \sigma_{i+1}$ with $f(a_i) = \beta$. As f is activation-respecting, we know that $a_i = \alpha$ for each $i \in \omega$ and P thus satisfies the trace condition of $\mathcal{T}_{\mathcal{A}}$. □

With these notions, we can begin analyzing activation algebras. First, we show that the Booleans $\mathbb{B}$ occupy a special position when it comes to activation algebras.

Proposition 5.20. *Any injective $f : \mathcal{A} \to \mathcal{B}$ is activation respecting.*

Proof. We know that $f(\alpha) = \beta$ thus, by injectivity, $f(a) = \beta$ implies $\alpha = \beta$. □

Proposition 5.21. *For every activation algebra $\mathcal{A}$ there exists a unique $!_{\mathcal{A}} : \mathbb{B} \to \mathcal{A}$. Furthermore, the $!_{\mathcal{A}}$ are activation-respecting.*

Proof. Take $!_{\mathcal{A}}(\bot) \coloneqq 0$ and $!_{\mathcal{A}}(\top) \coloneqq \alpha$. Uniqueness follows as these choices are forced by the definition of activation algebra homomorphisms. It is activation-respecting as it is injective. □

Remark. In other words, $\mathbb{B}$ is initial in the category of activation algebras.

We continue with an investigation of linear activation algebras, that is, activation algebras whose Hasse diagram is a line. The three point failure algebra $\mathbb{F}$ can be seen as "most powerful" linear activation algebra as it can simulate any other linear AA.

Definition 5.22. An activation algebra $\mathcal{A}$ is called *linear* if $\leq$ is a linear order.

Proposition 5.23. *If $\mathcal{A}$ is linear, there exists a unique activation-respecting $!_{\mathcal{A}} : \mathcal{A} \to \mathbb{F}$.*

Proof. Take

$$!_{\mathcal{A}}(x) \coloneqq \begin{cases} 0 & x < \alpha \\ 1 & x = \alpha \\ 2 & \alpha < x \end{cases}$$

which is clearly activation-respecting. Furthermore, it is monotone and thus preserves joins as both activation algebras, and thus the underlying semi-lattices, are linear. For uniqueness, observe every $x < \alpha$ needs to be mapped to some $y < 3$ with $y \leq 1$, as $f(\alpha) = 1$ for any homomorphism and to guarantee monotonicity, and with $y \neq 1$ to guarantee that the homomorphism is activation-respecting, together forcing $y < 1$ and thus the choice of $f(x) = 0$. Similar reasoning shows that $f(x) = 2$ needs to be the case for $\alpha < x$. □

Remark. In other words, $\mathbb{F}$ is terminal in the subcategory of linear activation algebras and activation-preserving maps.

Remark. In contrast to the initiality of $\mathbb{B}$, the homomorphisms into $\mathbb{F}$ are only unique if we restrict ourselves to activation-respecting homomorphisms. That is, there is a unique activation-respecting homomorphism $!_{\mathcal{A}} : \mathcal{A} \to \mathbb{F}$ for any linear $\mathcal{A}$ whereas there is a unique homomorphism $!_{\mathcal{A}} : \mathbb{B} \to \mathcal{A}$ for any $\mathcal{A}$ which happens to also always be activation-respecting. For an example of a homomorphism $f : \mathbb{F} \to \mathbb{F}$ with $f \neq !_{\mathbb{F}}$, consider one with $f^{-1}(\alpha) = \{1, 2\}$.

Our study of linear activation algebras concludes with a somewhat surprising result: up to inter-simulation, $\mathbb{B}$ and $\mathbb{F}$ are the only linear activation algebras. In practice, this means that there is little reason to ever use $\mathcal{T}_{\mathcal{A}}$ for some other linear activation algebra $\mathcal{A}$

to formalize the trace condition of a cyclic derivation system, as it could just as well be formulated in terms of either $\mathbb{B}$ or $\mathbb{F}$.

Proposition 5.24. *For any $\mathcal{A}$ which has some $a \in A$ with $\alpha < a$ there exists an activation-respecting $f : \mathbb{F} \to \mathcal{A}$.*

Proof. Simply take $f(0) = 0$, $f(1) = \alpha$ and $f(2) = a$. It clearly is an activation algebra homomorphism and it respects activations as it is an injection. □

Proposition 5.25. *If $\mathcal{A}$ is linear and has no $a \in A$ with $\alpha < a$ then there is a unique, activation-respecting $f : \mathcal{A} \to \mathbb{B}$.*

Proof. Simply take

$$f(x) \coloneqq \begin{cases} \bot & x < \alpha \\ \top & x = \alpha \end{cases}$$

The reasoning as to why this is an activation algebra homomorphism, activation-respecting and unique is analogous to that in Proposition 5.23. □

Theorem 5.26. *If $\mathcal{A}$ is linear, $\mathcal{T}_{\mathcal{A}}$ is inter-simulated with either $\mathcal{T}_{\mathbb{B}}$ or $\mathcal{T}_{\mathbb{F}}$.*

Proof. If there is no $a \in A$ with $\alpha < a$ then the unique map $!_{\mathcal{A}} : \mathbb{B} \to \mathcal{A}$ and the map $f : \mathcal{A} \to \mathbb{B}$ granted by Proposition 5.25 induce the two trace-simulations. If there is an $\alpha < a$, then the unique map $!_{\mathcal{A}} : \mathcal{A} \to \mathbb{F}$ and the map $f : \mathbb{F} \to \mathcal{A}$ granted by Proposition 5.24 induce them. □

Lastly, we investigate activation algebras with failure points, that is, activation algebras which have elements $a \not\leq \alpha$. The result below shows that "one failure point is enough", i.e. that adding more failure points to an activation algebra which already possesses at least one does not increase its expressivity.

Definition 5.27. For a given activation algebra $\mathcal{A} = (A, \leq, \vee, 0, \alpha)$ we define its *failure-simplification* $\mathcal{A}^F = (A^F, \leq^F, \vee, 0, \alpha)$ by taking

$$A^F \coloneqq \{a \in A \mid a \leq \alpha\} \cup \{1\} \qquad \leq^F \coloneqq (\leq \cap A^F \times A^F) \cup \{(a, 1) \mid a \in A^F\}$$

Proposition 5.28. *For any $\mathcal{A}$, there exists an activation-respecting $f : \mathcal{A} \to \mathcal{A}^F$.*

Proof. For this, simply take

$$f(x) \coloneqq \begin{cases} x & x \leq \alpha \\ 1 & \text{otherwise} \end{cases}$$

Again, it is easy to see it respects activations. For monotonicity, observe that for $x \leq y \in A$, either $y \leq \alpha$ or not. In the former case, $x \leq y \leq \alpha$ and thus $f(x) = x, f(y) = y$, in the latter case $f(y) = 1$ and thus $f(x) \leq f(y)$ trivially. For join distributivity, pick $x, y \in A$. We again need to cover two cases: If $x, y \leq \alpha$ then $x \vee y \leq \alpha$ as well and thus $f(x) \vee f(y) = x \vee y = f(x \vee y)$. If, say $y \not\leq \alpha$ then $x \vee y \not\leq \alpha$ as well, meaning $f(x) \vee f(y) = f(x) \vee 1 = 1 = f(x \vee y)$. 0 and α are clearly preserved. □

Corollary 5.29. *Any $\mathcal{T}_{\mathcal{A}}$ is simulated by $\mathcal{T}_{\mathcal{A}^F}$.*

We close this section by demonstrating that, when dropping the restrictions on simulations we have imposed so far, namely that they should be functors generated by homomorphisms, it is possible to obtain "simulation-like" functions between all $\mathcal{A}$-activated trace categories.

Theorem 5.30. *For any activation algebra $\mathcal{A}$ there exists a function I mapping objects of $\mathcal{T}_{\mathcal{A}}$ to objects of $\mathcal{T}_{\mathbb{B}}$ and maps $R : X \to Y$ in $\mathcal{T}_{\mathcal{A}}$ to maps $I(R) : I(X) \to I(Y)$ in $\mathcal{T}_{\mathbb{B}}$. Furthermore, pick a path $P : \omega \to \mathcal{T}_{\mathcal{A}}$ and consider the path $\hat{P} : \omega \to \mathcal{T}_{\mathbb{B}}$ given by $\hat{P}(i) \coloneqq I(P(i))$ and $\hat{P}(i < i+1) \coloneqq I(P(i < i+1))$. Then P satisfies the trace condition iff $\hat{P}$ does.*

Proof. For this, take $I(X) \coloneqq X \times \mathcal{A}$ and, for $R : X \to Y$,

$$\begin{aligned} I(R) \coloneqq &\{((x,a), \bot, (y, a \vee b)) \mid a \in \mathcal{A}, xR^b y\} \\ &\cup \{((x,a), \top, (y, 0)) \mid a \in \mathcal{A}, xR^b y, a \vee b = \alpha\} \end{aligned}$$

First suppose that P satisfied the trace condition. That means there is a subpath $P \circ S$ and a sequence $\sigma : \Pi i \in \omega . P(S(i))$ such that $\sigma_i P(S(i))^\alpha \sigma_{i+1}$. We claim that $\hat{P} \circ S$ is the witnessing subpath of P with the sequence $\hat{\sigma} : \Pi i \in \omega . \hat{P}(S(i))$ given by $\hat{\sigma}_i \coloneqq (\sigma_i, 0)$. It remains to prove that $\hat{\sigma}_i \hat{P}(S(i < i+1))^\top \hat{\sigma}_{i+1}$. By Lemma 5.4 we know that there are $\sigma_i = x_0, \ldots, x_n = \sigma_{i+1}$ and $a_0, \ldots, a_{n-1}$ for $n \coloneqq S(i+1) - S(i)$ and $b \coloneqq S(i)$ such that $x_j P(b+j)^{a_j} x_{j+1}$ and $\alpha = \bigvee a_j$. Then define $\hat{x}_j \coloneqq (x_j, \bigvee_{k=0}^{j-1} a_k)$ and observe that $\hat{\sigma}_i = \hat{x}_0 \hat{P}(b)^\bot \hat{x}_1 \hat{P}(b+1)^\bot \ldots \hat{x}_{n-1} \hat{P}(b+n-1)^\top (x_n, 0) = \hat{\sigma}_{i+1}$.

Now suppose that $\hat{P}$ satisfied the trace condition. Then there is a subpath $\hat{P} \circ S$ and a sequence $\hat{\sigma} : \Pi i \in \omega . \hat{P}(S(i))$ such that $\hat{\sigma}_i \hat{P}(S(i < i+1))^\top \hat{\sigma}_{i+1}$. Let, without loss of generality, $\pi_2(\hat{\sigma}_i) = 0$ (this can always be achieved by moving the $S(i)$ appropriately). By taking $\sigma_i \coloneqq \pi_1(\hat{\sigma}_i)$ we have that $\sigma_i P(S(i < i+1))^\alpha \sigma_{i+1}$ as desired. □

Corollary 5.31. *For any two activation algebras $\mathcal{A}, \mathcal{B}$ there exists a function I mapping objects of $\mathcal{T}_{\mathcal{A}}$ to objects of $\mathcal{T}_{\mathcal{B}}$ and maps $R : X \to Y$ in $\mathcal{T}_{\mathcal{A}}$ to maps $I(R) : I(X) \to I(Y)$ in $\mathcal{T}_{\mathcal{B}}$. Furthermore, pick some path $P : \omega \to \mathcal{T}_{\mathcal{A}}$ and consider the path $\hat{P} : \omega \to \mathcal{T}_{\mathcal{B}}$ given by $\hat{P}(i) \coloneqq I(P(i))$ and $\hat{P}(i < i+1) \coloneqq I(P(i < i+1))$. Then P satisfies the trace condition iff $\hat{P}$ does.*

Proof. Simply take $I \coloneqq !_{\mathcal{B}} \circ I'$ where I' is obtained via Theorem 5.30. □

Remark. This result is somewhat hard to make sense of. In a way, it states that all $\mathcal{A}$-activated trace categories are inter-simulated, albeit by a somewhat unclean, non-functorial simulation. One interesting consequence of this fact is that the CYCLICST generic theorem prover developed by Brotherston et.al. [6], which can work with any cyclic system whose trace condition is (essentially) specified in terms of $\mathcal{T}_{\mathbb{B}}$, can be extended to accept and work with cyclic systems whose trace condition is specified in terms of some finite activation algebra $\mathcal{A}$.

These results also raise the question why the concept of activation algebras is useful in the first place, if they cannot be used to formalize any trace conditions that are not expressible in $\mathcal{T}_{\mathbb{B}}$. The answer to this question is somewhat subtle: If all one cares about is a generic

notion of trace condition for which one can develop abstract results such as those in Chapter 4, there indeed seems no need to use activation algebras. For example, the Büchi automatons for some $\mathcal{T}_{\mathcal{A}}$ given by Proposition 5.11 are essentially of the same size as the ones obtained by first applying Theorem 5.30 and then constructing the automaton for $\mathcal{T}_{\mathbb{B}}$. However, activation algebras often allow for trace conditions to be modeled in a trace category in a manner much more similar to the "original formulation" from the respective papers introducing them. They thus separate the information of traces into two parts: The elements of the finite sets signify *which objects* is being tracked, typically terms, inductive predicates or fixed-point quantifiers, while the elements of the activation algebra describe *how progress* of these objects is detected. This separation is muddled by the transformation in Theorem 5.30, which models both the objects being tracked and their progress via the underlying sets. Apart from the clear aesthetic benefit of this separation, it might also help solve certain kinds of problems, some of which we outline in Section 7.2.

Remark. The functions given by Theorem 5.30 fail to be functors on both accounts: preserving identities and distributing over composition. The failure of identity preservation is easily observed, simply notice that any $I(1_X)$ will contain triples of the form $((x, \alpha), \top, (x, 0))$ which the identities of $\mathcal{T}_{\mathbb{B}}$ do not contain. This issue could, in theory, be alleviated by taking the definition to be

$$\begin{aligned} I(R) \coloneqq &\{((x,a), \bot, (y, a \vee b)) \mid a \in \mathcal{A}, xR^b y\} \\ &\cup \{((x,a), \top, (y, 0)) \mid a \in \mathcal{A} \setminus \{\alpha\}, xR^b y, a \vee b = \alpha\} \end{aligned}$$

instead, i.e. explicitly excluding that kind of transition. Indeed, this would still preserve the property asserted of I. However, as this still does not resolve the more pressing matter, the distributivity over composition, we chose to forgo this to keep the definition and proof simpler.

The failure of distributivity over composition is a bit more subtle. For this, suppose there were $a < b < c < \alpha \in \mathcal{A}$ such that $a \vee b = \alpha$ and take

$$R \coloneqq \{(\star, b, \star)\} \qquad\qquad R' \coloneqq \{(\star, c, \star)\}$$

Then clearly, $(\star, a)\, I(R)^\top\, (\star, 0)\, I(R')^\bot\, (\star, c)$. However, as $R' \circ R = \{(\star, b \vee c, \star)\}$, the only two transitions possible via $I(R' \circ R)$ are $(\star, a)\, I(R' \circ R)^\bot\, (\star, \alpha)$ and $(\star, a)\, I(R' \circ R)^\top\, (\star, 0)$. Thus $I(R') \circ I(R) \neq I(R' \circ R)$.

5.3 Subcategories of Trace Categories

In this section, we prove that many properties of trace categories are inherited by some of their subcategories. Motivation these results is the subcategory of a trace category $\mathcal{T}_{\mathcal{A}}^I$ which is restricted to injective trace maps. In the early stages of research for this thesis, some results, including Theorem 4.14, turned out to be significantly easier to obtain when restricting one's attention to these subcategories. We thus believe the subcategory could serve as a sort of "staging area" in which partial results for some open problems, such as the ones listed in Section 7.2, could be obtained first before proving theorems in full generality. Even results restricted to $\mathcal{T}_{\mathcal{A}}^I$ should enjoy wide applicability. For example, all global trace conditions of the cyclic derivation systems from the literature we analyze in Chapter 6 can be formalized in them.

Definition 5.32. Let $\mathcal{A}$ be an activation algebra. The *$\mathcal{A}$-activated injective trace category* $\mathcal{T}_{\mathcal{A}}^{I}$ has as its objects the finite sets. The morphisms between sets X, Y are given by relation $R \subseteq X \times \mathcal{A} \times Y$ such that for every $y \in Y$ there exists at most one $x \in X$ and at most one $b \in \mathcal{A}$ such that $(x, b, y) \in R$. Given $R : X \to Y, R' : Y \to Z$ they compose as follows

$$(x, b \vee b', z) \in R' \circ R \qquad \text{iff} \qquad \exists y \in Y.\ (x, b, y) \in R \text{ and } (y, b', z) \in R'$$

For any set X, the identity is given by $1_X \coloneqq \{(x, 0, x) \mid x \in X\}$.

The precise kind of subcategory we study in this section are those which can be embedded into the full category. Of course, there always exists an embedding $I : \mathcal{T}_{\mathcal{A}}^{I} \to \mathcal{T}_{\mathcal{A}}$.

Definition 5.33. An *embedding* is a faithful functor which is injective on objects.

Proposition 5.34. *For any activation algebra $\mathcal{A}$ there is an embedding $I : \mathcal{T}_{\mathcal{A}}^{I} \to \mathcal{T}_{\mathcal{A}}$.*

Proof. For this, simply take $I(X) \coloneqq X$ on objects and $I(R) \coloneqq R$ on morphisms. □

The first property we transfer from the full category to the subcategory is the trace condition, making $\mathcal{T}_{\mathcal{A}}^{I}$ a trace category.

Proposition 5.35. *If $F : \mathcal{C} \to \mathcal{T}$ is a functor and $\mathcal{T}$ is a trace category then there exists a trace condition for $\mathcal{C}$ such that $F : \mathcal{C} \to \mathcal{T}$ is a trace-simulation.*

Proof. The functor $F : \mathcal{C} \to \mathcal{T}$ induces the following condition on paths

$P : \omega \to \mathcal{C}$ satisfies the trace condition iff $F \circ P : \omega \to \mathcal{T}$ satisfies the trace condition

This trivially makes $F : \mathcal{C} \to \mathcal{T}$ a trace-simulation. Observe that if $P' \subseteq P$ then $I \circ P' \subseteq I \circ P$ as well, thus we have that P' satisfies the induced trace condition iff P does, making this a proper trace condition. □

Definition 5.36. For an activation algebra $\mathcal{A}$ the trace condition of $\mathcal{T}_{\mathcal{A}}^{I}$ is the one induced by the embedding $I : \mathcal{T}_{\mathcal{A}}^{I} \to \mathcal{T}_{\mathcal{A}}$.

Next, we prove that $\mathcal{T}_{\mathcal{A}}^{I}$ has finite products using a well-known fact about embeddings.

Proposition 5.37. *Let $I : \mathcal{C} \to \mathcal{D}$ be an embedding and let $im(I)$ have finite products. Then $\mathcal{C}$ has finite products as well.*

Corollary 5.38. *For any activation algebra $\mathcal{A}$, $\mathcal{T}_{\mathcal{A}}^{I}$ has finite products.*

Proof. Observe that the morphisms chosen in Proposition 5.10 are all injective. The claim thus follows by Proposition 5.37. □

Furthermore, we want to show that a trace condition being recognizable by some infinite word automaton transfers through certain trace-simulations.

Proposition 5.39. *If $G : \mathcal{T} \to \mathcal{T}'$ is a trace-simulation which is injective on objects and the trace condition of $\mathcal{T}'$ is non-deterministically Büchi-recognizable, then so is the trace condition on $\mathcal{T}$.*

Proof. Pick $S \in \mathrm{Ob}(\mathcal{T})$ and morphisms $M = \{\tau_i : X_i \to Y_i \mid 1 \leq i \leq n\}$. As the trace condition of $\mathcal{T}'$ is non-deterministically Büchi-recognizable, there exists a non-deterministic Büchi automaton $\mathfrak{A} = (G(M), Q, \Delta, q_0, F)$ which recognizes paths consisting of morphisms from the set $G(M) \coloneqq \{G(\tau_i) : G(X_i) \to G(Y_i)\}$ with starting state $G(S)$. Consider the automaton $\mathfrak{B} \coloneqq (M, Q, \Delta', q_0, F)$ where $\Delta' \coloneqq \{(q, G(\tau), q') \mid (q, \tau, q') \in \Delta\}$. Then clearly, for some sequence $\sigma \in M^\omega$, we know that $\mathfrak{B}$ accepts σ if and only if $\mathfrak{A}$ accepts $G(\sigma) \coloneqq (i \mapsto G(\sigma_i)) \in G(M)^\omega$ which in turn is the case if and only if $P(i) \coloneqq G(\mathrm{dom}(\sigma_i))$ and $P(i < i+1) \coloneqq G(\sigma_i)$ defines a path $P : \omega \to \mathcal{T}'$ with $P(0) = G(S)$ which satisfies the trace condition of $\mathcal{D}$. By the injectivity of G on objects, this can happen if and only if $P' : \omega \to \mathcal{T}$ with $P'(i) \coloneqq \mathrm{dom}(\sigma_i)$ and $P'(i < i+1) \coloneqq \sigma_i$ is a path through $\mathcal{C}$ with $P(0) = S$. As then $P = G \circ P'$, we know that P satisfies the trace condition if and only if P' does since G is a trace-simulation. □

Remark. This proof strategy extends to other notions kinds of automatons, such as deterministic parity automatons.

Corollary 5.40. *For finite $\mathcal{A}$, the trace condition of $\mathcal{T}_{\mathcal{A}}^{I}$ is non-deterministicall Büchi- and deterministically parity-recognizable.*

Lastly, we derive the properties needed for to apply the results from Section 4.2.

Proposition 5.41. *If $I : \mathcal{C} \to \mathcal{D}$ is an embedding and the set $\mathrm{HOM}_{\mathcal{D}}(I(X), I(Y))$ is finite for $X, Y \in Ob(\mathcal{C})$ then so is $\mathrm{HOM}_{\mathcal{C}}(X, Y)$.*

Proof. This follows from the fact that any embedding is faithful, and thus injective on morphims, and injections preserve finiteness in this manner. □

Proposition 5.42. *Let $F : \mathcal{C} \to \mathcal{D}$ be a computable functor. Then*

(1) Let F be an embedding and $F^{-1} : im(F) \to \mathcal{C}$ be computable as well. If $X, Y, Z \in Ob(\mathcal{C})$ are such for any $\tau : F(X) \to F(Y)$ and $\tau' : F(Y) \to F(Z)$ one can compute $\tau' \circ \tau : F(X) \to F(Y)$ in $\mathcal{D}$, then for any $\tau : X \to Y, \tau' : Y \to Z$, $\tau' \circ \tau : X \to Z$ is computable as well.

(2) If F is an embedding and for any $\tau, \tau' : F(X) \to F(Y)$ it is decidable whether $\tau = \tau'$, then the same is true for any $\tau, \tau' : X \to Y$.

(3) If F is a trace-simulation and for every idempotent $\tau : F(X) \to F(X)$ one can decide whether $\tau^\omega : \omega \to \mathcal{D}$ satisfies the trace condition, then the same can be decided for idempotent $\tau : X \to Y$.

Proof.

1. Simply compute $F^{-1}(F(\tau') \circ F(\tau)) = \tau' \circ \tau$.
2. Deciding whether $F(\tau) = F(\tau')$ is sufficient to determine whether $\tau = \tau'$ by the faithfulness of F.
3. Observe that $F \circ \tau^\omega = F(\tau)^\omega$ and thus τ^ω satisfies the trace condition iff $F(\tau)^\omega$ does. Furthermore, $F(\tau)$ is idempotent as $F(\tau) = F(\tau \circ \tau) = F(\tau) \circ F(\tau)$. Thus the decision can be made for $F(\tau)$ in $\mathcal{D}$. □

Chapter 6

Concrete Cyclic Derivation Systems

This chapter relays the definitions of a selection of cyclic derivation systems from the literature and demonstrates how their global trace condition can be represented in terms of trace categories. Taken together, these further examples serve as an illustration of the fact that the formalism proposed within this thesis is applicable to a wide range of cyclic derivation systems. Each cyclic derivation system in this chapter has been included for a certain reason:

- Section 6.1 covers a cyclic derivation system [20] for a higher-order fixed-point logic whose terms refer to the natural numbers. Together with cyclic arithmetic from Section 1.3, it serves as an example of the broad range of logics (other examples can be found in [4, 9, 11, 34]) whose GTC is straightforwardly formalized in terms of $\mathcal{T}_{\mathbb{B}}$ or $\mathcal{T}_{\mathbb{B}}^{I}$. Its main point of interest thus lies in the ordinarity of its GTC formalization.
- Section 6.2 covers a cyclic derivation system [1, 28] for the modal μ-calculus. Its most interesting aspect, in the context of this thesis, is that its GTC is most naturally expressed by the three-point failure algebra $\mathbb{F}$.
- Section 6.3 covers a cyclic derivation system [31] for the Grzegorczyk modal logic. It is of interest because of the great simplicity of its GTC which translates to a very simple representation in terms of $\mathcal{T}_{\mathbb{B}}^{I}$.

As each section of this chapter serves as a self-contained example, the reader is encouraged to only read those sections which peak their curiosity and skip those which do not. Only the contents Section 6.2 is referred to in Chapter 7 in relation to future work.

6.1 Higher-Order Fixed-Point Logic with Natural Numbers

The higher-order fixed-point logic with natural numbers $\mathrm{HFL}_{\mathbb{N}}$ was put forward by Kobayashi et. al. [18, 19] as an extension of higher-order fixed-point modal logic [40] for the purpose of automated program verification. Due to its higher-order nature, it lends itself well to formalizing the behavior and properties of higher-order programs, such as those written in purely functional programming languages. Kori et. al. [20] have recently introduced a cyclic derivation system for $\mathrm{HFL}_{\mathbb{N}}$ which we present in this section. Unless stated otherwise, all definitions and propositions not related to the abstract framework put forward

in this thesis were taken directly from [20].

Definition 6.1. The *terms and formulas of HFL*$_{\mathbb{N}}$ are typed with the following types

$$A ::= N \mid T \qquad\qquad T ::= \Omega \mid A \to T$$

their syntax being specified by the grammar

$$s, t \in \textsc{Term} ::= x \mid Z \mid S\,t$$
$$\varphi, \psi \in \textsc{Form} ::= x \mid s = t \mid \varphi \vee \psi \mid \varphi \wedge \psi \mid \lambda x^A.\varphi \mid \varphi\,\psi \mid \varphi\,t \mid \mu x^T.\varphi \mid \nu x^T.\varphi$$

The typing rules are as follows, Ξ representing mappings from variables to types

$$\frac{\Xi(x) = a}{\Xi \vdash x : A} \qquad \frac{}{\Xi \vdash Z : N} \qquad \frac{\Xi \vdash t : N}{\Xi \vdash S\,t : N} \qquad \frac{\Xi \vdash s : N \quad \Xi \vdash t : N}{\Xi \vdash s = t : \Omega}$$

$$\frac{\Xi \vdash \varphi : \Omega \quad \Xi \vdash \psi : \Omega}{\Xi \vdash \varphi \vee \psi : \Omega} \qquad \frac{\Xi \vdash \varphi : \Omega \quad \Xi \vdash \psi : \Omega}{\Xi \vdash \varphi \wedge \psi : \Omega} \qquad \frac{\Xi[x \mapsto A] \vdash \varphi : T}{\Xi \vdash \lambda x^A.\varphi : A \to T}$$

$$\frac{\Xi \vdash \varphi : T \to T' \quad \Xi \vdash \psi : T}{\Xi \vdash \varphi\,\psi : T'} \qquad \frac{\Xi \vdash \varphi : N \to T \quad \Xi \vdash t : N}{\Xi \vdash \varphi\,t : T}$$

$$\frac{\Xi[x \mapsto T] \vdash \varphi : T}{\Xi \vdash \mu x^T.\varphi : T} \qquad \frac{\Xi[x \mapsto T] \vdash \varphi : T}{\Xi \vdash \nu x^T.\varphi : T}$$

Definition 6.2. Each type A is interpreted as a poset $[\![A]\!] = ([\![A]\!], \leq_A)$ as follows

$$[\![N]\!] \coloneqq \mathbb{N} \qquad x \leq_N y :\Leftrightarrow x = y$$
$$[\![\Omega]\!] \coloneqq \mathbb{B} \qquad x \leq_\Omega y :\Leftrightarrow x = 0 \text{ or } y = 1$$
$$[\![A \to T]\!] \coloneqq \{f : [\![A]\!] \to [\![A]\!] \mid f \text{ monotone}\} \qquad f \leq_{A \to T} g :\Leftrightarrow \forall x \in [\![A]\!].\ f(x) \leq_T g(x)$$

Furthermore, type environments Ξ are interpreted as posets as follows

$$[\![\Xi]\!] \coloneqq \Pi x \in \mathrm{dom}(\Xi).\ [\![\Xi(x)]\!] \qquad \rho \leq \rho' :\Leftrightarrow \forall x \in \mathrm{dom}(\Xi).\ \rho(x) \leq_T \rho'(x)$$

Formulas are interpreted as monotone functions $[\![\Xi \vdash \varphi : A]\!] \in [\![\Xi]\!] \to [\![A]\!]$

$$\begin{aligned}
[\![\Xi \vdash x : A]\!](\rho) &\coloneqq \rho(x) \\
[\![\Xi \vdash Z : N]\!](\rho) &\coloneqq 0 \\
[\![\Xi \vdash S\,t : N]\!](\rho) &\coloneqq 1 + [\![\Xi \vdash t : N]\!](\rho) \\
[\![\Xi \vdash s = t : \Omega]\!](\rho) &\coloneqq ([\![\Xi \vdash s : N]\!](\rho) = [\![\Xi \vdash t : N]\!](\rho)) \\
[\![\Xi \vdash \varphi \wedge \psi : \Omega]\!](\rho) &\coloneqq [\![\Xi \vdash \varphi : N]\!](\rho) \wedge [\![\Xi \vdash \varphi : N]\!](\rho) \\
[\![\Xi \vdash \varphi \vee \psi : \Omega]\!](\rho) &\coloneqq [\![\Xi \vdash \varphi : N]\!](\rho) \vee [\![\Xi \vdash \varphi : N]\!](\rho) \\
[\![\Xi \vdash \lambda x^A.\varphi : A \to T]\!](\rho) &\coloneqq ([\![A]\!] \ni v \mapsto [\![\Xi \vdash \varphi : A \to T]\!](\rho[x \mapsto v])) \\
[\![\Xi \vdash \varphi\,\psi : T]\!](\rho) &\coloneqq [\![\Xi \vdash \varphi : A \to T]\!](\rho)([\![\Xi \vdash \psi : A \to T]\!](\rho)) \\
[\![\Xi \vdash \varphi\,t : T]\!](\rho) &\coloneqq [\![\Xi \vdash \varphi : N \to T]\!](\rho)([\![\Xi \vdash t : N \to T]\!](\rho)) \\
[\![\Xi \vdash \mu x^T.\varphi : T]\!](\rho) &\coloneqq \textsc{lfp}([\![\Xi \vdash \lambda x^T.\varphi : T \to T]\!](\rho)) \\
[\![\Xi \vdash \nu x^T.\varphi : T]\!](\rho) &\coloneqq \textsc{gfp}([\![\Xi \vdash \lambda x^T.\varphi : T \to T]\!](\rho))
\end{aligned}$$

where LFP and GFP refer to the least and greatest fixed-point operations, respectively. These fixed-points always exists as any poset $[\![T]\!]$, and thus by extension $[\![\Xi]\!] \to [\![T]\!]$, forms a complete lattice and each formula interpretation is monotone.

Example 6.3. $\mathrm{HFL}_{\mathbb{N}}$ is quite expressive. For example, a universal quantifier $\forall$ of type $(N \to \Omega) \to \Omega$ can be defined as follows

$$\forall \coloneqq \lambda p^{N\to\Omega}.\ (\nu a^{N\to\Omega}.\lambda x^{N}.\ P\,x \wedge a\,(S\,x))\,Z$$

For some $\varphi : N \to \Omega$, the expression $\forall\varphi$ can be transformed into $\varphi\,0 \wedge \varphi\,1 \wedge \varphi\,2 \wedge \ldots$ by repeated β-reductions and fixed-point unfoldings, which perfectly captures the universal quantification on natural numbers. Readers curious about further examples of $\mathrm{HFL}_{\mathbb{N}}$ expressions are encouraged to consult [20].

Definition 6.4. We write $\Xi \mid \Gamma \Rightarrow \Delta$ for lists of formulas Γ, Δ if for every formula $\varphi \in (\Gamma, \Delta)$ we have $\Xi \vdash \varphi : \Omega$. Formally, all sequents of the derivation system for $\mathrm{HFL}_{\mathbb{N}}$ are such $\Xi \mid \Gamma \Rightarrow \Delta$, although we do not write out the Ξ for the sake of legibility. The derivation rules of the sequent calculus $\mathcal{H}$ for $\mathrm{HFL}_{\mathbb{N}}$ are as follows

$$\textsc{Ax}\frac{}{\varphi \Rightarrow \varphi} \qquad \textsc{Cut}\frac{\Gamma \Rightarrow \varphi, \Delta \quad \Gamma, \varphi \Rightarrow \Delta}{\Gamma \Rightarrow \Delta} \qquad \textsc{WkL}\frac{\Gamma \Rightarrow \Delta}{\Gamma, \varphi \Rightarrow \Delta}$$

$$\textsc{WkR}\frac{\Gamma \Rightarrow \Delta}{\Gamma \Rightarrow \varphi, \Delta} \qquad \textsc{CtrL}\frac{\Gamma, \varphi, \varphi \Rightarrow \Delta}{\Gamma, \varphi \Rightarrow \Delta} \qquad \textsc{CtrR}\frac{\Gamma \Rightarrow \varphi, \varphi, \Delta}{\Gamma \Rightarrow \varphi, \Delta}$$

$$\textsc{ExL}\frac{\Gamma, \psi, \varphi, \Gamma' \Rightarrow \Delta}{\Gamma, \varphi, \psi, \Gamma' \Rightarrow \Delta} \qquad \textsc{ExR}\frac{\Gamma \Rightarrow \Delta, \psi, \varphi, \Delta'}{\Gamma \Rightarrow \Delta, \varphi, \psi, \Delta'} \qquad \textsc{Sub}\frac{\Gamma \Rightarrow \Delta}{\Gamma[\varphi/x] \Rightarrow \Delta[\varphi/x]}$$

$$\textsc{Mono}\frac{\begin{array}{c}x \text{ occurs in } \varphi \text{ precisely once}\\ \Gamma, \varphi, \psi\,\vec{y} \Rightarrow \varphi, \theta\,\vec{y}, \Delta\end{array}}{\Gamma, \varphi[\psi/x] \Rightarrow \varphi[\theta/x], \Delta} \qquad =\!\mathrm{L}\frac{\Gamma[t/x, s/y] \Rightarrow \Delta[t/x, s/y]}{\Gamma[s/x, t/y], s = t \Rightarrow \Delta[s/x, t/y]}$$

$$=\!\mathrm{R}\frac{}{\Gamma \Rightarrow t = t, \Delta} \qquad \vee\mathrm{L}\frac{\Gamma, \varphi \Rightarrow \Delta \quad \Gamma, \psi \Rightarrow \Delta}{\Gamma, \varphi \vee \psi \Rightarrow \Delta} \qquad \vee\mathrm{R}\frac{\Gamma \Rightarrow \varphi, \psi, \Delta}{\Gamma \Rightarrow \varphi \vee \psi, \Delta}$$

$$\wedge\mathrm{L}\frac{\Gamma, \varphi, \psi \Rightarrow \Delta}{\Gamma, \varphi \vee \psi \Rightarrow \Delta} \qquad \wedge\mathrm{R}\frac{\Gamma \Rightarrow \varphi, \Delta \quad \Gamma \Rightarrow \psi, \Delta}{\Gamma \Rightarrow \varphi \vee \psi, \Delta} \qquad \lambda\mathrm{L}\frac{\Gamma, \varphi[\psi/x]\,\vec{\theta} \Rightarrow \Delta}{\Gamma, (\lambda x.\varphi)\,\psi\,\vec{\theta} \Rightarrow \Delta}$$

$$\lambda\mathrm{R}\frac{\Gamma \Rightarrow \varphi[\psi/x]\,\vec{\theta}, \Delta}{\Gamma \Rightarrow (\lambda x.\varphi)\,\psi\,\vec{\theta}, \Delta} \qquad \sigma\mathrm{L}\frac{\Gamma, \varphi[(\sigma x.\varphi)/x]\,\vec{\psi} \Rightarrow \Delta}{\Gamma, (\sigma x.\varphi)\,\vec{\psi} \Rightarrow \Delta} \qquad \sigma\mathrm{R}\frac{\Gamma \Rightarrow \varphi[(\sigma x.\varphi)/x]\,\vec{\psi}, \Delta}{\Gamma \Rightarrow (\sigma x.\varphi)\,\vec{\psi}, \Delta}$$

$$\frac{\Gamma, N\,x^{N} \Rightarrow \Delta}{\Gamma \Rightarrow \Delta} \qquad \frac{}{S\,t = S \Rightarrow} \qquad \frac{\Gamma, s = t \Rightarrow \Delta}{\Gamma, S\,s = S\,t \Rightarrow \Delta}$$

where $N \coloneqq \mu X.\lambda x.(x = Z) \vee (\exists x'.\ x = S\,x' \wedge X\,x')$ and $\sigma \in \{\mu, \nu\}$. We generally work with ∞-derivations of this derivation system and call regular ∞-derivations *cyclic derivations*.

Remark. The Mono-rule we give is slightly different from that of [20]. We chose this presentation as it was proposed to us by the authors of [20] in an e-mail exchange because it leads to a slightly clearer formulation of the GTC.

Remark. In [20], cyclic derivations are defined as finite derivation trees in which non-axiomatic leaves are assigned an inner node companion. We chose to forgo such technicalities and simply, equivalently, consider regular ∞-derivations as cyclic.

Definition 6.5. Let $\Gamma' \Rightarrow \Delta'$ be a premise of derivation rule $R \in \mathcal{H}$ concluding $\Gamma \Rightarrow \Delta$ as below

$$\mathrm{R}\frac{\dots \quad \Gamma' \Rightarrow \Delta' \quad \dots}{\Gamma \Rightarrow \Delta}$$

Pick some occurrence of a formula φ from either Γ or Δ. We call an occurrence of a formula φ' in Γ' or Δ' the *precursor of* φ, writing $\varphi' \prec_R \varphi$ if

(a) φ is not principal in R, that is, if φ is not "altered by R", and $\varphi = \varphi'$ with φ' being in the same position as φ

(b) φ is principal in $\mathcal{R}$ and φ' is one of the formula occurrences resulting from φ via R

Remark. Technically, being "an occurrence resulting from φ via R" in (b) needs to be defined on a case-by-case basis for each possible rule. However, the notion is quite intuitive, so we simply give a few illustrative examples:

$$\vee\mathrm{R}\frac{\Gamma \Rightarrow \varphi, \psi, \Delta}{\Gamma \Rightarrow \varphi \vee \psi, \Delta} \qquad \textsc{Mono}\frac{\Gamma, \varphi, \psi\,\vec{y} \Rightarrow \varphi, \theta\,\vec{y}, \Delta}{\Gamma, \varphi[\psi/x] \Rightarrow \varphi[\theta/x], \Delta}$$

In ($\vee$R), both φ and ψ are occurrences resulting from $\varphi \vee \psi$. In (Mono), both φ and $\psi\,\vec{y}$ are occurrences resulting from $\varphi[\psi/x]$. A definition of the concept in full detail can be also found in [20, Appendix A].

Definition 6.6. Let $(\Gamma_i \vdash \Delta_i)_{i\in\omega}$ be a branch through an ∞-derivation according to Definition 6.4. A sequence of formulas $(\varphi_i)_{i\in\omega}$ is called a *pre-trace* along $(\Gamma_i \vdash \Delta_i)_{i\in\omega}$ if $\varphi_{i+1} \prec_{\mathcal{R}} \varphi_i$ for the rule R deriving $\Gamma_i \vdash \Delta_i$ from $\Gamma_{i+1} \vdash \Delta_{i+1}$ for every $i \in \omega$. Such a pre-trace is a *trace* if case (b) from Definition 6.5 occurs for infinitely many $\varphi_{i+1} \prec_R \varphi_i$. If the φ_i always occur in Γ_i (in Δ_i), we call $(\varphi_i)_{i\in\omega}$ a *left trace* (*right trace*).

Definition 6.7. Given a path $(\Gamma_i \vdash \Delta_i)_{i\in\omega}$ and a trace $(\varphi_i)_{i\in\omega}$ along it, its *unfolding tree* is defined by by annotating the fixed-point quantifiers occurring in the φ_i with words from ω^* as follows:

- Label each occurrence of a fixed-point quantifier in φ_0 with ε and take $U_0 \coloneqq \{\varepsilon\}$
- If φ_{i+1} is results from the σ-rule, meaning $\varphi_i = (\sigma^s x.\psi)\,\vec{\theta}$ for $s \in U_i$, then label φ_{i+1} as $\psi[\sigma^{si}x.\psi/x]\,\vec{\theta}$, i.e. label each of the instances of σ "substituted in" by the unfolding with $si \in \omega^*$. Take $U_{i+1} \coloneqq U_i \cup \{si\}$.
- If φ_i is not principal in a σ-rule, simply "leave the labels on" in the transformation from φ_i to φ_{i+1}. For example, if $\varphi_{i+1} = \psi$ results from $\varphi_i = \psi \vee \theta$ via $\vee$L then simply label the fixed-point quantifiers in φ_{i+1} as they were labeled in φ_i. Take $U_{i+1} \coloneqq U_i$.

The unfolding tree of φ is defined as $U(\varphi) \coloneqq \bigcup_{i\in\omega} U_i$.

Lemma 6.8 ([20, Lemma 22])**.** *For every trace $(\varphi_i)_{i\in\omega}$ along a path $(\Gamma_i \vdash \Delta_i)_{i\in\omega}$ the unfolding tree $U(\varphi)$ has precisely one infinite path.*

Definition 6.9. Given a trace $(\varphi_i)_{i\in\omega}$ along a path $(\Gamma_i \vdash \Delta_i)_{i\in\omega}$ we call it a *good trace* if

- $(\varphi_i)_{i\in\omega}$ is a left trace and the infinite branch of $U(\varphi)$ follows a μ-quantifier
- $(\varphi_i)_{i\in\omega}$ is a right trace and the infinite branch of $U(\varphi)$ follows a ν-quantifier

An ∞-derivation (cyclic derivation) of a sequent $\Gamma \vdash \Delta$ is a *∞-proof* (*cyclic proof*) if every infinite branch through it has a suffix which has a good trace running along it.

Remark. This formulation of the trace condition in terms of unfolding trees is inspired by the winning condition of $\mathrm{HFL}_{\mathbb{N}}$ games in [39] rather than the trace condition of [20].

We now demonstrate how the global trace condition of $\mathrm{HFL}_{\mathbb{N}}$ can be captured in terms of the $\mathbb{B}$-activated injective trace category $\mathcal{T}^I_{\mathbb{B}}$. This formalization of the trace condition allows us to apply theorems from Chapter 4 to $\mathrm{HFL}_{\mathbb{N}}$.

Definition 6.10. We call $p \in \mathbb{B}^*$ a *subformula path* and define a partial addressing function on formulas φ and sequences of formulas Γ as follows:

$$\varphi@\varepsilon := \varepsilon \qquad (\varphi_0 \Box \varphi_1)@ip := \varphi_i@p \qquad (\lambda x.\varphi)@0p := \varphi@p \qquad (\varphi_0\,\varphi_1)@ip := \varphi_i@p$$

$$(\varphi\,t)@0p := \varphi@p \qquad (\sigma x.\varphi)@0p := \varphi@p \qquad (\Gamma,\varphi)@0p := \varphi@p \qquad (\Gamma,\varphi)@1p := \Gamma@p$$

where $\Box \in \{\vee, \wedge\}$ and $\sigma \in \{\mu, \nu\}$. The set of *fixed-point paths* is defined as

$$\Pi(\varphi) := \{p \in \mathbb{B}^* \mid \varphi@p = \sigma x.\psi\}$$

and extended analogously to $\Pi(\Gamma)$ for sequences Γ.

Definition 6.11. Let $\mathcal{H}$ be as in Definition 6.4, then we define a trace-interpretation $I : \mathcal{D}^{\mathrm{op}}_{\mathcal{H}} \to \mathcal{T}^I_{\mathbb{B}}$ via Lemma 3.25 with the mapping $i : \textsc{Seq} \to \mathrm{Ob}(\mathcal{T}^I_{\mathbb{B}})$ given by

$$i(\Gamma \Rightarrow \Delta) := \{\Gamma@p = \mu x.\varphi \mid p \in \Pi(\Gamma)\} + \{\Delta@p = \nu x.\varphi \mid p \in \Pi(\Delta)\}$$

and for each $R \in \mathcal{H}$ where

$$\mathrm{R}\frac{\Gamma_0 \Rightarrow \Delta_0 \quad \dots \quad \Gamma_n \Rightarrow \Delta_n}{\Gamma \Rightarrow \Delta}$$

we define the trace maps $\tau^R_j : i(\Gamma \Rightarrow \Delta) \to i(\Gamma_j \Rightarrow \Delta_j)$ such that $(p, b, q) \in \tau^R_j$ if the occurrence q is a predecessor of the occurrence of p, where we take $b = \top$ if R is a σ-rule and p is its principal occurrence and $b = \bot$ otherwise.

Remark. Similar to Definition 6.5, the notion of "predecessor" used in this definition would need to be defined on a case-by-case basis. Instead, we again opt to give a few illustrative examples. Annotating each quantifier with its position, the rule

$$\lambda\mathrm{L}\frac{(\mu^c y.y) \vee (\mu^d y.y) \Rightarrow \nu^e y.y}{(\lambda x.\ x \vee x)\,\mu^a y.y \Rightarrow \nu^b y.y}$$

would yield $\tau = \{(a, \bot, c), (a, \bot, d), (b, \bot, e)\}$. The rule

$$\nu\mathrm{R}\frac{\Rightarrow (\nu^b x.\ x \vee x) \vee (\nu^c x.\ x \vee x)}{\Rightarrow \nu^a x.\ x \vee x}$$

would yield $\tau = \{(a, \top, b), (a, \top, c)\}$. Lastly, the rule

$$\textsc{Mono}\frac{(\mu^c x.x) \vee y, \mu^d x.x \Rightarrow (\mu x.x) \vee y, \mu x.x}{(\mu^a x.x) \vee (\mu^b x.x) \Rightarrow (\mu x.x) \vee (\mu x.x)}$$

would yield $\tau = \{(a, \bot, c), (b, \bot, d)\}$.

Proposition 6.12. *Given some ∞-derivation D of a sequent $\Gamma \Rightarrow \Delta$, it is an ∞-proof according to Definition 6.9 if and only if $D : \omega^{op} \to \mathcal{D}^{\mathcal{P}}_{\mathcal{H}}$ is an ∞-proof according to the trace condition induced by $I : \mathcal{D}^{op}_{\mathcal{H}} \to \mathcal{T}^{I}_{\mathbb{B}}$ from Definition 6.11.*

Proof. Simply observe that the $\mathcal{T}^{I}_{\mathbb{B}}$ trace condition induced by $I : \mathcal{D}^{\mathrm{op}}_{\mathcal{H}} \to \mathcal{T}^{I}_{\mathbb{B}}$ is satisfied precisely if there exists a μ-quantifier to the left of the $\Rightarrow$ or a ν-quantifier to the right of the $\Rightarrow$ which is, heredetarily, unfolded infinitely often. □

Corollary 6.13. *Given a cyclic derivation of $\mathcal{H}$ it is decidable whether it is a proof.*

Proof. Every cyclic derivation D is a regular ∞-derivation $D : \omega^{\mathrm{op}} \to \mathcal{D}^{\mathcal{P}}_{\mathcal{H}}$. From this, we can generate an equivalent abstract cyclic derivation D' on $\mathcal{T}^{I}_{\mathbb{B}}$ via Proposition 3.37. Now, either the algorithm from Theorem 4.4 or Theorem 4.14 can be applied to decide whether D' is an abstract cyclic proof. □

Remark. This result is also derived in [20, Section 4] via an explicit automata construction.

6.2 Modal μ-Calculus

The modal μ-calculus μML, introduced by Kozen [21], extends modal logic with fixed-point quantifiers. These quantifiers grant μML great expressive strength, notably allowing for many logics from the field of program verification to be embedded into μML. A cyclic derivation system for μML was first given by Niwinski and Walukiewicz [28] in 1996, although they presented it as a tableaux system yielding a decision procedure for a fragment of μML. The cyclic derivation system for μML we present in this section is essentially the same as that given by Niwinski and Walukiewicz, although we follow the modern presentation of it given by Afshari and Leigh in their technical report [1]. Unless explicitly stated otherwise, all definitions and theorems in this section which do not involve the abstract framework of this thesis stem from their report.

Definition 6.14. For a set Π of propositional letters, the syntax of the modal μ-calculus is given by the following grammar

$$\varphi, \psi \in \textsc{Form} ::= p \mid \overline{p} \mid x \mid \varphi \wedge \psi \mid \varphi \vee \psi \mid \Box\varphi \mid \Diamond\varphi \mid \mu x.\varphi \mid \nu x.\varphi$$

where $p \in \Pi$ and $x \in \textsc{Var}$ are formal variables.

Definition 6.15. A *Kripke frame* is a pair (W, R) where W is a set of *worlds* and $R \subseteq W \times W$ is a relation. A *Kripke model* is a triple (W, R, ρ) where (W, R) is a Kripke frame and $\rho : \Pi \cup \textsc{Var} \to \mathcal{P}(W)$ is an *assignment*.

For a Kripke frame (W, R) and an assignment $\rho : \Pi \cup \textsc{Var} \to \mathcal{P}(W)$ the *evaluation of* φ

under ρ, denoted by $[\![\varphi]\!]^\rho$, is recursively defined by

$$\begin{aligned}
[\![p]\!]^\rho &:= \rho(p) \\
[\![\overline{p}]\!]^\rho &:= W \setminus [\![p]\!]^\rho & [\![\Box\varphi]\!]^\rho &:= \{w \in W \mid \forall v \in [\![\varphi]\!]^\rho.\ w\,R\,v \text{ entails } v \in [\![\varphi]\!]^\rho\} \\
[\![x]\!]^\rho &:= \rho(x) & [\![\Diamond\varphi]\!]^\rho &:= \{w \in W \mid \exists v \in [\![\varphi]\!]^\rho.\ w\,R\,v \wedge v \in [\![\varphi]\!]^\rho\} \\
[\![\varphi \wedge \psi]\!]^\rho &:= [\![\varphi]\!]^\rho \cap [\![\psi]\!]^\rho & [\![\mu x.\varphi]\!]^\rho &:= \bigcap\{S \subseteq W \mid [\![\varphi]\!]^{\rho[x\mapsto S]} \subseteq S\} \\
[\![\varphi \vee \psi]\!]^\rho &:= [\![\varphi]\!]^\rho \cup [\![\psi]\!]^\rho & [\![\nu x.\varphi]\!]^\rho &:= \bigcup\{S \subseteq W \mid S \subseteq [\![\varphi]\!]^{\rho[x\mapsto S]}\}
\end{aligned}$$

Note that by the Knaster-Tarski theorem, $[\![\mu x.\varphi]\!]^\rho$ and $[\![\nu x.\varphi]\!]^\rho$ are the least and greatest fixed-point of $(S \mapsto [\![\varphi]\!]^{\rho[x\mapsto S]}) : P(W) \to P(W)$, respectively.

A formula φ is *a validity of the modal μ-calculus* if for every Kripke model (W, R, ρ) we have $[\![\varphi]\!]^\rho = W$.

Definition 6.16. Taking $\textsc{Var}(\varphi) := \{x \in \textsc{Var} \mid x \text{ occurs in } \varphi\}$ the *subsumption order* $<_\varphi \subseteq \textsc{Var}(\varphi) \times \textsc{Var}(\varphi)$ is the smallest partial order such that $x <_\varphi y$ if $\sigma y.\psi$ occurs as a subformula of φ for some $\sigma \in \{\mu, \nu\}$ and ψ, and further x is free in $\sigma y.\psi$. A formula is *well-named* if $<_\varphi$ is irreflexive.

Remark. Note that every formula φ can be α-renamed into a well-named formula, and that for a well-named $\varphi = \sigma x.\psi$ we have $<_\varphi = <_{\varphi'}$ where $\varphi' := \psi[\sigma x.\psi/x]$ for $\sigma \in \{\mu, \nu\}$.

Definition 6.17. The sequents of μML are given by finite sets Γ of well-named formulas. As usual, Γ, φ is short for $\Gamma \cup \{\varphi\}$. The derivation rules $\mathcal{M}$ of μML are given below

$$\textsc{Ax}\frac{}{p, \overline{p}} \qquad \textsc{Wk}\frac{\Gamma}{\Gamma, \varphi} \qquad \vee\frac{\Gamma, \varphi, \psi}{\Gamma, \varphi \vee \psi} \qquad \wedge\frac{\Gamma, \varphi \quad \Gamma, \psi}{\Gamma, \varphi \wedge \psi} \qquad \textsc{Mod}\frac{\Gamma, \varphi}{\Diamond\Gamma, \Box\varphi}$$

$$\mu\frac{\Gamma, \varphi[\mu x.\varphi/x]}{\Gamma, \mu x.\varphi} \qquad \nu\frac{\Gamma, \varphi[\nu x.\varphi/x]}{\Gamma, \nu x.\varphi}$$

where $\Diamond\Gamma = \{\Diamond\varphi \mid \varphi \in \Gamma\}$. We generally consider ∞-derivations in that sequent calculus, calling regular ∞-derivations *cyclic derivations*.

Definition 6.18. Let Γ be the conclusion of the derivation rule $R \in \mathcal{M}$ and let Γ' be one of R's premises. We call $\varphi' \in \Gamma'$ a *precursor of* $\varphi \in \Gamma$, writing $\varphi' \prec \varphi$

(i) if φ is not principal in R, that is, if φ is not "altered by R", and $\varphi = \varphi'$

(ii) or if φ is principal in R and φ' is one of the formula occurrences resulting from φ via R

Definition 6.19. Let $(\Gamma_i)_{i\in\omega}$ be a branch through an ∞-derivation formed according to Definition 6.17. A sequence of formulas $(\varphi_i)_{i\in\omega}$ is called a *trace along* $(\Gamma_i)_{i\in\omega}$ if $\varphi_{i+1} \prec \varphi_i$ for all $i \in \omega$ according to the rule deriving Γ_i from Γ_{i+1}. Such a trace is called a ν-trace if there exists an $x \in \textsc{Var}(\varphi_0)$ such that

(i) $\varphi_i = \nu x.\psi$ and $\varphi_{i+1} = \psi[\nu x.\psi/x]$ for infinitely many $i \in \omega$, and

(ii) for any $y \in \textsc{Var}(\varphi_0)$ if there are infinitely many $\varphi_i = \mu y.\theta$ then $x <_{\varphi_i} y$

Remark. Note that whenever $\varphi' \prec \varphi$ and φ is well-named then so is φ'. Furthermore, this means that $<_{\varphi'} = <_\varphi \cap \mathrm{VAR}(\varphi') \times \mathrm{VAR}(\varphi')$. We often simply write $<_\varphi$ for the subsumption relation of a trace $(\varphi_i)_{i\in\omega}$ which should be taken as $<_{\varphi_i}$ for the "contextually sensible" $i \in \omega$. This abuse of notation is appropriate because the facts stated above mean that all $<_{\varphi_i}$ are "consistent with each other".

Definition 6.20. An ∞-derivation of μML is an ∞-proof if every infinite branch running through it has a ν-trace.

The global trace condition of μML can be captured in terms of $\mathcal{T}^I_{\mathbb{F}}$. This allows us to apply theorems from Chapter 4 to μML.

Definition 6.21. We define a trace-interpretation $I : \mathcal{D}^{\mathrm{op}}_{\mathcal{M}} \to \mathcal{T}^I_{\mathbb{F}}$ according to Lemma 3.25 with the mapping $i : \mathrm{SEQ} \to \mathrm{Ob}(\mathcal{T}^I_{\mathcal{M}})$ given by the function

$$i(\Gamma) \coloneqq \Sigma\varphi \in \Gamma.\mathrm{VAR}(\varphi)$$

That is, pairs of formulas φ occurring in Γ and variables occurring in φ. For each $R \in \mathcal{M}$ where

$$\mathrm{R}\frac{\Gamma_0 \quad \dots \quad \Gamma_n}{\Gamma}$$

then we define the trace maps $\tau^R_j : i(\Gamma) \to i(\Gamma_j)$ as

$$\tau^R_j \coloneqq \left\{ ((\psi, x), a, (\psi', x)) \;\middle|\; \psi' \prec \psi, a = \begin{cases} 2 & \text{if } R = \mu, \psi = \mu y.\theta \text{ and } x \not<_\psi y \\ 1 & \text{if } R = \nu, \psi = \nu x.\theta \\ 0 & \text{otherwise} \end{cases} \right\}$$

Proposition 6.22. *Given an ∞-derivation D of a sequent Γ, it is an ∞-proof according to Definition 6.19 if and only if $D : \omega^{op} \to \mathcal{D}^{\mathcal{P}}_{\mathcal{M}}$ is an ∞-proof according to the trace condition induced by $I : \mathcal{D}^{op}_{\mathcal{M}} \to \mathcal{T}^I_{\mathbb{F}}$ from Definition 6.21.*

Proof. First of all, observe that if an infinite branch $(\Gamma_i)_{i\in\omega}$ of D has a ν-trace $(\varphi_i)_{i\in\omega}$, then there exists an $x \in \mathrm{VAR}(\varphi_0)$ such that

(i) the formula $\nu x.\psi$ associated with x is unfolded infinitely often along $(\varphi_i)_{i\in\omega}$

(ii) if for some $x \not<_\varphi y$ the associated formula is $\mu y.\theta$, it is unfolded only finitely often

That means there exists some index j from which point on no $\mu y.\theta$ for $x \not<_\varphi y$ is unfolded anymore. Then the sequence $\sigma_i \coloneqq (\varphi_{i+j}, x)$ witnesses that the path $P : \omega \to \mathcal{T}^I_{\mathbb{F}}$ generated by $(\Gamma_i)_{i\in\omega}$ via $I : \mathcal{D}^{\mathrm{op}}_{\mathcal{M}} \to \mathcal{T}^I_{\mathbb{F}}$ satisfies the trace condition of $\mathcal{T}^I_{\mathbb{F}}$ as the activation state 2 never occurs along it.

A similar argument can be made to show that if the branch $P : \omega \to \mathcal{T}^I_{\mathbb{F}}$ through D generated by a branch $(\Gamma_i)_{i\in\omega}$ satisfies the trace condition of $\mathcal{T}^I_{\mathbb{F}}$ this means there exists a suffix of $(\Gamma_i)_{i\in\omega}$ along which a ν-trace runs, which also means a ν-trace runs along $(\Gamma_i)_{i\in\omega}$ starting at Γ_0. However, the actual argument for this is very combinatorial as the trace condition of $\mathcal{T}^I_{\mathbb{F}}$ is defined in terms of arbitrary subpaths $Q \subseteq P$, which is why we chose not elaborate it any further. □

Corollary 6.23. *For every cyclic derivation of $\mathcal{M}$ it is decidable whether it is a proof.*

Proof. Every cyclic derivation D is a regular ∞-derivation $D : \omega^{\mathrm{op}} \to \mathcal{D}^{\mathcal{P}}_{\mathcal{M}}$. From this, we can generate an equivalent abstract cyclic derivation D' on $\mathcal{T}^{I}_{\mathbb{F}}$ via Proposition 3.37. Now, either the algorithm from Theorem 4.4 or Theorem 4.14 can be applied to decide whether D' is an abstract cyclic proof. □

Proposition 6.24. *Every ∞-derivation of $\mathcal{M}$ is derived using a finite fragment of the derivation rules given in Definition 6.17.*

Proof. The Fischer-Ladner closure $\mathrm{Clos}(\varphi)$ of a formula φ is the smallest set such that

- $\varphi \in \mathrm{Clos}(\varphi)$
- if $\psi \circ \theta \in \mathrm{Clos}(\varphi)$ then $\psi, \theta \in \mathrm{Clos}(\varphi)$ for $\circ \in \{\wedge, \vee\}$
- if $\bigcirc\psi \in \mathrm{Clos}(\varphi)$ then $\psi \in \mathrm{Clos}(\varphi)$ for $\bigcirc \in \{\Box, \Diamond\}$
- if $\sigma x.\psi \in \mathrm{Clos}(\varphi)$ then $\psi[\sigma x.\psi/x] \in \mathrm{Clos}(\varphi)$ for $\sigma \in \{\mu, \nu\}$

It is well known that for any φ the set $\mathrm{Clos}(\varphi)$ is finite, as for example argued in [21]. For a set of formulas Γ we extend $\mathrm{Clos}(\Gamma) \coloneqq \bigcup_{\varphi\in\Gamma} \mathrm{Clos}(\varphi)$.

Now consider some ∞-proof $D : \omega^{\mathrm{op}} \to \mathcal{D}^{\mathcal{P}}_{\mathcal{M}}$ such that $D(0) = [\Gamma]$. We first claim that for any sequent Γ' occurring within D, we have $\Gamma' \subseteq \mathrm{Clos}(\Gamma)$. This is the case as every derivation rule only adds new formulas to the sequents in its premises which are in the closures of formulas already present in the premise, as is easily observed. This, in turn, means that all sequents which may ever occur in D are collected in the finite set

$$\{\Gamma' \mid \Gamma' \subseteq \mathrm{Clos}(\Gamma)\}$$

As only a finite number of derivation rules can be applied to any given sequent Γ', there also exist only a finite number of derivation rules which can be applied within D. □

Corollary 6.25. *If Γ can be proven by an ∞-proof of μML, it can also be proven via a cyclic proof.*

Proof. We may apply Theorem 4.6 by viewing the ∞-proof $D : \omega^{\mathrm{op}} \to \mathcal{D}^{\mathcal{P}}_{\mathcal{M}}$ as $D : \omega^{\mathrm{op}} \to \mathcal{D}^{\mathcal{P}}_{\mathcal{M}'}$ where $\mathcal{M}'$ is the category of derivations generated by the finite fragment of $\mathcal{M}$ used within D according to Proposition 6.24. □

Corollary 6.26. *It is decidable whether any given sequent of μML is provable.*

Proof. Follows from Corollary 4.7 in the same manner as the result above. □

6.3 Grzegorczyk Modal Logic

The Grzegorczyk modal logic **Grz**, introduced by the eponymous author in [14], is the modal logic of reflexive, transitive and conversely well-founded modal Kripke frames. It is also the provability logic of Peano arithmetic obtained when interpreting $\Box\varphi$ as "φ is both provable and true". In the context of cyclic proof theory, it is remarkable in featuring neither inductively defined objects nor fixed-point quantifiers explicitly, the staples of logics admitting cyclic derivation systems, but nonetheless having a cyclic derivation system, which was put forward by Savateev and Shamkanov in [31].

Definition 6.27. For a set Π of propositional letters, the syntax of modal logic is

$$\varphi, \psi \in \textsc{Form} ::= \bot \mid p \mid \varphi \to \psi \mid \Box\varphi \qquad\qquad p \in \Pi$$

the remaining connectives can be obtained as their DeMorgan translations

$$\neg\varphi := \varphi \to \bot \qquad \varphi \wedge \psi := \neg(\varphi \to \neg\psi) \qquad \varphi \vee \psi := \neg\varphi \to \psi \qquad \Diamond\varphi := \neg\Box\neg\varphi$$

Definition 6.28. A *Kripke frame* is a pair (W, R) where W is a set of *worlds* and $R \subseteq W \times W$ is a relation. A Kripke frame is *reflexive* or *transitive* if R is such, respectively. A Kripke frame is *conversely well-founded* if $R^{-1} := \{(y, x) \mid x\, R\, y\}$ is well-founded. A *Kripke model* is a triple (W, R, ρ) where (W, R) is a Kripke frame and $\rho : W \to \mathcal{P}(\Pi)$ is an *assignment*.

Fix a Kripke model (W, R, ρ) and some $x \in W$. The *satisfaction of formula φ at world x*, denoted by $x \vDash \varphi$, is defined as follows

$$\begin{aligned} x \vDash \bot &:\Leftrightarrow \text{never} & x \vDash \varphi \to \psi &:\Leftrightarrow x \vDash \varphi \text{ entails } x \vDash \psi \\ x \vDash p &:\Leftrightarrow p \in \rho(x) & x \vDash \Box\varphi &:\Leftrightarrow y \vDash \varphi \text{ for every } x\, R\, y \end{aligned}$$

A formula φ is *a validity of Grzegorczyk modal logic* if for every Kripke frame (W, R) which is reflexive, transitive and conversely well-founded, every assignment $\rho : W \to \mathcal{P}(\Pi)$ and every world $x \in W$ we have $x \vDash \varphi$.

Definition 6.29. The sequents of Grzegorczyk modal logic are given by $\Gamma \Rightarrow \Delta$ where Γ, Δ are sets of formulas. The expression Γ, φ is short for $\Gamma \cup \{\varphi\}$ and the expression Γ, Δ is short for $\Gamma \cup \Delta$. The derivation rules $\mathcal{G}$ of Grzegorczyk modal logic are

$$\frac{}{\Gamma, p \Rightarrow p, \Delta} \qquad \frac{}{\Gamma, \bot \Rightarrow \Delta} \qquad \to\text{L}\frac{\Gamma, \psi \Rightarrow \Delta \quad \Gamma \Rightarrow \varphi, \Delta}{\Gamma, \varphi \to \psi \Rightarrow \Delta}$$

$$\to\text{R}\frac{\Gamma, \varphi \Rightarrow \psi, \Delta}{\Gamma \Rightarrow \varphi \to \psi, \Delta} \qquad \text{refl}\frac{\Gamma, \Box\varphi, \varphi \Rightarrow \Delta}{\Gamma, \Box\varphi \Rightarrow \Delta} \qquad \Box\frac{\Gamma, \Box\Gamma' \Rightarrow \varphi, \Delta \quad \Box\Gamma' \Rightarrow \varphi}{\Gamma, \Box\Gamma' \Rightarrow \Box\varphi, \Delta}$$

where $\Box\Gamma' := \{\Box\varphi \mid \varphi \in \Gamma'\}$. We generally consider ∞-derivations in that sequent calculus, calling regular ∞-derivations *cyclic derivations*.

Definition 6.30. An ∞-derivation of Grzegorczyk modal logic is an ∞-proof if every infinite branch running through it passes through the right premise ($\Box\Gamma' \Rightarrow \varphi$) of the $\Box$-rule infinitely often.

Remark. The cyclic derivation system of Grzegorczyk modal logic stands apart from the other cyclic derivation systems we cover in this thesis by the fact that its soundness condition is not formulated as a property of traces which run along each infinite branch but instead a property of the branches themselves.

The global trace condition of **Grz** can be captured in terms of $\mathcal{T}^I_{\mathbb{B}}$. In this, the simplicity of Definition 6.30 extends to Definition 6.31. Capturing the trace condition of **Grz** via a trace category also allows us to apply theorems from Chapter 4.

Definition 6.31. Denoting the derivation system from Definition 6.29 by $\mathcal{G}$, we define a trace-interpretation $I : \mathcal{D}_{\mathcal{G}}^{\mathrm{op}} \to \mathcal{T}_{\mathbb{B}}^{I}$ according to Lemma 3.25 with the mapping $i : \textsc{Seq} \to \mathrm{Ob}(\mathcal{T}_{\mathbb{B}}^{I})$ given by the constant function $i(\Gamma \Rightarrow \Delta) \coloneqq \{\star\}$. For each $R \in \mathcal{G}$ which is not an instance of the $\Box$-rule, we take $\tau_j^R \coloneqq \{(\star, \bot, \star)\}$. For any $\Box$ instance

$$\Box \frac{\Gamma, \Box\Gamma' \Rightarrow \varphi, \Delta \qquad \Box\Gamma' \Rightarrow \varphi}{\Gamma, \Box\Gamma' \Rightarrow \Box\varphi, \Delta}$$

we fix $\tau_0^{\Box} \coloneqq \{(\star, \bot, \star)\}$ for the left side and for the right side we take $\tau_1^{\Box} \coloneqq \{(\star, \top, \star)\}$.

Proposition 6.32. *Given some ∞-derivation D of a sequent $\Gamma \vdash \Delta$, it is an ∞-proof according to Definition 6.30 if and only if $D : \omega^{op} \to \mathcal{D}_{\mathcal{G}}^{\mathcal{P}}$ is an ∞-proof according to the trace condition induced by $I : \mathcal{D}_{\mathcal{G}}^{op} \to \mathcal{T}_{\mathbb{B}}^{I}$ from Definition 6.11.*

Proof. Simply observe that a branch's induced path $P : \omega^{\mathrm{op}} \to \mathcal{T}_{\mathbb{B}}^{I}$ satisfies the trace condition only if the branch passes through the right side of a $\Box$-rule application infinitely often. $\Box$

Corollary 6.33. *For every cyclic derivation of $\mathcal{G}$ it is decidable whether it is a proof.*

Proof. Every cyclic derivation D is a regular ∞-derivation $D : \omega^{\mathrm{op}} \to \mathcal{D}_{\mathcal{G}}^{\mathcal{P}}$. From this, we can generate an equivalent abstract cyclic derivation D' on $\mathcal{T}_{\mathbb{B}}^{I}$ via Proposition 3.37. Now, either the algorithm from Theorem 4.4 or Theorem 4.14 can be applied to decide whether D' is an abstract cyclic proof. $\Box$

Proposition 6.34. *Every ∞-derivation of $\mathcal{G}$ is derived using a finite fragment of the derivation rules given in Definition 6.29.*

Proof. For a set of formulas Γ, we define $\textsc{Sub}(\Gamma) \coloneqq \{\psi \mid \varphi \in \Gamma, \psi \text{ subformula of } \varphi\}$. Now consider some ∞-proof $D : \omega^{\mathrm{op}} \to \mathcal{D}_{\mathcal{G}}^{\mathcal{P}}$ such that $D(0) = [\Gamma \Rightarrow \Delta]$. We first claim that all of the sequent $\Gamma' \Rightarrow \Delta'$ occurring within D we have $\Gamma' \cup \Delta' \subseteq \textsc{Sub}(\Gamma \cup \Delta)$. This is the case as every derivation rule only adds new formulas to the sequents in its premises which are subformulas of formulas already present in the premise, as is easily observed. This, in turn, means that all sequents which may ever occur in D are collected in the finite set

$$\{\Gamma' \Rightarrow \Delta' \mid \Gamma', \Delta' \in \textsc{Sub}(\Gamma \cup \Delta)\}$$

As only a finite number of derivation rules can be applied to any given sequent $\Gamma' \Rightarrow \Delta'$, there also exist only a finite number of derivation rules which can be applied within D. $\Box$

Corollary 6.35. *If $\Gamma \Rightarrow \Delta$ can be proven by an ∞-proof of $\mathcal{G}$, it can also be proven via a cyclic proof.*

Proof. We may apply Theorem 4.6 by viewing the ∞-proof $D : \omega^{\mathrm{op}} \to \mathcal{D}_{\mathcal{G}}^{\mathcal{P}}$ as $D : \omega \to \mathcal{D}_{\mathcal{G}'}^{\mathcal{P}}$ where $\mathcal{G}'$ is the category of derivations generated by the finite fragment of $\mathcal{G}$ used within D according to Proposition 6.34. $\Box$

Corollary 6.36. *It is decidable whether any given sequent of **Grz** is provable.*

Proof. Follows from Corollary 4.7 in the same manner as the result above. $\Box$

Remark. Another modal logic worth mentioning in the context of cyclic proof theory is Gödel-Löb modal logic, the logic of transitive, conversely well-founded frames. Gödel-Löb modal logic also admits a cyclic derivation system [32]. Notably, its CUT-free cyclic derivation system does not require a GTC, which is why we did not pick it as the example of this section. If one was so inclined, one could model such a trivial global trace condition in $\mathcal{T}^I_{\mathbb{B}}$ by mapping all sequents to the set $\{\star\}$ and all traces through derivations to the map $\tau := \{(\star, \top, \star)\}$, although there seems little to be gained by such an exercise.

Chapter 7

Discussion

7.1 Related Work

Cyclic proofs whose soundness was guaranteed by a global trace condition (GTC) were first put forward by Sprenger and Dam [34]. They present a cyclic derivation system for first-order fixed-point logic with explicit ordinal approximations. Their trace condition is based on infinitely decreasing sequences of ordinals and, from a modern perspective, very similar to that of cyclic arithmetic [33]. This paper also pioneered the popular GTC decision procedure based on Büchi automata inclusion we present in Theorem 4.4.

Similarly to this thesis, Santocanale and Fortier [12, 30] work at the intersection of cyclic proofs and category theory. Santocanale [30] gives a Cut-free, cyclic derivation system describing the arrows of free μ-bicomplete categories, a categorical generalization of complete lattices and thus the μ-calculus. However, the system is incomplete as there exist arrows in free μ-complete categories which are not described by any cyclic proof. This problem is alleviated by Fortier and Santocanale [12] via the addition of Cut-rule to the derivation system. The way in which category theory and cyclic proof interact in their work and in ours is perfectly opposite. They use cyclic derivations to study an aspect of category theory, namely which kinds of arrows exist in free μ-bicomplete categories. Our work, on the other hand, uses the notion of trace categories to study the properties of various cyclic derivation systems.

To the best of our knowledge, Brotherston's [4, 6] is the only previous work examining cyclic systems from an abstract point of view. While the overarching goal of his dissertation [4] is the definition and study of a cyclic derivation system for first-order logic with mutually inductive definitions à la Martin-Löf [25], he also defines a generic notion of trace condition which is equivalent to the Boolean-activated trace category $\mathcal{T}_{\mathbb{B}}$ we present in Section 5.1. He then studies various properties purely in terms of the induced notion of cyclic derivation. This line of work is continued by the Cyclist generic automatic theorem prover for cyclic derivation systems which was developed by Brotherston et. al. [6]. It is designed in such a way that it can easily be extended to support any system whose trace condition can be described in terms of $\mathcal{T}_{\mathbb{B}}$ from the perspective of our framework. There are a few points of comparison between our works:

– The point of view Brotherston takes is slightly different from ours. His abstract notion of cyclic and ∞-proof still consists of derivation trees labeled by sequents

which were derived by a rule-set and along whose branches the traces run. While we consider similarly generic ∞-proofs, we also study abstract cyclic derivations (ACDs) which are removed from concrete derivation systems.

- Some theorems are proven in generalized manners by both theses. Both Brotherston and ourselves prove the decidability of GTC satisfaction via Büchi-automata. Furthermore, we both formalize the unfolding of cyclic proofs into an equivalent representation with backward edges.
- Brotherston studies some properties which we do not consider. He introduces the notion of ordinal trace functions, assigning ordinals to semantic interpretations along a trace, which allow him to give a generic variant of the soundness proof for ∞-proofs. He also studies derivation graph homomorphisms, which can describe transformations between derivations which preserve certain paths (and thus traces). Furthermore, he describes an alternative, possibly weaker, soundness condition on cyclic proofs which is witnessed by finite objects called trace-manifolds.
- We also prove results Brotherston does not cover. Most notable are the result about regularizing ∞-derivations (Theorem 4.6) and the decision procedure justified via Ramsey's theorem (Theorem 4.14). We also consider a transformation for "compressing ACDs" in Proposition 2.17 which would not make sense in Brotherston's framework.
- Our notion of trace categories does not have an analogue in Brotherston's work. However, Brotherston's abstract notion of traces and our $\mathcal{A}$-activated traces lend themselves well to a direct comparison. As demonstrated in Theorem 5.30, any cyclic derivation system whose trace condition can be expressed in terms of some $\mathcal{A}$-activated trace category can also be formalized in terms of $\mathcal{T}_{\mathbb{B}}$ and thus Brotherston's notion of trace. Our more complicated formalism does thus not add any mathematical expressivity when compared to Brotherston's. However, trace conditions can sometimes be expressed more naturally in terms of activation algebras, as exhibited in Section 6.2, making them easier to reason about. Indeed, we point out some points of future work in Section 7.2 for which expressing the trace condition more naturally may be of help.

In a more general sense, every cyclic derivation system that has been put forward in the literature may be considered related work. Such systems have been proposed for a breadth of logics such as those dealing with inductively defined structures [4, 20, 33], fixed point logics [20, 28, 34, 35] and modal logics [10, 28, 31]. The systems presented in Chapter 6 are merely intended to be a representative sample. During our literature search, we have yet to come across a cyclic system whose trace conditions cannot be naturally expressed in terms of one of the $\mathcal{A}$-activated trace categories we define in Section 5.1, including all systems mentioned earlier in this paragraph.

7.2 Future Work

Continuing Chapter 6, one could continue looking into which cyclic derivation systems fit into the formalism of this thesis. In this direction, there are two particularly interesting questions. First, are there cyclic systems in the literature which cannot be described in terms of $\mathcal{A}$-activated trace categories? Such a system could inspire new kinds of trace

categories or changes to the formalism as a whole. For the second question, note that so far, only two activation algebras ($\mathbb{B}$ and $\mathbb{F}$) were needed to naturally formalize the trace conditions of all cyclic systems we have surveyed. This raises the question whether there exist cyclic derivation systems whose GTC is most naturally expressed in terms of a more exotic activation algebra, such as one of the binomial $\binom{n}{k}$ algebras. We conjecture that such a cyclic system might manifest for a logic whose central concern is not induction but some other principle, such as fairness conditions.

A considerable part of this thesis was concerned with demonstrating the aptitude of our framework by deriving generalized variants of well-known results about said cyclic systems. Moving beyond this, we believe that the formalism of trace categories and abstract cyclic derivations can also be used to derive new results which apply to many concrete cyclic derivation systems uniformly. We close this section by listing some problems suited to this line of inquiry.

Checking whether a cyclic derivation constitutes a proof is computationally expensive. While verifying that every derivation rule has been applied correctly only takes the usual $O(n)$, the automaton based method of verifying the GTC is at least $2^{\Omega(n \log n)}$ as it involves an emptiness check for an automaton of that size [37]. To alleviate this, it is interesting to search for soundness conditions different from the usual "every infinite path has a good trace". Jungteerapanich [17] and Stirling [36] propose one such condition for the cyclic derivation system of the modal μ-calculus we cover in Section 6.2. They present it in the form of an alternative derivation system in which the sequents and modal quantifiers occurring in sequents are annotated with sequences of the quantifiers that have been unfolded, requiring certain "progress" to be made between buds and their companions. Notably, proof checking in their system is of complexity $O(n)$. So far, their approach has not been transferred to other cyclic systems. We believe that by connecting their system with the trace condition in terms of the injective $\mathbb{F}$-activated trace category we give in Section 6.2, we should be able to capture the idea behind their system in terms of an abstract notion of trace and thus be able to extent their approach to other cyclic derivation systems. Our presentation in terms of the $\mathbb{F}$ activation algebra is uniquely well-suited for this task as the objects in the finite sets along which the traces run are precisely the quantifier variables which are also the basis of the annotations in their derivation system. The ability to cleanly separate "what is being tracked" from "how the tracked objects activate" afforded by $\mathcal{A}$-activated trace categories should thus be crucial to tackling this problem.

The literature contains quite a few alternative soundness conditions, such as those put forward by Sprenger and Dam [34, 35], Brotherston [4], Stratulat [37], Jungteerapanich [17] and Stirling [36]. An important property of such soundness conditions is their completeness. That is, does every formula (or, more generally, every sequent) provable via a cyclic proof satisfying the usual global trace condition also have a cyclic proof satisfying the alternative soundness condition? Examples of complete soundness conditions are Stirling's proofs with names for the modal μ-calculus [36] and Sprenger and Dam's tree-dischargeable proofs for the first-order μ-calculus with explicit approximations [35]. While there have been abstract accounts of alternative soundness conditions [4], their completeness proofs are always tied to specific systems, often relying on semantics properties of the logic embodied by the cyclic system. It is thus of interest to study, either based on these already existing soundness conditions or by finding a novel one, whether and to what extend the completeness of alternative soundness conditions can be proven

in an abstract framework of cyclic derivation systems, such as the one we present in this thesis.

One of the problems of cyclic derivation systems in general is that nontrivial proof transformations, such as CUT-elimination procedures, are very difficult to prove correct. In non-cyclic, finite derivation systems, it suffices to check that every derivation rule is applied correctly to guarantee that the derivation produced by such a procedure always constitutes a proof. When working with cyclic systems, it must also be verified that the transformation does not "disturb" the global trace condition. Thus proof transformations involving cyclic proofs have either been derived via semantic methods [11], forgoing an explicit transformation algorithm, or spend considerable effort verifying that the given procedure maintains the GTC [3, 12, 31]. We believe that by analyzing the proof techniques used in the latter case through the lens of our abstract framework, we may find conditions in terms of trace categories which guarantee that certain transformation steps do not disturb the GTC, thereby making it easier to derive proof transformation procedures for cyclic derivations systems. This could facilitate deriving CUT-elimination procedures for systems for which none have been found so far.

Bibliography

[1] Bahareh Afshari and Graham E. Leigh. "Finitary Proof Systems for Kozen's μ". In: *Mathematisches Forschungsinstitut Oberwolfach* (Dec. 30, 2016).

[2] Bahareh Afshari, Graham E. Leigh, and Guillermo Menéndez Turata. "Uniform interpolation from cyclic proofs: the case of modal mu-calculus". In: *TABLEAUX 2021* (Sept. 2021).

[3] David Baelde, Amina Doumane, and Alexis Saurin. "Infinitary Proof Theory: The Multiplicative Additive Case". In: *25th EACSL Annual Conference on Computer Science Logic (CSL 2016)*. June 29, 2016.

[4] James Brotherston. "Sequent Calculus Proof Systems for Inductive Definitions". PhD thesis. University of Edinburgh. College of Science and Engineering. School of Informatics., Nov. 2006.

[5] James Brotherston, Dino Distefano, and Rasmus Lerchedahl Petersen. "Automated Cyclic Entailment Proofs in Separation Logic". In: *Automated Deduction – CADE-23* (2011).

[6] James Brotherston, Nikos Gorogiannis, and Rasmus L. Petersen. "A Generic Cyclic Theorem Prover". In: *Programming Languages and Systems*. Lecture Notes in Computer Science. 2012.

[7] Giovanna D'Agostino. "Interpolation in non-classical logics". In: *Synth.* 164.3 (2008).

[8] Anupam Das. "On the Logical Complexity of Cyclic Arithmetic". Version 1. In: *Logical Methods in Computer Science* 16 (Jan. 6, 2020).

[9] Anupam Das. *A Circular Version of Gödel's T and Its Abstraction Complexity*. Jan. 16, 2021. URL: http://arxiv.org/abs/2012.14421 (visited on 05/18/2021).

[10] Anupam Das, Rajeev Goré, and Sonia Marin. "On Cut-Elimination for Non-Wellfounded Proofs: The Case of PDL". In: *Circularity in Syntax and Semantics* (Nov. 20, 2019).

[11] Anupam Das and Damien Pous. "A Cut-Free Cyclic Proof System for Kleene Algebra". In: *Automated Reasoning with Analytic Tableaux and Related Methods*. Lecture Notes in Computer Science. 2017.

[12] Jérôme Fortier and Luigi Santocanale. "Cuts for Circular Proofs: Semantics and Cut-Elimination". In: *Computer Science Logic 2013 (CSL 2013)*. Vol. 23. Leibniz International Proceedings in Informatics (LIPIcs). 2013.

[13] Erich Grädel, Wolfgang Thomas, and Thomas Wilke, eds. *Automata, Logics, and Infinite Games: A Guide to Current Research*. Lecture Notes in Computer Science. Berlin Heidelberg: Springer-Verlag, 2002.

[14] Andrzej Grzegorczyk. "Some relational systems and the associated topological spaces". In: *Journal of Symbolic Logic* 34.4 (1969).

[15] Martin Hofmann. "Syntax and Semantics of Dependent Types". In: *Extensional Constructs in Intensional Type Theory*. Ed. by Martin Hofmann. CPHC/BCS Distinguished Dissertations. London: Springer, 1997.

[16] J. M. E. Hyland. "Proof Theory in the Abstract". In: *Annals of Pure and Applied Logic*. Troelstra Festschrift 114.1 (Apr. 15, 2002).

[17] Natthapong Jungteerapanich. "A Tableau System for the Modal μ-Calculus". In: *Automated Reasoning with Analytic Tableaux and Related Methods*. Ed. by Martin Giese and Arild Waaler. Lecture Notes in Computer Science. 2009.

[18] Naoki Kobayashi, Takeshi Nishikawa, Atsushi Igarashi, and Hiroshi Unno. "Temporal Verification of Programs via First-Order Fixpoint Logic". In: *Static Analysis*. Lecture Notes in Computer Science. 2019.

[19] Naoki Kobayashi, Takeshi Tsukada, and Keiichi Watanabe. "Higher-Order Program Verification via HFL Model Checking". In: *Programming Languages and Systems*. Lecture Notes in Computer Science. Cham: Springer International Publishing, 2018.

[20] Mayuko Kori, Takeshi Tsukada, and Naoki Kobayashi. "A Cyclic Proof System for HFLN". In: *29th EACSL Annual Conference on Computer Science Logic (CSL 2021)*. Vol. 183. Leibniz International Proceedings in Informatics (LIPIcs). 2021.

[21] Dexter Kozen. "Results on the Propositional μ-Calculus". In: *Theoretical Computer Science* 27.3 (1983).

[22] F. William Lawvere. "Adjointness in Foundations". In: *Dialectica* 23.3/4 (1969).

[23] Tom Leinster. *Basic Category Theory*. Cambridge University Press, July 24, 2014.

[24] Johannes Marti and Yde Venema. *Focus-Style Proof Systems and Interpolation for the Alternation-Free mu-Calculus*. Apr. 30, 2021. URL: `http://arxiv.org/abs/2103.01671` (visited on 07/14/2021).

[25] Per Martin-Löf. "Hauptsatz for the Intuitionistic Theory of Iterated Inductive Definitions". In: *Studies in Logic and the Foundations of Mathematics* 63 (Jan. 1, 1971).

[26] Robert McNaughton. "Testing and Generating Infinite Sequences by a Finite Automaton". In: *Information and Control* 9.5 (Oct. 1, 1966).

[27] Koji Nakazawa, Makoto Tatsuta, Daisuke Kimura, and Mitsuru Yamamura. "Cyclic Theorem Prover for Separation Logic by Magic Wand". In: *ADSL 18* (2018).

[28] Damian Niwiński and Igor Walukiewicz. "Games for the μ-Calculus". In: *Theoretical Computer Science* 163.1-2 (Aug. 30, 1996).

[29] F. P. Ramsey. "On a Problem of Formal Logic". In: *Proceedings of the London Mathematical Society* 1 (1930).

[30] Luigi Santocanale. "A Calculus of Circular Proofs and Its Categorical Semantics". In: *Foundations of Software Science and Computation Structures*. Lecture Notes in Computer Science. 2002.

[31] Yury Savateev and Daniyar Shamkanov. "Non-Well-Founded Proofs for the Grzegorczyk Modal Logic". In: *The Review of Symbolic Logic* 14.1 (Mar. 2021).

[32] Daniyar Shamkanov. "Circular Proofs for the Gödel-Löb Provability Logic". In: *Mathematical Notes* 96.3 (Sept. 1, 2014).

[33] Alex Simpson. "Cyclic Arithmetic Is Equivalent to Peano Arithmetic". In: *Foundations of Software Science and Computation Structures*. Lecture Notes in Computer Science. 2017.

[34] Christoph Sprenger and Mads Dam. "On Global Induction Mechanisms in a μ-Calculus with Explicit Approximations". In: *RAIRO - Theoretical Informatics and Applications* 37.4 (4 Oct. 1, 2003).

[35] Christoph Sprenger and Mads Dam. "On the Structure of Inductive Reasoning: Circular and Tree-Shaped Proofs in the μ-Calculus". In: *Foundations of Software Science and Computation Structures*. Lecture Notes in Computer Science. 2003.

[36] Colin Stirling. "A Proof System with Names for Modal Mu-Calculus". In: *Electronic Proceedings in Theoretical Computer Science* 129 (Sept. 19, 2013).

[37] Sorin Stratulat. "Cyclic Proofs with Ordering Constraints". In: *Automated Reasoning with Analytic Tableaux and Related Methods*. Lecture Notes in Computer Science. 2017.

[38] Gadi Tellez and James Brotherston. "Automatically Verifying Temporal Properties of Pointer Programs with Cyclic Proof'. In: *Automated Deduction – CADE 26*. Lecture Notes in Computer Science. 2017.

[39] Takeshi Tsukada. "On Computability of Logical Approaches to Branching-Time Property Verification of Programs". In: *Proceedings of the 35th Annual ACM/IEEE Symposium on Logic in Computer Science*. LICS '20: 35th Annual ACM/IEEE Symposium on Logic in Computer Science. July 8, 2020.

[40] Mahesh Viswanathan and Ramesh Viswanathan. "A Higher Order Modal Fixed Point Logic". In: *CONCUR 2004 - Concurrency Theory*. Lecture Notes in Computer Science. 2004.

Appendix A

Purely Categorical Matters

We begin by defining the categorical concepts employed throughout this thesis which we consider somewhat uncommon.

Definition A.1. A *semi-category* $\mathcal{C}$ consists of

(i) A collection of *objects* $\mathrm{Ob}(\mathcal{C})$;

(ii) For each pair $X, Y \in \mathrm{Ob}(\mathcal{C})$ a collection $\textsc{Hom}_{\mathcal{C}}(X, Y)$ of *morphisms*;

(iii) A *composition operation* which, given $f \in \textsc{Hom}_{\mathcal{C}}(X, Y), g \in \textsc{Hom}_{\mathcal{C}}(Y, Z)$, yields a $g \circ f \in \textsc{Hom}_{\mathcal{C}}(X, Z)$.

Furthermore, composition operation is required to be associative. In other words, a semi-category is a category which may lack identity morphisms.

Definition A.2. Given two semi-categories $\mathcal{C}, \mathcal{D}$, a *semi-functor* $F : \mathcal{C} \to \mathcal{D}$ consists of

(i) A *mapping on objects* $F : \mathrm{Ob}(\mathcal{C}) \to \mathrm{Ob}(\mathcal{D})$;

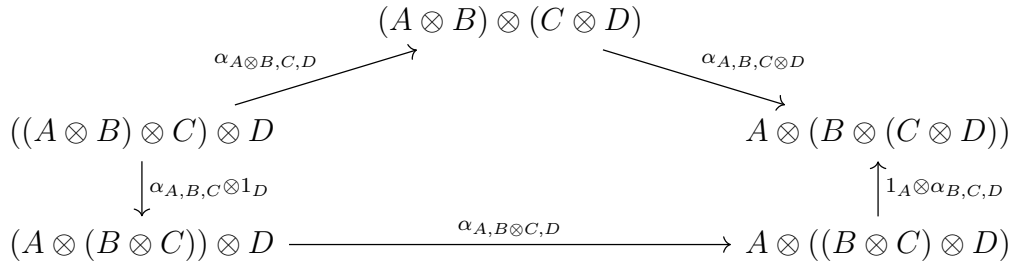
(ii) A *mapping on morphisms* $F : \textsc{Hom}_{\mathcal{C}}(X, Y) \to \textsc{Hom}_{\mathcal{D}}(F(X), F(Y))$ for each $X, Y \in \mathrm{Ob}(\mathcal{C})$

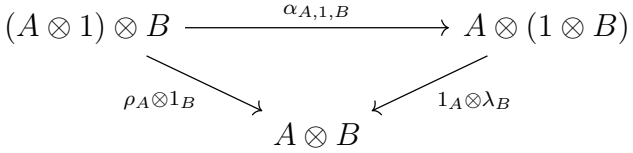
Furthermore, we require $F(g) \circ F(f) = F(g \circ f)$ for every $f : X \to Y, g : Y \to Z$.

Definition A.3. A *monoidial category* is a category $\mathcal{C}$ together with a binary functor $- \otimes - : \mathcal{C} \times \mathcal{C} \to \mathcal{C}$, a unit object $1 \in \mathrm{Ob}(\mathcal{C})$ and the following natural isomorphisms

- *associativity:* $\alpha : (A \otimes B) \otimes C \simeq A \otimes (B \otimes C)$ natural in $A, B, C \in \mathrm{Ob}(\mathcal{C})$
- *left unit:* $\lambda : (1 \otimes A) \simeq A$ natural in $A \in \mathrm{Ob}(\mathcal{C})$
- *right unit:* $\rho : (A \otimes 1) \simeq A$ natural in $A \in \mathrm{Ob}(\mathcal{C})$

such that the following diagrams commute for all $A, B, C, D \in \mathrm{Ob}(\mathcal{C})$:





Definition A.4. Let $(\mathcal{C}, \otimes_{\mathcal{C}}, 1_{\mathcal{C}})$ and $(\mathcal{D}, \otimes_{\mathcal{D}}, 1_{\mathcal{D}})$ be monoidal categories. A functor $F : \mathcal{C} \to \mathcal{D}$ is *strongly monoidal* if the following isomorphisms exist

tensor: $\eta : F(A \otimes_{\mathcal{C}} B) \simeq F(A) \otimes_{\mathcal{D}} F(B)$ natural in $A, B \in \mathrm{Ob}(\mathcal{C})$

unit: $\epsilon : F(1_{\mathcal{C}}) \simeq 1_{\mathcal{D}}$

and the following diagrams commute for all $A, B, C, D \in \mathrm{Ob}(\mathcal{C})$

$$\begin{array}{ccc}
(F(A) \otimes_{\mathcal{D}} F(B)) \otimes_{\mathcal{D}} F(C) & \xrightarrow{\alpha^{\mathcal{D}}_{F(A),F(B),F(C)}} & F(A) \otimes_{\mathcal{D}} (F(B) \otimes_{\mathcal{D}} F(C)) \\
\downarrow{\scriptstyle \eta^{-1}_{A,B} \otimes 1_{F(C)}} & & \downarrow{\scriptstyle 1_{F(A)} \otimes \eta^{-1}_{B,C}} \\
F(A \otimes_{\mathcal{C}} B) \otimes_{\mathcal{D}} F(C) & & F(A) \otimes_{\mathcal{D}} F(B \otimes_{\mathcal{C}} C) \\
\downarrow{\scriptstyle \eta^{-1}_{A \otimes_{\mathcal{C}} B, C}} & & \downarrow{\scriptstyle \eta^{-1}_{A, B \otimes_{\mathcal{C}} C}} \\
F((A \otimes_{\mathcal{C}} B) \otimes_{\mathcal{C}} C) & \xrightarrow{F(\alpha^{\mathcal{C}}_{A,B,C})} & F(A \otimes_{\mathcal{C}} (B \otimes_{\mathcal{C}} C))
\end{array}$$

$$\begin{array}{ccc}
1_{\mathcal{D}} \otimes_{\mathcal{D}} F(A) & \xrightarrow{\epsilon^{-1} \otimes_{\mathcal{D}} 1_{F(A)}} & F(1_{\mathcal{C}}) \otimes_{\mathcal{D}} F(A) \\
\downarrow{\scriptstyle \lambda^{\mathcal{D}}_A} & & \downarrow{\scriptstyle \eta^{-1}_{1_{\mathcal{C}}, A}} \\
F(A) & \xleftarrow[F(\lambda^{\mathcal{C}}_A)]{} & F(1_{\mathcal{C}} \otimes A)
\end{array}
\qquad
\begin{array}{ccc}
F(A) \otimes_{\mathcal{D}} 1_{\mathcal{D}} & \xrightarrow{1_{F(A)} \otimes \epsilon^{-1}} & F(A) \otimes_{\mathcal{D}} F(1_{\mathcal{C}}) \\
\downarrow{\scriptstyle \rho^{\mathcal{D}}_A} & & \downarrow{\scriptstyle \eta^{-1}_{A, 1_{\mathcal{C}}}} \\
F(A) & \xleftarrow[F(\rho^{\mathcal{C}}_A)]{} & F(A \otimes_{\mathcal{C}} 1_{\mathcal{C}})
\end{array}$$

We now proceed by proving Lemma A.8, which is required by the proof of Lemma 3.30. Towards this, we first prove a few lemmas yielding various useful properties of the natural isomorphisms $\eta_{A_0,\ldots,A_n} : I(A_0 \otimes \ldots \otimes A_n) \to I(A_0) \times \ldots \times I(A_n)$.

Proposition A.5. *Pick $A_0, \ldots, A_n, A_{n+1}, \ldots, A_{n+m} \in Ob(\mathcal{D}_{\mathcal{R}})$ then*

$$\eta^{n+m} = \sigma \circ (1^n \times \eta^m) \circ \eta^n$$

where $\sigma : I(A_0) \times \ldots I(A_n) \times (I(A_{n+1}) \times \ldots \times I(A_{n+m})) \to I(A_0) \times \ldots I(A_{n+m})$ and further $\eta^{n+m} := \eta_{A_0,\ldots,A_{n+m}}$ and $\eta^m := \eta_{A_{n+1},\ldots,A_m}$

Proof. Proof per induction on m. For $m = 1$ the claim holds per definition. Thus let $m > 1$, then we have

$$\begin{aligned}
&\eta^{n+m+1} \\
&\coloneqq \sigma_1 \circ (1^{n+m-1} \times \eta_{A_m, A_{m+1}}) \circ \eta^{n+m} \\
&= \sigma_1 \circ (1^{n+m-1} \times \eta_{A_m, A_{m+1}}) \circ \sigma_2 \circ (1^n \times \eta^m) \circ \eta^n && \text{IH} \\
&= \sigma_1 \circ \sigma_3 \circ (1^n \times (1^m \times \eta_{A_m, A_{m+1}})) \circ (1^n \times \eta^m) \circ \eta^n && \text{naturality of } \sigma_2 \\
&= \sigma_4 \circ (1^n \times ((1^m \times \eta_{A_m, A_{m+1}}) \circ \eta^m)) \circ \eta^n && (*) \\
&= \sigma_5 \circ (1^n \times \sigma_6) \circ (1^n \times ((1^m \times \eta_{A_m, A_{m+1}}) \circ \eta^m)) \circ \eta^n && (*) \\
&= \sigma_5 \circ (1^n \times (\sigma_6 (1^m \times \eta_{A_m, A_{m+1}}) \circ \eta^m)) \circ \eta^n \\
&\coloneqq \sigma_5 \circ (1^n \times \eta^{m+1}) \circ \eta^n
\end{aligned}$$

where σ_i denote the following instances of the natural isomorphisms between product functors

$$\begin{aligned}
&\sigma_1 : A^{n+m-1} \times (I(A_m) \times I(A_{m+1})) \to A^{n+m+1} \\
&\sigma_2 : A^n \times (A^{m-1} \times I(A_m \otimes A_{m+1})) \to A^{n+m-1} \times I(A_m \otimes A_{m+1}) \\
&\sigma_3 : A^n \times (A^{m-1} \times (I(A_m) \times I(A_{m+1}))) \to A^{n+m-1} \times (I(A_m) \times I(A_{m+1})) \\
&\sigma_4 : A^n \times (A^{m-1} \times (I(A_m) \times I(A_{m+1}))) \to A^{n+m+1} \\
&\sigma_5 : A^n \times (A^{m+1}) \to A^{n+m+1} \\
&\sigma_5 : A^{m-1} \times (I(A_m) \times I(A_{m+1}) \to A^{m+1}
\end{aligned}$$

with $A^n \coloneqq I(A_0) \times \ldots \times I(A_n)$ and so forth. The steps marked with $(*)$ follow by the uniqueness of said isomorphisms which in turn follow from the fact that they are induced by the universal properties of the various products involved. □

Proposition A.6. *Pick $A_0, \ldots, A_n, B, C \in Ob(\mathcal{D}_\mathcal{R})$ then*

$$\eta_{A_0,\ldots,A_n,B\otimes C} \circ \eta^{-1}_{A^n_\otimes \otimes B,C} = (1^n \times \eta^{-1}_{B,C}) \circ \sigma \circ (\eta_{A_0,\ldots,A_n,B} \times 1_C)$$

where $A^n_\otimes \coloneqq A_0 \otimes \ldots \otimes A_n$ and $\sigma : (A^n \times I(B)) \times I(C) \simeq A^n \times (I(B) \times I(C))$.

Proof. Proof per induction on n.

$n = 0$: This is precisely the commuting diagram for the associativity induced by the fact that $I : \mathcal{D}^{\text{op}} \to \mathcal{T}$ is a strong monoidal functor.

$n > 0$: Then

$$\begin{aligned}
&\eta_{A^{n+1},B\otimes C} \circ \eta^{-1}_{A^{n+1}_\otimes \otimes B,C} \\
&= \sigma_1 \circ (1^n \times \eta_{A_{n+1},B\otimes C}) \circ \eta_{A^n, A_{n+1}\otimes B\otimes C} \circ \eta^{-1}_{A^n_\otimes \otimes (A_{n+1}\otimes B)} && (1) \\
&= \sigma_1 \circ (1^n \times \eta_{A_{n+1},B\otimes C}) \circ (1^n \times \eta^{-1}_{A_{n+1}\otimes B,C}) \circ \sigma_2 \circ (\eta_{A^n, A_{n+1}\otimes B} \times 1_C) && \text{IH} \\
&= \sigma_1 \circ (1^n \times (\eta_{A_{n+1},B\otimes C} \circ \eta^{-1}_{A_{n+1}\otimes B,C}) \circ \sigma_2 \circ (\eta_{A^n, A_{n+1}\otimes B} \times 1_C) \\
&= \sigma_1 \circ (1^n \times ((1_{A_{n+1}} \times \eta^{-1}_{B,C}) \circ \sigma_3 \circ (\eta_{A_{n+1},B} \times 1_C)) \circ \sigma_2 \circ (\eta_{A^n, A_{n+1}\otimes B} \times 1_C) && (2) \\
&= (1^{n+1} \times \eta^{-1}_{B,C}) \circ \sigma_4 \circ (1^n \times (\sigma_3 \circ (\eta_{A_{n+1},B} \times 1_C)) \circ \sigma_2 \circ (\eta_{A^n, A_{n+1}\otimes B} \times 1_C) && (3)
\end{aligned}$$

$$
\begin{aligned}
&= (1^{n+1} \times \eta^{-1}_{B,C}) \circ \sigma_4 \circ (1^n \times \sigma_3) \circ \sigma_5 \circ ((1^n \times \eta_{A_{n+1},B}) \times 1_C) \circ (\eta_{A^n, A_{n+1}\otimes B} \times 1_C) && (4)\\
&= (1^{n+1} \times \eta^{-1}_{B,C}) \circ \sigma_6 \circ (((1^n \times \eta_{A_{n+1},B}) \circ \eta_{A^n, A_{n+1}\otimes B}) \times 1_C) && (5)\\
&= (1^{n+1} \times \eta^{-1}_{B,C}) \circ \sigma_7 \circ (\sigma_8 \times 1_C) \circ (((1^n \times \eta_{A_{n+1},B}) \circ \eta_{A^n, A_{n+1}\otimes B}) \times 1_C) && (5)\\
&= (1^{n+1} \times \eta^{-1}_{B,C}) \circ \sigma_7 \circ ((\sigma_8 \circ (1^n \times \eta_{A_{n+1},B}) \circ \eta_{A^n, A_{n+1}\otimes B}) \times 1_C)\\
&\coloneqq (1^{n+1} \times \eta^{-1}_{B,C}) \circ \sigma_7 \circ (\eta_{A^{n+1},B} \times 1_C)
\end{aligned}
$$

where (1) follows by Proposition A.5, (2) by the strong monoidality of I, (3) and (4) by the naturality of σ_1 and σ_2, respectively, and (5) again by the uniqueness of product natural isomorphisms. These isomorphisms are

$$
\begin{aligned}
&\sigma_1 : A^n \times (I(A_{n+1}) \times I(B \otimes C)) \simeq A^{n+1} \times I(B \otimes C)\\
&\sigma_2 : (A^n \times (I(A_{n+1} \otimes B))) \times I(C) \simeq A^n \times (I(A \otimes B) \times I(C))\\
&\sigma_3 : (A^{n+1} \times I(B)) \times I(C) \times A^{n+1} \times (I(B) \times I(C))\\
&\sigma_4 : A^n \times (I(A_{n+1}) \times (I(B) \times I(C))) \simeq A^{n+1} \times (I(B) \times I(C))\\
&\sigma_5 : (A^n \times (I(A_{n+1}) \times I(B))) \times I(C) \simeq A^n \times ((I(A) \times I(B)) \times I(C))\\
&\sigma_6 : (A^n \times (I(A_{n+1}) \times I(B))) \times I(C) \simeq A^{n+1} \times (I(B) \times I(C))\\
&\sigma_7 : (A^{n+1} \times I(B)) \times I(C) \simeq A^{n+1} \times (I(B) \times I(C))\\
&\sigma_8 : A^n \times (I(A_{n+1}) \times I(B)) \simeq A^{n+1} \times I(B)
\end{aligned}
$$

□

Corollary A.7. *Pick $A_0, \ldots, A_n, B \in Ob(\mathcal{D}_{\mathcal{R}})$ then*

$$\eta_{A^n,B} \circ \eta^{-1}_{A^n_{\otimes},B} = \sigma \circ (\eta_{A^n} \times 1_B)$$

where $\sigma : (A^n) \times I(B) \simeq A^n \times I(B)$.

Proof. Observe that

$$
\begin{aligned}
&\eta_{A^n,B} \circ \eta^{-1}_{A^n_{\otimes},B}\\
&= \sigma_1 \circ (1^{n-1} \times \eta_{A_n,B}) \circ \eta_{A^{n-1},A_n\otimes B} \circ \eta^{-1}_{A^{n-1}_{\otimes}\otimes A_n,B} && \text{Proposition A.5}\\
&= \sigma_1 \circ (1^{n-1} \times \eta_{A_n,B}) \circ (1^{n-1} \times \eta^{-1}_{A_n,B}) \times \sigma_2 \circ (\eta_{A^n} \times 1_B) && \text{Proposition A.6}\\
&= \sigma \circ (\eta_{A^n} \times 1_B)
\end{aligned}
$$

where

$$
\begin{aligned}
&\sigma_1 : A^{n-1} \times (I(A_n) \times I(B)) \simeq A^n \times I(B)\\
&\sigma_2 : A^n \times I(B) \simeq A^{n-1} \times (I(A_n) \times I(B))
\end{aligned}
$$

□

Lemma A.8. *Choose $A_0, \ldots, A_{i_0}, A_{i_0+1}, \ldots, A_{i_1}, \ldots, A_{i_n} \in Ob(\mathcal{D}_{\mathcal{R}})$ then*

$$\eta_{A^0,\ldots,A^n} \circ \eta^{-1}_{A^0_{\otimes},\ldots,A^n_{\otimes}} = \sigma \circ (\eta_{A^0} \times \ldots \times \eta_{A^n})$$

where $A^j \coloneqq A_{i_{j-1}+1}, \ldots, A_{i_j}$ and $A^j_{\otimes} \coloneqq A_{i_{j-1}+1} \otimes \ldots \otimes A_{i_j}$ and $\sigma : A^0 \times \ldots A^n \simeq A_0 \times \ldots \times A_{i_n}$.

Proof.

$n = 1$: Then

$$\begin{aligned}
&\eta_{A^0,A^1} \circ \eta^{-1}_{A^0_\otimes,A^1_\otimes} \\
&= \sigma_1 \circ (1^0 \times \eta_{A^1}) \circ \eta_{A^0,A^1_\otimes} \circ \eta^{-1}_{A^0_\otimes,A^1_\otimes} && \text{Proposition A.5} \\
&= \sigma_1 \circ (1^0 \times \eta_{A^1}) \circ \sigma_2 \circ (\eta_{A^0} \times 1_{A^1_\otimes}) && \text{Corollary A.7} \\
&= \sigma_1 \circ \sigma_3 \circ (1_{A^0} \times \eta_{A^1}) \circ (\eta_{A^0} \times 1_{A^1_\otimes}) && \text{naturality of } \sigma_2 \\
&= \sigma \circ (\eta_{A^0} \times \eta_{A^1})
\end{aligned}$$

where

$$\begin{aligned}
\sigma_1 &: A^0 \times (A^1) \simeq A^0 \times A^1 \\
\sigma_2 &: (A^0) \times A^1_\otimes \simeq A^0 \times A^1_\otimes \\
\sigma_3 &: (A^0) \times (A^1) \simeq A^0 \times (A^1)
\end{aligned}$$

$n > 1$: Then

$$\begin{aligned}
&\eta_{A^0,\ldots,A^{n+1}} \circ \eta^{-1}_{A^0_\otimes,\ldots,A^{n+1}_\otimes} \\
&= \sigma_1 \circ (1^n \times \eta_{A^{n+1}}) \circ \eta_{A^0,\ldots,A^n,A^{n+1}_\otimes} \circ \eta^{-1}_{A^0_\otimes,\ldots,A^n_\otimes \otimes A^{n+1}_\otimes} \circ (1^n_\otimes \times \eta^{-1}_{A^n_\otimes,A^{n+1}_\otimes}) \circ \sigma_2 && (1) \\
&= \sigma_1 \circ (1^n \times \eta_{A^{n+1}}) \circ \sigma_3 \circ (\eta_{A^0} \times \ldots \times \eta_{A^n,A^{n+1}_\otimes}) \circ (1^{n-1}_\otimes \times \eta^{-1}_{A^n_\otimes,A^{n+1}_\otimes}) \circ \sigma_2 && (2) \\
&= \sigma_1 \circ (1^n \times \eta_{A^{n+1}}) \circ \sigma_3 \circ (\eta_{A^0} \times \ldots \times (\eta_{A^n,A^{n+1}_\otimes} \circ \eta^{-1}_{A^n_\otimes,A^{n+1}_\otimes})) \circ \sigma_2 \\
&= \sigma_1 \circ (1^n \times \eta_{A^{n+1}}) \circ \sigma_3 \circ (\eta_{A^0} \times \ldots \times (\sigma_4 \circ (\eta_{A^n} \times 1_{A^{n+1}_\otimes}))) \circ \sigma_2 && (3) \\
&= \sigma_1 \circ \sigma_5 \circ (1_{A^1} \times \ldots \times (1^{A^n} \times \eta_{A^{n+1}})) \circ (\eta_{A^0} \times \ldots \times (\sigma_4 \circ (\eta_{A^n} \times 1_{A^{n+1}_\otimes}))) \circ \sigma_2 && (4) \\
&= \sigma_1 \circ \sigma_5 \circ (\eta_{A^0} \times \ldots \times ((1^{A^n} \times \eta_{A^{n+1}}) \circ \sigma_4 \circ (\eta_{A^n} \times 1_{A^{n+1}_\otimes}))) \circ \sigma_2 \\
&= \sigma_1 \circ \sigma_5 \circ (\eta_{A^0} \times \ldots \times (\sigma_6 \circ (1_{A^n} \times \eta_{A^{n+1}}) \circ (\eta_{A^n} \times 1_{A^{n+1}_\otimes}))) \circ \sigma_2 && (5) \\
&= \sigma_1 \circ \sigma_5 \circ (\eta_{A^0} \times \ldots \times (\sigma_6 \circ (\eta_{A^n} \times \eta_{A^{n+1}}))) \circ \sigma_2 \\
&= \sigma_1 \circ \sigma_5 \circ (1_{A^0} \times \ldots \times 1_{A^{n-1}} \times \sigma_6) \circ (\eta_{A^0} \times \ldots \times (\eta_{A^n} \times \eta_{A^{n+1}})) \circ \sigma_2 \\
&= \sigma_1 \circ \sigma_5 \circ (1_{A^0} \times \ldots \times 1_{A^{n-1}} \times \sigma_6) \circ \sigma_7 \circ (\eta_{A^0} \times \ldots \times \eta_{A^n} \times \eta_{A^{n+1}}) && (6) \\
&= \sigma \circ (\eta_{A^0} \times \ldots \times \eta_{A^n} \times \eta_{A^{n+1}}) && (7)
\end{aligned}$$

where (1) follows from Proposition A.5 and the inverse of the definition from Proposition 3.26, (2) by the inductive hypothesis, in which we take $A^n \coloneqq (A^n, A^{n+1})$, (3) follows by Corollary A.7, (4), (5) and (6) by the naturality of σ_3, σ_4 and σ_2, respectively, and (7) once again by the uniqueness of the product isomorphisms. Furthermore, we have

$$\begin{aligned}
\sigma_1 &: A^n \times \ldots \times A^n \times (A^{n+1}) \simeq A^0 \times \ldots \times A^{n+1} \\
\sigma_2 &: A^0_\otimes \times \ldots \times A^{n+1}_\otimes \simeq A^0_\otimes \times \ldots \times (A^n_\otimes \times A^{n+1}_\otimes) \\
\sigma_3 &: (A^0) \times \ldots \times (A^{n-1}) \times (A^n \times A^{n+1}_\otimes) \simeq A^0_\otimes \times \ldots \times A^n \times A^n \times A^{n+1}_\otimes \\
\sigma_4 &: (A^n) \times A^{n+1}_\otimes \simeq A^n \times A^{n+1}_\otimes \\
\sigma_5 &: (A^0) \times \ldots \times (A^{n-1}) \times (A^n \times (A^{n+1})) \simeq A^0_\otimes \times \ldots \times A^n \times A^n \times (A^{n+1}) \\
\sigma_6 &: (A^n) \times (A^{n+1}) \simeq A^n \times (A^{n+1})
\end{aligned}$$

□