Establishing a Connection Between Graph Structure, Logic, and Language Theory

by

Alexis Hunt

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Master of Mathematics
in
Combinatorics and Optimization

Waterloo, Ontario, Canada, 2015

Author's Declaration

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Abstract

The field of graph structure theory was given life by the Graph Minors Project of Robertson and Seymour, which developed many tools for understanding the way graphs relate to each other and culminated in the proof of the Graph Minors Theorem. One area of ongoing research in the field is attempting to strengthen the Graph Minors Theorem to sets of graphs, and sets of sets of graphs, and so on.

At the same time, there is growing interest in the applications of logic and formal languages to graph theory, and a significant amount of work in this field has recently been consolidated in the publication of a book by Courcelle and Engelfriet.

We investigate the potential applications of logic and formal languages to the field of graph structure theory, suggesting a new area of research which may provide fruitful.

Acknowledgements

To properly thank everyone who got me to where I am today, I would have to embark on another project as long as this thesis. The collective effort of everyone who helped me be ready to write it cannot be understated. At the same time, there is one person who really helped me go from ready to write a thesis to having written one: my supervisor. Thank you, Bruce, for getting me through this.

Table of Contents

List of Figures

List of Symbols

Notation	Description	Page List
$\mathrm{cl}_{\downarrow}(X)$	Downwards closure of X	4
$\downarrow(\Omega)$	Downwards-closed subsets of Ω	4
$\mathrm{cl}_{\uparrow}(X)$	Upwards closure of X	4
$\uparrow(\Omega)$	Set of upwards-closed subsets of Ω	4
$\mathrm{br}(v)$	Branch rooted at v	8
$\mathrm{chldrn}(v)$	Children of v	8
$\triangle G$	Cone of G	45
$\curlywedge G$	Apex of $\triangle G$	45
$\mathcal{G}_{\triangle}$	Cone graphs	46
$G \setminus E$	Deletion of a set of edges	8
$G \setminus e$	Deletion of an edge	8
$G \,/\, E$	Contraction of a set of edges	8
$G \,/\, e$	Contraction of an edge	8
$F_{\cup}$	Powerset signature of F	14
$\mathrm{fg}_a(G)$	Graph obtained from G by forgetting the a-source, if any	19
$\mathrm{forb}(F)$	Ideal with obstruction set F	4
$\mathcal{G}$	Collection of all graphs	8
$\mathfrak{S}(G)$	Set of all subgraphs of G	8
$\mathcal{G}_{\mathrm{conn}}$	Collection of connected graphs in $\mathcal{G}$	45
$\iota(G)$	Set of trees ideal-generated by G	23
$\mathcal{JS}$	Set of all finite graphs with sources in $\mathbb{N}$	19
$\mathbb{JS}_{\mathrm{t}}$	Sorted algebra of graphs with sources	38
$\mathrm{L}(S)$	Least solution to S	15
$\mathscr{L}(v)$	Vertex label of v	10
$\mathcal{LRT}(Q)$	Set of rooted trees fully labeled from Q	26
$\mathrm{MOD}(\phi)$	Set of all models of ϕ	30
$\mathbb{M}/\!\sim$	Quotient algebra of M	17
MS_2	Monadic second-order logic of all $\mathcal{R}^{\mathcal{G}}$ formulas	31
$\mathrm{obs}(C)$	Obstruction set of C	4
$\mathrm{ow}(\mathcal{I})$	Obstruction-width of $\mathcal{I}$	48
$\mathcal{P}(\mathbb{M})$	Powerset algebra of $\mathbb{M}$	14
$\mathcal{P}^{\alpha}(S)$	Generalized powerset of S	5
$\mathrm{r}(T)$	Root vertex of T	8

Notation	Description	Page List
$\mathrm{ren}_{a\leftrightarrow b}(G)$	Graph obtained from G by renaming the a- and b-sources	19
$\mathcal{R}^{\mathcal{G}}$	Relational signature of graphs	31
$\mathbb{STR}$	Algebra of relational signatures	37
$\mathrm{STR}(\mathcal{R})$	Collection of all $\mathcal{R}$-structures	29
$\mathbb{STR}_{\mathrm{pres}}$	Quasi-domain preserving subalgebra of $\mathbb{STR}$	37
$\mathrm{T}(F)$	Set of all terms over F	14
$\tau(G)$	Set of trees generated by G	23
$\mathcal{TG}(\Omega)$	Set of Ω-tree-generators	22
$\mathfrak{TG}(G)$	Set of sub-tgs of G	22
$\mathfrak{TG}_\alpha(G)$	Set of local sub-tgs of G at α	22
$\mathrm{tw}(G)$	Tree-width of G	11
$\mathrm{val}_{\mathbb{M}}(t)$	Value of t in $\mathbb{M}$	13
$\otimes$	Tree-generator self-reference	22
$\otimes^n$	Tree-generator recursive reference	43
$H \preceq_{\mathrm{I}} G$	Componentwise minor relation	46
$H \preceq_{\mathrm{L}} G$	label-respecting minor relation	10
$H \preceq_{\mathrm{Lr}} G$	label-respecting rooted topological minor relation	10
$H \preceq_{\mathrm{Lt}} G$	label-respecting topological minor relation	10
$H \preceq_{\mathrm{m}} G$	Minor relation	8
$G_1 /\!\!/ G_2$	Parallel composition of G_1 and G_2	19
$H \preceq_{\mathrm{r}} G$	Rooted topological minor relation	9
$H \preceq_{\mathrm{t}} G$	Topological minor relation	9
$S \sim^h_{\mathrm{L}} T$	Logical congruence of height h	37
$X \preceq^\alpha Y$	Powerset ordering	5
$G_1 \preceq_\tau G_2$	Tree-generator generated set relation	24

1 Introduction

1.1 Motivation

The field of graph structre theory has grown considerably as a result of the Graph Minors Project of Robertson and Seymour. Robertson and Seymour explored the structural properties and relationships of graphs, and the fruits of their research have led to advances in other fields, one of which is the study of fixed-parameter tractable algorithms in complexity theory.

Possibly the most famous result to come out of the project was the proof of Wagner's Conjecture, now called the Robertson-Seymour Theorem or simply the Graph Minor Theorem. The theorem states that, under the minor relation, the collection of finite graphs is well-quasi-ordered [17]. That graphs are a well-quasi-order implies a number of distinct yet equivalent properties, but perhaps the most striking and well-known of these is that any minor-closed family of graphs is characterized by finitely minimal excluded graphs.

Many interesting consequences follow from the Graph Minor Theorem. It is independently known that, given a fixed graph G, there exists an $O(n^2)$-time algorithm to test whether a graph H with n vertices contains G as a minor [5]. It follows that for any minor-closed family of graphs, there is an $O(n^2)$-time algorithm to test membership in that class.

Explicitly writing such an algorithm requires, however, a characterization of the excluded minors, and it is not always easy to find the excluded minors of a class of graphs. One well-studied minor-closed class of graphs, called the $Y\Delta Y$-reducible graphs, are known to have over 68 billion excluded minors and it is not known whether or not there are more [20]. There is no general way to find the set of excluded minors of a class of graphs [3].

The Graph Minor Theorem provides deep insights into the structure of graphs, but it also naturally raises further questions about the ordering properties of graphs. One can consider the ideals in the powerset of the set of graphs—the minor-closed families—and order them by the inclusion relation. If these form a well-quasi-order, then the process can be repeated with second-order ideals, and so on *ad nauseam*. (Strictly speaking, this is not correct: there is no set of all graphs and we must instead choose a representative of each isomorphism class to get a set. The additional pedanticism bears little benefit, however.)

The Graph Minor Theorem is built upon a result of Kruskal which shows that the trees are well-quasi-ordered under the topological minor relation (which is sometimes called homeomorphic embedding and which coincides with the minor relation for graphs with maximum degree at most 3) [7]. Nash-Williams showed that they are in fact better-quasi-ordered, in order to extend the result to infinite trees [12]. The concept of better-quasi-ordering unfortunately does not admit nearly as simple a description as that of well-quasi-ordering, but it is equivalent to, after suitable definitions for limit ordinals, the ideals of order up to the first uncountable ordinal being well-quasi-ordered [10]. Thomas later used category theory to extend the result to apply to larger classes of graphs. In particular, for any k, the class of graphs with tree-width at most k is bqo [19].

Bqo is a more desirable property than wqo, because it is much more easily preserved by various operations, and it is easier to perform inductions as a result. For instance, a simple counterexample due to Rado shows that there exist instances of wqos whose ideals are not wqo, but the same cannot happen for a bqo. In the context of graph minors, this leads us to an important problem: are the (finite) graphs better-quasi-ordered by the minor relation? If not, where in the taking of ideals does this fail?

The purpose of this work is to study the structure of graphs using techniques from language theory and logic. The intersection of these fields and graph theory is a growing area of study. Engelfriet and Courcelle recently published a book [2] outlining many major results from their work in the area, and it is this book which will serve as the basis for a lot of the present work. A large part of the work in graph grammars consists of defining context-free graph equation systems (essentially a form of grammar, although perhaps not as a language theorist might see it) and characterizing the sets defined by them. Of particular interest are the HR equation systems, which are capable of defining any minor-closed class of graphs of bounded tree-width.

The interest in this area comes from unpublished work of Christian, Richter, and Salazar. They

proved a novel well-quasi-ordering result on the ideals of labeled trees by recursive structures called tree-generators. It will be shown that a tree-generator is in fact a form of HR equation system that generates a particular ideal, and this observation forms the main motivation of this work. The interest in such a generalization arises from the fact that any minor-closed class of graphs with bounded tree-width can be generated by an HR equation system. This may be an avenue to find an alternate proof that the ideals of minor-closed classes of bounded tree-width are well-quasi ordered.

Unforunately, as we will see, this is not so easy, as there are HR equation systems for which this fails. Thus the present objective will be to generalize the result to some particular classes of HR grammars, and determining how such generalizations apply in particular to minor-closed families of graphs. In reality the bulk of this work is devoted to exposition, to tie the two branches of graph theory together, and only baby steps are taken towards this objective.

1.2 Overview

We begin with exposition of the various fields and works which have led to this thesis. In Section 2, we will cover the basics of quasi-ordering and well-quasi-ordering, which form the basis for the questions we wish to answer. In Section 3, we cover our definitions for graphs and existing results in graph structure, particularly those of Robertson & Seymour.

We then move away from the foundations to the work of Courcelle and Engelfriet with Sections 4 and 5, where we first present the abstract algebraic systems needed to understand the subsequent definition of the $\mathbb{JS}$ algebra on graphs, a focal point of this work.

In Section 6, we present the results of Christian et al. pertaining to tree-generators, built from a more understandable definition and with errors corrected from the original presentation. We then return to the logical world of Courcelle and Engelfriet in Section 7, where we present a small bit of the framework for doing formal logic with graphs.

Section 8 is devoted to a proof of the Recognizability Theorem, a powerful and deep theorem which links the algebras defined in Section 4 to the formal logic of Section 7.

Finally, we will explore potential new directions. In Section 9, we will explore the connections between tree-generators and the HR grammars, and in particular how the former arise as a special case of the latter. Then, in Section 10, we will investigate ways to represent graph ideals in terms of formal logic.

The present author's novel contributions to the area are found in Section 6, where defects in original proofs were repaired and the entire work recast in fashion that is easier to work with, and in Sections 9 and 10, which are original research except where otherwise noted. The remainder of this thesis is predominantly an elaboration of existing work.

1.3 A Note About Abstract Expressions

In much of modern mathematics, we make use of abstract expressions with variables without much of a second thought. The notation $f(x) = x^2 + 3$ is understood to mean that f is a function that squares its argument and adds 3 to it. But it may not be immediately clear what $x^2 + 3$ is, or whether it indeed has any independent existence as a mathematical object.

There is a school of logical thought which holds that $x^2 + 3$ is, on its own, merely a string in some language with no inherent meaning. It is purely a *syntactic* construct. Mathematically, we can treat this as an object on its own. By assigning meaning to the symbols used (x, ^{2}, $+$, and 3), we can express the *semantics* of the mathematical objects denoted by the expression.

One does not have to look far to see where the dichotomy proves useful. The definition $f(x) = x^2 + 3$ does not properly define a function because essential information is missing, most notably the domain of the function. $f(x)$ would be well-defined on any ring, for instance, and on many other objects besides. Some mathematicians would probably consider it an abuse of notation to view $f(x)$ as a function defined on the space of 1×1 matrices over the real numbers, yet they would have no difficulty interpreting 3 as actually meaning the matrix $[3]$.

By formally treating syntax and semantics separately, it becomes possible to talk about $x^2 + 3$ as an independent mathematical object, without relying on any additional context. Additional context

can be supplied as necessary and, importantly, it is possible to assign multiple different contexts as is convenient. If we let E be the expression $x^2 + 3$, then we could write $f_{\mathbb{R}}(x) = E$ as a function on the real numbers, and $f_{\mathbb{N}}(x) = E$ as a function on the natural numbers. We could even define $f_{1\times 1}(x) = E$ by officially interpreting 3 as meaning [3].

In this thesis, we will make use of this distinction in two different places. The first is with algebraic terms (Definition 4.3), and the second is with logical formulas (Definition 7.3). These correspond to expressions, which compute a value, and statements, which express a truth or a falsehood.

Unfortunately, to properly treat these syntactical objects, there is a very great deal of technical machinery that is required. First off, every syntactic expression needs an associated set of variables. The expression $x^2 + 3$ is a different object when it is in the variables $\{x\}$ from when it is in the variables $\{x, y\}$—this is analogous to how the codomain of a function is a significant part of its definition. As an example, two functions $f : \mathbb{R} \to \mathbb{R}$ and $f' : \mathbb{R} \to \mathbb{C}$ are always distinct mathematical objects, even if $f(x) = f'(x)$ for all $x \in \mathbb{R}$.

Secondly, the process of substituting one expression into another is fraught with difficulties. If we have $E_1 = x + y$, $E_2 = 3x$, and $E_3 = x^2$, what do we get by replacing x with E_2 and y with E_3 in E_1? Is it $3x + x^2$ or $3x + y^2$? In other words, are the x in E_2 and E_3 the same variable or not?

This question becomes difficult in first-order logic, where a quantification can change the meaning of a sentence. If $S_1 = x \geq 3$ and $S_2 = \forall x.\phi(x)$ is a logical statement, and we substitute S_1 for ϕ in S_2, the result is $\forall x.x \geq 3$, and now the variable x is captured in the quantification.

In this work, we shall gloss over the technical details, as we do not require their full versatility and precision. Terms and formulas will be substituted in a way that is hopefully clear from the context. It will be assumed that, unless otherwise indicated, all variables are distinct and no substitution leads to a change of meaning as in these examples. This assumption carries with it no loss of generality, as we can always rename variables in such a way that there is no repeated occurrences of the same variable in different terms and thus no change of meaning.

2 Well-Quasi-Ordering

We begin with a brief survey of results and definitions pertaining to well-quasi-orders. Questions relating to the quasi-ordering of graphs drove a lot of the development of graph structure theory; this is expanded in more detail in Section 3.

Readers familiar with the field will likely recognize most of the results, however, some of the notation and terminology used in this thesis is novel.

This section presents only a brief introduction to the rich theory of well-quasi-orders such as is necessary to understand the rest of the thesis. Readers interested in learning more are directed to Kruskal's 1970 survey paper [8], which is still an excellent, if slightly dated, resource for the basics in the area.

2.1 Well-Quasi-Orders

Definition 2.1. A relation $\Omega = (S, \preceq)$ is a *quasi-order* (sometimes called a *preorder*) if it is reflexive and transitive.

We write $s \prec t$ to mean that $s \preceq t$ but $t \not\preceq s$.

Let $\Omega = (S, \preceq)$ be a quasi-order. A subset $C \subseteq S$ is *upwards-closed* if, for all $x \in C$ and $y \in S$, if $x \preceq y$, then $y \in C$.

We denote by $\uparrow(\Omega)$ the set of upwards-closed subsets of S. We also denote by $\mathrm{cl}_\uparrow(X)$ the *upwards closure* of X, which is the intersection of all upwards-closed sets that contain X.

Analagously, C is *downwards-closed* if, for all $x \in C$ and $y \in S$, if $y \preceq x$, then $y \in C$. We also say that C is an *ideal* if C is downwards-closed and we have the corresonding sets $\downarrow(\Omega)$ and $\mathrm{cl}_\downarrow(X)$.

Definition 2.2. Let $\Omega = (S, \preceq)$ be a quasi-order. An infinite sequence $(s_i)_{i=1}^\infty$ in Ω is *ascending*, *strictly ascending*, *descending*, or *strictly descending* if, respectively, for every i, $s_i \preceq s_{i+1}$, $s_i \prec s_{i+1}$, $s_{i+1} \preceq s_i$, or $s_{i+1} \prec s_i$.

The sequence is *bad* if, for all $1 \leq j < i$, $s_j \not\preceq s_i$.

A quasi-order $\Omega = (S, \preceq)$ is a *well-quasi-order* or *wqo* if it contains no bad sequences.

Theorem 2.3 (Higman, [4])**.** *Let $\Omega = (S, \preceq)$ be a quasi-order. Then the following are equivalent:*

1. *Ω is wqo.*
2. *Every non-empty subset of Ω contains at least one, but finitely many, minimal elements.*
3. *Every infinite sequence in Ω has an infinite ascending subsequence.*
4. *There are no infinite strictly descending sequences in Ω and no infinite sets of pairwise incomparable elements.*
5. *If $s_1, s_2, \ldots$ is an infinite sequence in Ω, then there exist $1 \leq j < i$ such that $s_j \preceq s_i$.*
6. *Every upwards-closed subset of Ω is the closure of a finite subset.*
7. *The upwards-closed subsets of Ω satisfy the ascending chain condition (every ascending sequence $I_1 \subseteq I_2 \subseteq I_3 \subseteq \ldots$ is eventually constant).*

Corollary 2.4. *Let $\Omega = (S, \preceq)$ be a wqo and let C be an upwards-closed set in Ω. Then the set F of minimal elements of C is the unique smallest set F' such that $C = \mathrm{cl}_\uparrow(F')$.*

Proof. Suppose that F' is such that $\mathrm{cl}_\uparrow(F') = C$. Since each element of F is minimal in C, $F \subseteq F'$. It follows that F is the smallest such set. □

Definition 2.5. Let $C \subseteq S$ be an ideal. Let F be the minimal set such that $S \setminus C = \mathrm{cl}_\uparrow(F)$.

F is called the *obstruction set* of C (sometimes also the *h*forbidden or *excluded* elements of C). We write $F = \mathrm{obs}(C)$ and $C = \mathrm{forb}(F)$.

2.2 Derived WQOs

As one might expect, finite Cartesian products of wqos are themselves wqo:

Proposition 2.6. *Let $\Omega_1 = (S_1, \preceq_1)$ and $\Omega_2 = (S_2, \preceq_2)$ be well-quasi-orders. Then $\Omega_1 \times \Omega_2 = (S_1 \times S_2, \preceq)$ is a well-quasi-order, where $(x_1, x_2) \preceq (y_1, y_2)$ if and only if $x_1 \preceq_1 x_2$ and $y_1 \preceq_2 y_2$.*

Of critical interest to us are orderings on the recursive powersets.

Definition 2.7. Let $\Omega = (S, \preceq)$ be a quasi-order and let α be an ordinal. Given the usual notation that $\mathcal{P}(S)$ is the power set of S, we begin by defining general powersets $\mathcal{P}^\alpha(S)$ recursively:

1. $\mathcal{P}^0(S) = S$.
2. If α is the successor of β, then $\mathcal{P}^\alpha(S) = \mathcal{P}(\mathcal{P}^\beta(S))$.
3. If α is a limit ordinal, then $\mathcal{P}^\alpha(S) = \bigcup_{\beta<\alpha} \mathcal{P}^\beta(S)$.

We now wish to define, for each ordinal α, $\preceq^\alpha$ on $\mathcal{P}^\alpha(S)$. Note that if δ is a limit ordinal and $\alpha < \delta$, then $\mathcal{P}^\alpha(S) \subseteq \mathcal{P}^\delta(S)$. It is more convenient to recursively define $\preceq^\delta$ first, and then define $\preceq^\alpha$ by taking the restriction of $\preceq^\delta$ to $\mathcal{P}^\alpha(S)$.

Definition 2.8. We set $\preceq^0 = \preceq$.

Let α be a limit ordinal. For $X, Y \in \mathcal{P}^\alpha(S)$, there are least ordinals β and γ such that $X \in \mathcal{P}^\beta(S)$ and $Y \in \mathcal{P}^\gamma(S)$. Note that β and γ will always be either successors or 0, by the definition of $\mathcal{P}^\beta(S)$. If α is a successor, it may be the case that α is equal to either or both of β and γ.

We define $\preceq^\alpha$ recursively on β and γ:

1. If $\beta > \gamma$, then $X \not\preceq^\alpha Y$.
2. If $\beta = 0$ and $\gamma = 0$, then $X \preceq^\alpha Y$ if and only if $X \preceq Y$.
3. If $\beta = 0$ and $\gamma > 0$, then $X \preceq^\alpha Y$ if and only if there is a $y \in Y$ such that $X \preceq^\alpha y$.
4. If $\beta \preceq \gamma$ and $\beta \neq 0$, then $X \preceq^\alpha Y$ if and only if for every $x \in X$, there is some $y \in Y$ such that $x \preceq^\alpha y$.

Finally, we define $\mathcal{P}^\alpha(\Omega)$ to be the quasi-order $(\mathcal{P}^\alpha(S), \preceq^\alpha)$.

Lemma 2.9. *$\preceq^\alpha$ is well-defined when α is a successor ordinal.*

Proof. Let δ and ϵ be limit ordinals with $\alpha < \delta \leq \epsilon$. For any $X, Y \in \mathcal{P}^\alpha(S)$, when we define $X \preceq^\delta Y$, we will pick β and γ to be at most α, since $X, Y \in \mathcal{P}^\alpha(S)$. We will pick the same β and γ when defining $X \preceq^\epsilon Y$, and so, by transfinite induction on β and γ, the definitions coincide up to α. □

The intent of this definition is to effectively reduce the question $X \preceq^\alpha Y$ to a question about each of the members of X. Some difficulty arises in the limit ordinal case, because we may have, for instance, that $X \in \mathcal{P}^1(S)$ while $Y \in \mathcal{P}^2(S)$. The effect of the definition is to "unwrap" the levels of each ordinal successively until we make it back to S. Alternatively, we are comparing the recursive union of X and Y, with the caveat that if X is wrapped deeper in powersets than Y is, then we have $X \not\preceq^\alpha Y$.

Once we have established this definition for the limit ordinal case, it is easier to define the successor ordinal case in terms of this, rather than creating a separate, but equivalent, definition for successor ordinals.

Lemma 2.10. *If $\mathcal{P}^\alpha(\Omega)$ is wqo, and $\beta < \alpha$, then $\mathcal{P}^\beta(\Omega)$ is wqo.*

Proof. Suppose that $(s_i)_{i=1}^\infty$ is a bad sequence in $\mathcal{P}^\beta(\Omega)$. Let γ be the largest limit ordinal less than or equal to α, or 0 if no such ordinal exists. Then $\alpha = \gamma + n$ for some $n < \omega$.

Then for each i, let s_i' be s_i with n layers of singleton sets around it (so that e.g. if $n = 3$ and $s_i = x$, then $s_i' = \{\{\{x\}\}\}$). Thus, $s_i' \in \alpha$, and then $(s_i')_{i=1}^\infty$ is a bad sequence in $\mathcal{P}^\alpha(\Omega)$. □

As noted above, a better-quasi-order, or bqo, is a strengthening of well-quasi-ordering to uncountable analogues of sequences. There are a number of equivalent definitions, including combinatorial and topological ones, but the one of greatest relevance for this present work is as follows: a bqo is a quasi-order Ω such that $\mathcal{P}^{\omega_1}(\Omega)$ is well-quasi-ordered [10].

We will also need a powerset ordering on the set of all finite subsets of the underlying set.

Proposition 2.11. *Let $\Omega = (S, \preceq)$ be a quasi-order. Then $\Omega^{(<\omega)} = (S^{(<\omega)}, \preceq_1)$ is a quasi-order, where $S^{(<\omega)}$ is the set of all finite subsets of S.*

This next result allows us to limit our consideration of higher levels of quasi-ordering to the downwards-closed ideals, and with a simpler relation on them:

Proposition 2.12. *Let $\Omega = (S, \preceq)$ be a quasi-order. Then $\mathcal{P}(\Omega)$ is a wqo if and only if $\mathscr{I} = (\downarrow(\Omega), \subseteq)$ is.*

Proof. Suppose that $\mathcal{P}(\Omega)$ is a wqo, and let $(\mathcal{I}_i)_{i=1}^{\infty}$ be a sequence in $\mathscr{I}$. Since $\downarrow(\Omega) \subseteq \mathcal{P}(S)$, there exist $i > j \geq 1$ such that $\mathcal{I}_j \preceq^1 \mathcal{I}_i$. So for each $x \in \mathcal{I}_j$, there exists $y \in \mathcal{I}_i$ such that $x \preceq y$. Since $\mathcal{I}_i$ is downwards-closed, $x \in \mathcal{I}_i$, and hence $\mathcal{I}_j \subseteq \mathcal{I}_i$. Thus $\mathscr{I}$ is a wqo.

Conversely, suppose that $\mathscr{I}$ is a wqo, and let $(X_i)_{i=1}^{\infty}$ be a sequence in $\mathcal{P}(\Omega)$. Define $\mathcal{I}_1, \mathcal{I}_2, \ldots$ in $\mathscr{I}$ by $\mathcal{I}_i = \mathrm{cl}_{\downarrow}(X_i)$. By the hypothesis, we can find $i > j \geq 1$ such that $\mathcal{I}_j \subseteq \mathcal{I}_i$. Then, for any $x \in X_j$, it must be that $x \in \mathcal{I}_i$, so there must be $y \in X_i$ such that $x \preceq y$. It follows that $X_j \preceq^1 X_i$, and hence $\mathcal{P}(\Omega)$ is a wqo. □

2.3 Proving Existence of WQOs

Finally, we introduce the standard tool for proving the existence of a nontrivial wqo.

Definition 2.13. Let $\Omega = (S, \preceq)$ be a quasi-order. A *partial ranking* of Ω is a coarser partial order $\preceq_{\mathrm{p}}$ of S such that there are no infinite strictly descending sequences. In other words, if $x \preceq_{\mathrm{p}} y$, then $x \preceq y$, but not necessarily the other way around.

Let $f = (f_i)_{i=1}^{\infty}$ and $g = (g_i)_{i=1}^{\infty}$ be sequences in S, and let $\preceq_{\mathrm{p}}$ be a partial ranking of Ω. We write that $f \preceq_{\mathrm{p}}^* g$ (respectively $f \prec_{\mathrm{p}}^* g$) if g contains a subsequence $(g_{j_i})_{i=1}^{\infty}$, with $j_i < j_{i+1}$, such that $f_i \preceq_{\mathrm{p}} g_{j_i}$ (respectively $f_i \prec_{\mathrm{p}} g_{j_i}$).

A sequence f in S is called *minimal bad* with respect to $\preceq_{\mathrm{p}}$ if it is bad and there is no bad sequence g in Ω such that $g \prec_{\mathrm{p}}^* f$.

Note that it is not necessarily true that if $g \preceq_{\mathrm{p}}^* f$ and $g \neq f$, then $g \prec_{\mathrm{p}}^* f$. Although the definitions of $\preceq_{\mathrm{p}}^*$ and minimal bad do not actually require that $\preceq_{\mathrm{p}}$ be a partial ranking, and they work just as well for any quasi-order, we will only use them with the hypothesis that $\preceq_{\mathrm{p}}^*$ is a partial ranking.

The intent of a partial ranking is to allow us to pick a relation on S that we can more easily reason about. Usually we do this by taking the intersection of $\preceq$ and some other relation that we know more about. We assume that Ω is not wqo, take a minimal bad sequence f by the following lemma, and then build a contradiction by finding a bad sequence g such that $g \prec_{\mathrm{p}}^* f$.

Lemma 2.14 (Nash-Williams, reproduced by Thomas [19])**.** *Let $\Omega = (S, \preceq)$ be a quasi-order with no strictly descending infinite sequences, and let $f = (f_i)_{i=1}^{\infty}$ be bad. Then, for any partial ranking $\preceq_{\mathrm{p}}$ of Ω, there exists a minimal bad sequence $g = (g_i)_{i=1}^{\infty}$ such that $g \preceq_{\mathrm{p}}^* f$.*

Proof. Define g_1 to be the first term of some bad sequence $g' \preceq_{\mathrm{p}}^* f$ such that there is no $q \prec_{\mathrm{p}} g_1$ with the same property. This must exist as $f \preceq_{\mathrm{p}}^* f$, and Ω has no infinite strictly descending sequences.

Then for $i > 1$, define g_{i+1} such that $g_1, \ldots, g_{i+1}$ are the first $i+1$ terms of some bad sequence $g' \preceq_{\mathrm{p}}^* f$ such that there is no $q \prec_{\mathrm{p}} g_{i+1}$ with this property. Again, because we can pick $g_i = f_i$ and there are no strictly descending infinite sequences, this is always possible.

By construction, $g \preceq_{\mathrm{p}}^* f$. Also, g is bad because, for any $j < i$, $g_1, \ldots, g_i$ is the prefix of some bad sequence g' and hence $g_j \not\preceq g_i$. It remains to show that g is minimal bad.

Supppose there is a bad sequence h such that $h \prec_{\mathrm{p}}^{*} g$. Then there exists some j_0 such that $h_1 \prec_{\mathrm{p}} g_{j_0}$; pick j_0 to be as small as possible. We must have $j_0 > 1$, as if $j_0 = 1$, then $h_1 \prec_{\mathrm{p}} g_1$ and h is bad, which contradicts the choice of g_1.

Define $k = g_1, \ldots, g_{j_0-1}, h_1, h_2, \ldots$. Because g and h are bad, either k is bad or there exists some ℓ_0 with $1 \leq \ell_0 < j_0$ and some $i_0 \geq 1$ such that $g_{\ell_0} \preceq_{\mathrm{p}} h_{i_0}$. If the latter, then since $h \prec_{\mathrm{p}}^{*} g$, there exists a sequence $j_1 < j_2 < \ldots$ such that, for all $i \geq 1$, $h_i \prec_{\mathrm{p}} g_{j_i}$.

In particular, then, $h_{i_0} \prec_{\mathrm{p}} g_{j_{i_0}}$, which means $g_{\ell_0} \preceq_{\mathrm{p}} h_{i_0} \prec_{\mathrm{p}} g_{j_{i_0}}$. As j_0 is least such that $h_1 \prec_{\mathrm{p}} g_{j_0}$, and $h_1 \prec_{\mathrm{p}} g_{j_1}$, it follows that $j_0 \leq j_1$. Since $i_0 \geq 1$, $j_1 \leq j_{i_0}$, and we get that $\ell_0 < j_0 \leq j_1 \leq j_{i_0}$. Since $g_{\ell_0} \prec_{\mathrm{p}} g_{j_{i_0}}$, this contradicts the assumption that ℓ_0 and i_0 exist. So k must be bad after all.

So $k_{j_0} = h_1 \preceq_{\mathrm{p}} g_{j_0}$ and k is bad. This contradicts the choice of g_{j_0} made while constructing g. So g is minimal bad, as required. □

3 Graph Basics

In this section, we review the basic definitions of graph theory used in this thesis, and introduce a number of relations and labelling concepts that will be used later. A reader familiar with graph theory should still read this section, as some of the structural relations will likely be novel, and the mechanisms we use for graph labeling are important to understand.

We also touch on the basics of graph structure theory, including tree-decompositions and known well-quasi-ordering results. Most of these originate in the work of Robertson and Seymour.

3.1 Graphs

Definition 3.1. We will assume that the reader is familiar with the basics of graph theory. In this thesis, unless otherwise specified, all graphs will be finite, with loops and parallel edges permitted. We will sometimes write $e = \overline{uv}$ if e is a uv-edge, and $e = u^\ell$ if e is a u-loop. Additionally, for the sake of simplicity, we will assume that $V(G)$ and $E(G)$ are always disjoint.

A subgraph H of G is *induced* if $E(H)$ is precisely the subset of $E(G)$ containing all edges with both ends in $V(H)$. If $V' \subseteq V(G)$, then $G[V']$ denotes the subgraph *induced by* V': the induced subgraph of G with vertex set V'.

The collection of all graphs is denoted $\mathcal{G}$. The set of all subgraphs of G is denoted $\mathfrak{S}(G)$.

Definition 3.2. For any edge e or set E' of edges of a graph G, $G \setminus e$ and $G \setminus E'$ are the subgraphs of G obtained by deleting e or all edges of E', respectively, and likewise G / e and G / E' are the subgraphs of G obtained by contracting them.

Note that if $C, D \subseteq E(G)$ are disjoint, then $G / C \setminus D$ is isomorphic to $G \setminus D / C$.

Definition 3.3. A *rooted tree* is a tree T with a single distinguished vertex $\mathrm{r}(T)$ called the *root*.

Given verices $u, v \in V(T)$, we say that u is *above* v if v is in the (unique) path from $\mathrm{r}(T)$ to u, including if $v = u$. The *children* $\mathrm{chldrn}(v)$ of v are the neighbours of v which are above it, and the *branch* of v, denoted $\mathrm{br}(v)$, is the subgraph induced by all vertices above it.

Note that, for each vertex v of T, v is the root of $\mathrm{br}(v)$.

3.2 Minors

Definition 3.4. Let G and H be graphs. A *collapse* of G to H is an ordered pair (η_V, η_E) of injective functions $\eta_V : V(H) \rightarrowtail \mathfrak{S}(G)$ and $\eta_E : E(H) \rightarrowtail E(G)$ such that:

1. for every $v \in V(H)$, $\eta_V(v)$ is connected;
2. for distinct $v, u \in V(H)$, $V(\eta_V(v))$ and $V(\eta_V(u))$ are disjoint;
3. for every $e \in E(H)$, $\eta_E(e)$ is not a member of $E(\eta_V(v))$ for any $v \in V(H)$; and
4. for every $e = uv \in E(H)$, if $\eta_E(e) = u'v' \in E(G)$, then either $u' \in V(\eta_V(u))$ and $v' \in V(\eta_V(v))$ or vice-versa.

Example 3.5. An example of a graph collapse can be see in Figure 1.

Definition 3.6. A graph H is a minor of a graph G, denoted $H \preceq_{\mathrm{m}} G$, if there exists a collapse of G to H.

A set downwards-closed under $\preceq_{\mathrm{m}}$ is called *minor-closed*.

The conventional definition of a graph minor is that $H \preceq_{\mathrm{m}} G$ if H can be obtained from G via a sequence of contractions, deletions of edges, and deletions of isolated vertices. The subgraphs $\eta_V(v)$ of a collapse encode the edges to be contracted, and the edges $\eta_E(e)$ are the edges to be retained, which shows that the concepts are equivalent. The definition in terms of collapse is more readily extended to hypergraphs and other structures, however, and this motivates its use.

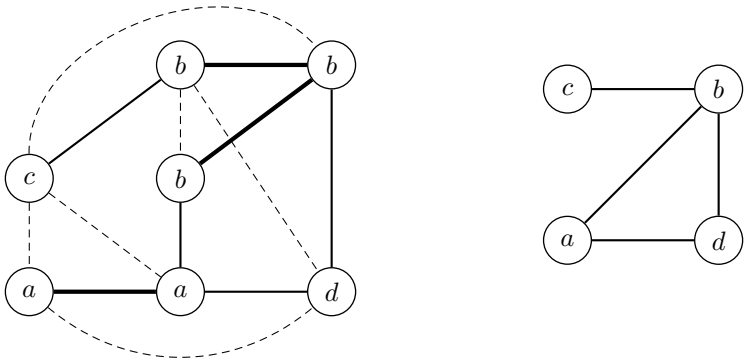

Figure 1: Example of a graph collapse of G (left) to H (right). Vertices with the same label and thick edges correspond to $\eta_V(v)$ for some $e \in V(H)$. Regular edges correspond to $\eta_E(e)$ for some $e \in E(H)$. Other edges are dashed. By contracting the thick edges and deleting the dashed edges, we retrieve H.

Theorem 3.7. *The following are equivalent:*

1. $H \preceq_{\mathrm{m}} G$.
2. *There exist disjoint* $C, D \subseteq E$ *such that* $H \cong G'$, *where* G' *is obtained from* $G \,/\, C \setminus D$ *by deleting any number of isolated vertices.*

There is also a more restricted notion of a topological minor, defined here in terms of a novel kind of collapse:

Definition 3.8. Let G and H be graphs. A *topological collapse* of G to H is an ordered pair $\eta = (\eta_V, \eta_E)$ of injective functions $\eta_V : V(H) \rightarrowtail V(G)$ and $\eta_E : E(H) \rightarrowtail \mathfrak{S}(G)$ such that:

1. for every $e = uv \in E(H)$, $\eta_E(e)$ is a $\nu_V(u)\nu_V(v)$-path in G; and
2. for every $e, f \in E(H)$, $\eta_E(e)$ and $\eta_E(f)$ are internally disjoint.

A graph H is a *topological minor* of G, or is *homeomorphically embeddable* in G, denoted $H \preceq_{\mathrm{t}} G$ if there exists a topological collapse of G to H.

We will generally avoid the term "homeomorphically embeddable" as potentially misleading, as homeomorphism is normally an equivalence relation. Similarly to how graph minors are usually defined by edge contraction and deletion, topological minors are usually defined in terms of graph subdivisions—obtained from a smaller graph by adding vertices in the middle of edges. Evidently, there is a topological collapse of G to H if and only if G contains a subgraph isomorphic to a subdivision of H. Moreover, by contracting only edges incident with a vertex of degree 2, we have the following standard result.

Proposition 3.9. *If* $H \preceq_{\mathrm{t}} G$, *then* $H \preceq_{\mathrm{m}} G$.

In general, topological minors do not admit the rich quasi-ordering structures of true minors; however, that does not mean that they are uninteresting.

Theorem 3.10. *If* G *has maximum degree at most 3, then* $H \preceq_{\mathrm{t}} G$ *if and only if* $H \preceq_{\mathrm{m}} G$.

Definition 3.11. Let S and T be rooted trees. A *rooted topological collapse* of T to S is a topological collapse $\eta = (\eta_V, \eta_E)$ such that whenever $\eta_V(v)$ is above $\eta_V(u)$ in T, then v is above u in S.

If there exists a rooted topological collapse of G to H, then H is a *rooted topological minor* of G, denoted $H \preceq_{\mathrm{r}} G$.

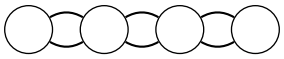

Figure 2: The graph $\mathbb{P}_4$.

This slightly restricted notion of a rooted topological minor was used by Nash-Williams [12] as a stepping-stone to prove that all trees are bqo under the topological minor relation. The definition presented here is different from, but equivalent to, Nash-Williams' original definition. This should not be confused with a weaker concept of the same name used by Kühn [9].

This relation will be explored further in Section 6.

Theorem 3.12 (Nash-Williams [12])**.** *The set of (possibly infinite) rooted trees is better-quasi-ordered under* $\preceq_{\mathrm{r}}$*.*

This theorem implies the following:

Theorem 3.13 (Nash-Williams [12])**.** *The set of (possibly infinite) trees is better-quasi-ordered under* $\preceq_{\mathrm{t}}$*.*

Recent work also gives this result:

Theorem 3.14 (Liu, Theorem 1.4.4 of [11])**.** *For every* $k \geq 1$*, the class of graphs* $\mathrm{forb}_{\preceq_{\mathrm{t}}}(\mathbb{P}_k)$ *is wqo under* $\preceq_{\mathrm{t}}$*, where* $\mathbb{P}_k$ *is the* Robertson chain of length k *obtained from the path of length* k *by replacing each edge with two parallel edges (see Figure 2).*

3.3 Labelings

We will also need to label vertices:

Definition 3.15. A *partial function* from A to B is a mapping from A to B which is not necessarily defined everywhere on A. If f is a partial function from A to B, we write $f : A \nrightarrow B$.

Definition 3.16. Let $G = (V, E)$, and let L be some set. Given a partial function $\mathscr{L} : V(G) \nrightarrow L$, G is *labeled* with labels from L with *label function* $\mathscr{L}$. A vertex v has label $\mathscr{L}(v)$. If $\mathscr{L}$ is defined everywhere on $V(G)$, then G is *fully labeled*.

A particular class of labeled graphs will be quite important, as the basis later for the HR algebra:

Definition 3.17. An *s-graph* (short for *graph with sources from* C) is a graph labeled from some label set C such that no two vertices get the same label. A labeled vertex is a *source*, a source labeled with $c \in C$ is a *c-source*, and a vertex that is not a source is *internal*.

Let G be an s-graph. Then the *type* of G is the set C such that the source labels of G are exactly C.

The set of all s-graphs of type C is denoted $\mathcal{G}_C$.

Note that a graph may be simultaneously labeled from multiple sets. In particular, a graph can both have sources from C and have additional labels from another set L.

Definition 3.18. Let $\Omega = (Q, \preceq)$ be a quasi-order and G and H be graphs fully labeled from Q by $\mathscr{L}_G$ and $\mathscr{L}_H$, respectively.

We write that:

- $H \preceq_{\mathrm{L}} G$ if there exists a collapse $\eta = (\eta_V, \eta_E)$ of G to H such that, for each $u \in V(H)$, there exists $v \in \eta_V(u)$ such that $\mathscr{L}_H(u) \preceq \mathscr{L}_G(v)$;
- $H \preceq_{\mathrm{Lt}} G$ if and only if there exists a topological collapse $\eta = (\eta_V, \eta_E)$ of G to H such that, for each $u \in V(H)$, $\mathscr{L}_H(u) \preceq \mathscr{L}_G(\eta_V(u))$; and
- $H \preceq_{\mathrm{Lr}} G$ if, additionally, H and G are rooted trees and η is rooted.

In any of these cases, we say that the ((rooted) topological) collapse η *respects labels*.

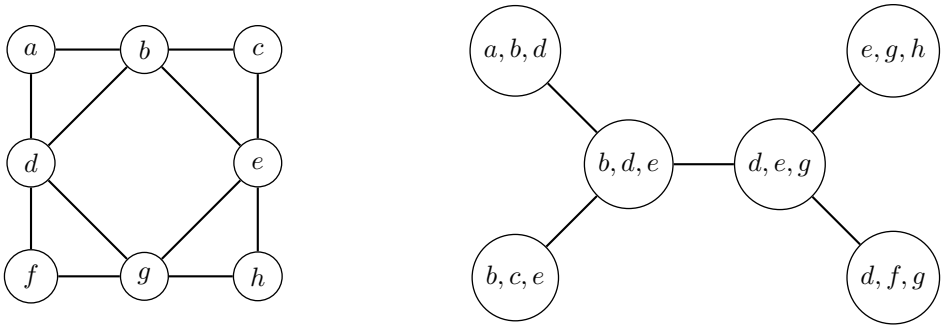

Figure 3: A graph G and a tree-decomposition T of G. T has width 2, because its largest bag has size 3.

3.4 Tree-Decompositions

Definition 3.19. Let $G = (V, E)$ be a graph, and let $T = (W, E')$ be a tree fully labeled from $\mathcal{P}(V)$ by the function f. For each $v \in V$, define $W(v) = \{w \in W : v \in f(w)\}$. Then the pair (T, f) is a *tree-decomposition* of G if and only if:

1. for each $v \in V$, $W(v)$ is nonempty and $T[W(v)]$ is connected; and
2. for each edge $e = uv \in V$, there is some $w \in W$ such that $u, v \in f(w)$.

The sets $f(w)$ are called the *bags* of (T, f). For $w, w' \in W$, we will write $w \equiv w'$ to mean that $f(w) = f(w')$.

The *width* of (T, f) is $\max\{|f(w)| : w \in W\} - 1$. The *tree-width* $\mathrm{tw}(G)$ of G is the least width of any tree-decomposition of G.

For any nonnegative integer k and class $\mathcal{G}$ of graphs, $\mathcal{G}_{\mathrm{tw}\leq k} = \{G \in \mathcal{G} : \mathrm{tw}(G) \leq k\}$.

The -1 term in the definition of width is so that a tree has tree-width 1, and not for any technical reason. We will sometimes refer to the bags and vertices of T interchangeably where there is no ambiguity.

Example 3.20. Figure 3 shows an example of a tree-decomposition.

The lemma below allows the imposition of some structure on a tree-decomposition to make transformations easier.

Definition 3.21. A tree-decomposition $(T = (W, E), f)$ is *nice* if we can root T such that, for each $w \in W$, exactly one of the following holds:

1. w is a leaf and $f(w)$ is size 1.
2. w has exactly one child u, and $f(w)$ can be obtained from $f(u)$ by either adding or removing a single element v, in which case w is an *introduce node* or a *forget node*, respectively.
3. w has exactly two children u_1, u_2, and $w \equiv u_1 \equiv u_2$, in which case w is a *join node*.

Lemma 3.22 (Lemma 13.1.3 in [6])**.** *If G has a tree-decomposition of width k, then it has a nice tree-decomposition of width at most k.*

Proof Sketch. If T is a tree-decomposition of G of width k, then we can reduce every vertex to degree at most 3 by replacing it by a tree of vertices with the same bag.

Then for any adjacent pair of vertices with different bag contents, we can insert a path in between with nodes that forget and/or introduce the necessary vertices. □

Example 3.23. Figure 4 shows a nice version of the earlier tree-decomposition.

The following is given without proof in [14]. The proof is straightforward: a tree-decomposition for G readily provides one for its minor H.

Proposition 3.24. *If $\mathrm{tw}(G) \leq k$, then $\mathrm{tw}(H) \leq k$ for every minor H of G. Equivalently, for any k, $\mathcal{G}_{\mathrm{tw}\leq k}$ is downwards closed under $\preceq_{\mathrm{m}}$.*

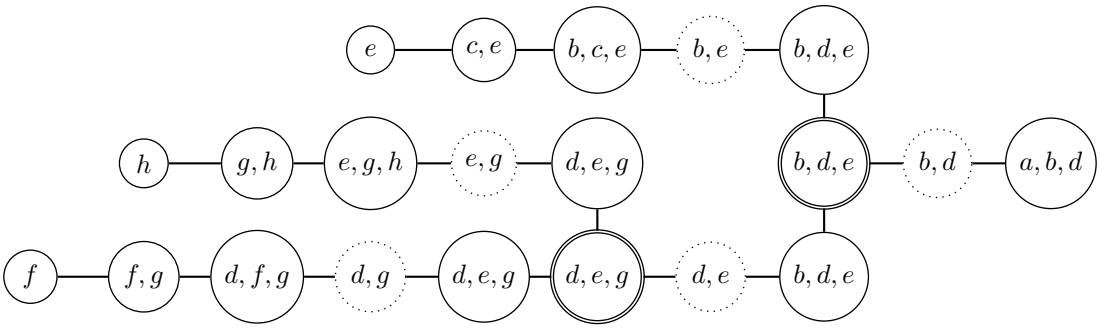

Figure 4: A nice tree-decomposition of the graph G from Figure 3 with the same width as T. The top-right vertex (a, b, d) is the root. Join nodes are double-struck and forget nodes are dotted. The remaining nodes are either leaves or introduce nodes.

3.5 Graph Minors Structure

The following is the Graph Minors Theorem:

Theorem 3.25 (Robertson & Seymour [17])**.** $(\mathcal{G}, \preceq_{\mathrm{m}})$ *is a well-quasi-order.*

We are interested in collections of graphs having an upper bound to their treewidth, and we will be referring to this property several times.

Definition 3.26. A set S of graphs has *bounded treewidth* if there exists k such that every $G \in S$ has tree-width at most k.

The notion of tree-width plays a crucial role in the proof of the Graph Minors Theorem. The theorem is built upon the following results:

Theorem 3.27 (Robertson & Seymour [16])**.** $(\mathcal{G}_{\mathrm{tw}\leq k}, \preceq_{\mathrm{m}})$ *is a wqo for all* k*.*

Theorem 3.28 (Robertson & Seymour [15])**.** *For any planar graph* H*, there exists* k *such that any graph* G *which does not contain* H *as a minor has tree-width at most* k*.*

Corollary 3.29. *If* $\mathcal{J}$ *is a minor-closed class that does not contain all planar graphs, then* $\mathcal{J}$ *has bounded tree-width.*

If the reader is interested in the proofs of these results, they are directed to a survey paper by Richter which presents them in a more digestible form [13].

The interest in the classes $\mathcal{G}_{\mathrm{tw}\leq k}$ towards strengthening the Graph Minors Theorem should thus be apparent. Thomas obtained a general strengthening of Nash-Williams' bqo result:

Theorem 3.30 (Thomas [19])**.** *For any planar graph* H_0*,* $(\mathrm{forb}(\{H_0\}), \preceq_{\mathrm{m}})$ *is bqo, even if we admit infinite graphs.*

Corollary 3.31. *For any* k*,* $(\mathcal{G}_{\mathrm{tw}\leq k}, \preceq_{\mathrm{m}})$ *is bqo, even if we admit infinite graphs.*

We would like to be able to apply logic and language theory to obtain similar results in this area of graph structure.

4 Abstract Algebras

In this section, we present the basics of formal language theory over arbtirary algebras. While we are only concerned with one particular algebra in this thesis, explored in the next section, it is both more convenient and more interesting to present the more general version.

Formal language theory is, in its simplest form, the study of the structure of sets of strings over a finite alphabet. This study can be generalized by taking an algebraic view of them (in the sense of universal algebra). The set of strings over an alphabet is in fact just the free monoid over that same alphabet. If we generalize the concepts by allowing the study of arbitrary algebras, including ones with different operations, or that are not free, we discover that many of the concepts generalize well and that there is indeed a rich theory to be discovered.

4.1 Signatures & Terms

In order to talk about abstract equation systems in an algebra, we must have a way to talk about expressions with unknowns. We begin by defining an abstract algebraic language to work with, without any reference to any concrete algebra. As discussed in Section 1.3, this allows us to treat expressions such as $x^2 + 1$ as distinct mathematical objects. At the same time, we define the actual algebras with real elements and operations, and how these relate to the abstract language.

Definition 4.1. A *functional signature* F is a set of *function symbol*, each with an associated natural number called its *arity*. We denote by $\rho(f)$ the arity of f. Symbols of arity 0, 1, and 2 are called *constant*, *unary*, and *binary* respectively.

An *F-algebra* $\mathbb{M}$ is a set M (the *underlying set*) and, for each $f \in F$, a function $f_{\mathbb{M}} : M^{\rho(f)} \to M$ called the *interpretation* of f. Here, M^n is the Cartesian product of M with itself n times.

Example 4.2. Consider the signature $R = (0, 1, -, +, \cdot)$, where $\rho(0) = \rho(1) = 0$, $\rho(-) = 1$, and $\rho(+) = \rho(\cdot) = 2$. This signature is the signature used to define an abstract ring.

Any ring is an R-algebra, but the converse is not necessarily true. For example, if M is some nonempty set, pick a constant $0_M = 1_M \in M$, and interpret $-$, $+$, and $\cdot$ as constant functions so that $-a = a + b = a \cdot b = 0_M$. The result would be a valid R-algebra but not a ring.

Our language has several purely syntactic operations. We define substitution of terms, where each ocurrence of an unknown is replaced by another term, and evaluation, where we take a concrete algebra and apply the functions corresponding to the function symbols in the term.

Definition 4.3. Let F be a functional signature and let X be some set of indeterminates, or *variable*. We will recursively define the F-terms as follows:

1. For any $x \in X$, x is an F-term; and
2. For any $f \in F$ with arity r, for any F-terms $e_1, \ldots, e_r$, $f(e_1, \ldots, e_r)$ is an F-term.

The *immediate subterms* of a term of the form $f(e_1, \ldots, e_r)$ are $\{e_1, \ldots, e_r\}$; a term in X has no subterms. The *subterms* of a term are itself and all the subterms of its immediate subterms.

The term x *contains* the variable x; the term $f(e_1, \ldots, e_r)$ *contains* the union of the variables contained in each of $e_1, \ldots, e_r$.

If $x_1, \ldots, x_n$ are variables, and $t_1, \ldots, t_n$ are F-terms, then $t(x_1/t_1, \ldots, x_n/t_n)$ is the term obtained from t by substituting each occurrence of x_i in t and its subterms with t_i.

Let $\mathbb{M}$ be an F-algebra over M and t an F-term. Let $x_1, \ldots, x_n$ be a list of variables including all variables contained in t, and possibly others. For any $a_1, \ldots, a_n \in M$, we define the *value* of t at $a_1, \ldots, a_n$ as:

$$\mathrm{val}_{\mathbb{M}}(t)(x_1/a_1, \ldots, x_n/a_n) = \begin{cases} a_i & : x = x_i \text{ for some } 1 \le i \le n \\ x & : \text{otherwise} \end{cases}$$

$$\mathrm{val}_{\mathbb{M}}(f(e_1, \ldots, e_r))(x_1/a_1, \ldots, x_n/a_n) = f_{\mathbb{M}}(\mathrm{val}_{\mathbb{M}}(e_1)(x_1/a_1, \ldots, x_n/a_n), \ldots, \mathrm{val}_{\mathbb{M}}(e_r)(x_1/a_1, \ldots, x_n/a_n))$$

Where the variables $x_1, \ldots, x_n$ are clear from context, we will abbreviate and write $t(t_1, \ldots, t_n)$ and $\mathrm{val}_{\mathbb{M}}(t)(a_1, \ldots, a_n)$ for $t(x_1/t_1, \ldots, x_n/t_n)$ and $\mathrm{val}_{\mathbb{M}}(t)(x_1/a_1, \ldots, x_n/a_n)$, respectively.

We denote by $\mathrm{T}(F)$ the set of all terms over F.

Example 4.4. Let $t = x_1 + (x_2 * -x_3)$ in the signature of rings. Then t's immediate subterms are $\{x_1, x_2 * -x_3\}$ and its subterms are $\{t, x_1, (x_2 * -x_3), x_2, -x_3, x_3\}$.

t contains the variables x_1, x_2, and x_3. If $t_2 = -x_2$, then $t(x_1/t_2) = -x_2 + (x_2 * -x_3)$.

Trying to evaluate in different rings, we get that $\mathrm{val}_{\mathbb{R}}(t)(x_1/3, x_2/5, x_3/7) = 3 + (5 * -7) = -32$. But if we let GF(11) denote the finite field of order 11, then $\mathrm{val}_{\mathrm{GF}(11)}(t)(x_1/3, x_2/5, x_3/7) = 3 + (5 * -7) = 1$.

Note that if c is a constant symbol in F, then $c()$ is a valid F-term. We will usually omit the parentheses and simply write c.

We must be careful to avoid assuming associativity when we are working with abstract terms—the terms $(e_1 + e_2) + e_3$ and $e_1 + (e_2 + e_3)$ are different, so we cannot write $e_1 + e_2 + e_3$ without introducing ambiguity. The same concern applies to precedence between different operators. When we are working in concrete algebras known to be associative, then we will write $e_1 + e_2 + e_3$, or the like, with the knowledge that it can represent $(e_1 + e_2) + e_3$ or $e_1 + (e_2 + e_3)$ and the distinction is immaterial.

Example 4.5. Let $R = (0, 1, -, +, \cdot)$ be the signature of a ring, as defined above. Then if x, y, and z are variables (in that order), x, $x + y$, $-x \cdot (y + z)$, and $x + (1 + (x + 0))$ are all R-terms.

The expressions $x + y + z$ and $x + y \cdot z$ are not, formally, R-terms, because we do not have defined associativity or precedence.

Let $t_1 = -x \cdot (y + z)$ and $t_2 = x + y$. Then $t_1(0, 1, t_2) = -0 \cdot (1 + (x + y))$. If we now consider the real numbers as an R-algebra, then $t_1(3, 4, 2) = -3 \cdot (4 + 2) = -18$.

4.2 Equational Sets

There are two very important classes of sets we care about when dealing with algebras. The first concept is that of an equational set—effectively a recursion set in an algebra.

In order to formalize this notion, we extend an algebra $\mathbb{M}$ to one on its powerset:

Definition 4.6. Let F be a functional signature. Then the signature $F_\cup$ is F with an added binary (that is, arity 2) symbol $\dot\cup$ and a constant symbol $\dot\emptyset$.

Let $\mathbb{M}$ be an F-algebra over M. Then its *powerset algebra* $\mathcal{P}(\mathbb{M})$ is an $F_\cup$-algebra with underlying set $\mathcal{P}(M)$.

For each nonconstant function symbol f of F, $f_{\mathcal{P}(\mathbb{M})}(L_1, \ldots, L_n)$ is defined to be the image of $L_1 \times \ldots \times L_n$ under $f_{\mathbb{M}}$. In the case of a constant symbol c, we instead set $c_{\mathcal{P}(\mathbb{M})} = \{c_{\mathbb{M}}\}$, the singleton of the constant. We define $\dot\emptyset_{\mathcal{P}(\mathbb{M})} = \emptyset$, and $L_1 \dot\cup_{\mathcal{P}(\mathbb{M})} L_2 = L_1 \cup L_2$—the empty set and the union, respectively.

The powerset algebra formalizes the natural notion of applying a function to a set, and adds unions and an empty set. This allows us to reason about the relationship between sets and their images under an equation set. The union is associative, so for our purposes we do need to add brackets when using multiple unions.

Definition 4.7. Let F be a functional signature. Then a *monomial* is an F-term, and a *polynomial* is either $\dot\emptyset$ or else $t_1 \dot\cup t_2 \dot\cup \ldots \dot\cup t_n$, where each t_i is a monomial (it may be that $k = 1$ in which case the polynomial is simply t_1). A polynomial is thus an $F_\cup$-term.

The terms *monomial* and *polynomial* are analogous to those used in numerical algebra, where a monomial such as $4x^2y^7$ is obtained by multiplying variables or constants, and a polynomial such as $x^2 + 4yx - 4$ is a (possibly empty) finite sum of monomials.

While it is not the case that every $F_\cup$-term is a polynomial, we do have the following:

Proposition 4.8 ([2])**.** *Let F be a functional signature and $\mathbb{M}$ an F-algebra over M. If t is an $F_\cup$-term, there is a polynomial p over $F_\cup$ such that, for any $A_1, \ldots, A_k$ in $\mathcal{P}(M)$, $\mathrm{val}_{\mathcal{P}(\mathbb{M})}(t)(A_1, \ldots, A_k) = \mathrm{val}_{\mathcal{P}(\mathbb{M})}(p)(A_1, \ldots, A_k)$.*

Proof. Let $Mon(t)$, the set of monomials defining t, be defined recursively as follows:

$$\begin{aligned} Mon(x) &= \{x\} \text{ for a variable } x \\ Mon(\dot{\emptyset}) &= \emptyset \\ Mon(t_1 \dot{\cup} t_2) &= Mon(t_1) \cup Mon(t_2) \\ Mon(f(t_1, \ldots, t_k)) &= \{f(m_1, \ldots, m_k) : m_i \in Mon(t_i), 1 \leq i \leq k\} \end{aligned}$$

By an easy induction, we get:

$$\mathrm{val}_{\mathcal{P}}(\mathbb{M})(t)(A_1, \ldots, A_n) = \bigcup \{\mathrm{val}_{\mathcal{P}(\mathbb{M})}(m)(A_1, \ldots, A_n) : m \in Mon(t)\}$$

Thus $p = \dot{\bigcup} Mon(t)$ is the desired polynomial. □

We now proceed to define an equation system, which relates several equations together. The relationship between the equations and the variables in them is not immediately obvious from the definition, but we will use a descriptive notation to make the relationship clear.

Definition 4.9. Let $x_1, \ldots, x_k$ be an ordered tuple of variables. Then an *equation system* S over F is an ordered tuple of polynomials $(p_1, \ldots, p_k)$ containing no variables other than $x_1, \ldots, x_k$.

We will also write:

$$\begin{aligned} x_1 &= p_1(x_1, x_2, \ldots, x_k) \\ x_2 &= p_2(x_1, x_2, \ldots, x_k) \\ &\vdots \\ x_k &= p_k(x_1, x_2, \ldots, x_k) \end{aligned}$$

to mean the equation system $(p_1, \ldots, p_k)$.

We say that S has *size* k.

In the definition of an equation system, it is possible that a polynomial p_i does not contain all variables $x_1, \ldots, x_k$. Recall that when we write $p_i(x_1, \ldots, x_k)$, we mean to always substitute for the appropriate variables. So if p_i only contains x_2, then $p_i(S_1, S_2)$ is a shorthand for $p_i(x_1/S_1, x_2/S_2)$, and thus means to substitute S_2 for x_2 and not for x_1.

Using the machinery of the powerset algebra, we can now easily define what a solution to an equation system is and, hence, the equational sets.

Definition 4.10. Let $\mathbb{M}$ be an F-algebra over M, and let $S = (p_1, \ldots, p_k)$ be an equation system over F of size k. Then a *solution* to S in $\mathbb{M}$ is a k-tuple of sets $L = (L_1, \ldots, L_k) \in M^k$ such that, for each i, $L_i = \mathrm{val}_{\mathcal{P}(\mathbb{M})}(p_i)(L_1, \ldots, L_k)$.

If $L = (L_1, \ldots, L_k)$ is a solution such that, for any solution $L' = (L'_1, \ldots, L'_k)$ and for $1 \leq i \leq k$, $L_i \subseteq L'_i$, then L is *least* and we write $L = \mathrm{L}(S)$. The sets $L_1, \ldots, L_k$ are *equational* in $\mathbb{M}$ and *generated* by S, denoted $L_i = \mathrm{L}(S, x_i)$.

Example 4.11. Using the signature $R = (0, 1, -, +, \cdot)$ of rings, consider the following equation system E:

$$S_1 = 1 \cup (S_2 * S_3)S_2 \qquad\qquad = 1 + 1 \cup -S_3S_3 = 0 \cup (S_2 + 1)$$

In $\mathbb{Z}$, the ring of integers, the following is a solution to E:

$$L_1 = \{-9, -4, -3, -1, 0, 1, 2, 6\} L_2 = \{-3, -1, 0, 2\} L_3 = \{-2, 0, 1, 3\}$$

We can verify that this is indeed a solution by evaluating each monomial with each possible substition of S_i by an element of L_i. For instance, for S_2, $1 + 1 = 2 \in L_2$, and for each $e \in L_3$, $-e \in S_2$.

The following result is very important, and arises as an instance of a general least fixed-point theorem on partial orders:

Proposition 4.12. *Let $S = (p_1, \ldots, p_k)$ be an equation system over F. Then S has a unique least solution.*

Proof. Let $L^0 = (L_1^0, L_2^0, \ldots, L_k^0) = (\emptyset, \emptyset, \ldots, \emptyset)$. For each positive n, let $L_i^n = \mathrm{val}_{\mathcal{P}(\mathbb{M})}(p_i)(L^{n-1}) \cup L_i^{n-1}$, and $L^n = (L_1^n, \ldots, L_k^n)$. To get our solution, set $L_i = \bigcup_{j=0}^{\infty} L_i^j$. Then take $L = (L_1, \ldots, L_k)$.

Now suppose we have some $x = (\{x_1\}, \ldots, \{x_k\})$ with $x_i \in L_i$ for $1 \leq i \leq k$. There must be some j such that, for all $1 \leq i \leq k$, $x_i \in L_i^j$. It follows that for any $1 \leq i \leq k$, we have $\mathrm{val}_{\mathcal{P}(\mathbb{M})}(p_i)(x) \subseteq L_i^{j+1} \subseteq L_i$. Hence $\mathrm{val}_{\mathcal{P}(\mathbb{M})}(p_i)(L) \subseteq L_i$.

Moreover, for each i with $1 \leq i \leq k$, there must be some $y_i \in L_1^j \times \ldots \times L_k^j$ such that $x_i \in \mathrm{val}_{\mathcal{P}(\mathbb{M})}(p_i)(y_i)$. We can regard y_i as an element of L in the natural way, giving that $x_i \in \mathrm{val}_{\mathcal{P}(\mathbb{M})}(p_i)(L)$. So $L_i \subseteq \mathrm{val}_{\mathcal{P}(\mathbb{M})}(p_i)(L)$.

Thus, $L_i = \mathrm{val}_{\mathcal{P}(\mathbb{M})}(p_i)(L)$, and so L is a solution to S. We wish to show that it is minimal.

Now suppose that $L' = (L_1', \ldots, L_k')$ is a solution to S. Then for any i, $\emptyset = L_i^0 \subseteq L_i'$, and hence it follows inductively that for any j, $L_i^j \subseteq \mathrm{val}_{\mathcal{P}(\mathbb{M})}(p_i)(L') = L_i'$. So $L_i \subseteq L_i'$ for each i, meaning that L' contains L. Hence L is in fact a minimal solution. □

As a result, we will often refer to the least solution of an equation system as *the* solution. Readers familiar with language theory should easily recognize that the equational sets of a free monoid over a finite alphabet are the context-free languages over that alphabet.

Definition 4.13. Let F be a functional signature. A *subsignature* of F is a functional signature F' such that $F' \subseteq F$ and, for each $f \in F'$, $\rho_{F'}(f) = \rho_F(f)$.

Let $\mathbb{M}$ be an F-algebra. Then the F'-algebra *induced* by $\mathbb{M}$ is the algebra $\mathbb{M}_{F'}$ obtained by taking the same underlying set and interpreting each symbol of F' identically as in $\mathbb{M}$.

Lemma 4.14. *Let F be a functional signature and let $\mathbb{M}$ be an F-algebra.*

If a set E is equational in $\mathbb{M}$, then there exists a finite subsignature F' of F such that E is equational in $\mathbb{M}_{F'}$.

Proof. Pick some equation system S defining E and let F' consist of all symbols appearing in S. Since S is finite, F' is too. □

4.3 Recognizable Sets

The second, less concrete classification of sets that we use is that of a recognizable set. Intuitively, a set is recognizable if there is a way to identify its members using a process that admits a finite description. There are several equivalent characterizations, but the simplest to state is in terms of algebra homomorphisms, which are defined in the obvious way. It is worth noting that one characterization is in terms of automaton theory. Under this characterization, the recognizable sets are those accepted by a class of finite automata obtained by generalizing determinisitic finite automata (DFAs) for terms of arity more than 1.

Definition 4.15. Let F be a functional signature and let $\mathbb{M}$ and $\mathbb{W}$ be F-algebras over M and W, respectively. A function $h : M \to W$ is an *F-homomorphism* if, for all symbols f of F and all elements $e_1, \ldots, e_{\rho(f)} \in M$, $h(f_{\mathbb{M}}(e_1, \ldots, e_{\rho(f)})) = f_{\mathbb{W}}(h(e_1), \ldots, h(e_{\rho(f)}))$.

If $h : M \to W$ is a F-homomorphism, we write $h : \mathbb{M} \to \mathbb{W}$.

Definition 4.16. Let F be a functional signature, $\mathbb{M}$ an F-algebra over M, and $R \subseteq M$. We say that R is *recognizable* if there exists an F-algebra $\mathbb{A}$ over a finite set A and a homomorphism $h : \mathbb{M} \to \mathbb{A}$ such that, for some $C \subseteq A$, $R = h^{-1}(C)$.

In this case h *witnesses* the recognizability of R.

Lemma 4.17. *Let F be a functional signature and F' a subsignature of F. Let $\mathbb{M}$ be an F-algebra. If R is recognizable in $\mathbb{M}$, then it is recognizable in $\mathbb{M}_{F'}$.*

Proof. Any F-homomorphism $h : \mathbb{M} \to \mathbb{A}$ is also an F'-homomorphism $\mathbb{M}_{F'} \to \mathbb{A}_{F'}$. □

We will directly work with one other characterization of recognizability, in terms of equivalence relations. Over regular languages of strings, this characterization is due to Myhill and Nerode and leads to several deep results.

Definition 4.18. Let F be a functional signature, $\mathbb{M}$ an F-algebra over M, and $\sim$ an equivalence relation on M. Let R be a subset of M, and let $f : M^r \to M$. Then:

1. R is *saturated* by $\sim$ if R is a union of $\sim$-equivalence classes.
2. $f_{\mathbb{M}}$ *respects* $\sim$ if, for all $m_1, \ldots, m_r$ and $m'_1, \ldots, m'_r$ such that $m_i \cong m'_i$, the following holds:
$$f_{\mathbb{M}}(m_1, \ldots, m_r) \sim f_{\mathbb{M}}(m'_1, \ldots, m'_r)$$
3. $\sim$ is a *congruence* on $\mathbb{M}$ if, for every function symbol f in F, the function $\mathbb{M}_f$ respects $\sim$.
4. If $\sim$ is a congruence, then the *quotient algebra* $\mathbb{M}/\sim$ is an F-algebra with underlying set the set of equivalence classes of $\sim$. Each constant $c_{\mathbb{M}/\sim}$ is defined as the equivalence class $[c_{\mathbb{M}}]$ containing $c_{\mathbb{M}}$, and each operation $f_{\mathbb{M}/\sim}$ is defined by:
$$f_{\mathbb{M}/\sim}([m_1], \ldots, [m_{\rho(f)}]) = [f_{\mathbb{M}}(m_1, \ldots, m_{\rho(f)})]$$

That $\sim$ is a congruence is necessary and sufficient for the quotient algebra to be well-defined, in a manner similar to the way quotients are defined over other algebras such as vector spaces and groups.

Lemma 4.19. *The natural map $h_\sim : m \mapsto [m]$ is an F-homomorphism $\mathbb{M} \to \mathbb{M}/\sim$.*

The relation between congruences and recognizability is a bit startling when seen at first, but helps to explain why recognizability is a very natural concept:

Proposition 4.20. *Let $\mathbb{M}$ be an F-algebra over M, and let L be a subset of M. Then L is recognizable if and only if there exists an M-congruence $\sim$ with finitely many equivalence classes such that $\sim$ saturates L.*

Proof. Suppose that $\sim$ exists as described. Then $h_\sim$ witnesses L's recognizability, because $\mathbb{M}/\sim$ has finitely many elements and there exists a set C of equivalence classes such that L is the union of C. From the definition of $h_\sim$, $L = h_\sim^{-1}(C)$.

Conversely, if L is recognizable with witness h, define $m \sim m'$ if and only if $h(m) = h(m')$. Then, since h is a homomorphism, $\sim$ is a congruence, and since the codomain of h is finite, there are only finitely many equivalence classes. □

This theory can be explored further. For a given set $L \subseteq M$, it is possible to define a canonical congruence that saturates it. In language theory, it is referred to as the Myhill-Nerode equivalence class. All congruences saturating L are refinements of the canonical one, and this congruence gives rise to the simplest automaton recognizing L. For more exploration of this, the reader should look to section 3.4.3 of [2] or, for applications in formal language theory, [18].

4.4 The Filtering Theorem

The Filtering Theorem expresses the major relationship between recognizable and equational sets, and it plays an important role in Section 10. We provide a sketch of the proof here; for the full proof, the reader is directed to [2].

Theorem 4.21 (Filtering Theorem [2])**.** *Let $\mathbb{M}$ be an F-algebra. Let R and E be a recognizable set and and equational set, respectively, in $\mathbb{M}$. Then $R \cap E$ is equational in M.*

Proof Sketch. Let F' be a finite subsignature of F such that E is equational in $\mathbb{M}_{F'}$. Then R is recognizable in $\mathbb{M}_{F'}$ by 4.17.

The terms over F' form an algebra $\mathbb{T}(F')$ in their own right, where each term is interpreted as itself. Then E corresponds to a $\mathbb{T}(F')$-equational set E' with the same equation system. Since val is a homomorphism $\mathbb{T}(F') \to \mathbb{M}_{F'}$, composing that with the witness h to R's recognizability shows that $R' = \mathrm{val}_{\mathbb{M}}^{-1}(R)$ is recognizable in $\mathbb{T}(F')$.

As F' is finite, it can be shown that the equational sets over $\mathbb{M}_{F'}$ are exactly those described by equational sets of terms. Furthermore, $E \cap R = \mathrm{val}_{\mathbb{M}}(E' \cap R')$.

Using a construction involving automata on terms, which is beyond the scope of this work, we can show that, as F' is finite, the equational and recognizable sets over $\mathbb{T}(F')$ coincide.

By combining two homomorphisms into a homomorphism onto a Cartesian product of the codomains, it follows that the intersection of any two recognizable sets is itself recognizable. Hence $E' \cap R'$ is recognizable and hence equational in $\mathbb{T}(F')$.

Finally, the image of an equational set of terms is also equational with the same equation system, and so it follows that $E \cap R$ is equational in $\mathbb{M}_{F'}$ and hence in $\mathbb{M}$. □

Readers familiar with language theory will recognize that, in the special case language of strings, the equational sets are the context-free languages, and the recognizable sets are the regular languages. The Filtering Theorem is thus a generalization of the theorem that the intersection of a context-free language with a regular language is context-free.

It is a standard result that every regular language is context-free, and so a reader might expect that, in the general case, every recognizable set is equational. This is not true, however. Consider the example of a signature with no function symbols. In this case, the only equational sets are the empty set and the entire underlying set, but every set is recognizable since any constant function is a homomorphism.

5 The HR Graph Algebra

In this section, we describe the algebra which forms the focus of much of this thesis. The so-called HR algebra is an algebra on graphs—that is, its elements are graphs—which has some deep connection to graph structure and, in particular, tree-width.

The HR algebra was first explored as a grammar on hypergraphs, where graphs are rewritten by replacing hyperedges with larger hypergraphs. This mechanism is still somewhat visible in the parallel composition operation, but the presentation as an abstract algebra is preferred so as to gain access to the theoretical framework explored in the previous section.

5.1 Definition

Definition 5.1. Let $n \in \mathbb{N}$. Then $[n] = \{0, 1, \ldots, n\}$.

Definition 5.2. The *HR signature* HR consists of the following symbols (denoted symbol : arity):

$$\begin{aligned}
\emptyset &: 0 & \\
a &: 0 & a \in \mathbb{N} \\
a^\ell &: 0 & a \in \mathbb{N} \\
\overline{ab} &: 0 & a, b \in \mathbb{N}, a \neq b \\
\mathrm{fg}_a &: 1 & a \in \mathbb{N} \\
\mathrm{ren}_{a \leftrightarrow b} &: 1 & a, b \in \mathbb{N}, a \neq b \\
/\!/ &: 2 &
\end{aligned}$$

The *HR algebra* or the *algebra of graphs with sources*, $\mathbb{JS}$, is the HR-algebra over the set $\mathcal{JS}$ of all finite graphs with sources in $\mathbb{N}$ where we interpret the symbols as follows:

- $\emptyset$ the null graph with no vertices;
- a the graph containing a single a-source and no edges;
- a^ℓ the graph containing a single a-source and a loop on that vertex;
- $\overline{ab}$ the graph containing an a-source and a b-source and an edge between them;
- $\mathrm{fg}_a(G)$ the graph obtained from G by converting its a-source, if any, to an internal vertex (*forgetting* the a-source);
- $\mathrm{ren}_{a \leftrightarrow b}$ the graph obtained from G by converting its a-source, if any, into a b-source, and vice-versa (*renaming* the a- and b-sources); and
- $G_1 /\!/ G_2$ the *parallel composition* of G_1 and G_2: the graph obtained from the disjoint union $G_1 \oplus G_2$ by identifying each pair of sources with same label.

The notation $\mathcal{JS}$ originates from [2], where they used $\mathcal{J}$ to denote the collection of all multigraphs, and $\mathcal{G}$ to denote the collection of all simple graphs. Because we use $\mathcal{I}$ and $\mathcal{J}$ to denote ideals in various orderings, we use $\mathcal{G}$ to denote the set of all multigraphs. We have retained the use of the notations $\mathcal{JS}$ and $\mathbb{JS}$ for s-graphs and the HR algebra, however.

Note that we restrict ourselves here to s-graphs with labels in $\mathbb{N}$. This restriction is by no means necessary, and the algebra works perfectly fine if any other infinite set is chosen. For our purposes, however, it is slightly simpler to restrict ourselves to the natural numbers, and we lose no generality by doing so. We will use bold-face numerals (e.g. $\mathbf{0}$ rather than 0) to denote source labels.

Additionally, the operation $/\!/$ is associative, so the notation $G_1 /\!/ G_2 /\!/ G_3$ is unambiguous.

Example 5.3. An example of a parallel composition can be see in Figure 5.

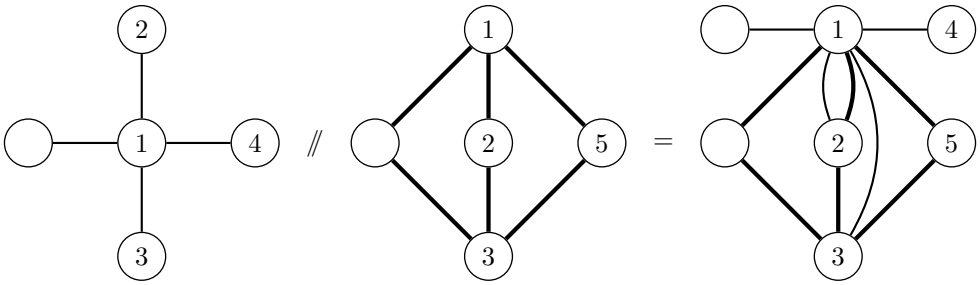

Figure 5: Example of parallel composition. The thin and thick edges correspond to edges of the first and second graphs, respectively.

Definition 5.4. For a positive integer n, we define:

1. HR^n is the signature consisting of those symbols in HR which include only natural numbers in $[n]$;
2. $\mathcal{JS}^n$ is the set of graphs with sources in $[n]$; and
3. $\mathbb{JS}^n$ is the HR^n-algebra over $\mathcal{JS}^n$, whose functions are the restrictions of the corresponding operations in $\mathbb{JS}$.

5.2 Relationship to Tree-width

The following theorem is a very important characterization of the HR algebra. It provides a complete description of the power of HR equations, and, while it is restricted in that it limits the applicability of the HR algebra to sets of graphs of bounded tree-width, it will later provide the basis for an assertion that, assuming we are willing to limit ourselves to graphs of bounded tree-width, the HR algebra is extremely rich and powerful.

Theorem 5.5 ([2])**.** *Let $G = (V, E)$ be a graph. Then $\mathrm{tw}(G) \leq k$ if and only if there is a term $t \in \mathrm{T}(\mathrm{HR}^k)$ such that $\mathrm{val}_{\mathbb{JS}^k}(t) \cong G$.*

Proof. First, suppose that $\mathrm{tw}(G) \leq k$. By Lemma 3.22, there is a nice tree-decomposition $T = ((W, E'), f)$ of G with width at most k. We will build a new tree-decomposition $T' = ((W', E''), f')$ from T, and use that to construct t.

For each $v \in V$ with a loop, we add to T a new vertex $b(v)$ and set $f'(b(v)) = \{v\}$. Likewise, for each pair of adjacent vertices u and v, we add a vertex $b(u, v)$ and set $f'(b(u, v)) = \{u, v\}$.

For each such added vertex $b(v)$ or $b(u, v)$, we pick some vertex w such that $v \in f(w)$ or $u, v \in f(w)$, as the case may be. We add a new vertex j in between w and its parent, with $j \equiv w$. Finally, we add a sequence of introduce nodes between $b(v)$ or $b(u, v)$ and j, such that j is a join node. Then T' is nice, except that the vertices $b(u, v)$ have two, rather than one, vertices in their bags.

For each $w \in W'$, we recursively define an HR^k-term $t(w)$, and a mapping $s_w : w \to [k]$, which tracks which vertices of G correspond to which sources, as follows. In order to understand what is going on here, recall that, for each vertex $v \in V(G)$, we know that it occurs in the bags of a contiguous subtree $T'[W(v)]$ of T'. Each leaf w of $T'[W(v)]$ is either a leaf of T' or a node that introduces v, and at each of these nodes, v will correspond to a source $s_w(v)$.

For each non-leaf node w of $T'[W(v)]$, it is either a forget or introduce node with child u, or a join node with children u_1 and u_2. If it is an introduce or forget node, we will have $s_w(v) = s_u(v)$ since v must occur in both bags. If it is an introduce node, then the $s_w(v)$-source in $t(w)$ will be obtained by identifying—through the permutation h below—the $s_{u_1}(v)$-source in $t(u_1)$ with the $s_{u_2}(v)$-source in $t(u_2)$. In this way, we keep track of v throughout all of $T'[W(v)]$.

We will use $/\!\!/^n(G)$ to mean $G /\!\!/ \ldots /\!\!/ G$ with n copies of G.

- If $w = b(v)$, then $t(w) = /\!\!/^n(\mathbf{0}^\ell)$ and $s_w(v) = 0$, where v has n loops.

- If $w = b(u, v)$, then $t(w) = /\!\!/^n(\overline{\mathbf{01}})$, $s_w(v) = 0$, and $s_w(u) = 1$, where there are n parallel edges between u and v.
- If w is a leaf with $f'(w) = \{v\}$, then $t(w) = \mathbf{0}$ and $s_w(v) = 0$.
- If w v to node u, then let m be distinct from $s_w(v')$ for any $v' \in f'(u)$. Then $t(w) = \mathbf{m} /\!\!/ t(u)$, $s_w(v) = m$, and $s_w(v') = s_u(v')$ for any $v' \in f'(u)$.
- If w forgets vertex v from node u, then $t(w) = \mathrm{fg}_{s_u(v)}(t(u))$ and s_w is the restriction of s_u to w.
- If w is a join node with children u_1 and u_2, then $t(w) = \mathrm{ren}_h(u_1) /\!\!/ u_2$, and $s_w(v) = s_{u_2}(v)$, where h is a permutation of $[k]$ such that $h(s_{u_1}(v)) = s_{u_2}(v)$ for any $v \in f'(w)$, and ren_h is a composition of rename operations (each of which operates as a single transposition) whose effect is to permute the labels as per h.

Claim 5.5.1. *For each $v \in V$, in the final product $\mathrm{val}_{\mathrm{HR}^k}(t(\mathrm{r}(T')))$, there is a single vertex corresponding to v.*

Proof. By construction, all nodes of T which have sources corresponding to v are eventually fused by parallel composition into one vertex at $r = \mathrm{r}(T'[W(v)])$.

If $r \neq \mathrm{r}(T')$, then r must have a parent p, and v is not in the bag $f'(w(p))$. So p must be a node that forgets v with corresponding term $t(p) = \mathrm{fg}_{s_r(v)}(t(r))$, so the source $s_r(v)$ corresponding to v becomes an internal vertex. Since the operations of $\mathbb{JS}$ leave internal vertices untouched, v will not be further changed.

Alternatively, $r = \mathrm{r}(T')$, and so v corresponds to the source $s_r(v)$ in $t(r)$. □

For each edge $e = uv$ in $E(G)$, we have a corresponding leaf $b(v)$ or $b(u, v)$. The term corresponding to $b(v)$ or $b(u, v)$ introduces e's parallel class into $t(w)$. Since there is only one such vertex for the loops of each vertex, and one bag for the edges between any given pair of vertices, this means that the edges of $\mathrm{val}_{\mathrm{HR}^k}(t(\mathrm{r}(T')))$ correspond exactly with the edges in G.

Thus $\mathrm{val}_{\mathrm{HR}^k}(t(\mathrm{r}(T'))) \cong G$.

Conversely, suppose that $G \cong \mathrm{val}_{\mathbb{JS}^k}(t)$. Then it is sufficient to prove that $G' = \mathrm{val}_{\mathbb{JS}^k}(t) = (V, E)$ has tree-width at most k.

We define a tree-decomposition $T = (W, E')$ of G' as follows. W is the set of all subterms of t. Each node u is adjacent to each of its immediate subterms, and contains the vertices which are sources of $\mathrm{val}_{\mathbb{JS}^k}(u)$. T is a tree by construction.

Each $v \in V(G)$ is obtained by identifying together a number of sources into a single vertex. Since internal vertices can never be fused together by a parallel composition, the terms where v occurs as a source must be connected in T. Every edge $e = uv$ of G must be introduced somewhere as a term $\overline{uv}$, and so there will be a bag containing both u and v. So T is in fact a tree-decomposition.

HR^k has only $k + 1$ source labels and there is at most one vertex with each label at a given stage. By construction, each bag contains only sources, so each bag contains at most $k + 1$ vertices. Thus T has width at most k and is our desired decomposition. □

Lemma 5.6. *Let E be an equational set in $\mathbb{JS}$. Then there exists k such that E is equational in $\mathbb{JS}^k$.*

Proof. E is defined by some finite HR equation system S. Let k be the largest label occurring in S. Then S is also an equation system in HR^k, and so E is equational in $\mathbb{JS}^k$. □

Corollary 5.7. *Let E be an equational set in $\mathbb{JS}$. Then E has bounded tree-width.*

Proof. Let k be such that E is equational in $\mathbb{JS}^k$. Then every graph in E can be expressed as a term in $\mathbb{JS}^k$, so it has tree-width at most k. □

6 Tree Generators

In this section, we present Christian et al.'s notion of tree generators and the labeled tree wqo result they obtained using this construction [1]. The definition presented here is not the original definition of Christian et al., but an alternate definition which makes the proofs easier to write and understand.

These results are largely reproductions of proofs in Christian et al.'s unpublished work, but during the process of rewriting them to the alternate definition, a defect was discovered. The present author, in correspondence with Richter and Salazar, identified a unstated assumption and has added the definition of a subtree-extending collection in order to make the assumption explicit and repair the defect.

6.1 Tree-Generators

Definition 6.1. Let $\Omega = (Q, \preceq)$ be a quasi-order. Let $\mathbb{N}^* = \mathbb{N} \cup \{\omega\} = \omega + 1$.

We define the set of Ω-*tree-generators* $\mathcal{TG}(\Omega)$ inductively as follows:

Set $\mathcal{TG}_0(\Omega) = \{\emptyset\}$.

Let $F_{i+1}(\Omega)$ be the set of functions $(\mathcal{TG}_i(\Omega) \cup \{\otimes\}) \setminus \{\emptyset\} \to \mathbb{N}^*$ which are nonzero on only finitely many points. In this case, $\otimes$ is a special symbol with no intrinsic meaning.

Then we define:

$$\mathcal{TG}_{i+1}(\Omega) = \mathcal{TG}_i \cup (\downarrow(\Omega) \times F_{i+1}(\Omega))^{(<\omega)}$$

where, as earlier, $S^{(<\omega)}$ means the finite subsets of S.

Then we take $\mathcal{TG}(\Omega) = \bigcup_{n\in\mathbb{N}} \mathcal{TG}_n(\Omega)$. If $G \in \mathcal{TG}(\Omega)$, we say that the *height* of G is the least k such that $G \in \mathcal{TG}_k(\Omega)$.

Given a tree-generator G, we define its set of *sub-tree-generators* (usually *sub-tgs*) $\mathfrak{TG}(G) = \{H : (Q, f) \in G, f(H) > 0\}$. If $G = \{(J_\alpha, f_\alpha) : \alpha \in I_G\}$ for some index set I_G, we also define $\mathfrak{TG}_\alpha(G) = \{H : f_\alpha(H) > 0\}$ to be a set of *local sub-tree-generators*.

The removal of the empty tree-generator $\{\emptyset\}$ in the definition of $F_{i+1}(\Omega)$ is to simplify the inductive properties somewhat, and without having to fudge over the empty tree generator. The inductive definition ensures that tree-generators are finite in nature.

The special symbol $\otimes$ is used for a tree-generator to refer to itself. This allows a tree-generator, which has a finite structure, to generate arbitrarily large trees without losing the ordering properties that we would like it to have. As we will see, this limitation to recursion is quite important. Note that as a result of $\otimes$, not every sub-tree-generator of G is itself a tree-generator, since $\otimes$ may be a sub-tree-generator.

Much like a grammar in language theory, a tree generator is used as a concise and structured description of a set of trees. We can generate trees using a generator in a fairly straightforward manner:

Example 6.2. Let Ω be a wqo and let J_a, J_b, J_c, and J_d be arbitrary ideals in Ω.

Let $G_1 = \{(J_a, f_a), (J_b, f_b)\}$, $G_2 = \{(J_c, f_c)\}$, and $G_3 = \{(J_d, f_d)\}$ as follows:

$$f_a(H) = \begin{cases} 1 & : H = \otimes \\ 1 & : H = G_2 \\ 0 & : \text{otherwise} \end{cases} \qquad f_b(H) = \begin{cases} \omega & : H = G_3 \\ 0 & : \text{otherwise} \end{cases}$$

$$f_c(H) = 0 \qquad f_d(H) = \begin{cases} 2 & : H = G_2 \\ 0 & : \text{otherwise} \end{cases}$$

To build a tree through G_1, we select either a or b, since $I_{G_1} = \{a, b\}$. We start with a root node labeled with J_a or J_b, as appropriate.

As the children of the root node, if we picked a, we will allow up to one child generated by G_1 and one child generated by G_2. We will always allow fewer children than are indicated—so our root could have no children, or only one child.

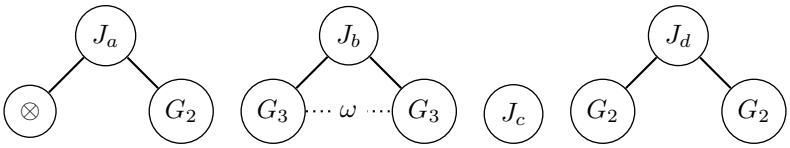

Figure 6: Four trees T_α, each corresponding to a pair (J_α, f_α). The root is labeled with J_α, and for each H, there are $f_\alpha(H)$ children of the root labeled H (note that, as $f_b(G_3) = \omega$, T_b is actually an infinite tree with ω children labeled G_3).

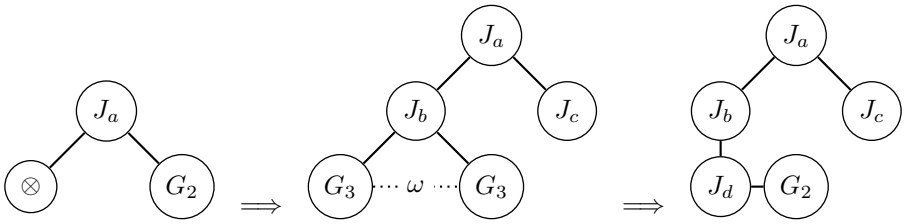

Figure 7: To generate a graph from a tree-generator G, pick T_α corresponding to some $(J_\alpha, f_\alpha) \in G$. Delete a selection of children of the root so that finitely many remain, and then for each remaining child with label H, replace that vertex with a tree generated by H (or by G if $H = \otimes$), so that the root of the tree inserted is adjacent to the root of the larger tree. Repeat this process until it terminates. In the diagram, the last step of replacing the remaining G_2 node with a J_c node has been omitted to preserve space.

If we picked b instead, then we could have any finite number of children—finite because we are dealing only with finite trees—each one generated by G_3.

We recurse into each child and repeat the process by choosing a member of I_{G_i}, labeling the vertex, and then generating its children. In this instance, a child generated by G_2 can have no children, and a child generated by G_3 can have up to two children generated by G_2.

This process could go indefinitely, by continually picking a and then adding a G_1 child, but again, we require a finite tree, so the process must halt in order to get a tree.

Figures 6 and 7 illustrate this graphically.

The formal definition is as follows:

Definition 6.3. Let $G = \{(J_\alpha, f_\alpha) : \alpha \in I_G\}$ be an Ω-tree generator.

We define the set $\iota(G)$, the set of trees *ideal-generated* by G, to contain all rooted trees T where there exists an $\alpha \in I_G$ and an injective function $g : \text{chldrn}(\text{r}(T)) \rightarrowtail \mathfrak{TG}(G)$ such that:

- the root r of T is labeled with J_α;
- for any $H \in \mathfrak{TG}(G)$, $|\{u \in \text{chldrn}(\text{r}(T)) : g(u) = H\}| \leq f_\alpha(H)$; and
- for each $u \in \text{chldrn}(\text{r}(T))$, $\text{br}(u) \in \iota'(g(u))$, where we say that $\iota'(\otimes) = \iota(G)$, and otherwise $\iota'(H) = \iota(H)$.

We say that T is *generated via* (J_α, f_α).

As a special case, we define ι to be empty on the empty tree-generator; that is, $\iota(\{\emptyset\}) = \emptyset$

We also define the set $\tau(G)$, the set of trees *generated* by G, to contain all trees T such that T is obtained from some $T' \in \iota(G)$ by replacing each node label with one of its elements.

Note that:

- the recursive definition that results from the use of $\iota'(\otimes)$ is still well-defined, as it could be defined inductively on the size of T;

- when $f_\alpha(H) = \omega$, then we are saying that there can be an unbounded number of subtrees generated by H;
- while the condition $\#H \leq f_\alpha(H)$ would allow for infinite subtrees in this case, we have restricted our definition to finite graphs;
- the trees in $\iota(G)$ and $\tau(G)$ are fully labeled from $\downarrow(\Omega)$ and Ω, respectively;
- the empty tree-generator always ideal-generates (and hence generates) the empty set, since we cannot label the root; and
- the empty tree is generated by every non-empty tree-generator.

The notion of $\iota(G)$ was not present in Christian et al.'s original work on tree generators, but it will be useful later.

6.2 Ordering Tree-Generators

The natural order on tree generators is by inclusion of the generated sets. We also define a structural property on collections of tree-generators which allows us to achieve our ordering results. This property was conceived of after discussion with Richter and Salazar to repair a defect in their original proof of Lemma 6.6, which attempted to prove 6.6 for all tree-generators.

Definition 6.4. Let Ω be a quasi-order. If G_1 and G_2 are Ω-tree generators, then $G_1 \preceq_\tau G_2$ if and only if $\tau(G_1) \subseteq \tau(G_2)$.

Definition 6.5. Let Ω be a quasi-order and let G be an Ω-tree generator. We say that G is *subtree-generating* if, for every $H \in \mathfrak{TG}(G) \setminus \otimes$, $\tau(H) \subseteq \tau(G)$ and, inductively, H is subtree-generating.

Let $\mathcal{T}$ be a collection of Ω-tree generators. We say that $\mathcal{T}$ is *subtree-extending* if, for every $G \in \mathcal{T}$, G is subtree-generating and, if H is a sub-tg of G, then $H \in \mathcal{T}$.

Recall that if T and T' are rooted, labeled trees, then $T \preceq_{\mathrm{Lr}} T'$ means that there exists a label-respecting rooted topological collapse from T' to T (see Definition 3.18).

Lemma 6.6. *Let $\Omega = (\preceq, S)$ be a wqo such that $\mathcal{P}^1(\Omega)$ is also wqo, and let $\mathcal{T}$ be a subtree-extending collection of Ω-tree-generators such that, for every $G \in \mathcal{T}$, $\tau(G)$ is downwards-closed under $\preceq_{\mathrm{Lr}}$. Then $(\mathcal{T}, \preceq_\tau)$ is a wqo.*

Proof. First, let $G, H \in \mathcal{T}$. We will say that $H \preceq_{\mathrm{p}} G$ if $H \preceq_\tau G$ and H is of equal or lesser height than G. By definition, $\preceq_{\mathrm{p}}$ is a partial ranking of $\mathcal{T}$ with respect to $\preceq_\tau$. Moreover, it has no infinite descending sequences since such a sequence would have infinite descending height, and tree-generator height is always finite and positive.

Suppose that $(G_i)_{i=1}^\infty$ is a bad sequence in $(\mathcal{T}, \preceq_\tau)$. We may assume that this sequence is minimal bad with respect to $\preceq_{\mathrm{p}}$ by Theorem 2.14.

Define $\mathcal{H} = \bigcup_{i=1}^\infty \mathfrak{TG}(G_i) \setminus \{\otimes\}$ to be the set of all sub-tgs of the tree generators G_i, less the special symbol $\otimes$.

Claim 6.6.1. *$(\mathcal{H}, \preceq_\tau)$ is wqo.*

Proof. We know that, as $\mathcal{T}$ is subtree-extending, $\mathcal{H} \subseteq \mathcal{T}$. Let us suppose that there exists a bad sequence $(H_i)_{i=1}^\infty$ in $(\mathcal{H}, \preceq_\tau)$. Inductively, we will create a subsequence $(H_{i_j})_{j=1}^\infty$ of H such that $(H_{i_j})_{j=1}^\infty \preceq_{\mathrm{p}}^* (G_i)_{i=1}^\infty$, contradicting the assumption that $(G_i)_{i=1}^\infty$ is minimal.

Set $i_1 = 1$, and let k_1 be smallest such that H_{i_1} is a sub-tg of G_{k_1}. Then for $j \geq 1$, pick i_{j+1} and k_{j+1} such that $i_{j+1} > i_j$, $k_{j+1} > k_j$, and $H_{i_{j+1}}$ is a sub-tg of $G_{k_{j+1}}$. This is possible because every H_i is a sub-tg of some G_k, and at each step the requirement that we move further along the sequences excludes only finitely many choices.

Then for each j, since H_{i_j} is a sub-tg of G_{k_j}, it must have lesser height. Moveover, since $\mathcal{T}$ is subtree-extending, we must have that $H_{i_j} \preceq_\tau G_{k_j}$. It follows that $H_{i_j} \prec_{\mathrm{p}} G_{k_j}$, so $(H_{i_j})_{j=1}^\infty$ is the desired subsequence. □

Define F to be the set of functions $\mathcal{H} \cup \{\otimes\} \to \mathbb{N}^*$ that are nonzero on a finite set (compare to the definition of $F_i(\Omega)$ in Definition 6.1).

Each tree-generator G_i is naturally identified with an element of $(\downarrow(\Omega) \times F)^{(<\omega)}$. We do this by identifying each function $f : (\mathcal{TG}_j(\Omega) \cup \{\otimes\}) \to \mathbb{N}^*$ occuring in G_i with the corresponding function $f' \in F$ where $f'(H) = f(H)$ if this is well-defined and $f'(H) = 0$ otherwise.If $f(H)$ is nonzero, then $H \in \mathcal{H} \cup \{\otimes\}$, so $f'(H)$ is well-defined. All we have really done is changed the domain of the functions by adding and removing zeroes.

By showing that $(\downarrow(\Omega) \times F)^{(<\omega)}$ is a well-quasi-order, we will contradict the assumption that $(G_i)_{i=1}^{\infty}$ is bad.

Given $f, g \in F$, we write that $f \preceq_F g$ if, for any $H \in \mathcal{H} \cup \{\otimes\}$, there exists $H' \in \mathcal{H} \cup \{\otimes\}$ such that $H \preceq'_\tau H'$ and $f(H) \leq g(H')$, where $\otimes \preceq'_\tau \otimes$ and otherwise $H \preceq'_\tau H'$ if and only if $H \preceq_\tau H'$.

Recall that one definition of a function $f : A \to B$ is as a subset of $A \times B$ where, if $(a_1, b_1), (a_2, b_2) \in f$, then $a_1 = a_2$ implies $b_1 = b_2$, and the a_is are unique. Using this definition, we can consider a function $f \in F$ to be a subset of $(\mathcal{H} \cup \{\otimes\}) \times \mathbb{N}^*$. We will then define $Z(f) = \{(H, n) \in f : n > 0\}$.

This gives rise to an equivalent formulation of $\preceq_F$. We can write that $f \preceq_F g$ by viewing $((\mathcal{H} \cup \{\otimes\}), \preceq'_\tau)$ as one wqo, and $(\mathbb{N}^*, \leq)$ as another, and then using the usual orderings on Cartesian product (recall Definition 2.6) and subsets (the powerset ordering of Definition 2.8). It follows that $\preceq_F$ is wqo, since Cartesian product and finite subsets preserve wqo.

We similarly define $\preceq_G$ on $\mathcal{G}$ by using $\preceq_F$ and the orderings on Cartesian product and subsets. Since $\preceq_F$ is wqo on F, $\preceq_G$ is wqo on $\mathcal{G}$.

It remains to be seen that if $G_i \preceq_G G_j$, then $\tau(G_i) \subseteq \tau(G_j)$. If $G_i = \{(J_\alpha, f_\alpha) : \alpha \in I_i\}$ and $G_j = \{(J_\beta, g_\beta) : \beta \in I_j\}$, then for any tree T generated by G_i via (J_α, f_α), by the definition of $\preceq_G$, there must be (J_β, g_β) such that $J_\alpha \subseteq J_\beta$ and $f_\alpha \preceq_F g_\beta$. $\mathscr{L}(\mathrm{r}(T)) \in J_\alpha$. It follows that $\mathscr{L}(\mathrm{r}(T)) \in J_\beta$. Moreover, since $f_\alpha \preceq_F g_\beta$, if a sub-tg H is used k times to generate k branches of T in G_i, then surely we can also use H at least k times to generate those same branches in G_j. Thus G_j generates T, completing the proof. □

6.3 Ideals of Labeled Trees

The next few lemmas will show us that we can represent every ideal of labeled trees with some tree-generator, which is critical to developing the well-quasi-ordering argument.

Lemma 6.7 (Christian, Richter, & Salazar)**.** *Let Ω be a quasi-order and G and H Ω-tree generators. Then there is an Ω-tree generator $G \cap H$ such that $\tau(G \cap H) = \tau(G) \cap \tau(H)$.*

Moreover, if G and H are subtree-generating, so is $G \cap H$.

Proof. We proceed by strong induction on the sum of the heights of G and H. If G or H is empty, then $G \cap H$ is empty too and we are done.

Suppose that $G = \{(J_\alpha, g_\alpha) : \alpha \in I_G\}$ and $H = \{(J_\beta, h_\beta) : \beta \in I_H\}$. We assume as the inductive hypothesis that the intersection is defined on $\mathfrak{TG}(G) \cup \mathfrak{TG}(H)$.

We define for the sake of this that $G' \cap \otimes = G' \cap H$, $\otimes \cap H' = G \cap H'$, and $\otimes \cap \otimes = \otimes$. If we define that $\mathcal{TG}(\otimes) = \mathcal{TG}(G)$ when it occurs on the left-hand side of an intersection and $\mathcal{TG}(\otimes) = \mathcal{TG}(H)$ when it occurs on the right-hand-side, this extends the inductive hypothesis to include $\otimes$.

Let $\alpha \in I_G$ and $\beta \in I_H$. We define $\mathcal{K}_{\alpha,\beta}$ to be the set of all bipartite graphs with bipartition $(\mathfrak{TG}_\alpha(G), \mathfrak{TG}_\beta(H))$ and where, for all $K \in \mathcal{K}_{\alpha,\beta}$:

- for every $G' \in \mathfrak{TG}_\alpha(G)$, $\deg_K(G') \leq g_\alpha(G')$;
- for every $H' \in \mathfrak{TG}_\beta(H)$, $\deg_K(H') \leq h_\beta(G')$; and
- if $g_\alpha(G') = h_\beta(H') = \omega$, there are no edges between G' and H'.

Note that multiple edges are permitted.

Then for $K \in \mathcal{K}_{\alpha,\beta}$, $G' \in \mathfrak{TG}_\alpha(G)$, and $H' \in \mathfrak{TG}_\beta(H)$, we define $\#(K, G', H')$ to be the number of edges between G' and H' in K. We then create a function $f_K : \mathcal{TG} \to \mathbb{N}^*$ by setting $f_K(\Gamma)$ as follows:

- if there exist G' and H' such that $g_\alpha(G') = h_\beta(H') = \omega$ and $\Gamma = G' \cap H'$, then $f_K(\Gamma) = \omega$;

- otherwise, $f_K(\Gamma) = \sum_{(G',H')} \#(K, G', H')$, where the sum ranges over all pairs of $G' \in \mathfrak{TG}_\alpha(G)$ and $H' \in \mathfrak{TG}_\beta(H)$ such that $G' \cap H' = \Gamma$; and
- otherwise, $f_K(\Gamma) = 0$.

The effect of K and f_K is to encode the various ways of "pairing up" the sub-tgs of G and H. For a graph to be generated by $G \cap H$, each branch must be generated by $G' \cap H'$ for sub-tgs G' and H' of G and H, respectively. G' and H' can be used $g_\alpha(G')$ and $h_\beta(H')$ times. The structure of K handles the case where $g_\alpha(G')$ and $h_\beta(H')$ are finite by imposing a limit on the corresponding degree of G' or H', and then f_K is defined so as to respect this limit while also accounting for the ω case, where there can be an unlimited number of edges. The sum in the finite case arises because we have have multiple ways of writing $\Gamma = G' \cap H'$.

We then finally define:

$$G \cap H = \{(J_\alpha \cap J_\beta, f_K) : \alpha \in I_G, \beta \in I_H, K \in \mathcal{K}_{\alpha,\beta}\}$$

We now need to show that $\tau(G \cap H) = \tau(G) \cap \tau(H)$.

Suppose that some T is generated by G and by H. Then there are $\alpha \in I_G$ and $\beta \in I_H$ such that T is generated via (J_α, g_α) and via (J_β, h_β).

Let $v = \mathrm{r}(T)$. For each $u \in \mathrm{chldrn}(v)$, there are tree-generators $G_u \in \mathfrak{TG}_\alpha(G)$ and $H_u \in \mathfrak{TG}_\beta(H)$ such that $\mathrm{br}(u)$ is generated by G_u and H_u (or by G or H itself, if G_u or H_u is $\otimes$).

Let K be the bipartite graph with bipartition $(\mathfrak{TG}_\alpha(G), \mathfrak{TG}_\beta(H))$, where for each $u \in \mathrm{chldrn}(v)$, if $g_\alpha(G_u) \neq \omega$ or $h_\beta(H_u) \neq \omega$, we have a distinct edge between G_u and H_u in K. Since G_u is only used at most $g_\alpha(G_u)$ times, $\deg_K(G_u) \leq g_\alpha(G_u)$; similarly $\deg_K(H_u) \leq h_\beta(H_u)$. If $g_\alpha(G_u) = h_\beta(H_u) = \omega$, there is no edge between G_u and H_u in k. So $K \in \mathcal{K}_{\alpha,\beta}$. It follows that, T is generated via $(J_\alpha \cap J_\beta, f_K)$. So $T \in \tau(G \cap H)$, hence $\tau(G) \cap \tau(H) \subseteq \tau(G \cap H)$.

Conversely, suppose that T is generated by $G \cap H$. Then there is $\alpha \in I_G$, $\beta \in I_H$, and $K \in \mathcal{K}_{\alpha,\beta}$ such that T is generated via $(J_\alpha \cap J_\beta, K)$. Let $v = \mathrm{r}(T)$. Then for each child u of v, $\mathrm{br}(u)$ is generated by $G_u \cap H_u$, and by induction it is generated by G_u and by H_u.

The total number of $\mathrm{br}(u)$ that are generated by a given G_u is limited to $g_\alpha(G_u)$ by the definition of $\mathcal{K}_{\alpha,\beta}$, and similarly, the number generated by a given H_u are limited to $h_\beta(H_u)$. If there exist multiple distinct possible G_u and/or H_u with the same intersection, then we can arbitrarily assign different G_u and H_u to each vertex in a way that ensures that the limits are respected, because of the sum in the definition of $f_K(G_u \cap H_u)$.

It follows immediately that T is generated via both (J_α, g_α) and (J_β, h_β), and hence $T \in \tau(G) \cap \tau(H)$. Thus $\tau(G \cap H) \subseteq \tau(G) \cap \tau(H)$ and we have equality.

Finally, as every sub-tree-generator of $G \cap H$ other than $\otimes$ is of the form $G' \cap H'$ for $G' \in \mathfrak{TG}(G)$ and $H' \in \mathfrak{TG}(H)$, it follows by induction that $G \cap H$ is subtree-generating. □

Definition 6.8. We denote by $\mathcal{LRT}(Q)$ the set of rooted trees fully labeled from Q.

Lemma 6.9 (Christian, Richter, & Salazar)**.** *Let $\Omega = (Q, \preceq)$ be a wqo. Then for any rooted tree T fully labeled from Q by $\mathscr{L}$, there is a subtree-generating Ω-tree generator G_T such that $\tau(G_T) = \mathrm{forb}_{\preceq_{\mathrm{Lr}}}(\{T\})$.*

Proof. If we can define G_T, it will necessarily be subtree-generating, since if S is a subtree of T, then $S \preceq_{\mathrm{Lr}} T$. Hence G_T will generate all graphs generated by G_S.

We proceed by induction on $|V(T)|$. If T is empty, then its forbidden set—which is empty—is generated by the empty tree-generator.

Otherwise, let $q = \mathscr{L}(\mathrm{r}(T))$ and $Q' = \mathrm{forb}_\preceq(q)$. Let the children of $\mathrm{r}(T)$ be $x_1, \ldots, x_k$, and define $T_i = \mathrm{br}(x_i)$.

By induction, there exists Ω-tree-generators G_{T_i} such that $\tau(G_{T_i}) = \mathrm{forb}_{\preceq_{\mathrm{Lr}}}(\{T_i\})$.

For each $K \subseteq [k]$ nonempty, define f_K by setting:

$$f_K(\otimes) = |K| - 1$$
$$f_K\left(\bigcap_{i \in K} G_{T_i}\right) = \omega$$

and f_K is zero everywhere else.

Additionally, we define $f_\otimes(\otimes) = \omega$ and $f_\otimes$ is zero everywhere else.

Then we define:

$$G_T = \{(Q', f_\otimes)\} \cup \{(Q, f_K) : K \subseteq [k], K \neq \emptyset\}$$

Claim 6.9.1. *Let $S \in \tau(G_T)$ and let $v \in V(S)$ be such that $q \preceq \mathscr{L}(v)$. Suppose there is a label-respecting rooted topological collapse $\eta = (\eta_V, \eta_E)$ of S to T such that $\eta_V(\mathrm{r}(T)) = v$. Then $\mathrm{br}(v) \in \tau(G_T)$.*

Proof. If $v = \mathrm{r}(S)$, then $\mathrm{br}(v) = S \in \tau(G_T)$.

Otherwise, let u be the child of $\mathrm{r}(S)$ such that $v \in V(\mathrm{br}(u))$. Since $\eta_V(\mathrm{r}(T)) = v$, the range of η_V must be entirely in $\mathrm{br}(v)$. Thus, for each i, we have $T_i \preceq_{\mathrm{Lr}} T \preceq_{\mathrm{Lr}} \mathrm{br}(v) \preceq_{\mathrm{Lr}} \mathrm{br}(u)$, the first and last relations coming because T_i is a subgraph of T and $\mathrm{br}(v)$ is a subgraph of $\mathrm{br}(u)$, respectively.

It follows that v cannot be in a subtree generated by G_{T_i} for any i, and hence not in $\bigcap_{i \in K} G_{T_i}$ for any $K \subseteq [k]$. So $\mathrm{br}(u)$ is generated via $\otimes$, and hence $\mathrm{br}(u) \in \tau(G_T)$. η is also a label-respecting rooted topological collapse of $\mathrm{br}(u)$ to T, so by induction on the distance from v to u, $\mathrm{br}(v) \in \tau(G_T)$. □

Claim 6.9.2. *Let $S \in \tau(G_T)$ such that $q \preceq \mathscr{L}(\mathrm{r}(S))$. Then there does not exist a label-respecting rooted topological collapse $\eta = (\eta_V, \eta_E)$ of S to T such that $\eta_V(\mathrm{r}(T)) = \mathrm{r}(S)$.*

Proof. Since, by assumption, $\mathscr{L}(\mathrm{r}(S)) \notin Q'$, S cannot be generated via $(Q', f_\otimes)$. If we look at the branches of the children of $\mathrm{r}(S)$, it follows that there is some $K \subseteq [k]$ nonempty such that at most $|K| - 1$ of them are generated via $\otimes$, and the rest are generated via $\bigcap_{i \in K} G_{T_i}$.

Consider the $|K|$ trees $\{T_i : i \in K\}$. Each of them must by mapped by η into the branch of a different child of $\mathrm{r}(K)$. But at most $|K| - 1$ of them are mapped to trees generated by $\otimes$, so at least one is mapped into a tree generated by $\bigcap_{i \in K} G_{T_i}$. This contradicts the inductive hypothesis that $\tau(G_{T_i}) = \mathrm{forb}_{\preceq_{\mathrm{Lr}}}(\{T_i\})$. □

Since, in any label-respecting rooted topological collapse $\eta = (\eta_V, \eta_E)$ of some $S \in \tau(G_T)$ to T, it would have to be the case that $q \preceq \mathscr{L}(\eta_V(\mathrm{r}(T)))$, it follows from the claims that no such collapse exists. Thus it is always the case that $T \not\preceq_{\mathrm{Lr}} S$, and hence $\tau(G_T) \subseteq \mathrm{forb}_{\preceq_{\mathrm{Lr}}}(\{T\})$.

Conversely, let $S \in \mathrm{forb}_{\preceq_{\mathrm{Lr}}}(\{T\})$. We will show that G_T generates S by induction on the number of vertices in S.

If S has one vertex v, then either T has two or more vertices, in which case G_T generates S via (Q, f_K) for any K, or T has a single vertex labeled with q, and so $q \not\preceq \mathscr{L}(v)$, giving that G_T generates S via $(Q', f_\otimes)$.

When S has more than one vertex, take $v = \mathrm{r}(S)$, $u_1, \ldots, u_m$ to be the children of v, and $S_i = \mathrm{br}(u_i)$. Note that $S_i \in \mathrm{forb}_{\preceq_{\mathrm{Lr}}}(\{T\})$, as $S_i \preceq_{\mathrm{Lr}} S$, and so by induction $S_i \in \tau(G_T)$. If $q \not\preceq \mathscr{L}(v)$, then S is generated by the $[Q', (\otimes^*)]$ branch of G_T. So we may assume that $q \preceq \mathscr{L}(v)$.

Let H be the bipartite graph with vertex set $\{T_1, \ldots, T_k\} \cup \{S_1, \ldots, S_n\}$ where T_i and S_j are adjacent if and only if $T_i \preceq_{\mathrm{Lr}} S_j$. If H had a matching saturating the vertices T_i, then we could take $\eta_j = (\eta_{V,j}, \eta_{E,j})$ to be the label-respecting rooted topological collapse of T_i to S_j, and then define $\eta = (\eta_V, \eta_E)$ as follows:

$$\eta_V(w) = \begin{cases} v & : w = \mathrm{r}(T) \\ \eta_{V,j}(w) & : w \in V(T_i) \end{cases}$$
$$\eta_E(e) = \begin{cases} P_j & : e = \mathrm{r}(T)x_j \\ \eta_{E,j}(e) & : e \in E(T_i) \end{cases}$$

where $x_1, \ldots, x_k$ are the children of $\mathrm{r}(T)$ (as above), and P_j is the path from $\mathrm{r}(S)$ to $\eta_{V,j}(\mathrm{r}(T_i))$ in S. Then η is a label-respecting rooted topological collapse of T to S, a contradiction.

In a beautiful application of Hall's bipartite matching theorem, there must thus exist some nonempty $K \subseteq [k]$ such that the vertices $\{T_i : i \in K\}$ of H have at most $|K| - 1$ distinct neighbours. This means that there are at most $|K| - 1$ different j such that $S_j \notin \mathrm{forb}_{\preceq_{\mathrm{Lr}}}(\{T_i : i \in K\})$.

For each such j, we can generate S_j by $\otimes$, and for every other j, we can generate S_j by $\bigcap_{i \in K} G_{T_i}$. We thus can generate S via (Q, f_K), and so $S \in \tau(G_T)$. It follows that $\mathrm{forb}_{\preceq_{\mathrm{Lr}}}(\{T\}) \subseteq \tau(G_T)$ and that the sets are therefore equal. □

This key corollary is essentially due to Christian, Richter, & Salazar, although their original statement did not require that $\mathcal{T}$ be subtree-extending and so was too weak to prove Theorem 6.11.

Corollary 6.10. *Let $\Omega = (Q, \preceq)$ be a wqo. Then there exists a subtree-extending collection of Ω-tree generators $\mathcal{T}$ such that $\{\tau(G) : G \in \mathcal{T}\} = {\downarrow}(\mathcal{LRT}(Q), \preceq_{\mathrm{Lr}})$.*

Proof. Let $\mathcal{I} \in {\downarrow}(\mathcal{LRT}(Q), \preceq_{\mathrm{Lr}})$. Take:

$$G_{\mathcal{I}} = \bigcap_{T \in \mathrm{obs}(\mathcal{I})} G_T$$

where G_T is the tree-generator found in Lemma 6.9

Clearly, $\tau(G_{\mathcal{I}}) = \mathcal{I}$. If we set $\mathcal{T} = \{G_{\mathcal{I}} : \mathcal{I} \in {\downarrow}(\mathcal{LRT}(Q), \preceq_{\mathrm{Lr}})\}$, then we just need to show that $\mathcal{T}$ is subtree-extending.

From the construction in Lemma 6.7, each proper sub-tg of $G \cap H$, other than $\otimes$, is $G' \cap H'$ for some sub-tgs G' of G and H' of H. Similarly, from the construction in Lemma 6.9, every sub-tg of G_T, other than $\otimes$, is of the form $\bigcap_{i \in K} G_{T_i}$. It follows that every proper sub-tg of $G_{\mathcal{I}}$ is going to be $G_{\mathcal{J}}$ for some $\mathcal{J} \in {\downarrow}(\Omega)$, since the intersection of two ideals is an ideal. Since all the trees are additionally subtree-generating, it follows that $G_{\mathcal{I}}$ is subtree-generating. □

Theorem 6.11 (Christian, Richter, & Salazar)**.** *Let $\Omega = (Q, \preceq)$ be a wqo such that $\mathcal{P}^1(\Omega)$ is also a wqo. Then $({\downarrow}(\mathcal{LRT}(Q), \preceq_{\mathrm{Lr}}), \subseteq)$ is a wqo.*

Proof. Let $\mathcal{T}$ be the collection of tree-generators from Corollary 6.10.

Suppose that $(\mathcal{I}_i)_{i=1}^{\infty}$ is a bad sequence of ideals in ${\downarrow}(\mathcal{LRT}(Q))$. We apply Lemma 6.6 to the tree generators $G_{\mathcal{I}_i}$ in $\mathcal{T}$, and find that there are $1 \leq i < j$ such that $G_{\mathcal{I}_i} \preceq_\tau G_{\mathcal{I}_j}$. But then, by definition, $\mathcal{I}_i = \tau(G_{\mathcal{I}_i}) \subseteq \tau(G_{\mathcal{I}_j}) = \mathcal{I}_j$, a contradiction.

So $({\downarrow}(\mathcal{LRT}(Q), \preceq_{\mathrm{Lr}}), \subseteq)$ has no bad sequences and is hence a wqo. □

7 Monadic Second-Order Logic

The other area of mathematics important for understanding our graph algebra and what we can do with it is that of formal logic, and that is what we explore in this section. While there exists a very large body of work by Courcelle and others exploring the intersection of graph theory and logic, our treatment here will be focused specifically on one type of logic—monadic second-order logic—and one particular way of modeling graphs using it.

Our treatment of monadic second-order logic is similar to the standard presentation of a formal logic, and echoes the way we defined algebras in Section 4.

7.1 The Logic

Our first step is to establish the definition of a relational signature, which is very similar in principle to the definition of a functional signature, with relations in place of functions. Much as an algebra is obtained by replacing the symbols with actual operations, a logic is obtained by replacing the symbols with actual relations.

The logic we use here is called *monadic second-order logic*, which is more expressive than first-order logic. In monadic second-order logic, there are, in addition to the first-order variables that range over individual elements, set variables which range over sets of elements. Quantifications are permitted over sets.

Monadic second-order logic is less expressive than second-order logic, which permits quantification over arbitrary relations (e.g. $\forall \sim .\exists x.\forall y.x \sim y$), but it has the advantage that the monadic second-order theories are less complicated, and thus less difficult to work with, than second-order theories.

Definition 7.1. A *relational signature* $\mathcal{R}$ is a finite set of symbols such that each symbol $R \in \mathcal{R}$ has an arity $\rho(R)$. We denote by $\mathcal{R}_i$ the set of symbols of arity i.

A symbol in $\mathcal{R}$ with arity 0 is a *constant symbol*; any other symbol is a *relation symbol*. We define the set of relation symbols $\mathcal{R}_+ = \mathcal{R} \setminus \mathcal{R}_0$, and denote by $\rho(\mathcal{R})$ the maximum arity of a symbol in $\mathcal{R}$.

Analagously to F-algebras, we define $\mathcal{R}$-structures.

Definition 7.2. Let $\mathcal{R}$ be a relational signature. An *$\mathcal{R}$-structure* S consists of an underlying set D, called the *domain*, together with interpretations of each of the symbols in $\mathcal{R}$.

For each constant symbol $c \in \mathcal{R}_0$, its interpretation c_S is a single element of D.

For each relation symbol $R \in \mathcal{R}_+$, its interpretation R_S is a relation on D with arity $\rho(R)$. In other words, R_S is simply a subset of $D^{\rho(R)}$.

We denote by $\mathrm{STR}(\mathcal{R})$ the collection of all $\mathcal{R}$-structures.

Exactly as with functional signatures and algebras, the relational structures provide different possible interpretations of a given signature. In this context, however, we are interested in logical formulas, using the relations of a signature to construct terms. As is typical, we will first define the formulas purely syntactically.

Definition 7.3. Let $\mathcal{R}$ be a relational signature.

Let $\bar{v}$ and $\bar{V}$ be countably infinite sets of variables, respectively the *first-order variables* and *set variables*. A *first-order term* is either some first-order variable $v \in \bar{v}$ or a constant symbol $c \in \mathcal{R}_0$.

The *atomic $\mathcal{R}$-formulas* are:

- True;
- False;
- $s = t$, where s and t are first-order terms;
- $R(t_1, \ldots, t_{\rho(R)})$, where R is a relation symbol in $\mathcal{R}_+$ and $t_1, \ldots, t_{\rho(R)}$ are first-order terms; and
- $t \in V$, where t is a first-order term and V is a set variable in $\bar{V}$.

Then the *$\mathcal{R}$-formulas* are recursively defined as:

- Any atomic $\mathcal{R}$-formula;
- $\neg\phi$, where ϕ is an $\mathcal{R}$-formula;
- $\phi \wedge \psi$, where ϕ and ψ are $\mathcal{R}$-formulas; and
- $\exists x.\phi$, where x is a variable in $\bar{v} \cup \bar{V}$ and ϕ is an $\mathcal{R}$-formula.

When a (first-order or set) variable x occurs in a formula ϕ, it is said to be a *bound ocurrence* if it occurs inside a quantification $\exists x.\psi$ of the same variable, and a *bound occurrence* otherwise. A *free variable* is a variable with at least one free occurrence. A formula ϕ is a *sentence* if it has no free variables, and is *quantifier-free* if it contains no quantifications $\exists x.\phi$.

We will use several common shorthands (and we may mix them in ways not explicitly described here):

- $x \notin X$ for $\neg(x \in X)$
- $X \subseteq Y$ for $\forall x.x \in X \Rightarrow x \in Y$
- $X \not\subseteq Y$ for $\neg(X \subseteq Y)$
- $\phi \vee \psi$ for $\neg(\neg\phi \wedge \neg\psi)$
- $\phi \Rightarrow \psi$ for $(\neg\phi) \vee \psi$
- $\phi \Leftrightarrow \psi$ for $(\phi \Rightarrow \psi) \wedge (\psi \Rightarrow \phi)$
- $\exists(x_1, \ldots, x_n).\phi$ for $\exists x_1.\exists x_2. \ldots . \exists x_n \phi$
- $\exists(x \in X).\phi$ for $\exists x.(x \in X \wedge \phi)$
- $\exists(X \subseteq Y).\phi$ for $\exists X.(X \subseteq Y \wedge \phi)$
- $\forall x.\phi$ for $\neg\exists x.\neg\phi$
- $\forall(x \in X).\phi$ for $\forall x.(x \in X \Rightarrow \phi)$
- $\forall(X \subseteq Y).\phi$ for $\forall X.(X \subseteq Y \Rightarrow \phi)$

By convention, the first-order variables will always be lowercase and the set variables will always be uppercase. When referring to a formula ϕ "in k variables", "in the variables $x_1, \ldots, x_n$", or the like, we mean to say that the variables of ϕ are at most k variables or are a subset of $x_1, \ldots, x_n$, or similar. In particular, ϕ does not need to use all of its variables. Finally, we will assume that, in all formulas, a variable never has both a bound and a free occurrence—if such a situation would arise, we can simply rename one of the bound occurrences to avoid such a collision.

Definition 7.4. Let S be an $\mathcal{R}$-structure and let ϕ be an $\mathcal{R}$-formula in the set variables $X_1, \ldots, X_n$, and first-order variables $x_1, \ldots, x_m$.

Given sets $E_1, \ldots, E_n \subseteq D$ and elements $d_1, \ldots, d_m \in D$, we write that:

$$S \models \phi(X_1/E_1, \ldots, X_n/E_n, x_1/d_1, \ldots, x_m/d_m)$$

if ϕ' is true, where ϕ' is the logical statement obtained from ϕ by replacing each free occurrence of a set variable X_i with the corresponding set E_i, replacing each free occurrence of a first-order variable x_j with the corresponding element d_j, and interpreting the relation symbols of $\mathcal{R}$ as in the structure S.

If ϕ is a sentence, and therefore $n = m = 0$, we may write $S \models \phi$ in place of $S \models \phi()$, and if this is the case we say that S *models* ϕ. We denote by $\mathrm{MOD}(\phi)$ all the models of ϕ.

As before, when we have established that $\phi(X_1, \ldots, X_n, x_1, \ldots, x_m)$ is a formula in the set variables $X_1, \ldots, X_n$ and first-order variables $x_1, \ldots, x_m$, we will write $S \models \phi(E_1, \ldots, E_n, d_1, \ldots, d_m)$ to mean $S \models \phi(X_1/E_1, \ldots, X_n/E_m, x_1/d_1, \ldots, x_m/d_m)$.

The final important logical concept is that of definability. This is how we associate a set or notion in the larger mathematics with a formal logical object.

Definition 7.5. Let $\mathcal{L}$ be some logical language (in this thesis, it will always be monadic second-order logic over a specific signature $\mathcal{R}$). Let $\mathcal{C}$ be a collection of $\mathcal{R}$-structures. We say that $\mathcal{C}$ is $\mathcal{L}$-*definable* if there exists a sentence ϕ in $\mathcal{L}$ such that $\mathrm{MOD}(\phi) = \mathcal{C}$.

We will often implicitly assume some universe of $\mathcal{R}$-structures, usually graphs of a certain nature. If $\mathcal{U}$ is our universe, then when considering the definability of a collection $\mathcal{C}$, we only really care that $\mathcal{C} = \mathrm{MOD}(\phi) \cap \mathcal{U}$.

We will occasionally abuse terminology and say that some property is definable to mean that the collection of all structures having that property is definable. For instance, we might say that being connected is definable when we mean that the collection of all connected graphs is definable.

7.2 Representing Graphs

Naturally enough, we wish to create relational signatures that define graphs. The most natural way to do so arises directly from one of the common ways of defining graphs: we use a signature consisting of a single binary relation symbol, $\mathrm{adj}(u, v)$, to mean that u and v are adjacent vertices.

This signature, unfortunately, has a critical limitation. We can indicate a loop with $\mathrm{adj}(u, u)$, but we cannot indicate multiple edges. Two vertices with a single edge between them are indistinguishable from two with a thousand edges between them. We also cannot create relations $\mathrm{adj}_1(u, v), \mathrm{adj}_2(u, v), \ldots$ to mean that there are n edges between u and v, because we may only have finitely many symbols.

The precise resolution of the issue with multigraphs is not of major importance, because all that matters is that we pick a signature for which the operations of $\mathbb{JS}$ are easily expressed by logic (in a sense that will be made formal in the next section). So we pick a very straightforward manner.

Definition 7.6. Let $\mathcal{R}^{\mathcal{G}}$ be the relational signature consisting of a single binary relation $\mathrm{in}(v, e)$. We will refer to the monadic second-order logic of all $\mathcal{R}^{\mathcal{G}}$-formulas as MS_2.

For any graph G, we define an $\mathcal{R}^{\mathcal{G}}$-structure $\lceil G \rceil$ whose domain is $V(G) \cup E(G)$, and where $\mathrm{in}(v, e)$ is true if and only if $v \in V(G)$, $e \in E(G)$, and v is incident to e.

Recall that an s-graph of type C is an s-graph whose sources are exactly the elements of C.

Definition 7.7. Let $C \subseteq \mathbb{N}$ be finite. We define $\mathcal{R}^{\mathcal{G}}_C$ to be the relational signature consisting of a single binary relation $\mathrm{in}(v, e)$, and, for each $c \in C$, a single constant symbol c.

For any s-graph G of type C, we augment the $\mathcal{R}^{\mathcal{G}}$-signature $\lceil G \rceil$ into an $\mathcal{R}^{\mathcal{G}}_C$-structure, also called $\lceil G \rceil$, where the constant $c_{\lceil G \rceil}$ is G's c-source.

Note that we cannot directly distinguish between an edge and a vertex in this logic. However, any edge is always incident to two vertices, so the formula $\exists v.\, \mathrm{in}(v, e)$ is true if and only if e is an edge. We can then define the formula $\phi_{\mathrm{adj}}(u, v)$ to mean $\exists e.\, \mathrm{in}(u, e) \wedge \mathrm{in}(v, e)$. We will also write $\phi_{\mathrm{e}}(e) = \exists v.\, \mathrm{in}(v, e)$ and $\phi_{\mathrm{v}}(v) = \neg\phi_{\mathrm{e}}(v)$.

If $\mathcal{G}$ is a collection of graphs, or a collection of s-graphs all of the same type, we will often implicitly identify $\mathcal{G}$ with the collection of structures $\{\lceil G \rceil : G \in \mathcal{G}\}$.

7.3 Definable Graph Properties

We now have a logical language, MS_2, which allows us to express properties of multigraphs. The full expressive power of this language may not, however, be immediately clear.

Example 7.8. For instance, suppose we want to express the property "there exists a path between u and v" in MS_2. The natural approach would be to try to define some sort of formula expressing the existence of vertices $u = v_0, v_1, \ldots, v_m = v$, with $v_{i-1} \sim v_i$.

Unfortunately, we can only use finitely many quantifications, and the length m of a uv-path in G could be unbounded. If we attempt to quantify $v_1, \ldots, v_{m-1}$ as first-order variables, then a single formula ϕ can only detect paths of length no longer than k, where k depends on ϕ. If we instead try to

use set quantifications, we run into the problem that we cannot distinguish between the vertices of our quantified set X in such a way as to get the ordering $v_1, \ldots, v_m$.

There is an alternative approach, however! We may instead rely on the basic fact that u and v have a path between them if and only if they lie in the same connected component.

We define:

$$\phi_{\text{sep}}(X) = \forall(x \in X, y \notin X).(\phi_{\text{v}}(x) \wedge \neg \operatorname{adj}(x, y))$$
$$\phi_{\text{conn}}(X) = \phi_{\text{sep}}(X) \wedge (\forall(Y \subseteq X).(Y = X \vee \neg\phi_{\text{sep}}(Y)))\phi_{\text{path}}(u, v) \quad = \exists X.(\phi_{\text{conn}}(X) \wedge u \in X \wedge v \in X)$$

It should be clear after a little thought that $\phi_{\text{sep}}(X)$ holds in G if and only if X is a *separation*—there are no edges between X and $V(G) \setminus X$. Then $\phi_{\text{conn}}(X)$ holds if and only if X is a separation, and it properly contains no separation. This is true if and only if X is a connected component. We then define $\phi_{\text{path}}(u, v)$ to hold if and if there exists a connected component containing u and v—equivalently if u and v are joined by a path.

We require in $\phi_{\text{sep}}(X)$ and $\phi_{\text{conn}}(X)$ that X be a set containing only vertices. We do not require this explicitly in $\phi_{\text{path}}(u, v)$, since $u, v \in X$ and X contains only vertices.

This example gives rise to the following:

Proposition 7.9. *The collection of all connected graphs is* MS_2*-definable.*

Proof. The collection is defined by the formula:

$$\forall X.\phi_{\text{conn}}(X) \Rightarrow (\forall x.(\phi_{\text{v}}(x) \Rightarrow x \in X))$$

which states that the only connected component of X contains all vertices. □

Given the subject-matter of this thesis, an astute reader may immediately leap to the question "Is $\preceq_{\text{m}}$ definable in MS_2?" This does not quite work, since $\preceq_{\text{m}}$ is not a collection of graphs.

We could, if we wanted, develop a new logic for pairs of graphs and attempt to determine if $\preceq_{\text{m}}$ is definable in that context. But there is an alternative approach of interest.

We shall fix a given graph H and ask the question "Is containing H as a minor definable?" In other words, given a graph H, we seek to know if there is a formula ϕ such that $\text{MOD}(\phi) = \{G : H \preceq_{\text{m}} G\}$ (note that the inverse question, "Is being a minor of H definable?" is trivially true as H has finitely many minors).

The answer, perhaps surprisingly, is yes. A fully technical proof won't be given here, but an English-language sketch of the logical formulas can be written quite easily.

Theorem 7.10. *Given a fixed graph H, there exists a formula ϕ_{m}^H such that $G \models \phi_{\text{m}}^H$ if and only if $H \preceq_{\text{m}} G$.*

Proof Sketch. Let $v_1, \ldots, v_n$ be the vertices of H, and $e_1, \ldots, e_m$ the edges. Then ϕ_{m}^H asserts the existence, in G, of disjoint sets $X_1, \ldots, X_n$ and distinct elements $f_1, \ldots, f_m$ such that, for all $1 \leq i \leq n$ and $1 \leq j \leq m$:

1. X_i is a connected set of vertices;
2. f_j is an edge; and
3. if $e_j = u_a u_b$, then there are $v \in X_a$ and $w \in X_b$ such that $f_j = vw$.

This is directly requiring that there exist a collapse of G to H, so ϕ_{m}^H will be true for G if and only if $H \preceq_{\text{m}} G$. □

We will later expand on this result to higher-order ideals of $\preceq_{\text{m}}$.

8 The Recognizability Theorem

The major result linking the monadic second-order logic to our HR algebra is very similar to the following statement

Pipe Dream 8.1 (Recognizability Theorem for the HR Algebra)**.** *Let D be a MS_2-definable set of s-graphs. Then D is recognizable in $\mathbb{JS}$.*

Sadly, this is not true. Instead, however, we will prove a slightly weakened form with a very similar algebra $\mathbb{JS}_\mathrm{t}$, defined later in this section.

Theorem 8.2 (Recognizability Theorem for the Sorted HR Algebra, [2])**.** *Let D be a MS_2-definable set of s-graphs. Then D is recognizable in $\mathbb{JS}_\mathrm{t}$.*

Corollary 8.3. *Let D be a MS_2-definable set of s-graphs. Then D is equational in $\mathbb{JS}$ if and only if it has bounded tree-width.*

Proof. If D is equational, then it is of bounded tree-width by Corollary 5.7.

If D is of bounded tree-width, there is some k such that $D \cap \mathcal{G}_{\mathrm{tw}\leq k} = D$. The set $\mathcal{G}_{\mathrm{tw}\leq k}$ is equational for all k, and D is recognizable by Theorem 8.2, so it follows by the Filtering Theorem (Theorem 4.21) that D is equational. □

The Recognizability Theorem does not hold for $\mathbb{JS}$ if we allow sets of unbounded tree-width. If, instead, we use the many-sorted variant of $\mathbb{JS}$ (which will be defined later in this section) we can remove this restriction. As was shown earlier, however, the requirement of bounded tree-width is necessary for equational sets. The upside to this is that, in the case of MS_2-definable sets, bounded tree-width is also sufficient for a set to be recognizable.

The proof of the Recognizability Theorem is long and technical, as presented in Section 5.3 of [2]. In this section, we will present a simplified sketch of the proof, leaving out many of the technical arguments. Rather than prove only the special case, however, most of the generality will be presented so as to give the reader an impression of the full power of the theorem. Throughout this section, if no proof is provided of a result or claim, the reader can find it in [2].

Two concessions will be made to avoid the complexity of a more general result. The first is that we will refer to a slightly weaker logical language than in the original presentation, which adds cardinality predicates. The proof itself is effectively unchanged in this regard. The second is that we will only prove the theorem with respect to a subalgebra of $\mathbb{STR}$. This subalgebra, $\mathbb{STR}_\mathrm{pres}$, is defined below. The more general statement of Theorem 8.2 can be proven by an additional step, but it adds even more technical work, and is any case not necessary for our application of the Recognizability Theorem to $\mathbb{JS}$.

The proof works by creating an algebra $\mathbb{STR}_\mathrm{pres}$ whose elements are finite relational structures. The constants will be single-element structures. Given any element $S \in \mathbb{STR}_\mathrm{pres}$, we can express S in terms of the constants and operations of S.

The large amount of technical work comes with showing that we can decompose a logical sentence about S into a series of equivalent sentences on each of the constants used to build S. From this, we can build a finite congruence on $\mathbb{STR}_\mathrm{pres}$ based on the equivalence of logical formulas. It follows that any definable set is saturated by this congruence, and such a set is recognizable. This will prove the general case of the theorem.

Since recognizability is preserved by taking subalgebras, to get the special case for the algebra $\mathbb{JS}$, it suffices to show that $\mathbb{JS}$ is in some sense a subalgebra of $\mathbb{STR}_\mathrm{pres}$. Other useful algebras arise as subalgebras of $\mathbb{STR}_\mathrm{pres}$ in a similar fashion, which shows the true power of the theorem.

We shall begin with a technical definition from logic, which will be used throughout this section.

Definition 8.4 (Informal)**.** Let ϕ be any logical formula. Then the *quantifier height* of ϕ is the greatest number of quantifiers appearing around a single atomic term.

For instance, the quantifier height of $(\exists x.x) \wedge (\forall y.y)$ is 1, and the quantifier height of $\exists x.(\forall y(x \vee y)) \wedge x$ is 2. Quantifier height is formally defined recursively over the structure of the formula.

8.1 Disjoint Union of Structures

There are two kinds of operations in the algebra $\mathbb{STR}_{\text{pres}}$. The first is the binary operations, all of which are expressed as taking the disjoint union of two relational structures.

Definition 8.5. The relational structures $\mathcal{R}$ and $\mathcal{R}'$ are *compatible* if:

1. they do not share any constant symbols, that is, $\mathcal{R}_0 \cap \mathcal{R}'_0 = \emptyset$;
2. they share all their relation symbols, that is, $\mathcal{R}_+ = \mathcal{R}'_+$; and
3. for every relation symbol $R \in \mathcal{R}_+$, the arity of R agrees in both signatures, that is, $\rho_{\mathcal{R}}(R) = \rho_{\mathcal{R}'}(R)$.

Let $\mathcal{R}$ and $\mathcal{R}'$ be compatible signatures. Let S be an $\mathcal{R}$-algebra over D, and let S' be an $\mathcal{R}'$ algebra over D'. We assume that D and D' are disjoint by possibly picking isomorphic algebras with disjoint domains.

We define the *disjoint union* of S and S' to be a new $(\mathcal{R} \cup \mathcal{R}')$-algebra $S \oplus S'$ over $D \cup D'$, where we interpret the symbols as in the original algebras:

1. for a constant symbol $c \in \mathcal{R}_0$, $c_{S \oplus S'} = c_S$;
2. for a constant symbol $c \in \mathcal{R}'_0$, $c_{S \oplus S'} = c_{S'}$; and
3. for a relation symbol $R \in \mathcal{R}_+$, $R_{S \oplus S'} = R_S \cup R_{S'}$.

In particular, note that elements of $S \cup S'$ are not related by any relation $R \in \mathcal{R}_+$ unless they are either all in S or all in S'.

The Splitting Theorem of [2], of which we present a special case, allows us to decompose any logical statement ϕ about a disjoint union $S \oplus T$ into one about S and one about T. Ideally this decomposition would not depend on S and T, but only on ϕ, but such a strong result does not hold generally. We can, however, limit ourselves to finitely many choices.

Theorem 8.6 (Special Case of the Splitting Theorem for Disjoint Union)**.** *Let $\mathcal{R}$ and $\mathcal{R}'$ be compatible relational signatures and let ϕ be a $(\mathcal{R} \cup \mathcal{R}')$-sentence. Then there exists $(\mathcal{R})$-sentences $\theta_1, \ldots, \theta_p$ and $(\mathcal{R}')$-sentences $\psi_1, \ldots, \psi_p$ such that, for all $\mathcal{R}$-signatures S and $\mathcal{R}'$-signatures T, $S \oplus T \models \phi$ if and only if there exists some i such that $S \models \theta_i$ and $T \models \psi_i$. Moreover, the formulas θ_i and ψ_i have quantifier height no greater than ϕ does.*

The proof of this theorem is technical and not particulary illuminating. It amounts to complex case analysis in an induction over the structure of ϕ.

8.2 Quantifier-Free Operations

The unary operations of the algebra $\mathbb{STR}_{\text{pres}}$ are defined in a fairly natural, if technical, fashion. The objective is to encode a mapping between structures in logical language. For technical reasons, we must restrict ourselves to the quantifier-free formulas.

This restriction to quantifier-free formulas is in some sense fundamental (quantifiers are, after all, what separates propositional logic from the much stronger first-order logic), and may seem at first blush to be overly limiting. In our application, however, we can still get the necessary mileage inside this constraint.

The actual definition we need is quite technical, as we need to jump through some hoops to express things correctly. The basic idea is to establish a *definition scheme*; a collection of logical formulas which describe the desired map.

Definition 8.7. Let $\mathcal{R}$ and $\mathcal{R}'$ be relational signatures.

A *quantifier-free operation definition scheme* (or *QFO definition scheme*, for short) $\mathcal{D}$ of type $\mathcal{R} \to \mathcal{R}'$ comprises the following quantifier-free $\mathcal{R}$-formulas:

1. a single formula $\delta(x)$ in a single first-order variable;
2. for each relation symbol $R \in \mathcal{R}'_+$, a formula $\theta_R(x_1, \ldots, x_{\rho(R)})$ in $\rho(R)$ first-order variables; and

3. for each constant symbol $c \in \mathcal{R}_0$ and each constant symbol $d \in \mathcal{R}'_0$, a sentence $\kappa_{c,d}$.

These formulas must satisfy the *constant mapping constraint*: for each $\mathcal{R}$-structure S and for each $d \in \mathcal{R}'_0$, there is exactly one $c \in \mathcal{R}_0$ such that $S \models \kappa_{c,d}$, and moreover, $S \models \delta(c)$.

Our goal is to use this definition scheme to define a map $\widehat{\mathcal{D}} : \mathrm{STR}(\mathcal{R}) \to \mathrm{STR}(\mathcal{R}')$ (recall that $\mathrm{STR}(\mathcal{R})$ is the collection of $\mathcal{R}$-structures), purely in terms of the logical formulas in $\mathcal{D}$.

Given an $\mathcal{R}$-structure S, the first step in evaluating $\widehat{\mathcal{D}}(S)$ is to determine the domain $D_{\widehat{\mathcal{D}}(S)}$. We shall require that it be a subset of D_S, to avoid introducing new elements with logical formulas. It is important, however, to allow for the possibility of $D_{\widehat{\mathcal{D}}(S)}$ being a proper subset of D_S.

To this end, the first formula is $\delta(x)$, used to define which elements of D_S are retained in $D_{\widehat{\mathcal{D}}(S)}$. For any $e \in D_S$, $e \in D_{\widehat{\mathcal{D}}(S)}$ if and only if $S \models \delta(e)$.

Next, for each relation symbol $R \in \mathcal{R}'_+$, we define its interpretation $R_{\widehat{\mathcal{D}}(S)}$ using the formula $\theta_R(x_1, \ldots, x_{\rho(R)})$: for $e_1, \ldots, e_{\rho(R)} \in D_{\widehat{\mathcal{D}}(S)}$, $R_{\widehat{\mathcal{D}}(S)}(e_1, \ldots, e_{\rho(R)})$ holds if and only if $S \models \theta_R(e_1, \ldots, e_{\rho(R)})$.

The technical difficulty comes in with the constant symbols. We must interpret each constant symbol $d \in \mathcal{R}'_0$ as a single element $d_{\widehat{\mathcal{D}}(S)} \in D_{\widehat{\mathcal{D}}(S)}$. Expressing it directly is not easy, so we will simplify the requirements a little by requiring that $d_{\widehat{\mathcal{D}}(S)}$ be a constant of S (though not necessarily d_S; it may be a different constant).

For every constant $c \in \mathcal{R}_0$ and $d \in \mathcal{R}'_0$, we have a sentence $\kappa_{c,d}$ which, when modeled by S, is taken to mean that $d_{\widehat{\mathcal{D}}(S)} = c_s$. For this to work properly, we need the constant mapping constraint defined above: the c such that $S \models \kappa_{c,d}$ is unique, and it must actually exist in $D_{\widehat{\mathcal{D}}(S)}$.

This interpretation of the definition scheme leads to the following formal definition:

Definition 8.8. Let $\mathcal{D}$ be a QFO definition scheme. The corresponding *quantifier-free operation* (or QFO) is a mapping $\widehat{\mathcal{D}} : \mathrm{STR}(\mathcal{R}) \to \mathrm{STR}(\mathcal{R}')$ defined by:

$$
\begin{aligned}
D_{\widehat{\mathcal{D}}(S)} &= \{e \in D_S : S \models \delta(e)\} \\
R_{\widehat{\mathcal{D}}(S)} &= \{(e_1, \ldots, e_{\rho(R)}) \in D_S^{\rho(R)} : S \models \theta_R(e_1, \ldots, e_{\rho(R)}) \wedge \delta(e_1) \wedge \ldots \wedge \delta(e_{\rho(R)})\} \\
d_{\widehat{\mathcal{D}}(S)} &= \text{the unique } c_S \text{ such that } S \models \kappa_{c,d}
\end{aligned}
$$

The full algebra $\mathbb{STR}$ includes all quantifier-free operations. It easier, however to work with a restricted class, giving the algebra $\mathbb{STR}_{\mathrm{pres}}$. As mentioned earlier, the Recognizability Theorem can be proven without this restriction, but the extra work to do so is beyond the scope of this thesis.

Definition 8.9. A QFO definition scheme $\mathcal{D} : \mathcal{R} \to \mathcal{R}'$ and its corresponding QFO $\widehat{\mathcal{D}}$ are *domain-preserving* if, for all $S \in \mathrm{STR}(\mathcal{R})$, $\widehat{\mathcal{D}}(S)$ has the same domain as S.

$\mathcal{D} : \mathcal{R} \to \mathcal{R}'$ and $\widehat{\mathcal{D}}$ are *quasi-domain-preserving* if, for all $S \in \mathrm{STR}(\mathcal{R})$, the domain of $\widehat{\mathcal{D}}(S)$ is obtained from that of S by possibly deleting constants, but no other elements.

We write *DP-QFO* and *QDP-QFO* as shorthand for "domain-preserving QFO" and "quasi-domain-preserving QFO", respectively.

Equivalently, a QFO definition scheme is domain-preserving if δ is equivalent to the formula True, and it is quasi-domain preserving if it is equivalent to some formula of the form $\bigwedge_{c \in \mathcal{R}_0}(\alpha_c \Rightarrow x \neq c)$, for a quantifier-free sentence α_c.

The next key theorem is similar to the Splitting Theorem, this time for QDP-QFO operations:

Theorem 8.10 (Special Case of the Backwards Translation Theorem)**.** *Let $\mathcal{D}$ be a QDP-QFO definition scheme of type $\mathcal{R} \to \mathcal{R}'$, and let ϕ be an $\mathcal{R}'$-sentence. Then there exists an $\mathcal{R}$-sentence ψ of quantifier height no greater than ϕ's such that, for all $S \in \mathrm{STR}(\mathcal{R})$, $S \models \psi$ if and only if $\widehat{\mathcal{D}}(S) \models \phi$.*

As with the Splitting Theorem, the proof of this theorem is mostly technical work and does not offer any great insight. It is somewhat simpler than the Splitting Theorem, though, since we produce only one formula ψ rather than a large number of formulas.

8.3 Many-Sorted Algebras

There is one problem with our approach towards the $\mathbb{STR}$ algebra so far. We wish to use QFOs as our operations, but a given QFO is not defined over all relational structures. So far, we have been avoiding this issue to simplify the definition of an algebra, but we cannot put this off any longer.

The basic idea is to partition the underlying set of the algebra into sets, each one identified as a *sort*. The domain of each function is limited so as to operate only between specified sorts, and similarly each constant is required to be of a given sort.

Definition 8.11. Let $\mathcal{Z}$ be an arbitrary index set. The elements of $\mathcal{Z}$ are called *sorts*.

A *functional $\mathcal{Z}$-signature* is a collection of function symbols where each symbol f has an arity $\rho(f)$, an *input type* $\alpha(f) \in \mathcal{Z}^{\rho(f)}$ and an *output type* $\sigma(f) \in \mathcal{Z}$. Constant symbols, having arity 0, receive the input type ().

An F-algebra $\mathbb{M}$ has, rather than a single domain M, one domain M_Z for each sort $Z \in \mathcal{Z}$. The domains must be pairwise disjoint.

For a function symbol f with input type $\alpha(f) = (Z_1, \ldots, Z_{\rho(f)})$, the corresponding interpretation $f_{\mathbb{M}}$ is, rather than a function $M^{\rho(f)} \to M$, a function from $M_{Z_1} \times M_{Z_2} \times \ldots \times M_{Z_{\rho(f)}} \to \sigma(f)$.

Most of the definitions and results already presented about single-sorted algebras are still applicable to many-sorted algebras, and we will not spend the time to go over the precise details of the generalizations here.

There is one particularly critical exception, however, that is used in the proof of the Recognizability Theorem:

Definition 8.12. Let F be a functional $\mathcal{Z}$-signature and let $\mathbb{A}$ be an F-algebra over domains A_Z (for $Z \in \mathcal{Z}$). $\mathbb{A}$ is *locally finite* if, for each sort $Z \in \mathcal{Z}$, the corresponding domain A_Z is finite.

Let $\mathbb{M}$ be an F-algebra and let $R \subseteq M$. We say that R is *recognizable* if there exists a locally finite F-algebra A and a homomorphism $h : \mathbb{M} \to \mathbb{A}$ (a homomorphism between many-sorted algebras must be sort-preserving) such that, for some $C \subseteq A$, $R = h^{-1}(C)$.

The critical difference between this definition and the definition for a single-sorted algebra is that in a many-sorted algebra we relax the restriction that $\mathbb{A}$ be finite to requiring that $\mathbb{A}$ be locally finite. On a single-sorted algebra, or indeed any algebra with only finitely many sorts, the notions coincide, so the distinction would have been academic in the earlier discussion.

The reason for this relaxation, which allows $\mathbb{A}$ to be infinite, is not immediately obvious nor is it easily explained. The technical reason is that this relaxation allows sufficient expressive power without removing the fact that, in the algebra $\mathbb{T}(F)$ discussed in the proof of the Filtering Theorem, the equational and recognizable sets coincide. The intuitive reason is that, given an element, we know its sort, and so when trying to identify members of R using the homomorphism h, we need only consider the members of C which share a sort. As long as there are only finitely many of those, we can still recognize elements of R.

We can and will do precisely the same loosening for the characterization of recognizability in terms of congruences:

Definition 8.13. Let F be a functional $\mathcal{Z}$-signature and let $\mathbb{M}$ be an F-algebra. Take $M = \bigcup_{Z \in \mathcal{Z}} M_Z$.

An equivalence relation $\sim$ on M is *sort-preserving* if $x \sim y$ implies that x and y have the same sort (that is, they are both in M_S for some $S \in \mathcal{Z}$). It follows that every equivalence class of $\sim$ has a single sort. $\sim$ is *locally finite* if there are only finitely many equivalence classes of each sort.

Proposition 8.14. *Let $\mathbb{M}$ be a many-sorted algebra. Then $\mathbb{M}$ is recognizable if and only if it is saturated by a locally finite congruence on $\mathbb{M}$.*

8.4 The Algebra of Relational Structures

We now have all of the pieces necessary to define our algebra $\mathbb{STR}$.

Definition 8.15. Let $\mathcal{Z}$ be the collection of all relational signatures. We define the $\mathcal{Z}$-sorted algebra $\mathbb{STR}$ as having the following properties:

1. For each relational signature $\mathcal{R} \in \mathcal{Z}$, the corresponding domain is $\mathrm{STR}(\mathcal{R})$.
2. For each QFO $\widehat{D}$ with type $\mathcal{R} \to \mathcal{R}'$, there is a unary function $\widehat{D}$ with type $\mathcal{R} \to \mathcal{R}'$.
3. For each pair of compatible signatures $\mathcal{R}$ and $\mathcal{R}'$, there is binary function $\uplus$ with type $\mathcal{R} \times \mathcal{R}' \to (\mathcal{R} \cup \mathcal{R}')$, defined as the disjoint union.
4. For each signature $\mathcal{R}$ with no constant symbols, a single constant $\emptyset_{\mathcal{R}}$ of type $\mathcal{R}$ denoting the empty structure.
5. For each signature $\mathcal{R}$ and set $\mathcal{B}$ with $\mathcal{R}_0 \subseteq \mathcal{B} \subseteq \mathcal{R}$, a constant $\diamond_{\mathcal{B},\mathcal{R}}$ denoting the structure with a single element $*$ where, for each constant $c \in \mathcal{R}_0$, $c_{\diamond_{\mathcal{B},\mathcal{R}}} = *$ and, for each relation $R \in \mathcal{R}_+$, $R(*, \ldots, *)$ holds if and only if $R \in \mathcal{B}$.
6. There are no other functions or constants.

We define $\mathbb{STR}_{\mathrm{pres}}$ to be the subalgebra of $\mathbb{STR}$ obtained by restricting the binary functions to the QDP-QFOs, rather than all QFOs.

8.5 Completing the Proof

To complete the proof of the Recognizability Theorem, we need a few more auxiliary definitions.

Definition 8.16. Let $h \in \mathbb{N}$. We define the equivalence relation $\sim_{\mathrm{L}}^{h}$, the *logical congruence of height* h, on $\mathbb{STR}$ by $S \sim_{\mathrm{L}}^{h} T$ if and only if there is a relational signature $\mathcal{R}$ such that S and T are both $\mathcal{R}$-structures and, for all $\mathcal{R}$-sentences ϕ of quantifier height at most h, $S \models \phi$ if and only if $T \models \phi$.

Lemma 8.17. *The relation $\sim_{\mathrm{L}}^{h}$ forms a congruence on $\mathbb{STR}_{\mathrm{pres}}$.*

Proof. First, let $\mathcal{R}$ be a relational signature and let S and S' be $\mathcal{R}$ structures such that $S \sim_{\mathrm{L}}^{h} S'$. Let $\widehat{D}$ be any QDP-QFO of type $\mathcal{R} \to \mathcal{R}'$.

Let ϕ be an $\mathcal{R}'$ formula of quantifier height at most h. Then by the Backwards Translation Theorem 8.10, there is a formula ψ of quantifier height at most h such that, for all $\mathcal{R}$-structures T, $T \models \psi$ if and only if $\widehat{D}(T) \models \phi$. $S \models \psi$ if and only if $S' \models \psi$, so $\widehat{D}(S) \models \phi$ if and only if $\widehat{D}(S') \models \phi$.

Hence $\widehat{D}$ respects $\sim_{\mathrm{L}}^{h}$.

The proof that the disjoint union respects $\sim_{\mathrm{L}}^{h}$ is done similarly, using the Splitting Theorem, but is more technical because the Splitting Theorem does not guarantee a unique pair of formulas θ and ψ. □

Theorem 8.18 (Recognizability Theorem for $\mathbb{STR}_{\mathrm{pres}}$)**.** *Let $\mathcal{R}$ be a relational signature, and let ϕ be a $\mathcal{R}$-formula. Then the set M of models of ϕ is recognizable in $\mathbb{STR}_{\mathrm{pres}}$.*

Proof. Let h be the quantifier height of ϕ.

By the lemma, $\sim_{\mathrm{L}}^{h}$ is a congruence on $\mathbb{STR}_{\mathrm{pres}}$.

$\sim_{\mathrm{L}}^{h}$ is locally finite because for any relational signature $\mathcal{R}'$, there are, up to logical equivalence, finitely many $\mathcal{R}'$-formulas. This is intuitively seen by considering that there are finitely many statements that can be expressed about h quantified variables and the finitely many constants of $\mathcal{R}'$.

Lastly, $\sim_{\mathrm{L}}^{h}$ saturates M because $S \sim_{\mathrm{L}}^{h} T$ implies that $S \models \phi$ if and only if $T \models \phi$.

It follows that $\sim_{\mathrm{L}}^{h}$ is a locally finite congruence on $\mathbb{STR}_{\mathrm{pres}}$ which saturates M, and hence M is recognizable. □

8.6 Application to the HR algebra

In order to get the Recognizability Theorem for the $\mathbb{JS}$ algebra, we only need to show that $\mathbb{JS}$ is a "subalgebra" of $\mathbb{STR}_{\text{pres}}$. In order to do so, the first step is to define the many-sorted version of $\mathbb{JS}$.

Definition 8.19. We define a many-sorted algebra $\mathbb{JS}_{\text{t}}$ by letting the set of sorts be the finite subsets of $\mathbb{N}$, and the sort of an s-graph $G \in \mathcal{JS}$ is its type (recall that G's type is the set of numbers which appear as source labels in G).

For each of the constant symbols a, $\overline{ab}$, and a^{ℓ}, the corresponding sorts are $\{a\}$, $\{a, b\}$, and $\{a\}$.

We replace the function fg_a with functions $\text{fg}_{a,C}$ for every finite $C \subseteq \mathbb{N}$ with $a \in C$. $\text{fg}_{a,C}$ is defined as the restriction of fg_a to the graphs of type C. The resulting graph always has type $C - a$.

Similarly, we replace the function $\text{ren}_{a\leftrightarrow b}$ with a function $\text{fg}_{a\leftrightarrow b,C}$ for every finite $C \subseteq \mathbb{N}$ with $a \in C$. We then define $\text{ren}_{a\leftrightarrow b,C}$ as the restriction of $\text{ren}_{a\leftrightarrow b}$ to the graphs of type C. The output type will be $(C - a) \cup \{b\}$.

Finally, we replace the function $/\!\!/$ with a function $/\!\!/_{C,D}$ for every finite $C, D \subseteq \mathbb{N}$, defined by the natural restriction to graphs of type C and type D. The resulting graph always has type $C \cup D$.

This is a very natural typing scheme, and it should be clear that we have not lost much expressive power. If we write out a graph as a $\mathbb{JS}_{\text{t}}$ term, then for each instance of fg_a or $\text{ren}_{a\leftrightarrow b}$ in the expression, we can either replace it with a corresponding sorted instance, or else remove it entirely because it is the identity function in that case.

The algebra $\mathbb{JS}_{\text{t}}$ is not a subalgebra of $\mathbb{STR}_{\text{pres}}$ in the traditional sense, because it does not have nearly the same richness of operations. We need define a more general notion in order to avoid this.

Definition 8.20. Let F be an $\mathcal{Z}$-signature and F' an $\mathcal{Z}'$-signature, where $\mathcal{Z}' \subseteq \mathcal{Z}$. Let $\mathbb{A}$ be an F-algebra and $\mathbb{B}$ an F'-algebra.

$\mathbb{B}$ is a *derived subalgebra* if:

1. for each $Z \in \mathcal{Z}'$, $B_Z \subseteq A_Z$;
2. for each operation $f_{\mathbb{B}}$ of $\mathbb{B}$ with arity ρ, there exists some F-term t in ρ variables such that $f_{\mathbb{B}}(x_1, \ldots, x_n) = \text{val}_{\mathbb{A}}(t)(x_1, \ldots, x_n)$ on the domain of $f_{\mathbb{B}}$.

In other words, we must have some way of composing the operations of $\mathbb{A}$ in order to make each of the operations of $\mathbb{B}$.

Example 8.21. Let $\mathbb{A}$ be the real numbers with constants 0 and 1, and two operations: addition, multiplication, and additive inverse.

Let $\mathbb{B}$ be the integers with constant 0 and two operations: subtraction and taking the square.

Then $\mathbb{B}$ is a derived subalgebra of $\mathbb{A}$. We can define subtraction in terms of the operations of $\mathbb{A}$ as $a - b = a + (-b)$. Similarly, we can define squaring as $x^2 = x \cdot x$. This is legal, because we allow a variable to appear more than once in the term t.

Recall that every s-graph G of type C corresponds to a $\mathcal{R}_C$-structure, where $\mathcal{R}_C$ is a relational signature containing the binary relation 'in' and one constant for every element of C. For the remainder of this section, we shall not draw a distinction between a (many-sorted) HR-term t, its value $G = \text{val}_{\mathbb{JS}_{\text{t}}}(t)$, and the relational structure $\lceil G \rceil$.

Theorem 8.22. *The algebra $\mathbb{JS}_{\text{t}}$ is a derived subalgebra of $\mathbb{STR}_{\text{pres}}$.*

Proof. Each of the constants of $\mathbb{JS}_{\text{t}}$ is a relational structure, and therefore in $\mathbb{STR}_{\text{pres}}$.

For the unary operation $\text{fg}_{a,C}$, we will create a corresponding QFO definition scheme $\mathcal{D}_{a,C}$ of type $\mathcal{R}_C \to \mathcal{R}_{C-a}$ (recall Definition 8.7), as follows, where c and d range over all constants of $\mathcal{R}_C$ and $\mathcal{R}_{C-a}$, respectively:

$$\delta(x) = \mathsf{True}$$

$$\theta_{\text{in}}(x_1, x_2) = \text{in}(x_1, x_2)$$

$$\kappa_{c,d} = \begin{cases} \mathsf{True} & : c = d \\ \mathsf{False} & : c \neq d \end{cases}$$

This definition may seem pointless, as it appears to do nothing at all. However, since a is not a constant symbol of $\mathcal{R}_{C-a}$, there are no formulas $\kappa_{c,a}$, and the effect of the corresponding operation $\widehat{D_{a,C}}$ is to drop the constant symbol a—precisely what we need to get $\mathrm{fg}_{a,C}$. Since $\delta(x) = \mathsf{True}$ for all x, $\widehat{D_{a,c}}$ is domain-preserving and hence quasi-domain-preserving. Note that $\delta(a) = \mathsf{True}$, because if $\delta(a)$ were False, then $\widehat{D_{a,c}}$ would remove a from the underlying set altogether, rather than only forgetting that it is a constant.

For the unary operation $\mathrm{ren}_{a\leftrightarrow b,C}$, we will similarly create a QFO definition scheme $\mathcal{D}_{a\leftrightarrow b,C}$ of type $\mathcal{R}_C \to \mathcal{R}_C$. The definition is as follows, where c and d range over all constants of $\mathcal{R}_C$ other than a and b:

$$\begin{aligned}
\delta(x) &= \mathsf{True} \\
\theta_{\mathrm{in}}(x_1, x_2) &= \mathrm{in}(x_1, x_2) \\
\kappa_{a,b} &= \kappa_{b,a} = \mathsf{True} \\
\kappa_{a,c} &= \kappa_{b,c} = \kappa_{c,a} = \kappa_{c,b} = \mathsf{False} \\
\kappa_{c,d} &= \begin{cases} \mathsf{True} & : c = d \\ \mathsf{False} & : c \neq d \end{cases}
\end{aligned}$$

The effect of $\widehat{D_{a\leftrightarrow b,C}}$ is to swap the constants a and b, exactly as desired, giving us the operation $\mathrm{ren}_{a\leftrightarrow b,C}$. It is again domain-preserving.

The final step is to define $G /\!\!/ H$ in terms of disjoint union and QDP-QFOs. Say that G has type C and H has type D. If C and D are not disjoint (which will often be the case, to make things interesting!), they are not compatible, so we cannot take the disjoint union as is.

To work around this, we pick some type $D' \subseteq \mathbb{N}$ with $|D'| = |D|$ such that D' and C are disjoint, and $(D' \cap D) \cup (C \cap D) = D$. This is the type of D after renaming the sources duplicated in C. Using applications of ren, we can turn H into an s-graph H' of type D'. If $a \in C \cap D$, we denote by a' the new label assigned to H's a-source in this relabeling. We will denote the composed renaming operations as ren_D, so that $H' = \mathrm{ren}_D(H)$.

Now that C and D' are disjoint, we can take the disjoint union $G' = G \uplus H'$. All that remains is to fuse a with a' for each $a \in C \cap D$.

To fuse a with a', we apply the QFO defined as follows:

$$\begin{aligned}
\delta(x) &= x \neq a' \\
\theta_{\mathrm{in}}(x_1, x_2) &= \mathrm{in}(x_1, x_2) \vee (x_1 = a \wedge \mathrm{in}(a', x_2) \\
\kappa_{c,d} &= \begin{cases} \mathsf{True} & : c = d \vee (c = a' \wedge d = a) \\ \mathsf{False} & : c \neq d \wedge (c \neq a' \vee d \neq a) \end{cases}
\end{aligned}$$

The effect of this operation is precisely to merge a and a' into a single vertex labeled with a, and have no other effects. Thus by fusing each $a \in C \cap D$ with a', we transform G' into $G /\!\!/ H$, using only QDP-QFOs and disjoint union. This shows that $\mathbb{JS}_{\mathrm{t}}$ is in fact a derived subalgebra of $\mathbb{STR}_{\mathrm{pres}}$. □

The following lemma is the final piece of the puzzle:

Lemma 8.23. *Let $\mathbb{N}$ be a derived subalgebra of $\mathbb{M}$. Then if L is recognizable in $\mathbb{M}$, it is recognizable in $\mathbb{N}$.*

Proof. Any congruence $\sim$ on $\mathbb{M}$ is a congruence on $\mathbb{N}$, since the operations of the latter are defined in terms of the former. Whether or not $\sim$ is locally finite and whether or not L is saturated by $\sim$ are independent of the choice of algebra. Thus a locally finite congruence on $\mathbb{M}$ which saturates L is also a locally finite congruence on $\mathbb{N}$ which saturates L. □

This completes the proof of the Recognizability Theorem for $\mathbb{JS}_{\mathrm{t}}$.

Theorem 8.2 (Recognizability Theorem for the Sorted HR Algebra, [2])**.** *Let D be a* MS_2*-definable set of s-graphs. Then D is recognizable in* $\mathbb{JS}_{\mathrm{t}}$*.*

Proof. By Theorem 8.22, $\mathbb{JS}_{\mathrm{t}}$ is a derived subalgebra of $\mathbb{STR}_{\mathrm{pres}}$. By Theorem 8.18, D is recognizable in $\mathbb{STR}_{\mathrm{pres}}$. The theorem follows from the lemma. □

9 Tree-Generators and HR Equation Systems

In our first true foray into new mathematics, we investigate the representation of tree-generators as a particular class of labeled HR equation systems. The ultimate goal would be to get an analog of Theorem 6.11 with applicability to general graph ideals. While this is promising, there is some theoretical machinery that still needs to be developed.

Once that technical machinery has been developed, we hope that it will be possible to extend the results of Section 6 to a more general class of tree-generators introduced here. It is clear that these results do not extend to HR-equational sets generally, and so there is a significant question of how far, exactly, we can extend the notion of a tree-generator until we lose those results. We present one concrete conjecture in this area and open the door to further investigation in the area.

9.1 HR Algebras for Labeled Graphs

First, in order to attach ideals to the vertices of graphs generated by HR equation systems, as is done with tree-generators, we need to extend the algebra $\mathbb{JS}$ to allow labeled graphs. Effectively, what we do is allow labels to be introduced on constants without affecting the structure of equation systems.

When we parallel-compose two graphs together and identify vertices, we need some way to combine the labels, so we require that they be a commutative monoid.

This definition is similar, but not identical to the one used by Courcelle and Engelfriet in [2] to handle labeled graphs: their definition did not allow for arbitrary commutative monoids and also accounted for edge labels. The choice of definitions is therefore mostly cosmetic.

Definition 9.1. Let $L = (S, \cdot)$ be a commutative monoid—that is, $\cdot$ is a commutative and associative binary operation on S with an identity element. Denote the identity element by e.

The *L-labeled HR signature* HR_L is the same as HR, except that each constant symbol a is replaced with the family of symbols $\{a_s : s \in S\}$.

The *L-labeled HR algebra* $\mathbb{JS}_L$ is an HR_L-algebra over the set $\mathcal{JS}_L$ of all s-graphs labeled from L. The graphs in the underlying set have two separate labelings: the source labeling, from $\mathbb{N}$, and the other labeling, from L. Unless otherwise specified, when referring to such a graph, the function $\mathscr{L}$ will refer to the label from L and not the source label.

The symbols are interpreted as in the unlabeled HR algebra $\mathbb{JS}$, with the following differences:

- a_s — the vertex introduced by this constant is labeled with s;
- a^ℓ — the vertex introduced by this constant is labeled with the identity element e;
- $\overline{ab}$ — the vertices introduced by this constant are labeled with e;
- $G_1 /\!\!/ G_2$ — when two vertices u and v are identified by the parallel composition to give w, we set $\mathscr{L}(w) = \mathscr{L}(v) \cdot \mathscr{L}(u)$, applying the operation of the commutative monoid.

We define $\mathbb{JS}_L^k$ analagously to $\mathbb{JS}^k$.

9.2 From Tree-Generators to HR

Proposition 9.2. *Let $\Omega = (Q, \preceq)$ be a quasi-order and let $G = \{(J_\alpha, f_\alpha) : \alpha \in I_G\}$ be an Ω-tree-generator. Then there exists a labeled $\mathbb{JS}_L$ equation system $\mathrm{EQ}(G)$ such that $\mathrm{L}(\mathrm{EQ}(G), S_1)$ (recall Definition 4.10) is equal to $\iota(G)$, if we consider the root of $T \in \iota(G)$ to be the $\mathbf{0}$-source of the corresponding graph of $\mathrm{L}(\mathrm{EQ}(G), S_1)$.*

Proof. We set $G_1 = G$, and let $G_2, \ldots, G_k$ be the sub-tgs of G other than $\otimes$. We will let our variable set be $\{S_1, \ldots, S_k\} \cup \{S_1^*, \ldots, S_k^*\} \cup \{S_1^?, \ldots, S_k^?\}$.

For $1 \leq i \leq k$, we will define HR^2 equations for S_i, S_i^*, and $S_i^?$ in the powerset algebra. S_i corresponds to a single graph generated by G_i, $S_i^?$ to zero or one such graphs, and S_i^* to any number of such graphs.

We can obtain the polynomials required for a valid equation system by applying Proposition 4.8. Suppose that $G_i = \{(J_\alpha, f_\alpha) : \alpha \in I_G\}$.

We define $\mathrm{EQ}(G)$ as the following equation system:

$$\begin{aligned} S_i &= \bigcup_{\alpha \in I_G} \left(\mathbf{0}_{J_\alpha} /\!\!/ S_i^{/\!\!/ f_\alpha(\otimes)} /\!\!/ \mathop{/\!\!/\!\!/}_{j \neq i} S_j^{/\!\!/ f_\alpha(G_j)} \right) \\ S_i^? &= \emptyset \cup \mathrm{fg}_{\mathbf{1}}(\mathrm{ren}_{\mathbf{0} \leftrightarrow \mathbf{1}}(S_i /\!\!/ \overline{\mathbf{01}})) \\ S_i^* &= \emptyset \cup (S_i^? /\!\!/ S_i^*) \end{aligned}$$

where for $1 \leq j \leq k$ (including when $j = i$), we define $S_j^{/\!\!/ \ell}$ to mean S_j^* if $\ell = \omega$, and $\overbrace{S_j^? /\!\!/ S_j^? /\!\!/ \ldots /\!\!/ S_j^?}^{\ell \text{ times}}$ otherwise.

Now we need to show that $\mathrm{EQ}(G)$ has the desired properties. First, suppose that $T \in \mathrm{L}(\mathrm{EQ}(G), S_1)$.

Then we wish to show that there exists some $\alpha \in I_G$ and an injective function $g : \mathrm{chldrn}(\mathrm{r}(T)) \rightarrowtail \mathfrak{TG}(G)$ satisfying the constraints in Definition 6.3.

However, we have the following equation in $\mathrm{EQ}(G)$:

$$S_1 = \bigcup_{\alpha \in I_G} \left(\mathbf{0}_{J_\alpha} /\!\!/ S_1^{/\!\!/ f_\alpha(\otimes)} /\!\!/ \mathop{/\!\!/\!\!/}_{j \neq 1} S_j^{/\!\!/ f_\alpha(G_j)} \right)$$

T will be generated by some element of the union, so set α such that T can be generated by:

$$\mathbf{0}_{J_\alpha} /\!\!/ S_1^{/\!\!/ f_\alpha(\otimes)} /\!\!/ \mathop{/\!\!/\!\!/}_{j \neq 1} S_j^{/\!\!/ f_\alpha(G_j)}$$

The root of T will be generated by $\mathbf{0}_{J_\alpha}$, satisfying the requirement that the root of r be labeled with J_α. If this is the only vertex of T, we are done. Otherwise, we can proceed inductively on the height of T.

Let $\mathrm{EQ}(G)$ be the equation system for G whose existence is guaranteed by Proposition 9.2. Let $G_1, \ldots, G_k$ be the sub-tgs of G other than $\otimes$, such that S_i is the equation in $\mathrm{EQ}(G)$ generating G_i.

Since every sub-tg of G_i is also a sub-tg of G, each equation in $\mathrm{EQ}(G_i)$ is also an equation of $\mathrm{EQ}(G)$, after renaming the indices. It follows that $\mathrm{L}(\mathrm{EQ}(G), S_i) = \mathrm{L}(\mathrm{EQ}(G_i), S_1)$. Thus, as our inductive hypothesis, if G_i is a proper sub-tg of G, then $\mathrm{L}(\mathrm{EQ}(G), S_i) = \iota(G_i)$.

By inspection, $\mathrm{L}(\mathrm{EQ}(G), S_i^?)$ contains exactly the graphs obtained from a graph of $\iota(G_i)$ by making the root and adding a new $\mathbf{0}$-source adjacent to only the root, plus the empty graph. Furthermore, $\mathrm{L}(\mathrm{EQ}(G), S_i^*)$ contains the parallel composition of any number of such graphs, such that the $\mathbf{0}$-source is adjacent to the root of any number of graphs of $\iota(G_i)$.

S_i is obtained by taking the parallel composition of $\mathbf{0}_{J_\alpha}$ with a number of terms of the form $S_j^{/\!\!/ k}$. Each child of the root is thus obtained from one of these terms. Thus, for any child u of $\mathrm{r}(T)$, define $g(u)$ to be the G_j such that u was obtained from the $S_j^{/\!\!/ k}$ term.

By construction, for any $H \in \mathfrak{TG}(G)$, $|\{u \in \mathrm{chldrn}(\mathrm{r}(T)) : g(u) = H\}| \leq f_\alpha(H)$. By the inductive hypothesis, for any $u \in \mathrm{chldrn}(\mathrm{r}(T))$, $\mathrm{br}(u) \in \iota'(g(u))$ (where ι' has the same meaning as in Definition 6.3). Thus $T \in \iota(G)$.

Conversely, if $T \in \iota(G)$, then let α be such that T is generated via (J_α, f_α), and let g be the function guaranteed by Definition 6.3. Then each child of $\mathrm{r}(T)$ will be generated in some sub-tg G_i of G. We perform the same induction, and so it is generated in $\mathrm{L}(\mathrm{EQ}(G_i), S_1)$. When we take $\{\mathrm{br}(u) : u \in \mathrm{chldrn}(\mathrm{r}(T))\}$ and join them all together with a single vertex, this gives us T, and we only need the S_i branch of the equation S_1 at most $f_\alpha(G_i)$ times. $\square$

This equation system provides a nice way to go from tree-generators to HR equations, but can we go in the opposite direction? We cannot do so generally, because tree-generators generate trees while HR-equational sets can contain graphs of arbitrary tree-width. The natural restriction is to HR^2, where all the equational sets are sets of trees, but this alone is not enough.

9.3 From HR Systems to Tree-Generators

Proposition 9.3. *There exists an* HR^2*-equational set which is not* $\iota(G)$ *for any tree-generator* G*.*

Proof. Let Ω be a nonempty wqo, and consider the following HR^2 equation system E, for some nonempty $a, b \in {\downarrow}(\Omega)$:

$$S_1 = \mathbf{0}_a \cup \mathrm{fg}_{\mathbf{1}}(\mathbf{0}_{\{a\}} /\!\!/ \overline{\mathbf{01}} /\!\!/ \mathrm{ren}_{\mathbf{0}\leftrightarrow\mathbf{1}}(S_2))$$
$$S_2 = \mathbf{0}_b \cup \mathrm{fg}_{\mathbf{1}}(\mathbf{0}_{\{b\}} /\!\!/ \overline{\mathbf{01}} /\!\!/ \mathrm{ren}_{\mathbf{0}\leftrightarrow\mathbf{1}}(S_1))$$

The set $\mathrm{L}(E, S_1)$ is equational by definition, and is exactly the paths with alternating labels a and b, where at least one of the endpoints is labeled with an a. It should be clear that each of these paths is in $\mathrm{L}(E, S_1)$, and because of the requirement that we pick the minimal solution, we cannot introduce any additional, unnecessary graphs.

But now, in order to have a tree-generator that ideal-generates $G(E, S_1)$, it cannot use the symbol $\otimes$, as that would require two consecutive vertices with the same label. As such, the paths it could generated would have bounded length, since a tree-generator must be finite, and this is not enough to create $G(E, S_1)$. □

It seems to us that this difference in expressive power originates because of the more powerful, general recursion available in general equation systems than in tree-generators. In a tree-generator, the only way it can refer to itself is immediately, using $\otimes$. No other sub-tg can refer back to the original. In contrast, an equation system can go arbitrarily deep.

9.4 Augmented Tree-Generators

We now present a more powerful alternative tree-generator which express do arbitrarily deep recursion. This allows us to overcome the first difficulty in relating HR equations with tree-generators.

Definition 9.4. Let $\Omega = (Q, \preceq)$ be a quasi-order, and let $k \geq 0$.

We define the set of *k-recursive Ω-tree-generators* $\mathcal{TG}^k(\Omega)$ inductively as follows:

Set $\mathcal{TG}^k_0(\Omega) = \{\emptyset\}$.

Let $F_{i+1}(\Omega)$ be the set of functions $(\mathcal{TG}^{k+1}_i(\Omega) \cup \{\otimes^0, \otimes^1, \ldots, \otimes^k\}) \setminus \{\emptyset\} \to \mathbb{N}^*$ which are nonzero on only finitely many points. As before, the symbols $\otimes^j$ have no intrinsic meaning.

Then we define:

$$\mathcal{TG}^k_{i+1}(\Omega) = \mathcal{TG}^k_i \cup ({\downarrow}(\Omega) \times F_{i+1}(\Omega))^{(<\omega)}$$

where, as earlier, $S^{(<\omega)}$ means the finite subsets of S.

Then we take $\mathcal{TG}^k(\Omega) = \bigcup_{n\in\mathbb{N}} \mathcal{TG}^k_n(\Omega)$.

An *augmented Ω-tree-generator* is a 0-recursive tree-generator, that is, an element of $\mathcal{TG}^0(\Omega) = \bigcup_{n\in\mathbb{N}} \mathcal{TG}^0_n(\Omega)$. Its *recursion depth* is the greatest k such that $\otimes^k$ appears as a sub-tg.

The symbols $\otimes^k$ represent recursion to k levels: when generating, if we encounter such a symbol, then rather than substituting a tree generated by the tree-generator immediately at hand, as with $\otimes$, we instead substitute a tree generated by the tree-generator k levels up. Thus, $\otimes^0$ indicates the current tree-generator, $\otimes^1$ indicates its immediate parent, and so on.

Note that an augmented tree-generator of recursion depth 0 is just a tree-generator.

Definition 9.5. Let $G = \{(J_\alpha, f_\alpha) : \alpha \in I_G\}$ be a k-recursive Ω-tree generator.

We define the set $\iota[G_1, \ldots, G_k](G)$ analogously to the definition of $\iota(G)$ for a plain tree-generator G.

$\iota[G_1, \ldots, G_k](G)$ contains all rooted trees T where there exists an $\alpha \in I_G$ and an injective function $G : \text{chldrn}(\text{r}(T)) \rightarrowtail \mathfrak{TG}(G)$ such that:

- the root r of T is labeled with J_α;
- for any $H \in \mathfrak{TG}(G)$, $|\{u \in \text{chldrn}(\text{r}(T)) : g(u) = H\}| \leq f_\alpha(H)$; and
- for each $u \in \text{chldrn}(\text{r}(T))$, $\text{br}(u) \in \iota'(g(u))$, where we say that $\iota'(\otimes^j) = \iota[G_{j+1}, \ldots, G_k](G_j)$ (with $G_0 = G$), and otherwise $\iota'(H) = \iota[G, G_1, \ldots, G_k](H)$.

As before, we say that T is *generated via* (J_α, f_α), and we define a special case $\iota[...](\{\emptyset\}) = \emptyset$. For augmented tree-generators (when $k = 0$), we also define $\tau(G)$ to contain all trees T such that T is obtained from some $T' \in \iota[](G)$ by replacing each node label with one of its elements.

The purpose of adding the $[G_1, \ldots, G_k]$ to the notation of $\iota(G)$ is to track the higher levels of tree-generators. When we descend to a lower level we add G to the list, and when we encounter $\otimes^j$, we jump back up by j levels. This additional context is not required when $k = 0$.

Now, returning to the above example, we can easily make an augmented tree-generator (of recursion depth 1) which generates alternating paths. The construction of Proposition 9.2 extends naturally to augmented tree-generators. The converse, however, is still not true in general, because, for instance, there is no augmented tree-generator which generates only graphs of size at least 2. There is much work to be done in characterizing which HR equation systems can be represented by (augmented) tree-generators.

It is unfortunately not immediately apparent how to extend the definition of a subtree-extending collection (Definition 6.5) to augmented tree-generators. We believe that this is the most important step in solving the following:

Conjecture 9.6. *Let $\Omega = (\preceq, S)$ be a wqo such that $\mathcal{P}^1(\Omega)$ is wqo, and let $\mathcal{T}$ be a subtree-extending collection of tree-generators such that:*

- *for every $G \in \mathcal{T}$, $\mathcal{TG}(G)$ is downwards-closed under $\preceq$; and*
- *there exists k such that, for every $G \in \mathcal{T}$, G has recursion-depth at most k.*

Then $(\mathcal{T}, \preceq_\tau)$ is a wqo.

This analog of Lemma 6.6 would be critical in developing new quasi-ordering results on generated sets, and we expect that there is more to be found in this area.

Another potential avenue of research is to reformulate the proof of Theorem 6.11 in terms of HR grammars directly, and thus bypass the need for tree-generators at all. Attempting to generalize such a proof may also lead to new results.

When posed as a statement about HR equation systems, Theorem 6.11 introduces many assumptions, included bounded recursion-depth and that the generated graphs are all trees (have tree-width 1). Ultimately, determining how we can weaken these assumptions while still preserving this result is likely a worthwhile endeavour.

10 Describing Ideals with HR Equations

Our other area of new work is in describing graph ideals using logic and, thus, HR equation systems. We describe a way, using the obstruction set, to encode the relationship of graph ideal inclusion in terms of a MSO-definable relation on graphs. We then investigate its consequences, including a new width parameter on graph ideals.

Our work was initially motivated as an attempt to find an alternate proof that, for any k, the graph ideals of width at most k are well-quasi-ordered. We discovered, however, that this logical technique is not well-suited to this problem. Instead, we describe a new width parameter on graph ideals specifically, and formulate a conjecture about well-quasi-ordering graph ideals bounded in this parameter.

10.1 Minors of Connected Graphs

Given a graph ideal $\mathcal{I}$, we wish to encode it into a graph $G(\mathcal{I})$. The ideal has a finite representation in its obstruction set $\text{obs}(\mathcal{I})$. We cannot, however, simply take $\oplus\,\text{obs}(\mathcal{I})$, as there may be a disconnected obstruction, and we would lose the ability to tell if two components came from the same original graph, or from different ones.

We will show that we can reduce our consideration to connected graphs. Given a collection of graphs $\mathcal{G}$, we denote by $\mathcal{G}_{\text{conn}}$ the connected graphs in $\mathcal{G}$.

Definition 10.1. Let $\triangle G$ denote the *cone* of G, created from G by adding a new vertex $\curlywedge G$, called the *apex*, and one edge from $\curlywedge G$ to each vertex of G.

Note, in particular, that $\triangle G$ is always connected.

Lemma 10.2. *Given graphs G, H, $H \prec_{\text{m}} G$ if and only if $\triangle H \prec_{\text{m}} \triangle G$.*

Proof. If $H \prec_{\text{m}} G$, take a collapse $\eta = (\eta_V, \eta_E)$ of G to H. Create a collapse $\triangle\eta = (\triangle\eta_V, \triangle\eta_E)$ of $\triangle G$ to $\triangle H$ as follows:

$$\triangle\eta_V(v) = \begin{cases} \curlywedge G & : v = \curlywedge H \\ \eta_V(v) & : \text{otherwise} \end{cases}$$

$$\triangle\eta_E(e) = \begin{cases} (\curlywedge G)\eta_V(u) & : e = (\curlywedge H)u \\ \eta_E(e) & : \text{otherwise} \end{cases}$$

Note that this is well-defined as $\curlywedge H$ has no loops. This demonstrates that $\triangle H \prec_{\text{m}} \triangle G$.

Conversely, suppose that $\triangle H \preceq_{\text{m}} \triangle G$. Let $\triangle\eta = (\triangle\eta_V, \triangle\eta_E)$ be a collapse of $\triangle G$ to $\triangle H$. We will construct a collapse $\eta = (\eta_V, \eta_E)$ from G to H. If there is no $u \in V(H)$ such that $\curlywedge G \in \triangle\eta_V(u)$, then we can take η_V to be the restriction of $\triangle\eta_V$ to $V(H)$ and likewise with η_E, and the result is the desired collapse.

Otherwise, there is some $u \in V(H)$ such that $\curlywedge G \in \triangle\eta_V(u)$. We take $U = \triangle\eta_V(\curlywedge H)$, and then for any $v \in V(H)$, $\triangle\eta_E((\curlywedge H)v)$ must go from some vertex in $\triangle\eta_V(v)$ to some vertex in U. We can then set:

$$\eta_V(v) = \begin{cases} U & : v = u \\ \triangle\eta_V(v) & : \text{otherwise} \end{cases}$$

$$\eta_E(e) = \begin{cases} \triangle\eta_E((\curlywedge H)v) & : e = uv \\ \triangle\eta_E(e) & : \text{otherwise} \end{cases}$$

For any $v \in V(H)$, $\eta_V(v)$ will be connected in G, either because $v \neq u$ and $\eta_V(v) = \triangle\eta_V(v)$, or because $\eta_V(v) = U = \triangle\eta_V(\curlywedge H)$. In either case, $\triangle\eta_V$ is a connected subgraph of $\triangle G$ not containing $\curlywedge G$, and hence is a connected subgraph of G.

Thus η is the desired collapse. □

Definition 10.3. Let $R_1 = (S_1, \sim_1)$ and $R_2 = (S_2, \sim_2)$ be relations. Then an *embedding* of R_1 into R_2 is an injective map $f : S_1 \rightarrowtail S_2$ such that $f(s) \sim_2 f(t)$ if and only if $s \sim_1 t$. Equivalently, the image of f is a subset of R_2 isomorphic to R_1.

Corollary 10.4. *The mapping $G \mapsto \triangle G$ is an injective embedding of $(\mathcal{G}, \prec_{\mathrm{m}})$ into $(\mathcal{G}_{\mathrm{conn}}, \prec_{\mathrm{m}})$.*

The identity mapping forms an embedding in the other direction, giving:

Corollary 10.5. *For any ordinal α, $\mathcal{P}^\alpha(\mathcal{G}, \preceq_{\mathrm{m}})$ is wqo if and only if $\mathcal{P}^\alpha(\mathcal{G}_{\mathrm{conn}}, \preceq_{\mathrm{m}})$ is wqo.*

Proof. Any bad sequence in $\mathcal{P}^\alpha(\mathcal{G}_{\mathrm{conn}}, \preceq_{\mathrm{m}})$ is also a bad sequence in $\mathcal{P}^\alpha(\mathcal{G}, \preceq_{\mathrm{m}})$, proving one direction.

For the other direction, for each ordinal α, recursively define maps $\triangle^\alpha : \mathcal{P}^\alpha(\mathcal{G}) \to \mathcal{P}^\alpha(\mathcal{G}_{\mathrm{conn}})$ as follows:

1. $\triangle^0 = \triangle$;
2. For each ordinal α which is the successor of an ordinal β, $\triangle^\alpha(X) = \{\triangle^\beta(Y) : Y \in X\}$.
3. For each ordinal α which is a limit ordinal:

$$\triangle^\alpha(X) = \{\triangle^\beta(Y) : Y \in X, \beta \text{ is the least ordinal such that } Y \in \mathcal{P}^\beta(\mathcal{G})\}$$

By induction, we show that α is an embedding of $\mathcal{P}^\alpha(\mathcal{G})$ into $\mathcal{P}^\alpha(\mathcal{G}_{\mathrm{conn}})$. The zero case is Corollary 10.4.

For α a successor of β, by induction, $\triangle^\beta$ is an embedding. It follows directly that $\triangle^\alpha$ is too.

Similarly for α a limit ordinal, by induction, for all $\beta < \alpha$, $\triangle^\beta$ is an embedding. Thus it follows directly again that $\triangle^\alpha$ is an embedding. □

This allows us to extend any ordering results on $\mathcal{G}_{\mathrm{conn}}$ to $\mathcal{G}$ in general, which is potentially quite powerful.

10.2 Ideal Graphs

The cone construction is a natural encoding of graphs into connected graphs, with which we can work more easily. We will begin with a little bit of notation:

Definition 10.6. A *graph ideal* is an ideal in $(\mathcal{G}, \preceq_{\mathrm{m}})$.

The set $\mathcal{G}_\triangle$ is defined as the set of all graphs which are cones.

Given a graph ideal $\mathcal{I}$, the corresponding *cone ideal* is defined as $\triangle\mathcal{I} = \mathrm{cl}_\downarrow(\{\triangle G : G \in \mathcal{I}\})$, where the closure is taken with respect to $(\mathcal{G}_\triangle, \preceq_{\mathrm{m}})$.

Note, however, that for some graph ideal $\mathcal{I}$, $\mathrm{forb}(\mathcal{I})$ may not bear any obvious relation to the corresponding set $\mathrm{forb}(\triangle\mathcal{I})$. However, either obstruction set will serve to uniquely identify $\mathcal{I}$, which is a critical property.

Lemma 10.7. *Let $\mathcal{I}$ and $\mathcal{J}$ be graph ideals. If $\triangle\mathcal{I} = \triangle\mathcal{J}$, then $\mathcal{I} = \mathcal{J}$.*

Proof. Suppose that $\triangle\mathcal{I} = \triangle\mathcal{J}$ but $\mathcal{I} \neq \mathcal{J}$. Without loss of generality, we may assume that there is a graph $H \in \mathcal{I}$ but $H \notin \mathcal{J}$.

Since $\triangle H \in \triangle\mathcal{I} = \triangle\mathcal{J}$, by definition of $\triangle\mathcal{J}$ there must be some $G \in \mathcal{J}$ such that $\triangle H \preceq_{\mathrm{m}} \triangle G$. But then $H \preceq_{\mathrm{m}} G$, and $\mathcal{J}$ is an ideal, so $H \in \mathcal{J}$, contradiction. □

Definition 10.8. Let $\mathcal{I}$ be a graph ideal. The *obstruction graph* $G(\mathcal{I})$ of $\mathcal{I}$ is defined to be $\bigoplus \mathrm{forb}(\triangle\mathcal{I})$.

We define a relation $\preceq_{\mathrm{I}}$ on $\mathcal{G}$ as follows: $H \preceq_{\mathrm{I}} G$ if and only if, for every component G' of G, there is some component H' of H such that $H' \preceq_{\mathrm{m}} G'$.

Lemma 10.9. *Let $\mathcal{I}$ and $\mathcal{J}$ be graph ideals. Then $\mathcal{J} \subseteq \mathcal{I}$ if and only if $G(\mathcal{J}) \preceq_{\mathrm{I}} G(\mathcal{I})$.*

Proof. Recall that $\mathcal{I}$ is the complement of $\mathrm{cl}_\uparrow(\mathrm{forb}_\mathcal{G}(\mathcal{I}))$ and similarly for $\mathcal{J}$. It follows that $\mathcal{J} \subseteq \mathcal{I}$ if and only if $\mathrm{cl}_\uparrow(\mathrm{forb}_\mathcal{G}(\mathcal{I})) \subseteq \mathrm{cl}_\uparrow(\mathrm{forb}_\mathcal{G}(\mathcal{J}))$.

This will occur exactly when, for every graph $G \in \mathrm{forb}_\mathcal{G}(\mathcal{I})$, there is some graph $H \in \mathrm{forb}_\mathcal{G}(\mathcal{J})$ such that $H \preceq_\mathrm{m} G$. Equivalently, since $\triangle$ is an embedding, for every graph $G' \in \mathrm{forb}_{\mathcal{G}_\mathrm{conn}}(\triangle\mathcal{I})$, there is some graph $H' \in \mathrm{forb}_{\mathcal{G}_\mathrm{conn}}(\triangle\mathcal{I})$ such that $H' \preceq_\mathrm{m} G'$.

Since all of the graphs in $\mathrm{forb}_{\mathcal{G}_\mathrm{conn}}(\triangle\mathcal{I})$ and $\mathrm{forb}_{\mathcal{G}_\mathrm{conn}}(\triangle\mathcal{J})$ are connected, this is exactly when $G(\mathcal{J}) \preceq_\mathrm{I} G(\mathcal{I})$. □

10.3 Defining Ideals

We can now tie logic into this by using our logic to identify ideals contained in other ideals:

Lemma 10.10. *Let H be a fixed graph. Then the collection of all graphs G such that $H \preceq_\mathrm{I} G$ is* MS_2*-definable.*

Proof. Let $H_1, \ldots, H_k$ be the connected components of H. Recall from Theorem 7.10 and Example 7.8 that for any fixed H', the collection of graphs G such that $H' \preceq_\mathrm{m} G$ is definable, as is the statement "X is the vertex set of a connected component of G".

We can encode the desired logical statement about a graph G as follows: "For all $X \subseteq V(G)$, if X is a connected component, then $H_1 \preceq_\mathrm{m} X$ or $H_2 \preceq_\mathrm{m} X$ or $H_3 \preceq_\mathrm{m} X$ or ... or $H_k \preceq_\mathrm{m} X$." Since k is fixed by H, this logical statement is of finite length and therefore well-defined. □

The combination of these two lemmas tells us that:

Proposition 10.11. *Let $\mathcal{J}$ be a fixed graph ideal. Then there is a definable set of graphs $\mathscr{S}$ such that, for any graph ideal $\mathcal{I}$, $\mathcal{J} \subseteq \mathcal{I}$ if and only if $G(\mathcal{I}) \in \mathscr{S}$.*

Proof. Let $\mathscr{S}$ be $\mathrm{obs}_{\preceq_\mathrm{I}}(\{G(\mathcal{J})\})$—that is, the set of all graphs G such that $G(\mathcal{J}) \preceq_\mathrm{I} G$. $\mathscr{S}$ is definable and has the desired properties. □

Note, unfortunately, that we cannot say that the set $\{G(\mathcal{I}) : \mathcal{J} \subseteq \mathcal{I}\}$ is definable, as $\mathscr{S}$ will in general contain graphs which are not $G(\mathcal{I})$ for some ideal $\mathcal{I}$. A graph will be expressible as such if and only if every component is a cone and no two components are minors of each other. The former condition is easily expressed in MS_2, but the latter condition is not expressible:

Theorem 10.12. *The property P, defined as "G contains two components, one of which is a minor of the other," is not* MS_2*-definable.*

Proof Sketch. Consider the monoid Σ^* of strings over the alphabet $\Sigma = \{\mathtt{a}, \mathtt{b}\}$. The elements are all finite sequences of symbols $\mathtt{a}$ and $\mathtt{b}$ and the only operation is concatenation. It is a standard result in language theory that the language (set) $L = \{\mathtt{a}^n\mathtt{b}^m : n \leq m\}$ is not regular (recognizable). We will assume that P is MS_2-definable and thereby create a contradiction by proving that L is regular. For the remainder of this proof, we will assume that MS_2 is the logical language being used.

Let $L' = \{\mathtt{a}^n\mathtt{b}^m : n, m \in \mathbb{N}\}$. Let $\mathcal{R}^\Sigma$ be the relational signature consisting of two unary symbols $\mathscr{L}_\mathtt{a}$ and $\mathscr{L}_\mathtt{b}$ and a single binary symbol 'adj'.

For each string $w \in \Sigma^*$, we denote by $|w|$ the length of w (the number of symbols). We then define a relational structure $\lceil w \rceil$ with $|w|$ elements, corresponding to each of the positions in the string. For each element e, $\mathscr{L}_\mathtt{a}(e)$ is true if the eth element is $\mathtt{a}$, and similarly for $\mathscr{L}_\mathtt{b}$ and $\mathtt{b}$. For two elements e and f, $\mathrm{adj}(e, f)$ is true if e comes immediately before f in the string.

We let C'_k denote the graph obtained from the cycle C_k by adding a pendant edge (see Figure 8), and define a map $f : L' \to \mathcal{G}$ by $f(\mathtt{a}^n\mathtt{b}^m) = C_n \oplus C'_m$ (where $\oplus$, as before, denotes disjoint union of graphs).

For any $w = \mathtt{a}^n\mathtt{b}^m \in L'$, then the property P is true for $f(w)$ if and only if the C_n component is a minor of the C'_m component, since the C'_m component cannot be a minor of the C_n component. This occurs if and only if $n \leq m$. Hence P is true for $f(w)$ if and only if $w \in L$.

At this point, we hit a roadblock, because we do not have the machinery to define f in a way that we can use. We cannot define it in a quantifier-free manner (or, more accurately, our attempts were met with

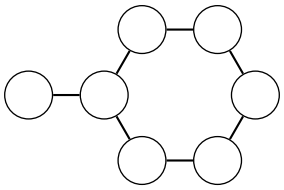

Figure 8: The graph C'_6.

failure). What we need is a more general theory of transformations: a generalization of Definition 8.7 and the Backwards Translation Theorem 8.10, applicable to f. The necessary generalizations can be found in Section 7 of [2].

The logical formulas defining f are straightforward once we have quantfiers. The formula $\phi(e) = \nexists f.\,\mathrm{adj}(f,e)$, for instance, asserts that e is the first element of the string, and $\psi(e) = \mathscr{L}_{\mathtt{a}}(e) \wedge \exists f.(\mathrm{adj}(e,f) \wedge \mathscr{L}_{\mathtt{b}}(f))$ asserts that e is the last $\mathtt{a}$.

Since L' is a definable subset of Σ^*, we can extend f to a map $f' : \Sigma^* \to \mathcal{G}$ by setting $f'(w) = f(w)$ if $w \in L'$, and $f'(w) = f(\mathtt{ab})$ otherwise. Then P holds for $f'(w)$ if and only if $w \in L$ as desired.

Finally, we need to find a structure in which $f'(L)$ is definable, in order to apply the Backwards Translation Theorem. But the range of f' is a subset of $\mathcal{G}' = \{C_n \oplus C'_m : n, m \in \mathbb{N}\}$, so we can actually just consider f' to be a map $\Sigma^* \to \mathcal{G}'$. Then, since P is definable, and since $f'(L)$ is exactly the subset of $\mathcal{G}'$ for which P holds, $f'(L)$ is definable and hence so is L.

Finally, since L is a definable subset of Σ^*, by the full Recognizability Theorem (of which Theorem 8.18 is a special case), it is recognizable, giving us the desired contradiction. □

We are not aware of any ocurrences of this proof in the literature, although the technique is standard. This proof was provided to us by Courcelle in private correspondence.

This result tells us that in order to restrict ourselves to the graphs of the form $G(\mathcal{I})$, we must do so as a post-processing step. We do not see any major obstacles in this, however, as we can simply regard the presence or absence of other graphs in $\mathscr{S}$ as irrelevant information.

10.4 Cone Tree-width

Turning to the implications for graph structure, we wish to generate equation systems which define sets of graphs analogous to $\mathscr{S}$ above. As with graphs in general, we are limited fundamentally by the requirement that all equational sets have bounded tree-width.

In this case, however, the limitation on tree-width is not on the ideals $\mathcal{I}$ themselves, because we are not attempting to describe them. Rather, our definability result is on the obstruction graphs. As a result, we are actually talking about a different criterion:

Definition 10.13. Let $\mathcal{I}$ be a graph ideal. The *obstruction-width* $\mathrm{ow}(\mathcal{I})$ is $\max\{\mathrm{tw}(G)+1 : G \in \mathrm{obs}(\mathcal{I})\}$. As a special case, we define $\mathrm{ow}(\mathcal{G}) = 0$, since the obstruction set of all graphs is empty.

We will also need a lemma about the tree-width of cones:

Lemma 10.14. *Let $G = (V, E)$ be any graph. Then* $\mathrm{tw}(\triangle G) = \mathrm{tw}(G) + 1$.

Proof. Given a tree-decomposition of G, we can turn it into a tree-decomposition of $\triangle G$ by adding $\curlywedge G$ to every bag, so $\mathrm{tw}(\triangle G) \leq \mathrm{tw}(G) + 1$.

Suppose $T = (W, E')$ is a minimum-width decomposition of $\triangle G$, and assume towards a contradiction that there exists $w \in W$ such that $f(w)$ is largest, but does not contain $\curlywedge G$. Assume without loss of generality that w is not labeled the same as any of its neighbours; that is, it has a bag distinct from its neighbours.

Only one component S of $T - w$ can contain bags with $\curlywedge G$, by the connectivity condition. As T is a tree, there is a unique neighbour w' of w which is in S. But $f(w') \neq f(w)$ and $f(w)$ is of largest size, so there is $v \in f(w)$ with $v \notin f(w')$.

It follows that v is not in any bag in S, but then v and $\curlywedge G$ never share a bag, which is a contradiction since $v \sim \curlywedge G$ by definition.

Accordingly, for every minimum-width tree-decomposition of $\triangle G$, $\curlywedge G$ is in every bag of largest size. So by removing $\curlywedge G$ from every bag, we get a tree-decomposition of G of width $\mathrm{tw}(\triangle G) - 1$, proving that $\mathrm{tw}(\triangle G) \geq \mathrm{tw}(G) + 1$. □

10.5 Second-Order Ideals

Now we wish to use HR equation systems to describe sets of ideals by describing their obstruction graphs. We will first observe that we cannot properly describe *any* second-order ideal (ideal of ideals) using this technique, however. This limitation seems somewhat arbitrary, and may be indicative of some deeper logic. It deserves more investigation.

Proposition 10.15. *Let $\mathcal{J}$ be a graph ideal and let $k \geq \mathrm{ow}(\mathcal{J})$. Then there exists a graph ideal $\mathcal{I}$ such that $\mathcal{J} \subseteq \mathcal{I}$ and $\mathrm{ow}(\mathcal{I}) = k$.*

Proof. Let G be a graph in $\mathrm{obs}(\mathcal{J})$ of highest tree-width. We define $\mathcal{F} = \{\triangle^{k-j} G\} \cup \{H \in \mathrm{obs}(\mathcal{J}) : H \not\preceq_{\mathrm{m}} \triangle^{k-j} G\}$, where $\triangle^i G$ is obtained from G by applying the cone operations i times, and $j = \mathrm{ow}(\mathcal{J})$.

In other words, $\mathcal{F}$ is obtained from $\mathrm{obs}(\mathcal{J})$ by increasing the tree-width of one of the graphs until it is k.

By construction, the graphs in $\mathcal{F}$ are all unrelated by the minor relation. It follows that taking $\mathcal{I} = \mathrm{forb}(\mathcal{F})$ is well-defined and gives us the desired obstruction width. Every graph in $\mathrm{obs}(\mathcal{J})$ is either in $\{H \in \mathrm{obs}(\mathcal{J}) : \mathcal{H} \not\preceq_{\mathrm{m}} \triangle^{k-j} G\}$. Thus, every graph which is a minor of a graph in $\mathrm{obs}\,\mathcal{J}$ is also a minor of a graph in $\mathcal{F}$, and so $\mathcal{J} \subseteq \mathcal{I}$. □

Corollary 10.16. *Let $\mathcal{J}$ be a graph ideal. Then there is no equational set of graphs containing $\{G(\mathcal{I}) : \mathcal{J} \subseteq \mathcal{I}\}$.*

It doesn't seem to us that this result is very dependent on the particular representation $G(\mathcal{I})$. Even if we find some other way to encode graph ideals into this algebra, the potentially unbounded tree-width of graphs in the obstruction set will likely be limiting.

10.6 Ideals of Bounded Obstruction-Width

We recall Thomas' result on graph ideals:

Theorem 10.17 (Thomas, [19])**.** *Let G_0 be a planar graph. Then the graphs not containing G_0 are bqo under taking of minors. In other words, $(\mathrm{obs}(G_0), \preceq_{\mathrm{m}})$ is bqo.*

This is a fairly powerful result, and it is not immediately clear that there is hope of coming close to it from this logic-based approach to first-order ideals. Any wqo result on first-order ideals must, in order to truly progress on the broader problem of graph ideals, find a way beyond this restriction of excluding a planar graph, because this result excludes ideals containing all planar graphs.

However, the notion of obstruction-width presents an alternate approach. Wagner's Theorem is well known and tells us that the set of all planar graphs is $\mathrm{forb}(K_{3,3}, K_5)$, which means that the planar graphs have obstruction-width 5. This leads us to the following conjecture:

Conjecture 10.18. *For each $k \in \mathbb{N}$, the collection of graph ideals with obstruction-width at most k is wqo under containment.*

Since $\mathrm{ow}(\mathcal{G}) = 0$, the set $\mathcal{G}$ of all graphs is always in this collection, and hence this conjecture is not implied for any k by Thomas' theorem. Proving it would represent a new advance. It is our belief that, by further development of logic and grammar-based techniques, this conjecture ought to be provable, and this may represent novel progress towards a result on all graph ideals.

We also believe that the technique of using cone graphs to define and make recognizable all the graph ideals represents a significant step towards higher-order ideals. The technique should extend readily to

ideals of any finite order, by simply taking more cones and defining an equivalent of $\preceq_{\mathrm{I}}$ at the higher levels. We could then define an obstruction-width property on each of the ideals of higher order, and attempt to prove the analog of this conjecture at that level. We leave this to future work, however.

11 Conclusion

The field of graph structure theory owes a lot of its development to the work of Robertson and Seymour in their attempt to prove the Graph Minors Theorem. Graph structure has found application in a number of fields, and has even led to development of entirely new areas of study, such as fixed-parameter tractable algorithms. It is unclear what applications may arise from the study of higher-order ideals of graphs, but that is of course no reason not to study them. We never know what we may find.

On the other side of things, the work of Courcelle and Engelfriet in graph logic and graph grammars is still a relatively isolated area of study. But as we have seen, the application of techniques from other areas of mathematics has given us a rich framework with which to begin a new attack on graph structure.

We have taken the tree-generators of Christian et al., shown that they arise as special cases of HR grammars, and proposed a way of generalizing them in hopes of strengthening existing well-quasi-ordering results to HR grammars. We have shown how to apply logical techniques to obtain new descriptions of graph ideals, including a width parameter on graph ideals which bears further investigation.

Most importantly, however, we hope that this thesis will serve as a basis for new and exciting developments where these two rich fields intersect. Each of these fields is complex and its own right, and so we also hope that this thesis can serve as an introduction to someone new to this area of graph theory. There is much, much more to be discovered here, and hopefully it will in fact be discovered!

References

[1] Robin Christian, R. Bruce Richter, and Gelasio Salazar. Tree generators. Unpublished research notes.

[2] Bruno Courcelle and Joost Engelfriet. *Graph Structure and Monadic Second-Order Logic: A Language-Theoretic Approach*. Number 138 in Encyclopedia of Mathematics and Its Applications. Cambridge University Press, 2012.

[3] M. R. Fellows and M. A. Langston. On search decision and the efficiency of polynomial-time algorithms. In *Proceedings of the Twenty-first Annual ACM Symposium on Theory of Computing*, STOC '89, pages 501–512, New York, NY, USA, 1989. ACM.

[4] Graham Higman. Ordering by divisibility in abstract algebras. *Proceedings of the London Mathematical Society*, s3-2(1):326–336, 1952.

[5] Ken ichi Kawarabayashi, Yusuke Kobayashi, and Bruce Reed. The disjoint paths problem in quadratic time. *Journal of Combinatorial Theory, Series B*, 102(2):424 – 435, 2012.

[6] Ton Kloks. *Treewidth - Computations and Approximations*. Number 842 in Lecture Notes in Computer Science. Springer-Verlag Berlin Heidelberg, 1994.

[7] J. B. Kruskal. Well-quasi-ordering, the tree theorem, and vazsonyi's conjecture. *Transactions of the American Mathematical Society*, 95(2):pp. 210–225, 1960.

[8] Joseph B Kruskal. The theory of well-quasi-ordering: A frequently discovered concept. *Journal of Combinatorial Theory, Series A*, 13(3):297 – 305, 1972.

[9] Daniela Kühn. On well-quasi-ordering infinite trees: Nash-williams's theorem revisited. *Mathematical Proceedings of the Cambridge Philosophical Soceity*, 130(3):401–408, 2001.

[10] Richard Laver. On fraisse's order type conjecture. *Annals of Mathematics*, 93(1):pp. 89–111, 1971.

[11] Chun-Hung Liu. *Graph Structures and Well-Quasi-Order*. PhD thesis, Georgia Institute of Technology, 2014.

[12] C. St. J. A. Nash-Williams. On well-quasi-ordering infinite trees. *Mathematical Proceedings of the Cambridge Philosophical Society*, 61(03), 7 1965.

[13] R. Bruce Richter. Graph minors: generalizing kuratowski's theorem. In Lowell W. Beineke and Robin J. Wilson, editors, *Topics in topological graph theory*, number 128 in Encyclopedia of Mathematics and Its Applications, chapter 5, pages 81–110. Cambridge University Press, 2013.

[14] Neil Robertson and Paul Seymour. Graph minors. ii. algorithmic aspects of tree-width. *Journal of Algorithms*, 7(3):309–322, 1986.

[15] Neil Robertson and Paul Seymour. Graph minors. v. excluding a planar graph. *Journal of Combinatorial Theory, Series B*, 41(1):92–114, 1986.

[16] Neil Robertson and Paul Seymour. Graph minors. iv. tree-width and well-quasi-ordering. *Journal of Combinatorial Theory, Series B*, 48(2):227–254, 1990.

[17] Neil Robertson and Paul Seymour. Graph minors. xx. wagner's conjecture. *Journal of Combinatorial Theory, Series B*, 92(2):325–357, 2004.

[18] Jeffrey Shallit. *A Second Course in Formal Languages and Automata Theory*. Cambridge University Press, 9 2008.

[19] Robin Thomas. Well-quasi-ordering infinite graphs with forbidden finite planar minor. *Transactions of the American Mathematical Society*, 312:279–313, 1989.

[20] Yaming Yu. More forbidden minors and wye-delta-wye reducibility. *The Electronic Journal of Combinatorics*, 2006.

Index