Definability and Hankel Matrices

Nadia Labai
Definability and Hankel Matrices

Research Thesis

Submitted in partial fulfillment of the requirements for the degree of Master of Science in Computer Science

Nadia Labai

Submitted to the Senate of the Technion — Israel Institute of Technology
Nisan 5775 Haifa April 2015
This research was carried out under the supervision of Prof. Johann A. Makowsky, in the Faculty of Computer Science.

Acknowledgments

Throughout my studies, Prof. Johann A. Makowsky has provided me with invaluable guidance. While always encouraging me to explore ideas independently, Prof. Makowsky also gave me the tools to spot and learn from my mistakes. I am fortunate to have had access to his vast knowledge in Computer Science and Mathematics, which was of help to me countless times. Above all else, his faith in my abilities made this thesis possible. I extend my grateful thanks to Prof. Makowsky for this privilege.

I would also like to thank Tomer Kotek, whose conversations with Prof. Makowsky made ideas in our research more precise, and Prof. Michael Kaminski for his time as acting supervisor during Prof. Makowsky’s sabbatical.

The generous financial help of the Technion, the Israel Science Foundation and the city of Haifa is gratefully acknowledged.
List of Publications

Parts of this thesis are based on results published in collaboration with Prof. J.A. Makowsky.


## Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Introduction</td>
<td>5</td>
</tr>
<tr>
<td>2</td>
<td>Definability in Monadic Second Order Logic</td>
<td>7</td>
</tr>
<tr>
<td>2.1</td>
<td>Graphs and Words as Structures</td>
<td>7</td>
</tr>
<tr>
<td>2.2</td>
<td>MSOL and CMSOL</td>
<td>9</td>
</tr>
<tr>
<td>2.3</td>
<td>Definable Graph Parameters</td>
<td>10</td>
</tr>
<tr>
<td>2.4</td>
<td>Definable Word Functions</td>
<td>11</td>
</tr>
<tr>
<td>2.5</td>
<td>MSOLEVAL and CMSOLEVAL</td>
<td>12</td>
</tr>
<tr>
<td>3</td>
<td>Meta-Theorems Using Logic</td>
<td>15</td>
</tr>
<tr>
<td>3.1</td>
<td>Common Notions of Width for Graphs</td>
<td>15</td>
</tr>
<tr>
<td>3.2</td>
<td>Meta-theorems for Clique-width and Tree-width</td>
<td>18</td>
</tr>
<tr>
<td>3.3</td>
<td>Generalized Notions of Width</td>
<td>18</td>
</tr>
<tr>
<td>4</td>
<td>Hankel Matrices and the Finite Rank Theorem</td>
<td>21</td>
</tr>
<tr>
<td>4.1</td>
<td>Hankel Matrices for General Parameters</td>
<td>21</td>
</tr>
<tr>
<td>4.2</td>
<td>Hankel Matrices for Graph Parameters and Word Functions</td>
<td>22</td>
</tr>
<tr>
<td>4.3</td>
<td>The Finite Rank Theorem</td>
<td>23</td>
</tr>
<tr>
<td>5</td>
<td>MSOL and Word Functions</td>
<td>25</td>
</tr>
<tr>
<td>5.1</td>
<td>Weighted Automata</td>
<td>25</td>
</tr>
<tr>
<td>5.2</td>
<td>A Finite Rank Theorem for Word Functions</td>
<td>27</td>
</tr>
<tr>
<td>5.3</td>
<td>Characterizing Recognizable Word Functions</td>
<td>30</td>
</tr>
<tr>
<td>MSOL-definable word functions are recognizable</td>
<td>30</td>
<td></td>
</tr>
</tbody>
</table>
Recognizable word functions are MSOL-definable ........................................ 31
5.4 Weighted MSOL and MSOLEVAL .......................................................... 34
   Syntax of WMSOL ...................................................................................... 34
   Semantics of WMSOL, and translation of RMSOL into MSOLEVAL\textsubscript{S} .... 35
   Translation from MSOLEVAL\textsubscript{S} to RMSOL ........................................ 39
   Comparison ............................................................................................... 39

6 Eliminating Logic .......................................................................................... 41
   6.1 Graph Algebras ...................................................................................... 41
       Quantum graphs ....................................................................................... 42
       Equivalence relations for quantum graphs .............................................. 43
       Computing graph parameters via dynamic programming ................. 44
   6.2 Linearly Linked Hankel Matrices ........................................................... 47
       A more general perspective .................................................................. 49

7 From Fields to Semirings .............................................................................. 51
   7.1 Motivating Examples ........................................................................... 51
   7.2 Semirings and Semimodules ................................................................. 52
   7.3 Graph Parameters Over Semirings ......................................................... 56
       Arbitrary semirings ............................................................................... 56
       Tropical semirings ............................................................................... 58
   7.4 LinEMSOL and the Tropical Semirings .................................................. 59

8 Conclusion and open questions ................................................................. 61
   8.1 Further Research .................................................................................. 61
       A different direction ............................................................................... 62

A Elaborations and Additional Proofs ............................................................ 63
Abstract

In automata theory, a Hankel matrix $H(f, \circ)$ is an infinite matrix where the rows and columns are labeled with words $w$ over a fixed alphabet $\Sigma$, and the entry $H(f, \circ)_{u,v}$ is given by $f(u \circ v)$ for words $u$ and $v$, where $f$ is a real-valued word function and $\circ$ denotes concatenation. A classical result of G.W. Carlyle and A. Paz (1971) in automata theory characterizes real-valued word functions $f$ recognizable by weighted automata as the functions for which the Hankel matrix has finite rank.

Hankel matrices for graph parameters were introduced by L. Lovász (2007) and used to characterize real-valued partition functions of graphs.

We study the use of Hankel matrices $H(f, \Box)$ for graph parameters $f$ and various binary operations $\Box$ on labeled and colored graphs.

We consider meta-theorems connecting the definability of graph parameters to their computational complexity on certain graph classes, such as Courcelle’s theorem, stating that definable graph properties are computable in linear time over graph classes of bounded tree-width, and generalizations thereof, and show how to eliminate logic from these theorems by replacing the definability assumption with a finiteness assumption on the rank of the Hankel matrices involved.

As a special case, we also consider word functions and show that word functions are definable in Monadic Second Order Logic (MSOL) if and only if their Hankel matrices for concatenation have finite rank.
Abbreviations and Notations

\[ |A| \]  \quad \text{The cardinality of the set } A
\[ [n] \]  \quad \text{The set } \{1, \ldots, n\}
\[ \mathcal{T}_{\text{max}} \]  \quad \text{The tropical semiring } (\mathbb{R} \cup \{-\infty\}, \max, +)
\[ \mathcal{T}_{\text{min}} \]  \quad \text{The tropical semiring } (\mathbb{R} \cup \{\infty\}, \min, +)

Logic and Model Theory

\[ \text{Str}(\tau) \]  \quad \text{The class of finite structures over the vocabulary } \tau
\[ \text{SOL}(\tau) \]  \quad \text{Second Order Logic over the vocabulary } \tau
\[ \text{MSOL}(\tau) \]  \quad \text{Monadic Second Order Logic over the vocabulary } \tau
\[ \text{CMSOL}(\tau) \]  \quad \text{Monadic Second Order Logic with modular counting over the vocabulary } \tau
\[ \text{MSOLEVAL}_F \]  \quad \text{The class of } F\text{-valued functions definable in MSOL}
\[ \text{CMSOLEVAL}_F \]  \quad \text{The class of } F\text{-valued functions definable in CMSOL}
\[ qr(f) \]  \quad \text{The quantifier rank of } f
\[ tv(\varphi) \]  \quad \text{The truth value of } \varphi

Graph Theory

\[ L\text{Graphs}_k \]  \quad \text{The class of } k\text{-labeled graphs}
\[ C\text{Graphs}_k \]  \quad \text{The class of } k\text{-colored graphs}
\[ \text{TW}(k) \]  \quad \text{The class of graphs of tree-width } k
\[ \text{CW}(k) \]  \quad \text{The class of graphs of clique-width } 2^k
\[ \text{PW}(k) \]  \quad \text{The class of graphs of path-width } k
\[ \text{LCW}(k) \]  \quad \text{The class of graphs of linear clique-width } 2^k
\[ \sqcup \]  \quad \text{The disjoint union operation}
\[ \sqcup_k \]  \quad \text{The } k\text{-connection operation}
\[ \bar{\eta}_{i,j} \]  \quad \text{The } (i,j)\text{-join operation}
<table>
<thead>
<tr>
<th>Notation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$H(f, \square)$</td>
<td>The Hankel matrix of $f$ with the operation $\square$</td>
</tr>
<tr>
<td>$H(f, \sqcup_k)$</td>
<td>The $k$-connection matrix of $f$</td>
</tr>
<tr>
<td>$H(f, \eta)$</td>
<td>The join matrix of $f$</td>
</tr>
<tr>
<td>$H(f, \circ)$</td>
<td>The concatenation matrix of $f$</td>
</tr>
<tr>
<td>$MH(f, \square)$</td>
<td>The (semi)module generated by the rows of $H(f, \square)$</td>
</tr>
<tr>
<td>$Q_k$</td>
<td>The space of $k$-colored quantum graphs</td>
</tr>
<tr>
<td>$\mathcal{S}^\Sigma^*$</td>
<td>The set of word functions $\Sigma^* \to \mathcal{S}$</td>
</tr>
</tbody>
</table>
Chapter 1

Introduction

The main theme of this thesis is the elimination of logic from meta-theorems relating definability to computational complexity. Some famous examples of meta-theorems include the Büchi-Elgot-Trakhtenbrot theorem, relaying the definability of languages in Monadic Second Order Logic (MSOL) to regularity, Fagin’s theorem, relating definability in Existential Second Order Logic to the complexity class NP (non-deterministic polynomial-time), and the Robertson-Seymour theorem, implying that minor-closed graph classes are definable in MSOL. See for a survey on finite model theory and descriptive complexity.

Another example of a meta-theorem is Courcelle’s celebrated theorem, which states that the definability of binary graph parameters in MSOL implies their verifiability in linear time for graph classes of bounded tree-width. The theorem’s generalization to graph classes of bounded clique-width, by B. Courcelle, J. A. Makowsky and U. Rotics in , states a similar implication for MSOL-definable graph parameters and polynomial time.

Informally, notions of width for graphs try to capture how much a graph “looks like” other graphs of a specific kind; thus having low tree-width or clique-width implies being very tree-like or clique-like, respectively. There are various notions of width for graphs, presented a detailed discussion. The graph classes of bounded tree-width and clique-width are defined inductively in Definitions and .

J.A. Makowsky gave a further generalized meta-theorem in , involving sum-like inductive graph classes. These graph classes are also defined inductively, by using binary operations which are sum-like. An operation is sum-like if it can be expressed as the result of a certain logical treatment of the graphs it is applied on. The details are given in Section 3.3.

The logic will be eliminated from these theorems by replacing their definability conditions on functions with finiteness conditions on Hankel matrices. In linear algebra, a Hankel matrix, named after Hermann Hankel, is a square matrix with constant skew-diagonals. Hankel matrices have many applications in numeric analysis, probability, and combinatorics. They appear in the theory and applications of Padé approximations.
theory of moments, coding theory (BCH codes, the Berlekamp-Massey algorithm), combinatorial enumerations (Lattice paths, Young tableaux, matching theory), and learning theory. In Chapter 4 the definition is generalized to Hankel matrices that describe (perhaps only partially) functions $f$ on structures, such as graphs.

A logic-free version of Courcelle’s theorem was given by L. Lovász in [68], which is stated here as Theorem 6.1. We prove the logic-free parallels of the clique-width case (Theorem 6.2) and of the sum-like inductive case (Theorem 6.11).

Another point of focus is word functions. Theorem 5.6 characterizes word functions recognizable by weighted automata using the formalism of [22], which was originally developed to deal with the definability of graph parameters and graph polynomials. In Section 5.4 we compare it to a different formalism introduced by M. Droste and P. Gastin in [30], called Weighted Monadic Second Order Logic (WMSOL), which was also used to characterize recognizable word functions.

Our logical framework is presented in detail in Chapter 2 where the logic MSOL is presented as well as the formalism MSOLEVAL from [22]. The meta-theorems we are interested in, along with the graph classes of bounded tree-width, bounded clique-width, and the sum-like inductive graph classes, are presented in Chapter 3. The infinite Hankel matrices, on which our finiteness conditions will be imposed, are discussed in Chapter 4. The main results of the thesis lie in Chapters 5 and 6 where in the latter the results are stated for graph parameters over fields. These results are extended in Chapter 7 to semirings.
Chapter 2

Definability in Monadic Second Order Logic

This chapter contains definitions necessary for our logical treatment of graph parameters and word functions. The representation of graphs and words as structures is given at the beginning of the chapter, and is followed by a presentation of the logics we are concerned with, namely Monadic Second Order Logic MSOL and (modular) Counting Monadic Second Order Logic CMSOL.

The precise definition of graph parameters and word functions in (C)MSOLEVAL has to be prefaced by a number of somewhat technical definitions, so we first develop an intuition by reviewing examples of CMSOL-definable word functions and graph parameters, and the exact notion of their definability is presented at the end of the chapter.

2.1 Graphs and Words as Structures

We begin with some model-theoretic definitions, taken partially from [33].

Definition 2.1. A vocabulary $\tau$ is a finite set consisting of relation symbols, function symbols, and constants. Each relation and function symbol is equipped with a natural number called its arity. If $\tau$ contains only relation symbols, we say it is relational.

Definition 2.2. A $\tau$-structure $A$ consists of a non-empty set $A$ called the universe of $A$, an $n$-ary relation on $A$ for each $n$-ary relation symbol in $\tau$, an $n$-ary function on $A$ for each $n$-ary function symbol in $\tau$, and an element in $A$ for each constant symbol in $\tau$.

$A$ is finite if its universe is finite. We assume all structures are finite.

A graph $G$ is given by its set of vertices $V_G$ and edge relation $E_G \subseteq V_G \times V_G$. In a straightforward way, we represent graphs as structures over $\tau_{\text{graphs}} = \langle E \rangle$, where $E$ is a binary relation.
Figure 2.1: A simple directed graph.

Given a graph $G = (V_G, E_G)$ where $V_G = \{v_1, \ldots, v_n\}$, we represent it by the structure

$$G = \langle G, E_G \rangle$$

where the universe $G$ is the set $[n] = \{1, \ldots, n\}$, and $(i, j) \in E_G$ if and only if $(v_i, v_j) \in E_G$. We denote the class of graphs by $\text{Graphs}$.

Example. The graph shown in Figure 2.1 is represented by the $\tau_{\text{graphs}}$-structure $G = \langle G, E_G \rangle$, where

$$G = \{1, 2, 3, 4, 5\} \quad E_G = \{(1, 2), (2, 1), (2, 4), (3, 1), (4, 3), (4, 5)\}.$$

It is worth mentioning that another natural way of representing graphs is via the vocabulary $\tau_{\text{hyper}} = \langle V, E, I \rangle$, where $V$ and $E$ are unary, and $I$ is ternary. A structure over $\tau_{\text{hyper}}$ is of the form


In order to represent a graph, we enforce that the vertex set $V_G$ and the edge set $E_G$ partition the universe $G$, and the incidence relation $I_G$ of the graph is a subset of $V_G \times E_G \times V_G$. This allows the representation of graphs with multiple edges. Graphs may also have labels or be colored, in which case the vocabulary would have constant symbols and unary predicates associated with the labels and colors. These graphs are addressed in [Chapter 3].

As for words, the representation requires more care. Let $\Sigma = \{\sigma_1, \ldots, \sigma_s\}$ be a finite alphabet. Since the positions in a word are ordered and hold letters, we represent words as structures over $\tau_{\text{words}}^\Sigma = \langle <, P_{\sigma_1}, \ldots, P_{\sigma_s} \rangle$, where $<$ is a binary relation enforced to be a linear order on the universe, and $P_{\sigma_i}$ is a unary relation, for $i = 1, \ldots, s$.

Were we to represent a word $w \in \Sigma^*$ of length $\ell(w)$ as a structure $W$ with universe $W = [\ell(w)]$, we would be unable to represent the empty word, as structures always have a non-empty universe. To overcome this, the universe of the structure representing a word $w$ will (additionally) contain the zero position, represented by $0 \in W$. So, strictly speaking, the size of the structure representing the empty word will be 1, and the size of the structure representing a word of length $n$ will be $n + 1$. 
Then, we have that a word $w \in \Sigma^*$ of length $\ell(w)$ is represented by the structure

$$W = (\{0\} \cup [\ell(w)], <^W, P^W_{\sigma_1}, \ldots, P^W_{\sigma_s}).$$

The zero element has no letter attached to it, and the other elements represent positions in the word which carry letters. For $i = 1, \ldots, s$, the positions represented by elements in $P^W_{\sigma_i}$ carry the letter $\sigma_i$.

In total, we have that for $W$,

- $P^W_{\sigma_1}, \ldots, P^W_{\sigma_s} \subseteq [\ell(w)]$,
- $\bigcap_{i=1}^s P^W_{\sigma_i} = \emptyset$,
- $\bigcup_{i=1}^s P^W_{\sigma_i} = [\ell(w)]$.

For readability, we will freely transition between graphs and words and the structures that represent them.

### 2.2 MSOL and CMSOL

We now define Monadic Second Order Logic (MSOL) and present its augmentation with modular counting, CMSOL, which are variants of (full) Second Order Logic (SOL). Although we will not be concerned with SOL, we mention its definition for completeness.

The logics will be defined over a vocabulary $\tau$, so we denote them $\text{MSOL}(\tau)$ and $\text{CMSOL}(\tau)$. We omit the vocabulary when it is clear from context.

The logic $\text{MSOL}(\tau)$ for $\tau$-structures involves first order variables, denoted by $x_i$ for $i \in \mathbb{N}$, and unary second order variables, denoted by $U_i$ for $i \in \mathbb{N}$. The first order variables will range over elements from the universe, and the second order variables will range over sets of elements from the universe. Our convention is to denote first order variables by lower-case letters and second order variables by upper-case letters.

**Definition 2.3.** A term is either

(i) a first order variable, or

(ii) a constant symbol from $\tau$, or

(iii) is obtained by an inductive application of function symbols from $\tau$.

**Definition 2.4** (MSOL($\tau$) formulas). An atomic MSOL formula is in one of the following forms, where all $t_i$ are terms:

(i) $t_1 \approx t_2$

(ii) $R(t_1, \ldots, t_s)$, for a relation symbol $R$ from $\tau$ of arity $s$
(iii) $U_i(t_1)$, for a second order variable $U_i$.

Formulas are inductively built using the connectives $\neg$, $\lor$, $\land$, $\rightarrow$, $\leftrightarrow$, and the quantifiers $\forall x_i$, $\exists x_i$, $\forall U_i$, $\exists U_i$, and are interpreted in the natural way.

We will sometimes call formulas with no free variables (variables which are not quantified) sentences.

Example. The following $\text{MSOL}(\tau_{\text{graphs}})$-sentence expresses that a graph is 3-colorable:

$$\exists U_1 \exists U_2 \exists U_3 \left[ \text{Partition}(U_1, U_2, U_3) \land \text{IndSet}(U_1) \land \text{IndSet}(U_2) \land \text{IndSet}(U_3) \right]$$

where $\text{Partition}(U_1, U_2, U_3)$ expresses that each vertex gets exactly one of the three colors:

$$\text{Partition}(U_1, U_2, U_3) = \forall x \left[ U_1(x) \lor U_2(x) \lor U_3(x) \right] \land \neg \exists x \left[ (U_1(x) \land U_2(x)) \lor (U_1(x) \land U_3(x)) \lor (U_2(x) \land U_3(x)) \right]$$

and $\text{IndSet}(U)$ expresses that the vertices colored $U$ form an independent set:

$$\text{IndSet}(U) = \forall x_1, x_2 \left[ U(x_1) \land U(x_2) \rightarrow \neg E(x_1, x_2) \right]$$

By allowing the usage of modular counting quantifiers, we obtain:

**Definition 2.5** (CMSOL$(\tau)$ formulas). Any MSOL$(\tau)$ formula is also a CMSOL$(\tau)$ formula. In addition, CMSOL$(\tau)$ allows formulas of the form $C_{a,b}\bar{x}.\varphi(\bar{x})$, which are satisfied when there are $a \mod b$ tuples $\bar{x}$ of elements that satisfy $\varphi$.

Example. The following CMSOL formula expresses that the universe is of even cardinality:

$$C_{0,2}x \left[ x \approx x \right]$$

CMSOL is strictly more expressive than MSOL, as it cannot be expressed in MSOL that the universe is of even cardinality. However, on ordered structures they have the same expressive power.

SOL$(\tau)$ formulas are defined as in [Definition 2.4](#) except the second order variables may range over sets of tuples of elements, not just sets of elements.

### 2.3 Definable Graph Parameters

A graph parameter $f$ is a function that is invariant under isomorphisms which takes graphs to a numeric domain. We now consider some examples of graph parameters which are in the (to be define) class MSOLEVAL, and take note of their presentation as sums and products ranging over expressions that are definable in MSOL.
Examples 2.6.

(i) For a graph $G$ with vertex set $V$, the cardinality $|V|$ of $V$ is MSOL-definable by

$$|V| = \sum_{v \in V} 1$$

(ii) The number of connected components of a graph $G$, $k(G)$, is MSOL-definable by

$$k(G) = \sum_{C \subseteq V : \text{component}(C)} 1$$

where $\text{component}(C)$ says that $C$ is a connected component.

(iii) The number of cliques $\#\text{Clique}(G)$ in a graph is MSOL-definable by

$$\#\text{Clique}(G) = \sum_{C \subseteq V : \text{clique}(C)} 1$$

where $\text{clique}(C)$ says that $C$ induces a complete graph.

(iv) Similarly “the number of maximal cliques” $\#\text{M clique}(G)$ is MSOL-definable by

$$\#\text{M clique}(G) = \sum_{C \subseteq V : \text{maxClique}(C)} 1$$

where $\text{maxClique}(C)$ says that $C$ induces a maximal complete graph.

2.4 Definable Word Functions

Let $\Sigma = \{0, 1\}$, and let $w$ be a word in $\Sigma^* = \{0, 1\}^*$ of length $\ell(w)$, represented by the structure

$$W = \langle \{0\} \cup [\ell(w)], <^W, P_0^W, P_1^W \rangle.$$ 

We denote by $w[i]$ the letter at position $i$ in $w$, and by $w[U]$ the word induced by a set $U$ of positions in $w$. We denote the concatenation of two words $u, v \in \Sigma^*$ by $u \circ v$.

Examples 2.7.

(i) The function $\#_1(w)$ counts the number of occurrences of 1 in a word $w$ and can be written as

$$\#_1(w) = \sum_{i \in [\ell(w) : P_1(i)} 1.$$ 

(ii) Let $L$ be a regular language defined by the MSOL-formula $\varphi_L$, [14, 34, 86]. The generating function of the number of (contiguous) occurrences of words $u \in L$ in
a word $w$ can be written as

$$\#_L(w) = \sum_{U \subseteq [n]: w[U] = \psi_L} \prod_{i \in U} X,$$

where $\psi_L(U)$ says that $U$ is an interval and that the relativization of $\varphi_L$ to $U$ holds.

The explicit definitions of the expressions mentioned here and in the previous section are given in Appendix A.

2.5 MSOLEVAL and CMSOLEVAL

The formalism MSOLEVAL was designed to deal with definability of graph parameters and graph polynomials, and it has been useful, since, in many applications in algorithmic and structural graph theory and descriptive complexity, \cite{22, 71, 73, 65}. MSOLEVAL can be seen as an analogue of the Skolem elementary functions aka lower elementary functions, \cite{84, 88}, adapted to the framework of meta-finite model theory as defined in \cite{47}.

As hinted by the examples in the previous sections, functions in MSOLEVAL are represented as terms that associate polynomials with structures. Here we precisely define the class of functions in MSOLEVAL for a commutative semiring $S$ which is assumed to contain the natural numbers $\mathbb{N}$.

The polynomials that are associated with the structures will be called MSOL-polynomials. The general notion, introduced in \cite{65} Section 4), is of SOL-polynomials, so we will present a restricted definition that fits our framework more closely.

Let $A$ be a $\tau$-structure, with domain $A$. We first define $\text{MSOL}(\tau)$-monomials as products of constants and indeterminates that ranges over elements of $A$, where $X$ denotes the set of indeterminates:

**Definition 2.8 (MSOL-monomials).** These monomials are defined inductively:

(i) Let $\varphi(x)$ be a formula in $\text{MSOL}(\tau)$, where $x$ is a first order variable. Let $r \in X \cup (S - \{0\})$ be either an indeterminate or an element of $S$. Then

$$\prod_{a: \langle A, a \rangle \models \varphi(a)} r$$

is an $\text{MSOL}(\tau)$-monomial whose value depends on the number of elements of $A$ which satisfy $\varphi$, that is, the cardinality of the set $\{ a \in A \mid \langle A, a \rangle \models \varphi(a) \}$.

(ii) Finite products of $\text{MSOL}(\tau)$-monomials are $\text{MSOL}(\tau)$-monomials.

The polynomials are now defined as sums of monomials where the sum ranges over unary relations $U \subseteq A$ satisfying an $\text{MSOL}(\tau)$-formula $\psi(U)$:
**Definition 2.9 (MSOL-polynomials).** The polynomials definable in MSOL(\(\tau\)) are defined inductively:

(i) MSOL(\(\tau\))-monomials are MSOL(\(\tau\))-polynomials.

(ii) Let \(\psi\) be a \(\tau \cup \{\overline{U}\}\)-formula in MSOL(\(\tau\)) where \(\overline{U} = (U_1, \ldots, U_m)\) is a finite sequence of unary relation symbols not in \(\tau\). Let \(t\) be an MSOL(\(\tau \cup \{\overline{U}\}\))-polynomial. Then

\[ \sum_{\mathcal{U}: \langle A, \mathcal{U} \rangle = \psi(\mathcal{U})} t \]

is an MSOL(\(\tau\))-polynomial.

For simplicity we refer to MSOL(\(\tau\))-polynomials as MSOL-polynomials when \(\tau\) is clear from context.

By substituting elements of \(S\) for the indeterminates in these polynomials, the graph parameters (or word functions) are obtained. Hence the dubbing MSOLEVAL - the parameters in the class are evaluations of MSOL-polynomials:

**Definition 2.10 (MSOLEVAL\(S\)).** Let \(\tau\) be a vocabulary and let \(f\) be a function mapping \(\tau\)-structures \(A\) into \(S\). If there exists an MSOL-polynomial \(p(A, \overline{X})\) and a tuple \(\overline{s}\) of elements from \(S\) such that \(f(A) = p(A, \overline{s})\) for all \(A \in \text{Str}[\tau]\), we say that \(f \in \text{MSOLEVAL}_S\).

CMSOL-monomials, CMSOL-polynomials, and CMSOLEVAL\(S\) are defined analogously.

To reduce notational clutter, we will sometimes say a word function or graph parameter \(f\) are definable in (C)MSOL instead of denoting \(f \in (C)\text{MSOLEVAL}\).
Chapter 3

Meta-Theorems Using Logic

A meta-theorem, in our context, is a statement about a logic \( \mathcal{L} \) relating the definability of a concept in \( \mathcal{L} \) to its mathematical properties. We will be interested in statements of the form:

If a graph parameter \( f \) is definable in \( \mathcal{L} \), then it can be efficiently computed for graph classes of a certain kind.

We focus on the logics MSOL and CMSOL.

We present common notions of width of graphs; the notions of tree-width and clique-width, cf. [58], and note that graphs can have unbounded width of either kind.

We then present Courcelle’s famous theorem for graph properties and tree-width, [28, 19], and its generalization to graph parameters and clique-width, [22], by B. Courcelle, J. A. Makowsky and E. Rotics, followed by J.A. Makowsky’s further generalized meta-theorem involving sum-like inductive classes, [71].

3.1 Common Notions of Width for Graphs

\( k \)-labeled graphs and \( k \)-colored graphs

Let \( k \in \mathbb{N} \), and let \([k]\) denote the set \( \{1, \ldots, k\} \).

**Definition 3.1.** A \( k \)-labeled graph is a graph \( G = (V, E) \) together with a (partial) mapping \( \ell : [k] \rightarrow V \). \( \ell \) is called a labeling and the images of \( \ell \) are called labels.

If \( \ell \) is partial not all labels in \([k]\) are assigned values in \( V \).

**Definition 3.2.** A \( k \)-colored graph is a graph \( G = (V, E) \) together with a mapping \( C : [k] \rightarrow 2^V \). \( C \) is called a coloring and the images of \( C \) are called colors.

\( C \) is often required to be injective, but this is not necessary.

The labeling \( \ell \) can be viewed as a special case of the coloring \( C \), where \( C(i) \) is a singleton for all \( i \in [k] \), or is sometimes empty in the case of a partial labeling.
We denote the class of $k$-colored graphs by $C_{\text{Graphs}}^k$, and the class of $k$-labeled graphs by $L_{\text{Graphs}}^k$. We represent $k$-colored graphs by structures over $\tau_{\text{graphs}} \cup \langle c_1, \ldots, c_k \rangle$ where $c_i$ is a constant symbol for $i = 1, \ldots, k$. We represent $k$-colored graphs by structures over $\tau_{\text{graphs}} \cup \langle C_1, \ldots, C_k \rangle$ where $C_i$ is a unary predicate symbol, for $i = 1, \ldots, k$.

A graph parameter for labeled (colored) graphs is required to be invariant under relabeling (recoloring), in addition to isomorphisms.

**Gluing and tree-width**

Two $k$-labeled graphs $(G_1, \ell_1)$ and $(G_2, \ell_2)$ can be glued together and produce a $k$-labeled graph $(G, \ell) = (G_1, \ell_1) \sqcup_k (G_2, \ell_2)$ by taking the disjoint union of $G_1$ and $G_2$ and $\ell_1$ and $\ell_2$ and identifying elements with the same label. An example can be seen in Figure 3.1.

This operation is also called a $k$-connection.

**Definition 3.3** (Graphs of tree-width $k$).

We start by defining the class $\text{TW}(k)$ of $k$-labeled graphs:

(i) Every $k$-labeled graph of size at most $k + 1$ is in $\text{TW}(k)$.

(ii) $\text{TW}(k)$ is closed under disjoint union $\sqcup$ and $k$-connections $\sqcup_k$.

(iii) Let $\pi : [k] \to [k]$ be a partial relabeling function. If $(G, \ell) \in \text{TW}(k)$ then also $(G, \ell') \in \text{TW}(k)$ where $\ell'(i) = \ell(\pi(i))$.

We say that a graph $G$ is of tree-width at most $k$ if there is a labeling $\ell$ such that $(G, \ell) \in \text{TW}(k)$.

This inductive definition, given in [71], is equivalent to the standard definition (found in [27], for example), cf. [23].

Note that there are graph classes of unbounded tree-width, for example the class of planar graphs includes the $n \times n$ grids which have tree-width $n$.

**Joining and clique-width**

For two $k$-colored graphs $(G_1, C_1)$ and $(G_2, C_2)$ we have similar operations. Let $i, j \in [k]$ be given, where $i \neq j$. We define their $(i, j)$-join, $\bar{\eta}_{i,j}((G_1, C_1), (G_2, C_2)) = (G, C)$, by
Figure 3.2: Joining two graphs with $k = 3$

(i) $V(G) = V(G_1) \sqcup V(G_2)$,

(ii) $C(i) = C_1(i) \sqcup C_2(i)$, for all $i \in [k]$, and

(iii) $E(G) = E(G_1) \sqcup E(G_2) \cup \{(u, v) \in V(G) : u \in C(i), v \in C(j)\}$, which is to connect in the disjoint union all vertices in $C(i)$ with all vertices in $C(j)$.

An example of a $(\bullet, \bullet)$-join is shown in Figure 3.2.

$\bar{\eta}_{i,j}$ is a binary version of the operation $\eta_{i,j}$ used in the definition of the clique-width, cf. [20].

**Definition 3.4** (Graphs of clique-width $2^k$).

We start by defining the class $CW(k)$ of $k$-colored graphs:

(i) Every single-vertex $k$-colored graph is in $CW(k)$.

(ii) $CW(k)$ is closed under disjoint union $\sqcup$ and $(i,j)$-joins $\bar{\eta}_{i,j}$ for $i,j \leq k$ and $i \neq j$.

(iii) Let $\rho : 2^{|k|} \to 2^{|k|}$ be a recoloring function. If $(G,C) \in CW(k)$ then also $(G,C') \in CW(k)$ where $C'(i) = \rho(C(i))$.

A graph $G$ is of **clique-width at most** $2^k$ if there is a coloring $C$ such that $(G,C) \in CW(k)$.

The discrepancy between $2^k$ and $k$ comes from the fact that we allow overlapping colorings. Note that in the original definition a unary operation $\eta_{i,j}$ is used instead of the binary $(i,j)$-join $\bar{\eta}_{i,j}$. However, the two are inter-definable with the help of disjoint union.

Any class of bounded tree-width is also of bounded clique-width, but not vice versa. For example, cliques are of bounded clique-width and unbounded tree-width.

**Definition 3.5.** A **parse tree for $G$** is a witness for the inductive definition describing how $G$ was constructed.

In the context of tree-width, parse trees are also referred to as **tree decompositions**.

It was shown in [12]:

**Theorem 3.6** (H. Bodlaender, 1993). Let $G$ be a graph of tree-width at most $k$. Then we can find a parse tree for $G \in TW(k)$ in polynomial time.

For $G \in CW(k)$ the situation seems slightly worse. It was shown in [77]:

**Theorem 3.7** (S. Oum, 2005). Let $G$ be a graph of clique-width at most $2^k$. Then we can find a parse tree for $G \in CW(3k)$ in polynomial time.
3.2 Meta-theorems for Clique-width and Tree-width

The inductive nature of $TW(k)$ and $CW(k)$, combined with decomposition properties of definable graph parameters which we will see in [Chapter 4] gives rise to some impressive meta-theorems:

**Theorem 3.8** (B. Courcelle, 1990). Let $f$ be an MSOL-definable graph property. Then $f$ can be computed in linear time on graph classes of bounded tree-width.

**Theorem 3.9** (B. Courcelle, J. A. Makowsky, and U. Rotics, 1998). Let $f$ be a CMSOL-definable graph parameter with values in a ring $R$. Then $f$ can be computed in polynomial time on graph classes of bounded clique-width.

*Remark.* For real-valued graph parameters we have to be careful about the model of computation. Either we work in a Turing computable subfield of $\mathbb{R}$, or we use the computational model of Blum-Shub-Smale BSS, cf. [11].

3.3 Generalized Notions of Width

More general inductive definitions of graph classes can be used to define other notions of width, such as sum-like inductive graph classes, which will be defined shortly. We need some definitions, which were introduced in [71] in a more general way. We present the parts relevant to our framework. We begin with the definition of translation schemes, and the transductions and translations induced by them. Informally, a translation scheme is a sequence of formulas which can be “applied” to either structures (transduction) or formulas (translation).

**Definition 3.10** (Translation scheme $\Phi$). Let $\tau$ and $\sigma = \langle R_1, \ldots, R_m \rangle$ be vocabularies, with $r(R_i)$ the arity of the relation $R_i$. Let $L$ be a fragment of SOL such as MSOL or CMSOL. A translation scheme from $\tau$ to $\sigma$ is a sequence $\Phi = \langle \phi, \psi_1, \ldots, \psi_m \rangle$ of formulas of $L(\tau)$ such that $\phi$ has exactly one free first order variable and each $\psi_i$ has $r(R_i)$ distinct free first order variables.

Originally, the definition included more details, but for our purposes this suffices.

**Definition 3.11** (Induced transduction $\Phi^*$). Given a translation scheme $\Phi$, the function $\Phi^* : \text{Str}[\tau] \rightarrow \text{Str}[\sigma]$ is a partial function from $\tau$-structures to $\sigma$-structures defined by $\Phi^*(A) = \mathcal{A}_\Phi$, and

1. the universe of $\mathcal{A}_\Phi$ is the set $A_\Phi = \{ a \in A \mid \langle A, a \rangle \models \varphi(a) \}$. $\mathcal{A}_\Phi$ is a $\sigma$-structure of cardinality at most $|A|$. In addition,
2. the interpretation of $R_i$ in $\mathcal{A}_\Phi$ is the set $\mathcal{A}_\Phi(R_i) = \{ \bar{a} \in A_\Phi^{(R_i)} \mid \langle A, \bar{a} \rangle \models \psi_i(\bar{a}) \}$.

As structures cannot have empty universes, $\Phi^*(A)$ is defined iff $A \models \exists x. \varphi(x)$. 

\[ W = \left\langle \{0\} \cup [6], \{1, 2, 5\}, \{3, 4, 6\} \right\rangle \]

\[ \Phi^*(W) = \begin{array}{c}
\begin{array}{ccc}
1 & 4 & \\
2 & & 5 \\
\end{array}
\end{array} \]

Figure 3.3: The transduction of the word 001101, represented by \( W \), to a graph.

**Example.** The sequence \( \Phi = \langle \varphi, \psi_E \rangle \), where

\[ \varphi(x) = x \approx x \]

and

\[ \psi_E(x, y) = (P_0(x) \land P_1(y)) \lor (P_1(x) \land P_0(y)) \]

is a translation scheme from the vocabulary of binary words, \( \tau_{\text{words}}^{\{0,1\}} = \langle <, P_0, P_1 \rangle \), to the vocabulary of graphs \( \tau_{\text{graphs}} = \langle E \rangle \).

It induces a transduction \( \Phi^* \) which transforms binary words \( w \) into (undirected) graphs, where two vertices are connected if and only if their positions in \( w \) carried different letters, see Figure 3.3.

Now we may present sum-like operations:

**Definition 3.12 (Sum-like operations).** Let \( \Box \) be a binary operation \( \Box : \text{Str}[\sigma] \times \text{Str}[\sigma] \rightarrow \text{Str}[\tau] \). We say that \( \Box \) is sum-like if there is a quantifier-free MSOL translation scheme \( \Phi \) such that for all \( \sigma \)-structures \( A_1, A_2 \), we have

\[ A_1 \Box A_2 = \Phi^*(A_1 \sqcup A_2). \]

**Example.** The \( (1, 2) \)-join of two graphs colored with colors \( \{1, 2, 3\} \) is sum-like, via the transduction induced by \( \Phi = \langle \varphi, \psi_E, \psi_1, \psi_2, \psi_3 \rangle \), where

- \( \varphi(x) = x \approx x \),
- \( \psi_i(x) = C_i(x) \) for \( i \in \{1, 2, 3\} \), and
- \( \psi_E(x, y) = E(x, y) \lor (C_1(x) \land C_2(y)) \lor (C_1(y) \land C_2(x)) \).

**Definition 3.13 (Sum-like inductive).** A graph class \( C \) is sum-like inductive if it can be inductively defined using a finite set of basic graphs \( B_j, j \leq J \) (which may be labeled or colored) and a finite set of sum-like binary operations \( \Box_i, i \leq I \). That is, each \( B_j, j \leq J \) is in \( C \), and whenever \( G_1, G_2 \in C \) then also \( G_1 \Box_i G_2 \in C \) for all \( i \leq I \).
The classes of graphs of fixed tree-width and clique-width are sum-like inductive, cf. [71]. Other examples of sum-like inductive classes of labeled graphs can be found using various graph grammars, cf. [44] [45] [71].

This framework allows the formulation of the following meta-theorem in model theoretic terms.

**Theorem 3.14** (JAM, 2004/14). Let $C$ be sum-like inductive, and let $f$ be a graph parameter in CMSOLEVAL. Then the computation of $f(G)$ is Fixed Parameter Tractable in the size of the parse tree witnessing that $G \in C$.

**Remarks.** Originally, the theorem was stated for MSOL-smooth operations, but the proof in [71] only works for sum-like operations. However, it is not known whether there are MSOL-smooth operations which are not sum-like. The difference between this theorem and Theorem 3.8 or Theorem 3.9 lies in the fact that a parse tree for a graph in a sum-like inductive class is not necessarily computable in polynomial time, as opposed to parse trees for graphs of bounded tree- or clique-width.

Additionally, a graph parameter is Fixed Parameter Tractable (FPT) if it can be computed in time $O(c(k) \cdot n^{d(k)})$ where $n$ is the size of the graph, and $c(k), d(k)$ are functions depending on the parameter $k$, but independent of the size of the graph, cf. [28] [40]. Here the parameter is hidden in the fact that $C$ is sum-like inductive.

We will see that the definability assumption on the graph parameters in these theorems can be replaced by a finiteness assumption on the rank of their Hankel matrices, which are discussed in the next chapter.

Lastly, the following measure of complexity for formulas and definable functions will be useful in formulating and proving bounds on these functions:

**Definition 3.15.** The quantifier rank $qr(\phi)$ of an $L$-formula $\phi$ is a function

$$qr : L(\tau) \rightarrow \mathbb{N}$$

such that:

(i) atomic formulas $\phi$ have $qr(\phi) = 0$, and

(ii) boolean operations and translations induced by quantifier-free translation schemes preserve maximal quantifier rank.

The quantifier rank $qr(f)$ of a definable function $f$ is the maximal quantifier rank of the formulas which appear in the definition of $f$.

The technical details are not needed here, we just note that there are, up to logical equivalence, only finitely many MSOL($\tau$) formulas of quantifier rank $q \in \mathbb{N}$ for a fix set of free variables.
Chapter 4

Hankel Matrices and the Finite Rank Theorem

This chapter surveys uses of Hankel matrices for graph parameters, such as in the characterization of real-valued partition functions in [41, 68], and in B. Godlin, T. Kotek and J. A. Makowsky’s [46], which related definability to rank-finiteness of Hankel matrices. As a preface to the next chapter, we illustrate a limitation of theorems involving definability conditions.

4.1 Hankel Matrices for General Parameters

Given a parameter $f$ for $\sigma$-structures along with a binary operation

$$\Box : \text{Str}[\tau] \times \text{Str}[\tau] \to \text{Str}[\sigma]$$

for vocabularies $\sigma$ and $\tau$, the Hankel matrix $H(f, \Box)$ is an infinite matrix whose rows and columns are indexed by all $\tau$-structures $A_1, A_2, \ldots$, whose entries are given by:

$$H(f, \Box)_{A_i, A_j} = f(A_i \Box A_j).$$

So it looks something like this:

<table>
<thead>
<tr>
<th>$H(f, \Box)$</th>
<th>$A_1$</th>
<th>$A_2$</th>
<th>$A_3$</th>
<th>$\cdots$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$A_1$</td>
<td>$f(A_1 \Box A_1)$</td>
<td>$f(A_1 \Box A_2)$</td>
<td>$f(A_1 \Box A_3)$</td>
<td>$\cdots$</td>
</tr>
<tr>
<td>$A_2$</td>
<td>$f(A_2 \Box A_1)$</td>
<td>$f(A_2 \Box A_2)$</td>
<td>$f(A_2 \Box A_3)$</td>
<td>$\cdots$</td>
</tr>
<tr>
<td>$A_3$</td>
<td>$f(A_3 \Box A_1)$</td>
<td>$f(A_3 \Box A_2)$</td>
<td>$f(A_3 \Box A_3)$</td>
<td>$\cdots$</td>
</tr>
<tr>
<td>$\vdots$</td>
<td>$\vdots$</td>
<td>$\vdots$</td>
<td>$\vdots$</td>
<td>$\ddots$</td>
</tr>
</tbody>
</table>
4.2 Hankel Matrices for Graph Parameters and Word Functions

Connection matrices

Connection matrices, introduced by L. Lovász in [69], are a kind of Hankel matrix that involve \( k \)-connections.

Let \( f : L\text{Graphs}_k \rightarrow \mathcal{F} \) be an \( \mathcal{F} \)-valued graph parameter on \( k \)-labeled graphs. That is, \( f \) is invariant under isomorphisms and labelings.

**Definition 4.1.** The **connection matrix** \( H(f, \sqcup_k) \) is the infinite matrix where rows and columns are indexed by \( k \)-labeled graphs and the entry \( H(f, \sqcup_k)_{G,G'} \) is given by \( f(G \sqcup_k G') \).

**Definition 4.2** (Connection rank). A graph parameter \( f \) has **finite connection rank** if all the connection matrices \( H(f, \sqcup_k) \) have finite rank.

Connection matrices were used by M. Freedman, L. Lovász and A. Schrijver in [41, 68] to study real-valued graph parameters and their relation to partition functions, which are defined by counting weighted homomorphisms.

**Definition 4.3** (Partition function). Let \( H = (V_H, E_H) \) be a fixed graph, and let \( \alpha : V_H \to \mathbb{R} \) and \( \beta : E_H \to \mathbb{R} \) be real-valued functions (weights). For a graph \( G = (V_G, E_G) \) we define the **partition function**

\[
Z_{H,\alpha,\beta}(G) = \sum_{h : G \to H} \prod_{v \in V_G} \alpha(h(v)) \prod_{(u,v) \in E_G} \beta(h(u), h(v)).
\]

Finite connection rank and partition functions are related in the following characterization:

**Theorem 4.4** (M. Freedman, L. Lovász and A. Schrijver, 2007).

A real-valued graph parameter \( f \) can be presented as a partition function \( f(G) = Z_{H,\alpha,\beta}(G) \) for some \( H, \alpha, \beta \), if and only if all its connection matrices \( H(f, \sqcup_k) \) have finite rank and are positive definite.

In [68] many variations of this theorem are discussed using different notions of connections of labeled graphs.

Join matrices

A similar definition arises for the join operations in the inductive definition of \( \text{CW}(k) \). In this case, there are seemingly many different operations, one for each pair of colors \( i, j \in [k], i \neq j \).
Definition 4.5. The \((i,j)\)-join matrix \(H(f, \eta_{i,j})\) is the infinite matrix where rows and columns are indexed by \(k\)-colored graphs and the entry \(H(f, \eta_{i,j})_{G,G'}\) is given by \(f(\eta_{i,j}(G, G'))\).

Remark. Consider the matrix \(H(f, \eta_{1,2})\). Note that all the join matrices \(H(f, \eta_{i,j})\) are submatrices of \(H(f, \eta_{1,2})\), since after a suitable recoloring the rows and columns correspond to all the graphs with all the possible \(k\)-colorings. We therefore denote \(\eta_{1,2} = \eta_k\), and speak of the join matrix of \(f\), \(H(f, \eta_k)\).

If a graph parameter has a join matrix of finite rank we say it has finite join rank.

Concatenation matrices

Let \(\Sigma\) be a finite alphabet, \(F\) a field and let \(\circ\) denote the concatenation operation on words: \(u \circ v = uv\).

Definition 4.6. Given a word function \(f : \Sigma^* \rightarrow F\), its concatenation matrix \(H(f, \circ)\) is the infinite matrix whose rows and columns are indexed by words in \(\Sigma^*\) and whose entries are given by \(H(f, \circ)_{u,v} = f(u \circ v)\).

If a word function has a concatenation matrix of finite rank, we say it has finite concatenation rank.

Hankel matrices of word functions were studied in [15, 18].

Proposition 4.6.1 (A. Cobham [18]). The Hankel matrix \(H(f, \circ)\) of a word function \(f\) with concatenation has rank 1 iff there is \(c \in F\) and a multiplicative word function \(g\) such that \(f = c \cdot g\).

Hankel matrices involving sum-like operations can be defined similarly.

4.3 The Finite Rank Theorem

In [46] the following Finite Rank Theorem is proved, relating definability in CMSOL to the rank of Hankel matrices:

Theorem 4.7 (Finite Rank Theorem). Let \(f\) be a real-valued graph parameter. If \(f\) is definable in CMSOL, then \(f\) has finite connection rank.

The same holds for a wider class of Hankel matrices arising from sum-like binary operations on labeled graphs.

The Finite Rank Theorem is very useful in showing non-definability, [66, 67], often in a more convenient way than the usual methods involving Ehrenfeucht-Fraissé games.
Uncountably many graph parameters with finite rank

Here we address the natural question of whether the converse of the finite rank theorem holds or not. We show that rank-finiteness of connection matrices of graph parameters does not imply definability in CMSOL.

We first need an observation.

**Definition 4.8.** A graph is \( k \)-connected, if there is no set of \( k \) vertices, such that their removal results in a graph which is not connected.

Obviously we have:

**Lemma 4.9.** Let \( G_1 \) and \( G_2 \) be two \( k \)-labeled graphs of size greater than \( k \) and \( G = G_1 \sqcup_k G_2 \). Then \( G \) is not \( k \)-connected.

For a subset \( A \subseteq \mathbb{N} \) we define the graph parameter

\[
  f_A(G) = \begin{cases} 
  |V(G)| & \text{if } G \text{ is } k_0\text{-connected and } |V(G)| \in A \\
  0 & \text{else}
  \end{cases}
\]

**Lemma 4.10.** Let \( \mathcal{F} \) be a field which contains \( \mathbb{N} \), and let \( k_0 \in \mathbb{N} \) and \( A \subseteq \mathbb{N} \). Let \( n_{k_0}(k) \) denote the number of \( k \)-labeled graphs of size \( \leq k_0 \). Then for every \( k \leq k_0 \), \( \text{H}(f_A, \sqcup_k) \) has rank at most \( n_{k_0}(k) + 2^{n_{k_0}(k)} \).

**Proof.** By Lemma 4.9, if the graph \( G_1 \sqcup_k G_2 \) is \( k_0 \)-connected, then either \( G_1 \) is \( k_0 \)-connected and \( G_2 \) is of size \( \leq k_0 \), or vice versa. So any non-zero entry in \( \text{H}(f_A, \sqcup_k) \) must be in the first \( n_{k_0}(k) \) rows and the first \( n_{k_0}(k) \) columns. Thus if we take all the \( n_{k_0}(k) \)-length binary vectors followed by infinitely many zeros, they would generate all the rows except the first \( n_{k_0}(k) \) rows, giving us rank at most \( n_{k_0}(k) + 2^{n_{k_0}(k)} \).

**Theorem 4.11.** Let \( k_0 \in \mathbb{N} \) and \( \mathcal{F} \) a field. There are continuum many graph parameters \( f \) with values in \( \mathcal{F} \) with \( r(f, \sqcup_k) \leq n_{k_0}(k) + 2^{n_{k_0}(k)} \) for each \( k \leq k_0 \).

**Proof.** There are continuum many subsets of \( \mathbb{N} \) and for two different sets \( A, B \subseteq \mathbb{N} \) the parameters \( f_A \) and \( f_B \) are different.

As there are only countably many definable graph parameters, we have that there are graph parameters with finite connection rank which are not definable in CMSOL. In the next chapter we take a small digression from the main theme of the thesis, and discuss when the converse of Theorem 4.7 does hold.
Chapter 5

MSOL and Word Functions

Classical automata theory gives various characterizations of regular languages; a characterization via Finite State Automata, Kleene’s characterization via regular expressions, [62], Büchi, Elgot, and Trakhtenbrot’s characterization via MSOL-definability, [14, 34, 86], and the Kleene-Schützenberger characterization via power series, [83]. In this chapter, we give an analogue of the Büchi-Elgot-Trakhtenbrot characterization of regular languages for word functions by proving that a word function is in MSOLEVAL if and only if it is recognizable by a weighted automaton.

This implies that the converse direction of the finite rank theorem holds in the case of word functions, giving us an “if and only if” theorem, in contrast to the general case, cf. Theorem 4.11.

In 2005, M. Droste and P. Gastin introduced a formalism of weighted monadic second order logic (WMSOL) [29], and the syntactic fragment RMSOL was used to characterize word functions recognizable by weighted automata [30]. We immediately conclude that MSOLEVAL and RMSOL have the same expressive power over words, and give another proof to this by presenting translations from MSOLEVAL to RMSOL and vice versa. We end the chapter with a comparison between the two formalisms.

5.1 Weighted Automata

Let $\mathcal{S}$ be a commutative semiring (see Definitions 7.3, 7.4, 7.5), $\Sigma$ a finite alphabet, and $w$ a word in $\Sigma^*$. $\mathcal{S}$-valued functions on words $f : \Sigma^* \rightarrow \mathcal{S}$ are called word functions, following [15, 18]. They are also called formal power series in [9], where indeterminates $X$ are indexed by words $w$ and the coefficient of $X_w$ is $f(w)$.

We will often abuse notation and denote both the set of row-vectors of length $r$ and the set of column-vectors of length $r$ over $\mathcal{S}$ by $\mathcal{S}^r$. We will, however, always make our intention clear.

**Definition 5.1.** A weighted automaton $A$ of size $r \in \mathbb{N}^+$ over $\mathcal{S}$ is given by:

(i) a set of states $Q$ of size $r$,
(ii) two row-vectors \( \alpha, \gamma \in S_r \), respectively called the initial vector and final vector, and

(iii) for each \( \sigma \in \Sigma \), a transition matrix \( \mu(\sigma) \in S_{r \times r} \), where the entry \( \mu(\sigma)_{p,q} \) indicates the weight of the transition \( p \xrightarrow{\sigma} q \) for \( p, q \in Q \).

For a matrix or vector \( M \) we denote by \( M^T \) the transpose of \( M \).

For a word \( w = \sigma_1 \sigma_2 \ldots \sigma_{\ell(w)} \), the automaton \( A \) defines the function \( f_A \) of \( A \):

\[
f_A(w) = \alpha \cdot \mu(\sigma_1) \cdot \ldots \cdot \mu(\sigma_{\ell(w)}) \cdot \gamma^T.
\]

\( f_A \) is also called the external function of \( A \), [15], or the behavior of \( A \), [29].

**Definition 5.2.** A word function \( f : \Sigma^* \to S \) is recognized by a weighted automaton \( A \) if \( f = f_A \), and \( f \) is recognizable if there exists a weighted automaton \( A \) which recognizes it.

If the initial vector \( \alpha \) and the transition matrices \( \mu(\sigma), \sigma \in \Sigma \) are stochastic, and \( \gamma \) is binary, then \( A \) is a usual probabilistic automaton, where \( f_A(w) \) computes the probability of acceptance of \( w \). If \( \gamma \) has entries \( \gamma_q \) with arbitrary real values interpreted as costs of return at state \( q \), \( f_A(w) \) is the expected cost of \( w \).

Recognizable word functions were first introduced in the study of stochastic automata by A. Heller [57]. Probabilistic automata were introduced in [79] and investigated in [78]. More recently, recognizable word functions have been used in machine learning [8, 53, 10], in verification of finite-state programs, [4, 75], in program synthesis, [16, 17], in digital image compression, [24], and in speech–to–text processing, [76]. For a comprehensive survey, see [30] and the Handbook of Weighted Automata [31].

**Example of a weighted automaton.** For the alphabet \( \Sigma = \{0, 1\} \), the function \( \#_1(w) \) which counts the number of ones in \( w \) is recognized by the automaton \( A \) which is given by:

(i) \( Q = \{q_1, q_2\} \),

(ii) \( \alpha = \begin{pmatrix} 0 & 1 \end{pmatrix}, \gamma = \begin{pmatrix} 1 & 0 \end{pmatrix} \),

(iii) \( \mu(0) = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \mu(1) = \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix} \).

We illustrate its computation on a word \( w = 110 \):

\[
f_A(110) = \alpha \cdot \mu(1) \cdot \mu(1) \cdot \mu(0) \cdot \gamma^T
= \begin{pmatrix} 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix}
= \begin{pmatrix} 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 2 & 1 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix}
= 2.
\]
A classical result of G.W. Carlyle and A. Paz [15] characterizes real-valued word functions $f$ recognizable by weighted automata in algebraic terms. The theorem was stated with a different notion of rank, called function rank, but the following formulation is equivalent:

**Theorem 5.3** (G.W. Carlyle and A. Paz, 1971). A real-valued word function is recognizable iff it has finite concatenation rank.

### 5.2 A Finite Rank Theorem for Word Functions

A finite rank theorem like [Theorem 4.7](#) also holds for definable word functions. It is proved through a bilinear decomposition theorem, which allows us to compute the value $f(w)$ for a word $w = u \circ v$ using only $f(u)$ and $f(v)$. Together with [Theorem 5.3](#) this immediately implies that MSOL-definable real-valued word functions are recognizable by weighted automata. We will see in Section 5.3 that this is actually true for arbitrary commutative semirings.

We first demonstrate this bilinear decomposition property with two examples. Let $\Sigma = \{0, 1\}$ and $u, v, w \in \Sigma^*$.

We consider again the function $\sharp_1(w)$, and the function $b_1(w)$ that counts the number of blocks of ones in $w$. A block of ones in $w$ is a maximal set of consecutive positions $i \in [\ell(w)]$ in the word $w$ for which $P_1(i)$ holds, i.e., carry the letter 1.

Clearly, we have the decomposition:

$$\sharp_1(u \circ v) = \sharp_1(u) + \sharp_1(v).$$

However,

$$b_1(u \circ v) = \begin{cases} 
 b_1(u) + b_1(v) - 1 & \text{if } P_1(\ell(u)) \text{ and } P_1(1) \\
 b_1(u) + b_1(v) & \text{else}
\end{cases}$$

To see the decomposition of $b_1$ we introduce auxiliary functions.

(i) $f_1(w)$ counts the number of blocks of ones in $w$ which include the first position.

(ii) $l_1(w)$ counts the number of blocks of ones in $w$ which include the last position.

(iii) $i_1(w)$ counts the number of blocks of ones in $w$ which exclude the first and last position.

(iv) $c(w) = 1$, the constant function with value 1.

It is shown in [Appendix A](#) that these function are in MSOLEVAL$_\mathbb{N}$.

Then we have the decomposition:

$$b_1(w) = f_1(w) + l_1(w) + i_1(w) - f_1(w)l_1(w).$$

Furthermore, we have $f_1(w), l_1(w) \in \{0, 1\}$ and $f_1, l_1, i_1, c$ can be decomposed
themselves:

\[ f_1(u \circ v) = f_1(u), \quad (5.1) \]
\[ l_1(u \circ v) = l_1(v), \quad (5.2) \]
\[ i_1(u \circ v) = i_1(u) + i_1(v) + f_1(v) + l_1(u) - f_1(v)l_1(u), \quad (5.3) \]
\[ c(u \circ v) = 1. \quad (5.4) \]

For a word \( w \), let \( B(w) = (f_1(w), l_1(w), i_1(w), c(w)) \).

**Proposition 5.3.1.** There are matrices \( M^f, M^l, M^i, M^c \in \mathbb{N}^{4 \times 4} \) such that:

\[ f_1(u \circ v) = B(u) \cdot M^f \cdot B(v)^T, \]
\[ l_1(u \circ v) = B(u) \cdot M^l \cdot B(v)^T, \]
\[ i_1(u \circ v) = B(u) \cdot M^i \cdot B(v)^T, \]
\[ c(u \circ v) = B(u) \cdot M^c \cdot B(v)^T. \]

**Proof.** Using the equations (5.1) - (5.4) one easily verifies that

\[ M^f = \begin{pmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}, \quad M^l = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}, \]
\[ M^i = \begin{pmatrix} 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 \end{pmatrix}, \quad M^c = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}. \]

Alternatively, we can replace \( i_1(w) \) by \( b_1(w) \) and use the decomposition

\[ b_1(u \circ v) = b_1(u) + b_1(v) - l_1(u)f_1(v), \]

which gives us

\[ M^b = \begin{pmatrix} 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}. \]

Then we replace the vector \( B \) in Proposition 5.3.1 by \( B'(w) = (f_1(w), l_1(w), b_1(w), c(w)) \), and the function \( i_1(w) \) by \( b_1(w) \).

These two examples are typical in the following sense:
**Theorem 5.4.** Let $f \in \text{MSOLEVAL}_S$ be a word function $\Sigma^* \rightarrow S$ of quantifier rank $qr(f)$. There are:

(i) a function $\beta : \mathbb{N} \rightarrow \mathbb{N}$,

(ii) a finite row-vector $F = (g_1, \ldots, g_{\beta(qr(f))})$ of functions in $\text{MSOLEVAL}_S$ of length $\beta(qr(f))$, with $f = g_i$ for some $i \leq \beta(qr(f))$,

(iii) and for each $g_i \in F$, matrices $M^{(i)} \in S^{\beta(qr(f)) \times \beta(qr(f))}$, such that $g_i(u \circ v) = F(u) \cdot M^{(i)} \cdot F^T(v)$, where $F(w)$ is shorthand for $(g_1(w), \ldots, g_{\beta(qr(f))}(w))$.

**Sketch of proof.** The first proof is in [22, Theorem 45]. The bilinear version was only formulated later in [71, Theorem 6.4], but uses the same proof. Here we merely note that $F$ can be chosen to consist of all the functions in $\text{MSOLEVAL}_S$ of quantifier rank at most $qr(f)$. This is a rough estimate for $\beta(qr(f))$. The examples of $\#_1$ and $b_1$ above show that $F$ can often be much smaller.

We will use the following version of the **Finite Rank Theorem** to characterize recognizable word functions:

**Theorem 5.5.** Let $f \in \text{MSOLEVAL}_S$ be a word function $f : \Sigma^* \rightarrow S$ of quantifier rank $qr(f)$. Then the Hankel matrix $H(f, \circ)$ has rank at most $\beta(qr(f))$.

**Proof.** Let $f : \Sigma^* \rightarrow S$ be a word function in $\text{MSOLEVAL}_S$ with quantifier rank $qr(f)$. Then by **Theorem 5.4** there exist a row-vector $F = (g_1, \ldots, g_{\beta(qr(f))})$

with $f = g_i$ for some $i \leq \beta(qr(f))$, and matrices $M^{(i)}, \ldots, M^{(\beta(qr(f)))} \in S^{\beta(qr(f)) \times \beta(qr(f))}$ such that $f(u \circ v) = F(u) \cdot M^f \cdot F^T(v) = F(u) \cdot M^{(i)} \cdot F^T(v)$.

Let $v_i, i \in \mathbb{N}$ be an enumeration of the words in $\Sigma^*$, and let $H(f, \circ)$ be the Hankel matrix whose rows and columns are indexed by this enumeration. We have that the entry $H(f, \circ)_{v_i,v_j}$ is given by:

$$f(v_i \circ v_j) = F(v_i) \cdot M^f \cdot F^T(v_j)$$

$$= \sum_{k=1}^{\beta(qr(f))} g_k(v_i) \cdot M^f_{k,1} \cdot g_1(v_j) + \cdots + \sum_{k=1}^{\beta(qr(f))} g_k(v_i) \cdot M^f_{k,\beta(qr(f))} \cdot g_{\beta(qr(f))}(v_j).$$
Therefore the $i$th row is given as a linear combination of the infinite row-vectors

$$B_k = (g_k(v_1), g_k(v_2), \ldots) \text{ for } k \leq \beta(qr(f)),$$

implying $H(f, \circ)$ has rank at most $\beta(qr(f))$. $\blacksquare$

In [71], the row-vector $F = (g_1, \ldots, g_{\beta(qr(f))})$ in Theorem 5.4 above is called a splitting set for $f$. In [46], graph parameters satisfying the conclusion of Theorem 5.4 above are called weakly multiplicative.

### 5.3 Characterizing Recognizable Word Functions

Let $S$ be a commutative semiring. In this section we prove the following characterization:

**Theorem 5.6.** Let $f$ be an $S$-valued word function. Then $f$ is in MSOLEVAL$_S$ iff $f$ is recognized by some weighted automaton $A$ over $S$.

First, we need a few definitions.

We denote by $S^\Sigma^*$ the set of word functions $\Sigma^* \rightarrow S$.

**Definition 5.7.** A module $M$ is a subset of $S^\Sigma^*$ closed under pointwise addition of word functions over $S$, and point-wise multiplication with elements of $S$.

Note that $S^\Sigma^*$ itself is a module.

**Definition 5.8.** A module $M \subseteq S^\Sigma^*$ is finitely generated if there is a finite set $F \subseteq S^\Sigma^*$ such that each element $f \in M$ can be written as a linear combination of elements in $F$.

Note that if $F$ generates $M$, then $F \subseteq M$.

These definitions are instances of more general notions presented in Chapter 7.

Let $w$ be a word and $f$ a word function. Following [9], we denote by $w^{-1}f$ the word function $g$ defined by:

$$g(u) = (w^{-1}f)(u) = f(w \circ u)$$

**Definition 5.9.** $M$ is stable if for all words $w \in \Sigma^*$ and for all $f \in M$, the word function $w^{-1}f$ is also in $M$.

**MSOL-definable word functions are recognizable.**

To prove the “only if” direction of Theorem 5.6 we use Theorem 5.4 and a theorem first proved by G. Jacob, [60]:

**Theorem 5.10** (G. Jacob 1975). Let $f$ be a word function $f : \Sigma^* \rightarrow S$. Then $f$ is recognizable by a weighted automaton over $S$ iff there exists a finitely generated stable module $M \subseteq S^\Sigma^*$ which contains $f$.
We reformulate this direction in order to apply Theorem 5.10.

**Theorem 5.11.** Let $S$ be a commutative semiring and let $f \in \text{MSOLEVAL}_S$ be a word function of quantifier rank $qr(f)$.

There are:

(i) a function $\beta : \mathbb{N} \to \mathbb{N}$, and

(ii) a finite row-vector $F = (g_1, \ldots, g_{\beta(qr(f))})$ of functions in $\text{MSOLEVAL}_S$ of length $\beta(qr(f))$, with $f = g_i$ for some $i \leq \beta(qr(f))$

such that the module $M[F]$ generated by $F$ is stable.

**Proof.** Consider $F$ and the matrices $M^{(i)}$ from Theorem 5.4.

We have to show that for every fixed word $w$ and $f \in M[F]$, the function $w^{-1}f$ is in $M[F]$. As $f \in M[F]$, there is a row-vector

$$B = (b_1, \ldots, b_{\beta(qr(f))}) \in S^{\beta(qr(f))}$$

such that

$$f(w) = B \cdot F^T(w)$$

for every fixed word $w$. Here $F(w)$ is again shorthand for $(g_1(w), \ldots, g_{\beta(qr(f))}(w))$.

Let $u$ be a word. We compute $(w^{-1}f)(u)$:

$$(w^{-1}f)(u) = f(w \circ u) = B \cdot F^T(w \circ u)$$

$$= \sum_{i=1}^{\beta(qr(f))} b_i \cdot g_i(w \circ u) = \sum_{i=1}^{\beta(qr(f))} b_i \cdot F(w) \cdot M^{(i)} \cdot F^T(u).$$

We put $C_i = b_i \cdot F(w) \cdot M^{(i)}$ and observe that $C_i \in S^{\beta(qr(f))}$.

By taking the row-vector $C = \sum_{i=1}^{\beta(qr(f))} C_i$, we get $(w^{-1}f)(u) = C \cdot F^T(u)$, hence

$$w^{-1}f \in M[F].$$

**Recognizable word functions are MSOL-definable**

For the “if” direction we consider a given weighted automaton and construct an appropriate MSOL-polynomial for it.

**Proof.** Let $A$ be a weighted automaton of size $r$ over $S$ for words in $\Sigma^*$. For a word $w$ of length $\ell(w) = n$, described by a function $w : [n] \to \Sigma$, the automaton $A$ defines the function

$$f_A(w) = \alpha \cdot \mu_{w(1)} \cdot \ldots \cdot \mu_{w(n)} \cdot \gamma^T. \quad (5.5)$$

We have to show that $f_A \in \text{MSOLEVAL}_S$. 

31
For notational convenience, we denote $M_{i,j}^a = (\mu_a)_{i,j}$.

Equation 5.5 is a product of $n$ matrices and two vectors. Let $P$ be the product of these matrices:

$$P = \prod_{k=1}^{n} \mu_{w(k)}.$$

Using matrix algebra, we get for the entry $P_{a,b}$ of $P$:

$$P_{a,b} = \sum_{i_{n-1}=1}^{r} \left( \sum_{i_{n-2}=1}^{r} \left( \ldots \left( \sum_{i_1=1}^{r} \left( \sum_{i_{n-1}=1}^{r} \left( M_{a,i_1}^{w(1)} \cdot M_{i_1,i_2}^{w(2)} \cdot M_{i_2,i_3}^{w(3)} \right) \ldots \right) M_{i_{n-1},b}^{w(n)} \right) \right) \right).$$

Let $\pi : [n-1] \rightarrow [r]$ be the function with $\pi(k) = i_k$. We rewrite $P_{a,b}$ as:

$$P_{a,b} = \sum_{\pi:[n-1] \rightarrow [r]} \left( M_{a,\pi(1)}^{w(1)} \cdot M_{\pi(1),\pi(2)}^{w(2)} \cdot \ldots \cdot M_{\pi(n-1),b}^{w(n)} \right)$$

Equation 5.6

Next we compute the $b$ coordinate of the vector $\alpha \cdot P$:

$$(\alpha \cdot P)_b = \sum_{i=1}^{r} \alpha_i \cdot P_{i,b}$$

Therefore,

$$f_A(w) = \alpha \cdot P \cdot \gamma^T = \sum_{b=1}^{r} (\alpha \cdot P)_b \cdot \gamma_b$$

$$= \sum_{b=1}^{r} \left( \sum_{a=1}^{r} \alpha_a \cdot P_{a,b} \right) \cdot \gamma_b = \sum_{a,b \leq r} \alpha_a \cdot P_{a,b} \cdot \gamma_b$$

and by using Equation 5.6 for $P_{a,b}$ we get:

$$f_A(w) = \sum_{a,b \leq r} \alpha_a \cdot \left( \sum_{\pi:[n-1] \rightarrow [r]} \left( M_{a,\pi(1)}^{w(1)} \cdot M_{\pi(1),\pi(2)}^{w(2)} \cdot \ldots \cdot M_{\pi(n-1),b}^{w(n)} \right) \right) \cdot \gamma_b$$

Now let $\pi' : [n] \cup \{0\} \rightarrow [r]$ be the function for which $\pi'(0) = a, \pi'(n) = b$ and $\pi'(k) = \pi(k) = i_k$ for $1 \leq k \leq n - 1$. Then we get:

$$f_A(w) = \sum_{\pi' : [n] \cup \{0\} \rightarrow [r]} \alpha_{\pi'(0)} \cdot \left[ M_{\pi'(0),\pi'(1)}^{w(1)} \cdot \ldots \cdot M_{\pi'(n-1),\pi'(n)}^{w(n)} \right] \cdot \gamma_{\pi'(n)}$$

$$= \sum_{\pi' : [n] \cup \{0\} \rightarrow [r]} \alpha_{\pi'(0)} \cdot \left( \prod_{k \in [n]} M_{\pi'(k-1),\pi'(k)}^{w(k)} \right) \cdot \gamma_{\pi'(n)}$$

Equation 5.7

32
To convert Equation 5.7 into an expression in MSOLEVAL$_S$ we use a few lemmas:

First, let $S$ be any set and $\pi : S \to [r]$ be any function. $\pi$ induces a partition of $S$ into sets $U_1^\pi, \ldots, U_r^\pi$ by $U_i^\pi = \{ s \in S : \pi(s) = i \}$. Conversely, every partition $\mathcal{U} = (U_1, \ldots, U_r)$ of $S$ induces a function $\pi_{\mathcal{U}}$ by setting $\pi_{\mathcal{U}}(s) = i$ for $s \in U_i$. To transition between functions $\pi$ with finite range $[r]$ and partitions into $r$ sets we use the following lemma:

**Lemma 5.12.** Let $E(\pi)$ be any expression depending on $\pi$. Then

$$\sum_{\pi : S \to [r]} E(\pi) = \sum_{\mathcal{U} : \text{Partition}(U_1, \ldots, U_r)} E(\pi_{\mathcal{U}}),$$

where $\mathcal{U}$ ranges over all partitions of $S$ into $r$ sets $U_i : i \in [r]$. We have seen in Section 2.2 that $\text{Partition}(U_1, \ldots, U_r)$ can be written in MSOL.

To convert the factors $\alpha_{\pi'\prime(0)}$ and $\gamma_{\pi'\prime(n)}$ we proceed as follows:

**Lemma 5.13.** Let $\alpha_i$ be the unique value of the coordinate of $\alpha$ such that $0 \in U_i$. Similarly, let $\gamma_i$ be the unique value of the coordinate of $\gamma$ such that $n \in U_i$. We have:

$$\alpha_{\pi'\prime(0)} = \prod_{i=1}^r \prod_{0 \in U_i} \alpha_i$$

$$\gamma_{\pi'\prime(n)} = \prod_{i=1}^r \prod_{n \in U_i} \gamma_i$$

**Proof.** First we note that as $\mathcal{U}$ is the partition induced by $\pi'$, the restriction of $\pi'$ to $U_i$ is constant for all $i \in [r]$. Next we note that the product ranging over the empty set gives the value 1.

Similarly, to convert the factor $\prod_{k \in [n]} M_{\pi'(k-1),\pi'(k)}^{w(k)}$ use the following lemma:

**Lemma 5.14.** Let $m_{i,j,w(v)}$ be the unique value of the $(i,j)$-entry of the matrix $\mu_{w(v)}$ such that $v \in U_i$ and $v + 1 \in U_j$. We have:

$$\prod_{k \in [n]} M_{\pi'(k-1),\pi'(k)}^{w(k)} = \prod_{i,j=1}^r \left( \prod_{v-1 \in U_i, v \in U_j} m_{i,j,w(v)} \right).$$

Using the fact that every element which is the interpretation of a term in $S$ can be written as an expression in MSOLEVAL$_S$, [64, Lemma 3.3.2], we can write $U_i(v)$ instead of $v \in U_i$, and see that the monomials of Lemmas 5.12, 5.13 and 5.14 are indeed in MSOLEVAL$_S$. Now we apply the fact that the pointwise product of two word functions in MSOLEVAL$_S$ is again a function in MSOLEVAL$_S$, [64, Proposition 3.3.6], to Lemmas 5.12, 5.13 and 5.14 and complete the proof of Theorem 5.6. 

---

33
5.4 Weighted MSOL and MSOLEVAL

Given a word function $f$, knowing the rank of its concatenation matrix $H(f, \circ)$ or finding an appropriate weighted automaton $A$ reveals very little about the function $f$, as opposed to having an MSOL expression describing $f$.

The need to have a descriptive language for word functions was recognized by others, e.g. M. Droste and P. Gastin. They introduced a weighted version of MSOL where the usual satisfaction relation, which associates with a pair $(w, \varphi)$ of a word $w$ and formula $\varphi$ a truth value 0 or 1, is replaced by a weight function, which associates with $(w, \varphi)$ an element in the numeric domain $S$. Then they identify a restricted subset $\text{RMSOL} \subset \text{MSOL}$ for which they prove the following:

**Theorem 5.15** (M. Droste, P. Gastin). A word function $f$ is recognizable iff it is the weight function of some formula in $\text{RMSOL}$.

In their approach $S$ does not have to be a field, but it suffices to assume that $S$ is a commutative semiring. If additionally the commutative semiring is locally finite, the theorem can be extended to full MSOL (fields of characteristic 0 are not locally finite).

In this section we present the formalism of weighted MSOL, $\text{WMSOL}$, and give a translation from it into $\text{MSOLEVAL}$ and vice versa. Additionally, we compare the formalisms.

In [29, 30], and [13], two fragments of weighted MSOL are discussed. One is based on unambiguous formulas (a semantic concept), the other on step formulas based on the boolean fragment of weighted MSOL (a syntactic definition). The two fragments have equal expressive power, as stated in [13], and characterize the functions recognizable by weighted automata. We denote both versions by $\text{RMSOL}$.

**Syntax of WMSOL**

The definitions and properties of WMSOL and its fragments are taken literally from [13].

**Definition 5.16.** The syntax of formulas $\varphi$ of weighted MSOL, denoted by $\text{WMSOL}$, is given inductively in Backus–Naur form by

\[
\varphi ::= k \mid P_a(x) \mid \neg P_a(x) \mid x \leq y \mid \neg x \leq y \mid x \in U \mid x \notin U \\
\mid \varphi \lor \psi \mid \varphi \land \psi \mid \exists x. \varphi \mid \exists U. \varphi \mid \forall x. \varphi \mid \forall U. \varphi
\]

where $k \in S$, $a \in \Sigma$.

The set of weighted MSOL-formulas over a commutative semiring $S$ and the alphabet $\Sigma$ is denoted by $\text{WMSOL}(S, \Sigma)$.

$b\text{MSOL}$ formulas and $b\text{MSOL}$-step formulas are defined below. $b\text{MSOL}$ is the boolean fragment of WMSOL, and its name is justified by Lemma 5.20.
Definition 5.17. The syntax of weighted bMSOL is given by

$$\varphi ::= 0 \mid 1 \mid P_a(x) \mid x \leq y \mid x \in U \mid \neg \varphi \mid \varphi \land \psi \mid \forall x. \varphi \mid \forall U. \varphi$$

where \(a \in \Sigma\).

In [13], \(L\)-step formulas are given by the grammar

$$\varphi ::= k \mid \alpha \mid \neg \varphi \mid \varphi \land \varphi \mid \varphi \lor \varphi,$$

where \(k \in S\), \(\alpha \in L\), and \(L \subseteq bMSOL\) is closed under \(\lor, \land,\) and \(\neg\).

Lemma 3 from [13] states that an equivalent formula \(\psi\), of the form described below, can always be constructed for every formula \(\varphi\) given by this grammar. Therefore we adopt the following definition:

Definition 5.18. A bMSOL-step formula \(\psi\) is a formula of the form

$$\psi = \bigvee_{i \in I} (\varphi_i \land k_i)$$

where \(I\) is a finite set, \(\varphi_i \in bMSOL\) and \(k_i \in S\).

A formula \(\varphi\) is restricted if it contains no universal set quantification of the form \(\forall X. \psi\), and whenever \(\varphi\) contains a universal first-order quantification \(\forall x. \psi\), then \(\psi\) is a bMSOL-step formula.

Definition 5.19. RMSOL is comprised of the WMSOL formulas which are restricted.

Semantics of WMSOL, and translation of RMSOL into MSOLEVAL\(_S\)

Next we define the semantics of WMSOL and, where it is straightforward, simultaneously also its translation \(\text{tr}\) into MSOLEVAL\(_S\).

Let \(w\) be a word and, given a formula \(\varphi\), let \(\sigma\) be an assignment of the first order variables in \(\varphi\) to positions in \(w\) and of the second order variables in \(\varphi\) to sets of positions in \(w\).

First, denote the evaluation of an MSOL-polynomial \(t\) for a word \(w\) and an assignment \(\sigma\) by \(E(t, w, \sigma)\). For a formula \(\varphi\), we will need an MSOL-polynomial whose evaluation on a word \(w\) will equal the truth value of \(\varphi\) given the assignment \(\sigma\). Thus we let \(\text{tv}(\varphi)\) be an abbreviation for

$$\text{tv}(\varphi) = \sum_{U: \psi(U) \land \varphi} 1$$

where \(\psi(U) = \forall x(U(x) \leftrightarrow (x \approx x))\) and \(U\) does not occur freely in \(\varphi\). Since only the set consisting of all the universe satisfies \(\psi(U)\), the result of the summation is given by:

$$E(\text{tv}(\varphi), w, \sigma) = \begin{cases} 1 & (w, \sigma) \models \varphi \\ 0 & \text{else} \end{cases}$$

35
We denote by \( \text{TRUE}(x) \) the formula \( x \approx x \) with free first order variable \( x \). Similarly, \( \text{TRUE}(U) \) denotes the formula \( \exists y(U(y)) \lor \neg \exists y(U(y)) \) with free set variable \( U \).

The evaluations of weighted formulas \( \varphi \in \text{WMSOL}(S, \Sigma) \) on a word \( w \) are denoted by \( WE(\varphi, w, \sigma) \), and are now defined inductively, along with their translations \( tr(\varphi) \). Note that indeed, the values \( WE \) of the function defined by \( \varphi \in \text{WMSOL}(S, \Sigma) \) identify with the evaluation \( E \) of our translation \( tr(\varphi) \):

(i) For \( k \in S \) we have \( tr(k) = k \) and

\[
WE(k, w, \sigma)) = E(tr(k), w, \sigma)) = k
\]

(ii) For atomic formulas \( \theta \) we have \( tr(\theta) = tv(\theta) \) and

\[
WE(\theta, w, \sigma) = E(tr(\theta), w, \sigma) = E(tv(\theta), w, \sigma)
\]

(iii) For negated atomic formulas we have

\[
tr(\neg \theta) = 1 - tr(\theta) = 1 - tv(\theta)
\]

and

\[
WE(\neg \theta, w, \sigma) = 1 - E(tv(\theta), w, \sigma)
\]

(iv) For the logical connective \( \lor \), we have:

\[
tr(\varphi_1 \lor \varphi_2) = tr(\varphi_1) + tr(\varphi_2)
\]

and

\[
WE(\varphi_1 \lor \varphi_2, w, \sigma) = E(tr(\varphi_1) + tr(\varphi_2), w, \sigma) = E(tr(\varphi_1), w, \sigma) + E(tr(\varphi_2), w, \sigma)
\]

and for \( \land \), we have:

\[
tr(\varphi_1 \land \varphi_2) = tr(\varphi_1) \cdot tr(\varphi_2)
\]

and

\[
WE(\varphi_1 \land \varphi_2, w, \sigma) = E(tr(\varphi_1) \cdot tr(\varphi_2), w, \sigma) = E(tr(\varphi_1), w, \sigma) \cdot E(tr(\varphi_2), w, \sigma)
\]

(v) For first order existential quantification we have:

\[
tr(\exists x. \varphi) = \sum_{x: \text{TRUE}(x)} tr(\varphi)
\]
and

\[ WE(\exists x.\varphi, w, \sigma) = E\left( \sum_{x: \text{TRUE}(x)} tr(\varphi, w, \sigma) \right) = \sum_{x: \text{TRUE}(x)} E(tr(\varphi, w, \sigma)) \]

(vi) For second order existential quantification we have:

\[ tr(\exists U.\varphi) = \sum_{U: \text{TRUE}(U)} tr(\varphi) \]

and

\[ WE(\exists U.\varphi, w, \sigma) = E\left( \sum_{U: \text{TRUE}(U)} tr(\varphi, w, \sigma) \right) = \sum_{U: \text{TRUE}(U)} E(tr(\varphi, w, \sigma)) \]

So far the definition of \( WE \) was given using the evaluation function \( E \) and the translation was straightforward. Problems arise with the universal quantifiers.

The unrestricted definition of \( WE \) for WMSOL given below gives us functions which are not recognizable by weighted automata, and the straightforward translation defined below gives us expressions which are not in MSOLEVALS:

(vii) First order universal quantification:

\[ tr(\forall x.\varphi) = \prod_{x: \text{TRUE}(x)} tr(\varphi) \]

and

\[ WE(\forall x.\varphi, w, \sigma) = E\left( \prod_{x: \text{TRUE}(x)} tr(\varphi, w, \sigma) \right) = \prod_{x: \text{TRUE}(x)} E(tr(\varphi, w, \sigma)) \].

The formula \( \varphi_{sq} = \forall x.\forall y.2 \) gives the function \( 2^L(w)^2 \) and is not a bMSOL-step formula. The straightforward translation \( tr \) gives the term

\[ \prod_{x: \text{TRUE}(x)} \left( \prod_{y: \text{TRUE}(y)} 2 \right) = \prod_{(x,y): \text{TRUE}(x,y)} 2 \]

which is a product over the tuples of a binary relation, hence not in MSOLEVALS.

(viii) Second order universal quantification:

\[ tr(\forall U.\varphi) = \prod_{U: \text{TRUE}(U)} tr(\varphi) \]
and

\[ WE(\forall U. \varphi, w, \sigma) = E(\prod_{U: \text{TRUE}(U)} \text{tr}(\varphi, w, \sigma)) = \prod_{U: \text{TRUE}(U)} E(\text{tr}(\varphi, w, \sigma)) \]

Here the translation gives a product \( \prod_{U: \text{TRUE}(U)} \) ranging over subsets, which is not an expression in MSOLEVAL_S.

In RMSOL, universal second order quantification is restricted to formulas of bMSOL, and first order universal quantification is restricted to bMSOL-step formulas.

In [13, page 590], after Figure 1, the following is stated:

**Lemma 5.20.** The evaluation \( WE \) of a bMSOL formula \( \varphi \) assumes values in \( \{0, 1\} \) and coincides with the standard semantics of \( \varphi \) as an unweighted MSOL formula.

Because the translation of universal quantifiers using \( \text{tr} \) leads outside of MSOLEVAL_S, we define a proper translation \( \text{tr}' : \text{RMSOL} \to \text{MSOLEVAL}_S \).

Using [Lemma 5.20] we set \( \text{tr}'(\varphi) = \text{tv}(\varphi) \) for \( \varphi \) a bMSOL formula, and for universal first order quantification of bMSOL-step formulas

\[ \psi = \bigvee_{i \in I} (\varphi_i \land k_i) \]

we proceed as follows:

\[ \text{tr}'(\forall x. \psi) = \text{tr}'(\forall x. \bigvee_{i \in I} (\varphi_i \land k_i)) = \prod_{x: \text{TRUE}(x)} \left( \sum_{i \in I} \text{tr}'(\varphi_i) \cdot k_i \right) \]

Clearly, the formula \( \prod_{x: \text{TRUE}(x)} \left( \sum_{i \in I} \text{tr}'(\varphi_i) \cdot k_i \right) \) is an expression in MSOLEVAL_S.

We compute \( WE(\forall x. \psi, w, \sigma) \) and \( E(\text{tr}(\forall x. \psi), w, \sigma) \) by:

\[ WE((\forall x. \psi), w, \sigma) = E(\text{tr}(\forall x. \psi), w, \sigma) = E(\text{tr}(\forall x. \bigvee_{i \in I} (\varphi_i \land k_i)), w, \sigma) \]

\[ = \prod_{x: \text{TRUE}(x)} E(\text{tr}(\bigvee_{i \in I} (\varphi_i \land k_i)), w, \sigma) \]

\[ = \prod_{x: \text{TRUE}(x)} \left( \sum_{i \in I} (E(\text{tr}(\varphi_i), w, \sigma) \cdot k_i) \right) \]

\[ = \prod_{x: \text{TRUE}(x)} \left( \sum_{i \in I} (E(\text{tv}(\varphi_i), w, \sigma) \cdot k_i) \right) \]

For universal second order quantification of bMSOL formulas \( \psi \) again we use [Lemma 5.20] and get the expression

\[ \text{tr}'(\forall U. \psi) = \text{tv}(\forall U. \psi) \]
which is clearly in $\text{MSOLEVAL}_S$, and

$$\text{WE}(\forall U.\psi, w, \sigma) = E(tr'(\forall U.\psi), w, \sigma) = E(\text{tv}(\forall U.\psi), w, \sigma)$$

Thus we have proved:

**Theorem 5.21.** Let $S$ be a commutative semiring. For every expression $\varphi \in \text{RMSOL}$ there is an expression $tr'(\varphi) \in \text{MSOLEVAL}_S$ such that $\text{WE}(\varphi, w, \sigma) = E(tr'(\varphi), w, \sigma)$, i.e., $\varphi$ and $tr'(\varphi)$ define the same word function.

**Translation from $\text{MSOLEVAL}_S$ to RMSOL**

It follows from our [Theorem 5.6](#) and the characterization in [30](#) of recognizable word functions as the functions expressible in $\text{RMSOL}$, that the converse of [Theorem 5.21](#) is also true. We give a direct proof of this, without using weighted automata.

**Theorem 5.22.** Let $S$ be a commutative semiring. For every expression $t \in \text{MSOLEVAL}_S$ there is a formula $\varphi_t \in \text{RMSOL}$ such that $\text{WE}(\varphi_t, w, \sigma) = E(t, w, \sigma)$, i.e., $\varphi_t$ and $t$ define the same word function.

**Proof.** (i) Let $t = \prod_{x:\varphi(x)} \alpha$ be an MSOL-monomial. We note that

$$\alpha \cdot \text{tv}(\varphi) + \text{tv}(\neg \varphi) = \begin{cases} \alpha & \text{if } \varphi \text{ is true} \\ 1 & \text{else} \end{cases}$$

Furthermore, by [Lemma 5.20](#) $\varphi \in \text{bMSOL}$. So we put

$$\varphi_t = \forall x.((\varphi(x) \land \alpha) \lor \neg \varphi(x)).$$

(ii) Let $t_1 = \sum_{U: \varphi(U)} t$ and let $\varphi_t$ be the translation of $t$. Then

$$\varphi_{t_1} = \exists U(\varphi_t \land \varphi(U)).$$

**Comparison**

The formalism $\text{WMSOL}$ of weighted logic was first invented in 2005 in [29](#) and since then used to characterize word and tree functions recognizable by weighted automata, [32](#). These characterizations need some syntactic restrictions which lead to the formalism of $\text{RMSOL}$. No such syntactic restrictions are needed for the characterization of recognizable word functions using $\text{MSOLEVAL}$. The weighted logic $\text{WMSOL}$ can also be defined for general relational structures. However, it is not immediate which syntactic
restrictions are needed, if at all, to obtain algorithmic applications similar to the ones obtained using MSOLEVAL, cf. [22] [21] [72].

There are some notational disadvantages in the WMSOL approach.

- The restrictions in the definition of RMSOL are not purely syntactic, since they include a restriction on the semantics of the formulas.

- The formulas themselves are hybrid objects, mixing constants from $S$ and logical expressions. For instance, $\forall x \cdot 2$ is a weighted formula which represents the function $2^{l(w)}$, and $\forall x \forall y \cdot 2$ is a weighted formula which represents the function $2^{2^{l(w)}}$.

- Logically equivalent formulas can represent different functions: $\exists x (P_1(x))$ represents the function $f_1(w)$ which counts the number of ones in $w$, but $\exists x (P_1(x) \lor P_1(x))$ represents the function $2 \cdot f_1(w)$.

- It may be rather difficult or unnatural to find a formula for a given word function.

In contrast, the MSOLEVAL approach has the following advantages.

- The expressions are natural and intuitive.

- The expressions are defined for all formulas of MSOL, without any restrictions.

- If we replace formulas occurring in an expression by logically equivalent formulas, the word function it represents remains the same.

In conclusion, we have given two proofs that RMSOL and MSOLEVAL with values in $S$ have the same expressive power over words. One proof uses model theoretic tools to show directly that MSOLEVAL captures the functions recognizable by weighted automata. The other proof shows how to translate the formalisms from one into the other. Adapting the translation proof, it should be possible to extend the result to tree functions as well, cf. [32].
Chapter 6

Eliminating Logic

In this chapter we replace the definability assumptions made in the meta-theorems of Chapter 3 by finiteness assumptions on the ranks of the Hankel matrices at hand, and obtain their logic-free analogues. This elimination of logic allows us to separate the algebraic from the logical, and give a more precise argument for the validity of these theorems. Additional, perhaps more concrete, motivation is given by Theorem 4.11.

We present L. Lovász’s logic-free version of Courcelle’s theorem proved in [68], followed by our analogous logic-free version of the general Theorem 3.9. Both theorems are proved using the formalism of graph algebras, which is presented in Section 6.1. In Section 6.2 we eliminate logic from the further general Theorem 3.14.

6.1 Graph Algebras

In [68], the formalism of graph algebras was used to prove a logic-free analogue of Theorem 3.8:

**Theorem 6.1** (L. Lovász, 2007). Let $\mathcal{F}$ be a field and let $f$ be an $\mathcal{F}$-valued graph parameter with finite connection rank. Then $f$ can be computed in polynomial time on graph classes of bounded tree-width.

In this section, we present an adaption of the formalism, which will be used to prove:

**Theorem 6.2.** Let $\mathcal{F}$ be a field and let $f$ be an $\mathcal{F}$-valued graph parameter with finite join rank. Then $f$ can be computed in polynomial time on graph classes of bounded clique-width.

We take an adaptation not only because the graph classes of bounded tree-width and bounded clique-width are defined using different operations, but also in forethought of including graph parameters over semirings in our framework later on; As the original formalism unfortunately does not readily translate to semirings, some amendments must be made to some of the definitions. We will point these out during the presentation.
For this section, unless stated otherwise, let $\mathcal{F}$ be a field, let $k \in \mathbb{N}$ fixed, and let $f$ be an $\mathcal{F}$-valued graph parameter.

### Quantum graphs

**Definition 6.3.** A $k$-labeled quantum graph is a formal linear combination of a finite number of $k$-labeled graphs $G_i \in L\text{Graphs}_k$ with coefficients from $\mathcal{F}$.

Note that some $k$-labeled graphs in the combination can have a zero coefficient, but they can be deleted without changing the quantum graph. $k$-labeled quantum graphs form an infinite dimensional linear space, denoted here by $Q_k$. Similarly we define:

**Definition 6.4.** A $k$-colored quantum graph is a formal linear combination of a finite number of $k$-colored graphs $G_i \in C\text{Graphs}_k$ with coefficients from $\mathcal{F}$.

We denote the linear space of $k$-colored quantum graphs by $Q_k$. As $k$-labeled graphs are a special case of $k$-colored graphs, this notation also includes $Q_k'$, and for the sequel we assume the context of $k$-colored graphs. We will also sometimes refer to $k$-colored quantum graphs simply as quantum graphs.

It is often convenient to use quantum graphs to express combinatorial facts and identities, e.g. [41, 82, 70].

Let $X, Y$ be quantum graphs, where $X = \sum_{i=1}^{m} x_i G_i$ and $Y = \sum_{i=1}^{n} y_i G_i$. It will be convenient to regard the coefficients which are not mentioned as zeros. Then $Q_k$ (as well as $Q_k'$) is a vector space over $\mathcal{F}$ with addition:

$$X + Y = \left( \sum_{i=1}^{m} x_i G_i \right) + \left( \sum_{i=1}^{n} y_i G_i \right) = \sum_{i=1}^{\max\{m,n\}} (x_i + y_i) G_i$$

and scalar multiplication with elements $a \in \mathcal{F}$:

$$a \cdot X = \sum_{i=1}^{m} (a \cdot x_i) G_i$$

Any binary operation $\sqcup$ on $k$-colored graphs is extended to quantum graphs by:

$$X \sqcup Y = \sum_{i=1}^{m} \sum_{j=1}^{n} (x_i \cdot y_j) (G_i \sqcup G_j)$$

We are concerned either with $k$-connections, where $\sqcup = \sqcup_k$, or with joining, where $\sqcup = \eta_k$.

Any graph parameter $f$ is extended to quantum graphs linearly:

$$f(X) = \sum_{i=1}^{m} x_i \cdot f(G_i)$$
Given a Hankel matrix $H(f, \Box)$, we turn $Q_k$ into a commutative algebra by defining an inner product on $X$ and $Y$:

$$\langle X, Y \rangle_{f, \Box} = f(X \Box Y) = \sum_{i=1}^{m} \sum_{j=1}^{n} (x_i \cdot y_j) \cdot f(G_i \Box G_j) \quad (6.1)$$

**Equivalence relations for quantum graphs**

Here the definitions diverge; in [68], the annihilator $N_f^\Box$ of the inner product induces an equivalence relation, where:

$$N_f^\Box = \{ X \in Q'_k \mid f(X \Box Y) = 0 \ \forall Y \in Q'_k \}.$$

**Definition 6.5.** Two quantum graphs $X, Y \in Q'_k$ are *equivalent with respect to $\Box$ and $f$* if

$$X - Y \in N_f^\Box.$$

**Remark.** The notation in [68] is slightly different. As $\Box$ is always fixed in that context, it is not explicitly mentioned in the notation of the annihilator or in the definition of the equivalence relation.

In our adaptation, on the other hand, we define an equivalence relation $Ker_f^\Box$ over $Q_k$ in a different way:

**Definition 6.6.** $Ker_f^\Box \subseteq Q_k \times Q_k$ is defined by:

$$(X, Y) \in Ker_f^\Box \iff \forall Z \in Q_k : f(X \Box Z) = f(Y \Box Z)$$

We will justify this discrepancy in [Section 7.3]. One also might justify this choice with how equivalent $X$ and $Y$ cannot be distinguished using $f$ and $\Box$, in resemblance to the equivalence relation used in the Myhill-Nerode Theorem characterizing regular languages, cf. [59]. The set of equivalence classes of $Ker_f^\Box$ is denoted by $Q_k / Ker_f^\Box$. For a quantum graph $X$, the equivalence class of $X$ in $Ker_f^\Box$ is denoted by $[X]_f^\Box$.

We have that $Q_k / Ker_f^\Box$ is a vector space with addition:

$$[X]_f^\Box + [Y]_f^\Box = [X + Y]_f^\Box$$

and scalar multiplication with elements $a \in \mathcal{F}$:

$$a \cdot [X]_f^\Box = [a \cdot X]_f^\Box$$
We turn $Q_k/Ker_f^\square$ into a quotient algebra by extending the inner product in Equation 6.1 to these equivalence classes as follows:

$$[X]_f^\square[Y]_f^\square = [X \diamond Y]_f^\square$$

The algebra $Q_k/Ker_f^\square$ has some nice properties that will prove useful:

**Proposition 6.6.1.** Let $\square$ be a commutative and associative operation on $k$-colored graphs, and let $X' \in [X]_f^\square$ and $Y' \in [Y]_f^\square$. Then

(i) $X' + Y' \in [X + Y]_f^\square = [X]_f^\square + [Y]_f^\square$

(ii) $a \cdot X' \in [a \cdot X]_f^\square = a \cdot [X]_f^\square$

(iii) $X' \diamond Y' \in [X \diamond Y]_f^\square = [X]_f^\square \diamond [Y]_f^\square$

The proof is given in Appendix A.

**Proposition 6.6.2.** The operations $\sqcup_k$ and $\eta_k$ are commutative and associative.

The following lemma relates the quotient algebra $Q_k/Ker_f^\square$ to the Hankel matrix $H(f, \square)$ used to define its inner product.

**Lemma 6.7.** Let $H(f, \square)$ be of rank $m$ and let $\{r_1, \ldots, r_m\}$ be $m$ linearly independent rows corresponding to the graphs $E_1, \ldots, E_m$. Then $B_k = \{[E_1]_f^\square, \ldots, [E_m]_f^\square\}$ forms a basis of $Q_k/Ker_f^\square$.

**Proof.** Let $[X]_f^\square \in Q_k/Ker_f^\square$, for $X \in Q_k$ where $X = \sum_{i=1}^n x_i G_i$. Each row in $H(f, \square)$ corresponding to the $k$-colored graph $G_i$ is a linear combination $\sum_{j=1}^m \alpha_{ij} E_j$ of the rows corresponding to the generators $E_1, \ldots, E_m$, therefore by the definition of $Ker_f^\square$, we have that

$$(G_i, \sum_{j=1}^m \alpha_{ij} E_j) \in Ker_f^\square$$

and by Proposition 6.6.1(i)-(ii), we have

$$[X]_f^\square = \sum_{i=1}^n x_i [G_i]_f^\square = \sum_{i=1}^n x_i \sum_{j=1}^m \alpha_{ij} [E_j]_f^\square$$

**Computing graph parameters via dynamic programming**

We are now ready to start proving Theorem 6.2. Since graph classes of bounded tree-width are also of bounded clique-width, the proof we will present, which is of the clique-width case, holds for the tree-width case as well. However, one can also obtain a direct proof of the tree-width case by following our proof while using the inductive definition of tree-width.
Let $k \in \mathbb{N}$ be fixed. Given a parse tree $pt(G)$ for a graph $G \in CW(k)$, any instance of an operation $\tilde{\eta}_{i,j}$ can be replaced by a sequence consisting of a recoloring, the operation $\tilde{\eta}_{1,2}$ and another recoloring. Performing this procedure takes polynomial time, therefore we safely assume in the sequel that all parse trees use only the operation $\tilde{\eta}_{1,2}$, which we simply denote by $\eta_k$, omitting $k$ when it is clear from context.

**Representing graphs in the graph algebra**

Given $G$, we want to compute $f(G)$, where $f$ is an $F$-valued graph parameter with finite join rank. We will present an algorithm which finds a representation $X_G$ of the equivalence class $[X_G]_f$ of a certain quantum graph $X_G$ associated with $G$, from which the value $f(G)$ is computed.

To validate the reality of this algorithm, we will show that there indeed exists a quantum graph $X_G$ such that $f(X_G) = f(G)$, and that there exists a computable representation of $[X_G]_f$ using a set $B$ of generators of $Q_k/Ker_\eta_f$.

Additionally, we will show that the algorithm runs in polynomial time.

**Lemma 6.8.** Let $G \in CW(k)$ be given together with its parse tree $pt(G)$, and let $B = \{[E_i]_f, \ldots, [E_m]_f\}$ be a basis of $Q_k/Ker_\eta_f$. Then there exists $[X_G]_f \in Q_k/Ker_\eta_f$ such that $f(X_G) = f(G)$, and $[X_G]_f$ can be represented as a linear combination of elements in $B$.

First, some notation.

- Let $S_1, \ldots, S_n \in Q_k$ be the single-vertex $k$-colored graphs, and
- let $S_1, \ldots, S_n$ denote the representations of the equivalence classes $[S_1]_f, \ldots, [S_n]_f$ using the basis $B$.

For $[E_i]_f, [E_j]_f \in B$,

- let $\eta(E_i, E_j)$ denote the representation of the equivalence class $\eta([E_i]_f, [E_j]_f)$ using $B$, and
- let $\rho(E_i)$ denote the representation of the equivalence class $[\rho(E_i)]_f$ using $B$.

**Proof.** Let $G \in CW(k)$ be given together with its parse tree $pt(G)$. We follow the definition of $CW(k)$, and proceed by induction on $pt(G)$:

- If $G = S_i$, then set $X_G = S_i$. We have that $G \in [X_G]_f$, and the graph $X_G = S_i$ is a single-vertex graph, so we have a representation $S_i$ for $[X_G]_f = [S_i]_f$. Obviously $f(G) = f(X_G)$.

Assume that for $G_1, G_2$, there exist $X_{G_1}$ and $X_{G_2}$, such that $G_1 \in [X_{G_1}]_f$ and $G_2 \in [X_{G_2}]_f$, and there exist representations $X_{G_1}$ and $X_{G_2}$ of $[X_{G_1}]_f$ and $[X_{G_2}]_f$ respectively.
• If $G = \eta(G_1, G_2)$, take $X_G = \eta(X_{G_1}, X_{G_2})$. By Proposition 6.6.1(iii), we have

$$X_G = \eta(X_{G_1}, X_{G_2}) \in [\eta(X_{G_1}, X_{G_2})]_f^\eta = \eta([X_{G_1}]_f^\eta, [X_{G_2}]_f^\eta)$$

and by the induction hypothesis, we have

$$\eta([X_{G_1}]_f^\eta, [X_{G_2}]_f^\eta) = \eta([G_1]_f^\eta, [G_2]_f^\eta) = [\eta(G_1, G_2)]_f^\eta = [G]_f^\eta$$

Therefore $G \in [X_G]_f^\eta$, and $f(G) = f(X_G)$.

As for the representation of $[X_G]_f^\eta$, consider the expansion of the expression $\eta(X_{G_1}, X_{G_2})$. It is a formal sum with summands that are the result of the $\eta$ operation on basis elements, for which there are representations of the form $\eta(E_i, E_j)$. Thus replacing the summands by these representations produces a representation $X_G$ of $[X_G]_f^\eta$ that only uses basis elements.

• If $G = \rho(G_1)$, take $X_G = \rho(X_{G_1})$. As $f$ is invariant under recoloring, we have $f(G) = f(X_G)$. Replace the basis elements $[E_i]_f^\eta$ in the representation $X_{G_1}$ of $X_G$, by the representations $\rho(E_i)$ of $[\rho(E_i)]_f^\eta$ and obtain a representation $X_G$ of $X_G$.

Proof of Theorem 6.2

We prove Theorem 6.2 by describing a dynamic programming algorithm that computes $f$ that runs in polynomial time.

Proof. Let $k \in \mathbb{N}^+$ be fixed and let a graph $G$ be given as input.

We first use Theorem 3.7 and find a parse tree $pt(G)$ for $G \in CW(3k)$.

Next, using $pt(G)$, we build a representation of the equivalence class $[X_G]_f^\eta$ associated with the given graph $G$ in the basis $B$ in order to obtain $f(G)$. The algorithm requires a finite amount of preprocessing:

• Find basis elements $B = \{[E_1]_f^\eta, \ldots, [E_m]_f^\eta\}$.
• Compute the representations $S_1, \ldots, S_n$ of all the single-vertex $k$-colored graphs.
• Compute the representations $\eta(E_i, E_j)$ of the operation $\eta$ for all basis elements.
• Compute the representations $\rho(E_i)$ of the operation $\rho$ for all basis elements.
• Compute the value of $f$ on all the basis elements.

The algorithm works with the parse tree $pt(G)$ and performs substitutions from the bottom up, following the inductive proof of Lemma 6.8. When the top of the tree is reached, we have a representation $X_G$ of $[X_G]_f^\eta$ which only uses basis elements $[E_i]_f^\eta \in B$. Then we plug the precomputed values of $f$ into $X_G$ and compute $f(X_G) = f(G)$. 

46
The computation of the parse tree $pt(G)$ takes polynomial time, and the preprocessing takes constant time. The substitutions performed during the traversal of $pt(G)$ are of constant size, since the number of summands in each representation is bounded by $|B| = m$. Therefore the total run time is polynomial in the size of $pt(G)$, which is polynomial in the size of $G$.

This proof is not very different from the proof of Theorem 6.1 in [68], however, the latter is rather sketchy in its part relating to tree decompositions. In particular, the role of relabelings, admittedly not very critical, is not spelled out at all. The role of parse trees for clique-width in the dynamic programming part of the algorithm was spelled out here as well.

6.2 Linearly Linked Hankel Matrices

The graph classes addressed in the previous section were definable using one binary operation, either $\eta_k$ or $\sqcup_k$, and treating graph classes which are inductively defined using more than one binary operation appears to be a natural next step.

However, a straightforward extension of our approach so far does not seem to work in the general case, as each binary operation defines its own set of equivalence classes, and these do not necessarily interact nicely with each other as the algorithm tries to work up the parse tree. Nonetheless, Theorem 3.14 suggests an extension should be possible for graph classes which are sum-like inductive. Following this, we introduce the notion of linearly linked Hankel matrices which we then use to prove a logic-free analogue of Theorem 3.14. We will see that sum-like inductive graph classes do fall within its scope.

For the sequel, let $O_I = \{\square_i\}_{i \leq I}$ be finitely many binary operations on labeled graphs (which are not necessarily sum-like), and let $B_J = \{B_j\}_{j \leq J}$ be finitely many labeled graphs, called basic graphs. Let $\mathcal{P}_K = \{p_k\}_{k \leq K}$ be finitely many graph parameters over a field, and for a labeled graph $G$ let $\bar{p}(G)$ denote the transpose of the row-vector $(p_1(G), \ldots, p_K(G))$.

**Definition 6.9** (Inductively defined classes of graphs).

A graph class $C$ is inductively defined using $B_J$ and $O_I$ if each $B_j \in B_J$ is in $C$, and for each $\square_i \in O_I$, whenever $G_1, G_2 \in C$ then also $G_1 \sqcup_i G_2 \in C$.

In particular, $TW(k)$ and $CW(k)$ are inductively defined classes of graphs which are also sum-like inductive.

**Definition 6.10** (Linearly linked Hankel matrices).

The Hankel matrices $\mathcal{H}(\mathcal{P}_K, O_I) = \{H(p_k, \square_i) \mid p_k \in \mathcal{P}_K, \square_i \in O_I\}$ are linearly linked if the following holds:

- Each $H(p_k, \square_i) \in \mathcal{H}(\mathcal{P}_K, O_I)$ is of finite rank.
• For each $\sqcup_i \in \mathcal{O}_I$ there is a $K \times K$ matrix $M_i$ such that for any two labeled graphs $G_1$ and $G_2$,
\[ \bar{p}(G_1 \sqcup_i G_2) = M_i \cdot \bar{p}(G_1 \sqcup_i G_2) \]

Note that $M_1$ is the $K \times K$ identity matrix.

For the rest of the section, we assume the notation $\mathcal{O}_I = \{ \sqcup_i \}_{i \leq I}$, $\mathcal{B}_J = \{ B_j \}_{j \leq J}$,
$\mathcal{P}_K = \{ p_k \}_{k \leq K}$, and $\mathcal{H}(\mathcal{P}_K, \mathcal{O}_I) = \{ \mathcal{H}(p_k, \sqcup_i) \mid p_k \in \mathcal{P}_K, \sqcup_i \in \mathcal{O}_I \}$. We write $\mathcal{H}$ when $\mathcal{P}_K$ and $\mathcal{O}_I$ are clear from context.

The Hankel matrices $\mathcal{H}$ being linearly linked will ensure that substitutions can be safely performed during the traversal of a parse tree of a graph in $C$, giving us:

**Theorem 6.11.** Let $C$ be inductively defined using $\mathcal{B}_J$ and $\mathcal{O}_I$, and let $\mathcal{P}_K$ be finitely many graph parameters, such that the Hankel matrices $\mathcal{H}$ are linearly linked. Then for a graph $G \in C$ with a given parse tree $pt(G)$, all the graph parameters $p_k \in \mathcal{P}_K$ can be computed in time polynomial in the size of $pt(G)$.

**Sketch of proof.** One describes an algorithm similar to the one in the proof of Theorem 6.2 which includes, in addition to the usual substitutions performed as we travel up the parse tree, substitutions of $\sqcup_i$ operations by the operation $\sqcup_1$ preaced by a multiplication with the appropriate matrix $M_i$.

In fact, Theorem 6.11 is a proper generalization of Theorem 3.14.

**Proposition 6.11.1.** Let $\mathcal{O}_I$ be a set of sum-like operations and $f$ be a CMSOL-definable graph parameter. Then there are finitely many graph parameters $\mathcal{P}_K$ with $f \in \mathcal{P}_K$ so that the Hankel matrices $\mathcal{H}(\mathcal{P}_K, \mathcal{O}_I)$ are linearly linked.

**Proof.** Assume w.l.o.g that $\sqcup_1 = \sqcup$ and let $q$ be the quantifier rank of $f$. Take $\mathcal{P}_K$ to be the set of all CMSOL-polynomials with quantifier rank $\leq q$, of which there are finitely many. By the Finite Rank Theorem we have that the Hankel matrices $\mathcal{H}(\mathcal{P}_K, \mathcal{O}_I)$ are of finite rank. It remains to show that for each $\sqcup_i \in \mathcal{O}_I$ there is a $K \times K$ matrix $M_i$ such that for any two graphs $G_1$ and $G_2$,
\[ \bar{p}(G_1 \sqcup_i G_2) = M_i \cdot \bar{p}(G_1 \sqcup_i G_2) \]

Let $\sqcup_i \in \mathcal{O}_I$. The operations in $\mathcal{O}_I$ are sum-like, therefore there is a quantifier-free translation scheme $\Phi_i$ associated with $\sqcup_i$. In addition, by the fundamental property, Theorem 2.6, for any CMSOL-polynomial $p_k$ of quantifier rank $\leq q$, $\Phi_i$ induces a reverse translation $\Phi_i^*(p_k) = p_k^\Phi$ such that
\[ p_k(G_1 \sqcup_i G_2) = p_k^\Phi(G_1 \sqcup_i G_2) = p_k^\Phi(G_1 \sqcup_i G_2) \]

Furthermore, $p_k^\Phi$ is CMSOL-definable with quantifier rank $\leq q$. Therefore, $p_k^\Phi$ appears in some entry $\sharp_i(k)$ of $\bar{p}$. Since $\bar{p}$ is composed only of (and all) CMSOL-definable graph
parameters of quantifier rank $\leq q$, we have in total

$$
\bar{p}(G_1 \square_i G_2) = \begin{bmatrix}
p_1^i(G_1 \sqcup G_2) \\
p_2^i(G_1 \sqcup G_2) \\
\vdots \\
p_K^i(G_1 \sqcup G_2)
\end{bmatrix}
$$

All that remains is to permute the parameters in $\bar{p}$ to the order they should appear in if the operation performed is $\sqcup$ instead of $\square_i$. Let $e^i_{z_i(k)}$ denote the $K$-length (column) vector that has $0$ in entries $\neq z_i(k)$ and $1$ in the entry $z_i(k)$.

Define the translation matrix composed of these vectors

$$M_i = \begin{pmatrix}
e^i_{z_i(1)} & e^i_{z_i(2)} & \cdots & e^i_{z_i(\beta)}
\end{pmatrix}
$$

and have that

$$\bar{p}(G_1 \square_i G_2) = M_i \cdot \bar{p}(G_1 \sqcup G_2) = M_i \cdot \bar{p}(G_1 \square_1 G_2)
$$

which makes the Hankel matrices $\mathcal{H}$ linearly linked.

---

**A more general perspective**

As much as it is natural to focus on graphs (labeled or colored) when having a computer scientific viewpoint, it should be mentioned that the fact that we were discussing graphs in particular was immaterial for much of our treatment. The notions of width and inductively defined classes readily extend to general $\tau$-structures, as do Definition 6.10 and Theorem 6.11.

Another expression of this viewpoint is the implicit, but important, assumption we made regarding computability. The preprocessing in the proof of Theorem 6.2 is described as a procedure that takes finite time, implying the families of algorithms in this chapter are uniformly generated. However, an algorithm that has the required information hardwired can be described just the same. Additionally, the parse trees in this chapter are computable, but this requirement may be unnecessarily restrictive, as the algorithms described only need to have them.
Chapter 7

From Fields to Semirings

In this chapter we present the extension of our work to commutative semirings. We start with examples of graph parameters that have infinite Hankel ranks when interpreted over a field, to which our work so far is not applicable. Then we present the notions of commutative semirings and semimodules, and define the framework which will be used to formulate and prove theorems analogous to Theorem 6.2 for the case of commutative semirings. We will see the value in reinterpreting real-valued graph parameters as graph parameters over tropical semirings.

7.1 Motivating Examples

In [3, 20] linear extremum problems definable in MSOL, called LinEMSOL, are studied. They deal with graphs equipped with weight functions for vertices and/or edges, where the weights take values in Q or R. The solution to these problems is the maximal or the minimal value of an expression involving vertices or edges that satisfies an MSOL formula. More precisely, let $f_1, \ldots, f_m$ be $m$ function symbols for some fixed integer $m$, and let $\varphi$ be an MSOL($\tau$) formula with free set-variables $U_1, \ldots, U_l$.

**Definition 7.1.** A LinEMSOL($\tau$) optimization problem over C is given as follows: for a $\tau$-structure $A \in C$ with universe $A$ and $m$ evaluation functions $f_1, \ldots, f_m$ associating integer values to the elements of $A$, find an assignment $z$ to the $l$ free variables in $\varphi$ such that:

$$\sum_{1 \leq i \leq l} \sum_{a \in z(U_i)} a_{ij} f_j(a) = \text{opt} \left\{ \sum_{1 \leq i \leq l} \sum_{a \in z'(U_i)} a_{ij} f_j(a) \mid \langle A, z' \rangle \models \varphi(U_1, \ldots, U_l) \right\}$$

where $\text{opt} \in \{\min, \max\}$ and $a_{ij}$, for $1 \leq i \leq l$ and $1 \leq j \leq m$, are any integers.

**Example.** The maximum weight clique problem is given by a graph $G$ with a weight function $\alpha$ for its vertices, with the objective of finding a clique $C$ in $G$ with maximal total weight. This is a LinEMSOL optimization problem as we can take the $\tau_{\text{graphs}}$-structure...
$G$ representing $G$, an evaluation function $f_1 = \alpha$ and a $\tau_{\text{graphs}}$-formula

$$\varphi(U_1) = \forall x_1, x_2 [(U_1(x_1) \land U_1(x_2) \land x_1 \neq x_2) \rightarrow E(x_1, x_2)]$$

and have that the solution is given by finding an assignment $z$ satisfying:

$$\sum_{a \in z(U_1)} f_1(a) = \max \{ \sum_{a \in z'(U_1)} f_1(a) \mid \langle G, z' \rangle \models \varphi(U_1) \}$$

Other examples include the vertex cover, dominating set, maximal independent set, and more, see [3].

If the solution of a LinEMSOL problem is thought of as a graph parameter, the result is often a $\sqcup$-maximizing or $\sqcup$-minimizing graph parameter:

**Definition 7.2.** A graph parameter $f$ is $\sqcup$-maximizing, where $\sqcup$ is a binary operation on (possibly labeled or colored) graphs, if for every two (labeled or colored) graphs $G$ and $G'$,

$$f(G \sqcup G') = \max \{ f(G), f(G') \}.$$ $\sqcup$-minimizing graph parameters are defined similarly.

Our treatment of graph parameters relied on the rank-finiteness of their Hankel matrices, so $\sqcup$-maximizing or $\sqcup$-minimizing graph parameters pose a challenge, as they have Hankel matrices of infinite rank:

**Proposition 7.2.1 ([64]).** Let $f$ be an unbounded real-valued graph parameter which is $\sqcup$-maximizing (-minimizing). Then $H(f, \sqcup)$ has infinite rank.

For example, the chromatic number, maximal degree, tree-width, and clique number are all $\sqcup$-maximizing. We will see that many graph parameters that have infinite rank when interpreted over a field have, when interpreted over the tropical semirings, finite row-rank (soon to be defined). Therefore, our motivation is mostly to move to the tropical semirings and formulate an extension of Theorem 6.2 but we will treat graph parameters over general semirings as well.

## 7.2 Semirings and Semimodules

We start with some needed definitions, mostly taken from [56].

**Definition 7.3.** A **monoid** is a set $S$ together with a binary operation $S \oplus S \rightarrow S$ such that:

- for all $a, b, c \in S$, it holds that $(a \oplus b) \oplus c = a \oplus (b \oplus c)$, and
- there exists an identity element $e \in S$ such that for all $a \in S$, it holds that $e \oplus a = a \oplus e = a$. 52
If the operation $\oplus$ is commutative, we say that $S$ is a **commutative monoid**.

**Definition 7.4.** A semigroup is a set $S$ together with a binary operation $S \otimes S \to S$ such that:

- for all $a, b, c \in S$, it holds that $(a \otimes b) \otimes c = a \otimes (b \otimes c)$.

Now we may define semirings:

**Definition 7.5.** A semiring is a set $S$ with two binary operations, addition $\oplus$ and multiplication $\otimes$, such that:

- $S$ is a commutative monoid under addition (with identity 0),
- $S$ is a semigroup under multiplication (with identity, if any, 1),
- multiplication is distributive over addition on both sides, and
- $s \otimes 0 = 0 \otimes s = 0$ for all $s \in S$.

If multiplication is commutative, we say $S$ is a **commutative semiring**.

Unless explicitly stated otherwise, we will assume all semirings mentioned in the sequel (including in definitions) are commutative.

As opposed to a ring, not every element in a semiring must have an additive inverse. Common examples of semirings (which are not rings) are the non-negative integers $\mathbb{Z}^+$, non-negative rationals $\mathbb{Q}^+$, and non-negative reals $\mathbb{R}^+$. A semiring does not have to be numeric; for example, for a set $U$, the set of binary relations over $U$ forms a semiring with the union of sets as addition and the composition of relations as multiplication. Other examples include the tropical semirings:

**Definition 7.6 (Tropical semiring).** The min-plus algebra, denoted $\mathcal{T}_{\text{min}}$ is the set $\mathbb{R} \cup \{\infty\}$ with addition $\oplus = \min$, multiplication $\otimes = +$, and multiplicative identity $\{\infty\}$. The max-plus algebra, also known as the arctic semiring, denoted $\mathcal{T}_{\text{max}}$, is defined similarly with $\mathbb{R} \cup \{-\infty\}$, max, $+$, and $\{-\infty\}$.

We refer to both semirings as the tropical semirings.

Before we introduce more definitions, let us highlight the gap created by moving from fields to semirings. For a field $\mathcal{F}$, the rank of a matrix over $\mathcal{F}$ is usually defined as the largest number of linearly independent rows (or columns) found in the matrix. If we try and translate this to semirings, we will find that there is no obvious analogue to the notion of linear dependence. A common definition of linear dependence for a set of vectors $\{\mathbf{u}_1, \ldots, \mathbf{u}_k\}$ over a field describes the existence of scalars $a_1, \ldots, a_k \in \mathcal{F}$ which are not all zero such that $\sum_{i=1}^k a_i \mathbf{u}_i = \mathbf{0}$. Since over a semiring this condition is no longer equivalent to a vector being a linear combination of other vectors in the set, a straightforward translation to semirings would lose the sense of redundancy we want to capture. The obvious amendment is to define a set of vectors to be linearly
dependent if there is a vector in the set that can be expressed as a linear combination of the other elements, which is indeed what use in Definition 7.10 but one might also consider a stronger condition, such as in [1] 43, where a set of vectors is said to be linearly dependent if it can be partitioned into two sets that generate a common vector.

With this freedom in mind, we proceed with the rest of the definitions.

**Definition 7.7.** A semimodule $U$ over a semiring $S$ is a commutative monoid with identity $0$, which is additionally equipped with a function

$$S \times U \rightarrow U$$

$$(s, u) \rightarrow s \cdot u$$

called scalar multiplication such that for all $u, v \in U$ and $r, s \in S$, we have

- $(s \otimes r) \cdot u = s \cdot (r \cdot u)$
- $(s \oplus r) \cdot u = s \cdot u \oplus r \cdot u$
- $s \cdot (u \oplus v) = s \cdot u \oplus s \cdot v$
- $1 \cdot u = u$
- $s \cdot 0 = 0 = 0 \cdot u$

Our interest will be in semimodules generated by the rows of Hankel matrices.

**Definition 7.8.** An element $u$ in a semimodule $U$ is called a linear combination of $u_1, \ldots, u_k \in U$ if there exist $s_1, \ldots, s_k \in S$ such that

$$u = \bigoplus_{i=1}^{n} s_i \cdot u_i$$

A set of vectors generates all of its linear combinations.

**Definition 7.9.** If there exist finitely many elements $g_1, \ldots, g_m \in U$ which generated $U$, we say that $U$ is finitely generated.

As mentioned before, in contrast to vector spaces, there are several ways of defining the notion of independence for semimodules. For our purposes we adopt the definition 3.4 used in [51] Section 3] and in [25], but see also [26] 2.

**Definition 7.10.** A set of elements $P$ from a semimodule $U$ over a semiring $S$ is linearly independent if there is no element in $P$ that can be expressed as a linear combination of other elements in $P$.

Using this notion of linear independence, we define the notions of basis and dimension as in [51] 25:

**Definition 7.11.** A basis of a semimodule $U$ over a semiring $S$ is a set $P$ of linearly independent elements from $U$ which generate it, and the dimension of a semimodule $U$ is the cardinality of its smallest basis.
As hinted in the definition, the cardinality of a certain basis of a semimodule is not necessarily equal to its dimension.

Finally, we define the notion parallel to rank for matrices over semirings:

**Definition 7.12.** The row-rank $r(M)$ of the matrix $M$ is the dimension of the semimodule generated by its rows. In addition, we say that $M$ has maximal row-rank $mr(M) = k$ if $M$ has $k$ linearly independent rows and any set of $k + 1$ rows is linearly dependent.

The differences between fields and semirings are stressed further when we compare vector spaces and semimodules. It may seem trivial that any subspace of a vector space is of equal or lower dimension, but over a semiring this is not granted. In fact, not only can a semimodule have a subsemimodule of higher dimension, the former may be of finite dimension and the latter of infinite dimension:

**Example 7.13.** It was shown in [25] that the vectors $(s_i, 0, -s_i) \in \mathbb{R}^3_\text{max}$ for $s_i \in \mathbb{R}$ and $i = 0, 1, \ldots, m$ are linearly independent for any $m$ if the $s_i$ are different. Expanding this, consider the infinite vectors $u_i = (s_i, 0, -s_i, 0, 0, \ldots) \in T^\infty_\text{max}$ with $s_i \in \mathbb{R}$. It holds that, using the vectors

- $v_1 = (0, -\infty, -\infty, -\infty, -\infty, \ldots)$,
- $v_2 = (-\infty, -\infty, 0, -\infty, -\infty, \ldots)$,
- $v_3 = (-\infty, 0, -\infty, 0, 0, \ldots)$,

we can express any $u_i$ as the linear combination

$$u_i = (s_i, 0, -s_i, 0, 0, \ldots) = (\max\{s_i, -\infty\}, \max\{-\infty, 0\}, \max\{-s_i, -\infty\}, \max\{-\infty, 0\}, \max\{-\infty, 0\}, \ldots) = (s_i \cdot v_1) \oplus (-s_i \cdot v_2) \oplus v_3.$$  

Therefore the semimodule associated with a matrix $M$ whose rows exclude $v_1$, $v_2$, and $v_3$ and include infinitely many different $u_i$ is of infinite dimension, meaning the row-rank of $M$ is infinite, while being a subsemimodule of the 3-dimensional semimodule generated by $v_1$, $v_2$, and $v_3$.

Even when the rows of a matrix do generate a finite dimensional semimodule, i.e. its row-rank is finite, it does not indicate the value of its maximal row-rank:
Example 7.14. Consider the row-vectors:

\[
\begin{align*}
\mathbf{u}_0 &= (-\infty, -\infty, 0) \\
\mathbf{u}_1 &= (0, -\infty, -\infty) \\
\mathbf{u}_2 &= (-\infty, 0, -\infty) \\
\mathbf{u}_3 &= (0, -\infty, 0) \\
\mathbf{u}_4 &= (-\infty, 0, 0)
\end{align*}
\]

As a specific instance of Lemma 3.11 in [6], we have that \( \mathbf{u}_1, \mathbf{u}_2, \mathbf{u}_3, \) and \( \mathbf{u}_4 \) are linearly independent over \( T_{\text{max}} \), but clearly they are spanned by \( \mathbf{u}_0, \mathbf{u}_1 \) and \( \mathbf{u}_2 \). Therefore the matrix composed of the rows \( \mathbf{u}_1, \mathbf{u}_2, \mathbf{u}_3, \) and \( \mathbf{u}_4 \) would have row-rank 3 and maximal row-rank 4.

7.3 Graph Parameters Over Semirings

The examples from the previous section urge caution as we extend Theorem 6.2 to semirings. Our frame of reference in this cause is the formalism of graph algebras. Quantum graphs and their algebra readily transfer to semirings, as do their quotient algebra and Proposition 6.6.1.

Given a Hankel matrix \( \mathcal{H}(f, \Box) \) of an \( S \)-valued parameter \( f \), we denote the semimodule generated by its rows by \( \mathcal{M}(\mathcal{H}(f, \Box)) \). Although it was not expressed, the algorithm in the proof of Theorem 6.2 relied on the fact that over a field, if \( \mathcal{H}(f, \Box) \) has finite rank then a basis of \( \mathcal{M}(\mathcal{H}(f, \Box)) \) resides in \( \mathcal{H}(f, \Box) \). As previously illustrated, over a semiring, the maximal row-rank of a matrix may exceed its row-rank, nullifying Lemma 6.7.

We present two ways to overcome this. The first is to discard the bound on clique-width and consider other notions of width, for which weaker assumptions suffice, and the second is to consider semirings wherein the lemma does hold.

Arbitrary semirings

Assuming the algorithm we previously used is no longer tangible, consider the approach in automata theory from Chapter 5. The theorem characterizing word functions recognizable by weighted automata applied to any commutative semiring, suggesting that perhaps a similar statement can be formulated for graph parameters.

Although Theorem 5.10 was stated without mentioning Hankel matrices, the following definition, first introduced in [60], is implied in it:

**Definition 7.15 (J-finiteness).** A Hankel matrix \( \mathcal{H}(f, \Box) \) of an \( S \)-valued graph parameter \( f \) is \( J \)-finite if \( \mathcal{M}(\mathcal{H}(f, \Box)) \) is finitely generated.

In the case of words, the operation \( \Box \) was concatenation, so in our setting, \( \sqcup_k \) or \( \eta_k \) may play this role. But what would play the role of letters? Let us draw more
similarities between the inductive definition of graph classes and the definition of words over a finite alphabet Σ, and consider what a parse tree witnessing that \( w \in \Sigma^* \) would look like. Clearly, a word \( w \) is its own witness, and if we lay out its construction the result is shaped more like a string than a tree, as there is only a finite number of possible values \( \sigma \in \Sigma \) for the second operand in the concatenation operation.

As for inductively defined graph classes, if we restrict the second operand in the \( \Box \) operation to be from a fixed finite set of graphs, the resulting graph class would have string-looking parse trees as well. Once said fixed set of graphs is viewed as an alphabet, we may naturally regard the constructed graphs as words and use virtually the same proof of Theorem 5.10 in [9]. We therefore restrict the inductive step in the definitions of the graph classes of bounded tree-width and clique-width and define the graph classes of bounded path-width and linear clique-width. Although the definitions are very similar, we present them in full for readability.

**Definition 7.16** (Graphs of path-width \( k \)).

We start by defining the class \( \text{PW}(k) \) of \( k \)-labeled graphs:

(i) Every \( k \)-labeled graph of size at most \( k + 1 \) is in \( \text{PW}(k) \).

(ii) \( \text{PW}(k) \) is closed under disjoint union \( \Box \) and \( k \)-connection \( \sqcup_k \) where the second operand is a \( k \)-labeled graph of size at most \( k + 1 \).

(iii) Let \( \pi : [k] \to [k] \) be a partial relabeling function. If \( (G,\ell) \in \text{PW}(k) \) then also \( (G,\ell') \in \text{PW}(k) \) where \( \ell'(i) = \ell(\pi(i)) \).

We say that a graph \( G \) is of **path-width at most** \( k \) if there is a labeling \( \ell \) such that \( (G,\ell) \in \text{PW}(k) \).

Note the closure under \( \sqcup_k \), where only \( k \)-labeled graphs of size at most \( k + 1 \) are allowed as the second operand.

**Definition 7.17** (Graphs of linear clique-width \( 2^k \)).

We start by defining the class \( \text{LCW}(k) \) of \( k \)-labeled graphs:

(i) Every single-vertex \( k \)-colored graph is in \( \text{LCW}(k) \).

(ii) \( \text{LCW}(k) \) is closed under disjoint union \( \Box \) and \((i,j)\)-joins for \( i,j \leq k \) and \( i \neq j \), where the second operand is a single-vertex \( k \)-colored graph.

(iii) Let \( \rho : 2^k \to 2^k \) be a recoloring function. If \( (G,C) \in \text{LCW}(k) \) then also \( (G,C') \in \text{LCW}(k) \) where \( C'(I) = C(\rho(I)) \).

A graph \( G \) is of **linear clique-width at most** \( 2^k \) if there is a coloring \( C \) such that \( (G,C) \in \text{LCW}(k) \).

Again note that the closure under \( \eta_k \) only allows single-vertex \( k \)-colored graphs as the second operand.

Finally, we state the theorem:
**Theorem 7.18.** Let $S$ be an arbitrary commutative semiring. Let $f$ be an $S$-valued graph parameter and $k \in \mathbb{N}$ be fixed.

(i) If $H(f, \sqcup) is J$-finite for all $i \leq k$, then $f$ can be computed in polynomial time on graphs of path-width at most $k$.

(ii) If $H(f, \eta_k)$ is J-finite, then $f$ can be computed in polynomial time on graphs of linear clique-width at most $2^k$.

As explained above, the role of concatenation is played by $\sqcup_k$ or $\eta_k$ and the role of the letters is played by $k$-labeled graphs of size at most $k + 1$ or single-vertex $k$-colored graphs, respectively, giving us a finite set playing the role of $\Sigma$ in both cases.

**Tropical semirings**

We show that the algorithm in the proof of Theorem 6.2 can be applied in the case of tropical semirings. We use the fact that the row-rank and maximal row-rank of Hankel matrices over the tropical semirings coincide, [51, 25], and prove:

**Lemma 7.19.** If a Hankel matrix $H(f, \Box)$ over a tropical semiring with real entries has row-rank $r(H(f, \Box)) = m$, then there are $m$ rows in $H(f, \Box)$ which form a smallest basis $B$ of $MH(f, \Box)$.

**Proof.** As $r(H(f, \Box)) = m$, the dimension of $MH(f, \Box)$ is $m$ by definition. Suppose the set $B = \{g_1, \ldots, g_m\}$ is a smallest basis for $MH(f, \Box)$. Each $g_p$ is in $MH(f, \Box)$, therefore there is a finite linear combination of $\ell_p$ rows from $H(f, \Box)$ expressing $g_p$:

$$g_p = \bigoplus_{i_p=1}^{\ell_p} \alpha_{i_p} r_{i_p}$$

Consider the set of all the rows that appear in any of these linear combinations:

$$R = \bigcup_{p=1}^{m} \left( \bigcup_{i_p=1}^{\ell_p} r_{i_p} \right)$$

Since $H(f, \Box)$ is over a tropical semiring and has real entries, it holds that

$$mr(H(f, \Box)) = r(H(f, \Box)) = m$$

Therefore, any set of $m + 1$ rows from $H(f, \Box)$ is linearly dependent. Consider the result of the following process:

- Set $i = |R|$, and $B_i = R$. Note that $B_i$ is of size $i$ and generates $B$, and repeat until $i = m$:  

58
• Let \( r' \in B_i \) be a row that can be expressed using other rows in \( B_i \). Such an element must exist, as \( |B_i| > m \). Set \( B' = B_i - \{r'\} \), set \( i = i - 1 \) and \( B_i = B' \). Note that \( B_i \) is still of size (now smaller) \( i \) and it still generates \( B \).

When \( i = m \) is reached, we have \( B = B_m \) of size \( m \) which generates \( B \). This set must be independent: if it were not, we could perform more iterations of the above process and obtain a linearly independent set of size \( < m \) which generates \( B \). But the existence of such a set contradicts \( B \) being a smallest basis. Therefore, \( B \) is linearly independent and generates \( B \). Since \( B \) generates \( \text{MH}(f, \Box) \), so does \( B \), making \( B \) a smallest basis for \( \text{MH}(f, \Box) \) which resides in \( H(f, \Box) \).

Lastly, after establishing the fact that there lies a smallest basis \( B \) of \( \text{MH}(f, \Box) \) in \( H(f, \Box) \), we note we can find it in finite time, due to [25, Theorems 2.4 and 2.5], and obtain the extension of Theorem 6.2 to tropical semirings:

**Theorem 7.20.** Let \( f \) be a real-valued tropical graph parameter with finite join row-rank. Then \( f \) can be computed in polynomial time on graph classes of bounded clique-width.

The assumption on the values of \( f \) being real may seem restrictive at first, but it is often supported by the graph parameters motivating us, which are the solutions of optimization problems involving weighted graphs. As weight functions on graphs usually assign real-valued weights to the vertices and edges, these graph parameters are naturally real-valued.

### 7.4 LinEMSOL and the Tropical Semirings

In [3] it is shown that for graphs and hypergraphs of tree-width at most \( k \), the problems in LinEMSOL can be solved in linear time. Similarly, in [20] it is shown that for graphs \( G \) (but not hypergraphs) of clique-width at most \( k \) the problems in LinEMSOL can be solved in time linear in the size of the parse tree of \( G \).

By showing that problems in LinEMSOL have finite connection-rank or finite join-rank when they are interpreted as graph parameters over the tropical semirings, we get alternative (but not so different) proofs of these results.

One way is via definability: if a problem in LinEMSOL is shown to be in \( \text{MSOLEVAL}_{\text{max}} \) or \( \text{MSOLEVAL}_{\text{min}} \) then we have finite row-rank for the connection matrix by the [Finite Rank Theorem]. For example, suppose the weight functions \( f_1, \ldots, f_m \) of an LinEMSOL problem all have finite range \( N^{(1)}, \ldots, N^{(m)} \), where \( N^{(i)} = \{n_1^{(i)}, \ldots, n_{c_i}^{(i)}\} \) for \( i = 1, \ldots, m \). Then the expression

\[
\sum_{1 \leq i \leq \ell} \sum_{a \in z(U_i)} a_{ij}f_j(a) = \text{opt}\{ \sum_{1 \leq i \leq \ell} \sum_{a \in z'(U_i)} a_{ij}f_j(a) \mid \langle A, z' \rangle \models \varphi(U_1, \ldots, U_m) \}
\]

59
can be written as an MSOL-polynomial by addressing each case: for each $\ell$-tuple of sets $U = U_1, \ldots, U_\ell$, we consider all the possible values of the functions $f_1, \ldots, f_m$ and take the values that were assigned to the elements in each $U_i \in U$:

$$\sum_{\mathcal{U}, \mathbf{\psi}(\mathcal{U}) \in \mathcal{U}} \prod_{a: \mathbf{\psi}_{1,1}(a)} a_{1,1} \cdots \prod_{a: \mathbf{\psi}_{l,m,c}(a)} a_{l,m} n_{c,m}$$

where $\mathbf{\psi}_{i,j,c}(a)$ enforces that the element $a$ is in $U_i$ and that the function $f_j$ assigns the value $c$ to it:

$$\mathbf{\psi}_{i,j,c}(a) = U_i(a) \land (f_j(a) \approx c)$$

Assume w.l.o.g that $opt = \max$. Then when the above expression is considered to be over $T_{\max}$, we have:

$$\max_{\mathcal{U}, \mathbf{\psi}(\mathcal{U}) \in \mathcal{U}} \sum_{a: \mathbf{\psi}_{1,1}(a)} a_{1,1} \cdots \sum_{a: \mathbf{\psi}_{l,m,c}(a)} a_{l,m} n_{c,m}$$

which gives us the solution.

However, not all problems in LinEMSOL have weight functions with finite ranges, in which case it is not clear whether they are in MSOLEVAL or not. In some of these cases we may still have finite row-rank over the tropical semirings, for instance if the graph parameter implied by the solution is $\square$-maximizing or $\square$-minimizing. In this case one can use:

**Proposition 7.20.1.** Let $f$ be a real-valued graph parameter which is $\square$-maximizing (-minimizing). Then the Hankel matrix $H(f, \square)$ has finite row-rank when interpreted over $T_{\max}$ ($T_{\min}$).

**Proof.** Let $f$ be $\square$-maximizing (-minimizing). Let $r_0$ denote the infinite vector corresponding to the first row in $H(f, \square)$, $r_1$ denote the infinite vector whose entries are all 0 and let $r_i$ denote the row in $H(f, \square)$ corresponding to a graph $G_i$. Recall that 0 is the multiplicative identity in the tropical semirings, and have that $r_i$ is given by the linear combination

$$r_i = (f(G_i) \otimes r_1) \oplus r_0$$

since for any entry $r_i[j] = H(f, \square)_{G_i, G_j}$,

$$H(f, \square)_{G_i, G_j} = f(G_i \square G_j) = \max\{f(G_i), f(G_j)\} = \min\{f(G_i), f(G_j)\} = (f(G_i) \otimes 0) \oplus f(G_j) = (f(G_i) \otimes r_1)[j] + r_0[j]$$

therefore the row-rank of $H(f, \square)$ is at most 2. 

A possible direction in handling the tropical interpretation of LinEMSOL problems which are not in MSOLEVAL or with $\square$-maximizing (-minimizing) solutions will be addressed in [Chapter 8](#)
Chapter 8

Conclusion and open questions

We have studied how meta-theorems relating the MSOL- and CMSOL-definability of graph parameters to their computational complexity over certain graph classes may be reformulated without definability assumptions. We have seen that finiteness assumptions on the rank of the Hankel matrices involved suffices for the cases of tree-width and for the case of clique-width, and presented the extension of our treatment to commutative semirings. This assumption proved much weaker, as there are only countably many definable graph parameters, but uncountably many graph parameters satisfying our finiteness condition. In order to eliminate logic also from the meta-theorem involving sum-like inductive classes, we introduced linearly linked Hankel matrices and proved a logic-free analogous theorem that is a proper generalization of the former.

In our digression into word functions, we presented a characterization of the word functions recognizable by weighted automata using the formalism MSOLEVAL, which gave us an “if and only if” version of the Finite Rank Theorem, and discussed another formalism used to make the same characterization, WMSOL. We proved the two formalisms have the same expressive power over words by giving explicit translations from one to the other, which comes in addition to the proof via weighted automata.

8.1 Further Research

The Finite Rank Theorem implies that the Hankel matrices $H(f, \Box)$ of CMSOL-definable parameters have finite rank for all sum-like operations $\Box$. We start by asking the natural question:

**Problem 1.** Is there a parameter $f$ whose Hankel matrices $H(f, \Box)$ all have finite rank for sum-like operations $\Box$ and is not definable in CMSOL?

The counting argument we saw in Theorem 4.11 does not answer this question, as it only presents graph parameters with finite connection rank.

In Chapter 7 it was not clear how can LinEMSOL problems be treated in MSOLEVAL when their evaluation functions have infinite range, as we are limited in what may
appear in our MSOL-monomials. This can perhaps be resolved by formulating an MSOLEVAL-equivalent for meta-finite structures. Meta-finite structures, introduced in [48], are structures that are equipped with a weight function. The rank finiteness of these problems can then possibly be shown via a parallel of the Finite Rank Theorem for parameters on meta-finite structures.

Another natural direction for further research is to search for broader notions of width of graphs for which theorems similar to the ones seen in this thesis may be formulated. One can also try and relax the finiteness condition on the Hankel matrices.

A different direction

We suggest exploring a model-theoretic direction. Recalling the motivation of this thesis, we take another look at the meta-theorems discussed, and notice that in their proofs, logic played a role on two fronts; one having to do with the specific graph parameter in question and one having to do with the operations used to define the graph class. On both these fronts, the proof relied on a decomposition property valid in the logic. This property is spelled out in a Feferman-Vaught-type theorem for MSOL, which states, informally, that whether a τ-structure \( A = B_1 \sqcup B_2 \) satisfies a given formula or not depends only on which formulas does \( B_1 \) satisfy, and which does \( B_2 \).

Feferman-Vaught-type theorems are the most famous instances, [39, 38, 36, 37, 50, 71], of the general reduction theorems discussed in [74] Chapter 4.

In light of how useful it can be to have a Feferman-Vaught-type theorem hold in a logic, we ask:

**Problem 2.** Is there a characterization of the logics which satisfy a Feferman-Vaught-type theorem for the disjoint union? How about for sum-like operations?

And follow-up with:

**Problem 3.** Can a logic which does not satisfy a Feferman-Vaught-type theorem be extended to a logic that does? Under what conditions?

In particular, we suggest that investigating the relationship between logics with Feferman-Vaught-type theorems and the Hankel matrices of the parameters definable in these logics could lead to some interesting model-theoretic results.
Appendix A

Elaborations and Additional Proofs

This appendix includes details omitted from the body of the thesis.

Explicit definitions of the formulas in Chapter 2

We start with the formulas in Examples 2.6. We freely transition between the vocabulary of graphs $\tau_{graphs}$ and its extensions with constants, and assume the graphs are undirected.

Item (ii):

$$component(C) = connected(C) \land maximal(C)$$

with

$$connected(C) = \neg [\exists U (\text{subset}(U, C) \land \text{nonTrivial}(U) \land \text{closed}(U, C))]$$

$$\text{subset}(U, C) = \forall x (x \in U \rightarrow x \in C)$$

$$\text{nonTrivial}(U) = \exists x (x \in U) \land \exists y (y \notin U)$$

$$\text{closed}(U, C) = \forall x, y [((x \in C) \land (y \in C) \land E(x, y)) \rightarrow (x \in U \leftrightarrow y \in U)]$$

$$\text{maximal}(C) = \forall x [x \notin C \rightarrow \forall y (y \in C \rightarrow \neg E(x, y))]$$

Item (iii):

$$\text{clique}(C) = \forall x, y [(x \in C \land y \in C) \rightarrow E(x, y)]$$

Item (iv):

$$\text{maxClique}(C) = \text{clique}(C) \land \forall x [x \notin C \rightarrow \exists y (y \in C \land \neg E(x, y))]$$

Next is the formula $\psi_L$ mentioned in Examples 2.7(ii), where $L$ is a language defined by an MSOL-formula $\varphi_L$. We give a formula over the vocabulary $\tau_{words}^\Sigma$ of words in $\Sigma^*$, for $\Sigma = \{0, 1\}$. 

63
\[ \psi_L(U) = \text{interval}(U) \land \text{relativization}_{\varphi_L}(U) \]

with

\[ \text{interval}(U) = \forall x, y [(x \in U \land y \in U) \rightarrow \forall z ((x > z \land z > y) \rightarrow z \in U)] \]

and \( \text{relativization}_{\varphi_L}(U) \) is the result of the recursive transformation which restricts the quantifiers in \( \varphi \) to range over elements of \( U \). For example, if \( \varphi_L = \forall x.\phi(x) \), we have the restriction \( \forall x(x \in U \rightarrow \phi(x)) \).

**Explicit definitions of the functions in Chapter 5**

We give the MSOL-polynomials associated with the functions mentioned in Section 5.2.

First we define the formula:

\[ \text{block}(U) = \text{interval}(U) \land \forall x(x \in U \rightarrow P_1(x)) \]

Now we have:

\[
\begin{align*}
\text{f}_1(w) &= \sum_{U: \text{block}(U) \land \text{first}(U)} 1 \\
\text{l}_1(w) &= \sum_{U: \text{block}(U) \land \text{last}(U)} 1 \\
\text{i}_1(w) &= \sum_{U: \text{block}(U) \land \neg(\text{first}(U) \lor \text{last}(U))} 1
\end{align*}
\]

with

\[
\begin{align*}
\text{first}(U) &= \forall x[(\forall z. x \neq z \rightarrow x < z) \rightarrow x \in U] \\
\text{last}(U) &= \forall x[(\forall z. x \neq z \rightarrow z < x) \rightarrow x \in U]
\end{align*}
\]

**Proof of Proposition 6.6.1**

We restate.

**Proposition.** Let \( \Box \) be a commutative and associative operation on \( k \)-colored graphs, and let \( X' \in [X]_f^\Box \) and \( Y' \in [Y]_f^\Box \). Then

(i) \( X' + Y' \in [X + Y]_f^\Box = [X]_f^\Box + [Y]_f^\Box \)

(ii) \( a \cdot X' \in [a \cdot X]_f^\Box = a \cdot [X]_f^\Box \)

(iii) \( X' \odot Y' \in [X \odot Y]_f^\Box = [X]_f^\Box \odot [Y]_f^\Box \)
Proof. Let $X' = \sum_{v=1}^{n'} \lambda'_v G_v$ be in $[X]_f^\square$, let $Y' = \sum_{v=1}^{n'} \gamma'_v G_v$ be in $[Y]_f$ and let $Z = \sum_{u=1}^{\delta_u} \delta_u G_u$. Then by definition,

$$
\langle X' + Y', Z \rangle_{f,\square} = f((X' + Y')^\square Z)
$$

\[
= f \left( \max_{\{n', p'\}} \sum_{v=1}^{n'} \sum_{u=1}^{\ell} ((\lambda'_v + \gamma'_v) \cdot \delta_u) (G_v \square G_u) \right)
\]

\[
= \sum_{v=1}^{\max \{n', p'\}} \sum_{u=1}^{\ell} ((\lambda'_v + \gamma'_v) \cdot \delta_u) \cdot f(G_v \square F_u)
\]

\[
= \sum_{v=1}^{\max \{n', p'\}} \sum_{u=1}^{\ell} ((\lambda'_v \cdot \delta_u) + (\gamma'_v \cdot \delta_u)) \cdot f(G_v \square G_u)
\]

\[
= \sum_{v=1}^{\max \{n', p'\}} \sum_{u=1}^{\ell} ((\lambda'_v \cdot \delta_u) \cdot f(G_v \square G_u)) + (\gamma'_v \cdot \delta_u) \cdot f(G_v \square G_u))
\]

\[
= f((X' \square Z) + (Y' \square Z)) = f(X' \square Z) + f(Y' \square Z)
\]

Similarly we have

$$
\langle X + Y, Z \rangle_{f,\square} = f(X \square Z) + f(Y \square Z)
$$

Since $X' \in [X]_f^\square$ and $Y' \in [Y]_f^\square$, we have $f(X' \square Z) = f(X \square Z)$, and $f(Y' \square Z) = f(Y \square Z)$. Therefore, for all $k$-colored quantum graphs $Z$, we have

$$
\langle X + Y, Z \rangle_{f,\square} = f(X \square Z) + f(Y \square Z) = f(X' \square Z) + f(Y' \square Z) = \langle X' + Y', Z \rangle_{f,\square}
$$

which gives us item (i).

As for item (ii), we have:

$$
f(\alpha X' \square Z) = f \left( \sum_{v=1}^{n'} \sum_{u=1}^{\ell} ((\alpha \cdot \lambda'_v) \cdot \delta_u) (G_v \square G_u) \right)
$$

\[
= \sum_{v=1}^{n'} \sum_{u=1}^{\ell} ((\alpha \cdot \lambda'_v) \cdot \delta_u) \cdot f(G_v \square G_u)
\]

\[
= \alpha \cdot \sum_{v=1}^{n'} \sum_{u=1}^{\ell} (\lambda'_v \cdot \delta_u) \cdot f(G_v \square G_u)
\]

\[
= \alpha \cdot f(X' \square Z)
\]

Again, since $X' \in [X]_f^\square$ we have that for all $k$-colored quantum graphs $Z$,

$$
\langle \alpha X', Z \rangle_{f,\square} = \alpha f(X' \square Z) = \alpha f(X \square Z) = \langle \alpha X, Z \rangle_{f,\square}
$$

Lastly, item (iii) is proved using the associativity and commutativity of $\square$. For any
\[ f((X' \Box Y' \Box Z) \Box Z) = f((X' \Box (Y' \Box Z)) = f((X \Box (Y' \Box Z)) = f((Y' \Box Z) \Box X) \\
= f(Y' \Box (Z \Box X)) = f((Y \Box (Z \Box X)) = f((X \Box Y) \Box Z) \]
Bibliography


מנב התחומים

התחומים הלימודיים והתחומים הנוגעים לתחום בפער, 2, הם נוגעים לתחומי מחקר ופיתוח MSOLEVAL ואה תופעתים (MSOL) המודלובים, ו-CMSOLEVAL. המתחמים של תחום מחקר והתחומים


The text on the page is in Hebrew and contains a discussion on the properties of functions $f$, $g_1$, $g_2$, and $h$ in the context of computer science. It mentions concepts such as computer science departments, M.Sc. theses, and the Technion, a renowned university in Israel. The text also includes mathematical expressions and references to specific works or theses.

Here is a rough translation of the content:

"The function $f$ is continuous, and $g_1$, $g_2$, and $h$ are functions of $f$. The properties of these functions are discussed in the context of computer science departments and M.Sc. theses at the Technion. Further details are provided in the referenced works and theses."
תקציר

המתכון שעשה בנחתית פרופ' יוחי מקובסקי בפקולטהلدעי הלימודים

תודות

בماتכון לימודינו פרופ' יוחי מקובסקי סיפק לנו הדרכה והרשים התחליק. בזמננו שאנו עוד יום אחד למח totalPrice ועינו עוזרים לנו נוספים ורעיונות שצמצמו. ואנו בזמננו ילדי של בדיעים המהוים ו_motionקחיים, ידנו ונוירוט רועים רוח עניני. מעבר לכל, האמונת שלם בימינו איפשרה את קים התנה הוהי. אנו אסירת תדるので פרופ' מקובסקי על ידיה ועניני התחילה. הברגוי להודות וב/generated הקומ, ששרתיינו עפ פרופ' מקובסקי עוצר לעמדת האוניה במדת עניני. והם בזמננו, בכלי ימי רומם עיבורי בזמנ התנה לימים פרופ' מקובסקי.

אני מודה לעצמי,ครบ בדיעו העברית ונועית חיפה על התוכנה והרשים הבוחקים.
גדירות ומשריגות הנקל

حيح בר מתחרי

לשם مليי חקיק של הדרישות לليبלי התואר
มงคลר למדעים במדעי המחשב

נואית לבא

הונש לסנטה טקניזיה – מרכז טכנולוגיה לישראל
היפה אפריל 2015
גירוט ומטרייזות הנקל

נאדיה לבראי