DESCRIPTIVE COMPLEXITY OF LEARNING

Von der Fakultät für Mathematik, Informatik und Naturwissenschaften der RWTH Aachen University zur Erlangung des akademischen Grades eines Doktors der Naturwissenschaften genehmigte Dissertation

vorgelegt von

STEFFEN VAN BERGEREM, MASTER OF SCIENCE

aus Goch

Berichter: Professor Dr. Martin Grohe
Professor Dr. Sebastian Siebertz

Tag der mündlichen Prüfung: 10. März 2023

Steffen van Bergerem: *Descriptive Complexity of Learning*
March 2023

ABSTRACT

Supervised learning is a field in machine learning that strives to classify data based on labelled training examples. In the Boolean setting, each input is to be assigned to one of two classes, and there are several fruitful machine-learning methods to obtain a classifier. However, different algorithms usually come with different types of classifiers, e. g. decision trees, support-vector machines, or neural networks, and this is cumbersome for a unified study of the intrinsic complexity of learning tasks.

This thesis aims at strengthening the theoretical foundations of machine learning in a consistent framework. In the setting due to Grohe and Turán (2004), the inputs for the classification are tuples from a relational structure and the search space for the classifiers consists of logical formulas. The framework separates the definition of the class of potential classifiers (the *hypothesis class*) from the precise machine-learning algorithm that returns a classifier. This facilitates an information-theoretic analysis of hypothesis classes as well as a study of the computational complexity of learning hypotheses from a specific hypothesis class.

As a first step, Grohe and Ritzert (2017) proved that hypotheses definable in first-order logic (FO) can be learned in sublinear time over structures of small degree. We generalise this result to two extensions of FO that provide data-aggregation methods similar to those in commonly used relational database systems. First, we study the extension FOCN of FO with counting quantifiers. Then, we analyse logics that operate on weighted structures, which can model relational databases with numerical values. For that, we introduce the new logic FOWA, which extends FO by weight aggregation. We provide locality results and prove that hypotheses definable in a fragment of the logic can be learned in sublinear time over structures of small degree.

To better understand the complexity of machine-learning tasks on richer classes of structures, we then study the parameterised complexity of these problems. On arbitrary relational structures and under common complexity-theoretic assumptions, learning hypotheses definable in pure first-order logic turns out to be intractable. In contrast to this, we show that the problem is fixed-parameter tractable if the structures come from a nowhere dense class. This subsumes numerous classes of sparse graphs. In particular, we obtain fixed-parameter tractability for planar graphs, graphs of bounded treewidth, and classes of graphs excluding a minor.

ZUSAMMENFASSUNG

Überwachtes Lernen ist ein Teilgebiet des maschinellen Lernens, in dem Daten anhand von gelabelten Trainingsbeispielen klassifiziert werden. Bei der Boole'schen Klassifikation werden die Eingaben in zwei Kategorien einsortiert und es gibt mehrere effektive Lernmethoden, um einen Klassifikator zu erhalten. Unterschiedliche Methoden liefern allerdings oft unterschiedliche Arten von Klassifikatoren, wie z.B. Entscheidungsbäume, Support Vector Machines oder neuronale Netzwerke, was eine einheitliche Untersuchung der intrinsischen Komplexität von Lernproblemen sehr erschwert.

Ziel dieser Dissertation ist ein Ausbau der theoretischen Grundlagen des maschinellen Lernens innerhalb eines konsistenten formalen Rahmens. Im von Grohe und Turán (2004) eingeführten Modell sind die zu klassifizierenden Eingaben Tupel aus einer relationalen Struktur und der Suchraum für Klassifikatoren besteht aus logischen Formeln. Der Ansatz trennt die Definition der Klasse von möglichen Klassifikatoren (die *Hypothesenklasse*) vom konkreten Lernalgorithmus. Dies ermöglicht eine informationstheoretische Analyse der Hypothesenklassen sowie eine Untersuchung der Komplexität des Problems, Hypothesen aus einer bestimmten Klasse zu lernen.

Grohe und Ritzert zeigten 2017, dass in Prädikatenlogik erster Stufe (FO) definierbare Hypothesen in sublinearer Zeit auf Strukturen von kleinem Grad lernbar sind. Wir verallgemeinern das Resultat auf zwei FO-Erweiterungen, die Methoden zum Aggregieren von Daten liefern, welche denen in relationalen Datenbanksystemen ähneln.

Zunächst untersuchen wir die Logik FOCN, die FO um Zählquantoren erweitert. Dann analysieren wir Logiken, die auf gewichteten Strukturen operieren, welche numerische Werte in relationalen Datenbanken modellieren können. Dazu führen wir die neue Logik FOWA ein, welche FO um Methoden zur Gewichtsaggregation erweitert. Wir präsentieren Lokalitätsergebnisse und zeigen, dass in einem Fragment der Logik definierbare Hypothesen in sublinearer Zeit auf Strukturen von kleinem Grad lernbar sind.

Um die Komplexität von Lernproblemen auf allgemeineren Strukturen besser zu verstehen, untersuchen wir dann die parametrisierte Komplexität der Probleme. Unter weit verbreiteten komplexitätstheoretischen Annahmen stellt sich heraus, dass das Lernproblem für FO-definierbare Hypothesen auf beliebigen relationalen Strukturen nicht effizient lösbar ist. Im Gegensatz dazu zeigen wir, dass es auf Klassen von Strukturen, die nirgends dicht (*nowhere dense*) sind, einen im Sinne der parametrisierten Komplexität effizienten Algorithmus für das Problem gibt. Dies umfasst zahlreiche Klassen von dünnen Graphen, darunter planare Graphen, Graphen mit beschränkter Baumweite und Klassen von Graphen, die je einen Minoren ausschließen.

ACKNOWLEDGMENTS

First and foremost, my sincere thanks go to my supervisor Martin Grohe for his guidance and support. He gave me lots of freedom and opportunities to grow as a researcher and he always had an open door whenever I needed advice.

Furthermore, I am grateful to Nicole Schweikardt for inviting me for a research visit to Berlin. I enjoyed the productive discussions and pleasant atmosphere during my visit as well as the fruitful collaboration with her.

Moreover, I would like to thank my colleagues during my time at i7. I am particularly grateful to Sandra Kiefer for guiding me on my path to the post-graduate academic world and for sharing her unchallenged language skills. In addition, I want to thank Martin Ritzert for the nice discussions in our shared office, including our quest for the ideal office temperature, and Patrick Landwehr for always cheering me up with his card tricks and his interesting maths and automata theory challenges.

Finally, I am grateful to my parents for their continuous loving support and belief in me.

CONTENTS

1 INTRODUCTION

Descriptive complexity theory studies links between the computational and the descriptive complexity of (computational) problems. That is, it analyses how the computational resources needed to solve a problem are linked to the richness of a language needed to define the problem [50, 65]. The inputs of the problems are usually modelled as finite relational structures, and the problems are defined using logics.

We adapt this approach to machine-learning problems, where the task is to learn an unknown target function from given input-output pairs. We explore links between the computational complexity of learning certain functions and the descriptive complexity of these functions, i. e., the logics needed to define the functions we want to learn.

In the following, we introduce the complexity-theoretic background. Then, we present the machine-learning framework that we consider in this thesis, and we review known results for it. The chapter concludes with an outline of the thesis as well as a discussion of the scientific contribution.

DESCRIPTIVE COMPLEXITY THEORY

In 1974, Fagin initiated the field of descriptive complexity theory with his groundbreaking result [35], which states that the complexity class NP consists of exactly those problems that can be defined in existential second-order logic. In 1980, Chandra and Harel [24] raised the question whether there is a "natural" query language for relational databases that is able to express precisely those queries that can be evaluated in polynomial time. Gurevich [59] restated this question in terms of logics, asking whether there is a logic that captures the complexity class P. Immerman [64] and Vardi [92] both gave a partial answer by showing that least fixed-point logic (LFP) captures P over ordered structures. Although there have been several extensions of this result, it is still open whether there is a logic for P on arbitrary finite relational structures.

While descriptive complexity theory yields powerful completeness results, which show that all problems from a certain complexity class can be expressed in a certain logic, the algorithmic insights obtained from these results are usually limited [49]. In contrast to this, Courcelle's Theorem [26] does not claim completeness. It states that every property of graphs definable in monadic second-order logic can be decided in linear time on graphs of bounded treewidth. More formally, for every sentence φ in monadic second-order logic and every class $\mathcal{C}$ of graphs of bounded treewidth, the *model-checking* problem for φ on $\mathcal{C}$ can be decided in linear time. That is, it can be checked in linear time in the size of the input graph whether $G \models \varphi$ holds, i. e., whether an input graph G from the class $\mathcal{C}$ is a model of φ. Therefore, Courcelle's Theorem is actually a meta-theorem, yielding an efficient algorithm for every property-defining sentence and every class of graphs of bounded treewidth. A more recent example of an algorithmic meta-theorem is a result due to Grohe, Kreutzer, and Siebertz [52], which shows that every graph property definable in first-order logic can be decided in almost linear time on nowhere dense graph classes.

For a more refined complexity-theoretic analysis of model-checking problems, we measure the running time of algorithms not only in the size of the input structure, but also in terms of the length of a logical sentence that defines the property we are supposed to check. The length of such a sentence is usually small compared to the size of the input structure. Thus, when studying model-checking problems, it makes sense to relax the classical notion of tractability and allow for a non-polynomial running time in terms of the length of the sentence, while the dependence in terms of the size of the input structure still needs to be polynomial. More formally, for a sentence φ and a relational structure $\mathfrak{A}$, we look for cases in which the model-checking problem can be decided in time $f(|\varphi|) \cdot p(|\mathfrak{A}|)$, where $f\colon \mathbb{N} \to \mathbb{N}$ is some computable function, p is a polynomial, and $|\varphi|$ and $|\mathfrak{A}|$ are the length of the sentence and the size of the structure, respectively. As a matter of fact, the model-checking results [26, 52] actually show this relaxed version of tractability, called *fixed-parameter tractability*.

LEARNING LOGICS

We study the descriptive complexity of Boolean classification problems in the framework introduced by Grohe and Turán in 2002 [57]. In these problems, we are given a sequence of labelled tuples from a relational structure, where the labels are Boolean-valued. The goal is to return a function, called a *hypothesis* or a *concept*, which is consistent with (almost all of) the labels given in the examples and, ideally, can also be used to predict the labels of so far unseen instances. In machine learning, this problem falls into the category of *supervised learning* tasks: we want to learn a function from given input-output pairs. In contrast to this, in unsupervised learning (e. g. clustering tasks), the goal is to learn patterns from unlabelled data [85].

We require our algorithms to return a hypothesis from a predefined *hypothesis class*. In their work [57], Grohe and Turán give information-theoretic learnability results for hypothesis classes that can be defined using first-order and monadic second-order logic on restricted classes of relational structures, such as the class of planar graphs or graphs of bounded degree.

Algorithmic aspects of the framework, including the running time of a learning algorithm, were first studied by Grohe and Ritzert in [55], which forms the basis for our research. They showed that concepts definable in first-order logic can be learned in sublinear time on structures of small degree. We describe these results in more detail in Chapter 3. Analogous results have been obtained for monadic second-order logic by Grohe, Löding, and Ritzert [53] on strings and by Grienenberger and Ritzert [47] on trees.

Several problems related to the one we consider have been studied. This includes the framework of inductive logic programming (ILP) [25, 28, 66, 76, 77] and, in the database literature, various approaches to learning queries from examples [1, 5, 9, 10, 15, 16, 20, 61, 63, 67, 86, 87]. We give an overview of related work in Section 3.5.

OBJECTIVES AND OUTLINE

We aim at extending the theoretical foundations of machine learning in the aforementioned framework [57] in two directions.

First, we want to generalise the results for first-order logic [55] to extensions that provide data-aggregation methods similar to those in commonly used relational database systems. Such database systems usually allow counting of selected entries in the database. Therefore, we study extensions of first-order logic with counting quantifiers. Moreover, databases often include numerical values, which can be aggregated when querying data from the database. To account for that, we consider logics over weighted structures, an extension of ordinary relational structures, which enable us to explicitly model numerical values in a database. The logics provide different methods to aggregate these values.

To gain more insights about the complexity of machine-learning tasks on richer classes of structures, our second objective for this thesis is to find logics and classes of structures with fixed-parameter tractable machine-learning problems. In the spirit of Grohe's, Kreutzer's, and Siebertz's aforementioned first-order model-checking result [52], we analyse the parameterised complexity of learning first-order logic, and we identify classes of structures with a fixed-parameter tractable learning problem.

This thesis is structured as follows.

In Chapter 2, we introduce the relevant logics, discuss locality results for first-order logic, and give a short introduction into parameterised complexity.

In Chapter 3, we formally introduce the learning framework. To exemplify it, we discuss the results Grohe and Ritzert [55] obtained for learning first-order logic on structures of small degree. We complement them with negative (i. e., non-learnability) results, some of which have been published in [11]. At the end of the chapter, we give an overview of related work regarding the learning framework.

We generalise the results of [55] to the extension FOCN of first-order logic with counting quantifiers in Chapter 4. After discussing locality results for FOCN, we show in Section 4.3 that concepts definable in FOCN can be learned in sublinear time over classes of structures of bounded degree. In Section 4.4, we extend this to classes of structures of unbounded but still small degree.

In Chapters 5 and 6, we generalise the learnability results of [55] to weighted structures, which extend ordinary relational structures by assigning weights to tuples present in the structure. Such weighted structures were recently considered by Toruńczyk [88], who studied the complexity of query evaluation on these structures. Inspired by commonly used relational database systems, we consider enriched hypothesis classes, which include different methods to aggregate the numerical values. For that, in Chapter 5, we introduce a new logic, called *first-order logic with weight aggregation* (FOWA). In the remainder of the chapter, we provide locality results for fragments of this logic, including Feferman-Vaught decompositions and a Gaifman normal form. In Chapter 6, we use those to prove that concepts definable in a fragment of FOWA can be learned in sublinear time on weighted structures of small degree.

To analyse learning problems on richer classes of structures, we study the parameterised complexity of learning logics in Chapter 7. In Section 7.1, we show that learning concepts definable in first-order logic is AW[$*$]-hard in general. Under common complexity-theoretic assumptions, this means that the problem is not fixed-parameter tractable. In Sections 7.2 and 7.3, we identify tractable cases of the problem and show that it is fixed-parameter tractable on nowhere dense graph classes.

We conclude this thesis in Chapter 8 with a summary of the contents and a discussion of potential directions for future work.

PERSONAL CONTRIBUTION

Apart from explicitly cited results, this thesis contains only research results to which I contributed significantly. It is based on the following publications.

[11] Steffen van Bergerem. 'Learning Concepts Definable in First-Order Logic with Counting'. In: *34th Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2019, Vancouver, BC, Canada, June 24-27, 2019*

[13] Steffen van Bergerem and Nicole Schweikardt. 'Learning Concepts Described By Weight Aggregation Logic'. In: *29th EACSL*

Annual Conference on Computer Science Logic, CSL 2021, Ljubljana, Slovenia (Virtual Conference), January 25-28, 2021

[12] Steffen van Bergerem, Martin Grohe and Martin Ritzert. 'On the Parameterized Complexity of Learning First-Order Logic'. In: *PODS 2022: International Conference on Management of Data, Philadelphia, PA, USA, June 12-17, 2022*

In the following, I describe my contribution towards the publications.

The first publication [11] is a single-author paper written by myself. It is mainly based on [55], which introduces the algorithmic framework we consider in this thesis, and on [69], which introduces the logic FOCN and provides the necessary locality results for this logic. While the basic structure of the learnability proofs might appear reminiscent of the one in [55], new obstacles had to be overcome due to the fact that the locality results for first-order logic used in [55] do not apply to the logic FOCN. Especially the generalisation of the results from structures of bounded degree to structures of unbounded but small degree required new techniques. The paper [11] also provides non-learnability results, which are included in Chapter 3 of this thesis.

The publication [13] is joint work with Nicole Schweikardt, which started during a research visit in Berlin. In numerous research meetings, we commonly designed the logic FOWA and its fragments FOWA_1 and FOW_1. It is based on weighted structures, which Toruńczyk considered in [88], and the logic FOC_1, which Grohe and Schweikardt introduced in [56]. In the initial design phase, I worked on the applicability of the logic in machine-learning scenarios as well as the locality properties needed for the learnability results, while Nicole Schweikardt focused on the features of the logic needed to obtain the locality results. After we had worked out the design of the logic, I developed the learnability results, i. e., the contents presented in Chapter 6 of this thesis, and the main ideas for application scenarios. Furthermore, I also proved a consequence of the Feferman-Vaught result, which we use in the proof regarding the Gaifman normal form (see Corollary 5.9).

The third publication [12] is joint work with Martin Grohe and Martin Ritzert. The ideas for the hardness result evolved over a series of group discussions, while the final statement is due to Martin Grohe. Together with Martin Ritzert, I worked on the formalisation and presentation of the result. The two smaller tractability results (Propositions 7.3 and 7.4) are mostly due to Martin Ritzert, while I helped with the formalisation. The proof sketch for the main tractability result (Theorem 7.5) is due to Martin Grohe. I formalised and completed the proof.

2 PRELIMINARIES

2.1 GENERAL NOTATION AND DEFINITIONS

We let $\mathbb{R}$, $\mathbb{Q}$, $\mathbb{Z}$, $\mathbb{N}$, and $\mathbb{N}_{\geqslant 1}$ denote the sets of reals, rationals, integers, non-negative integers, and positive integers, respectively. For $m, n \in \mathbb{Z}$, we let $[m, n] \coloneqq \{\ell \in \mathbb{Z} \mid m \leqslant \ell \leqslant n\}$ and $[n] \coloneqq [1, n]$.

For a k-tuple $\bar{v} = (v_1, \ldots, v_k)$, we write $|\bar{v}|$ to denote its *length* k. We denote the empty tuple, i. e. the tuple of length 0, by $()$. The *cardinality* or *size* of a set S is the number of elements it contains and we denote it by $|S|$. We denote the power set of a set S by 2^S. Furthermore, for every $k \in \mathbb{N}$, we write $\binom{S}{k}$ for set of all k-element subsets of S.

Groups and rings

A *group* $(G, \circ)$ is a set G equipped with a binary operator $\circ \colon G \times G \to G$ that is associative (i. e. $(a \circ b) \circ c = a \circ (b \circ c)$ for all $a, b, c \in G$) and has a neutral element $e_G \in G$ (i. e. $a \circ e_G = e_G \circ a = a$ for all $a \in G$) such that each $a \in G$ has an inverse $a' \in G$ (i. e. $a \circ a' = a' \circ a = e_G$); we write a^{-1} for this a'. A group is *abelian* if $\circ$ is commutative (i. e. $a \circ b = b \circ a$ for all $a, b \in G$). A *ring* $(R, +, \cdot)$ is a set R equipped with two binary operators $+$ (*addition*) and $\cdot$ (*multiplication*) such that $(R, +)$ is an abelian group with neutral element $0_R \in R$, $\cdot$ is associative and has a neutral element $1_R \in R$, and multiplication is distributive with respect to addition, i. e. $a \cdot (b + c) = (a \cdot b) + (a \cdot c)$ and $(a + b) \cdot c = (a \cdot c) + (b \cdot c)$ for all $a, b, c \in R$. A ring is *commutative* if $\cdot$ is commutative.

2.2 RELATIONAL STRUCTURES

A *(relational) signature* is a finite set of relation symbols. Every relation symbol R has an *arity* $\mathrm{ar}(R) \in \mathbb{N}$. Let σ be a signature. A *(relational) structure* $\mathfrak{A}$ *over* σ, also called a σ*-structure*, is a tuple consisting of a finite set $U(\mathfrak{A})$, the *universe* of $\mathfrak{A}$, and a relation $R(\mathfrak{A}) \subseteq \big(U(\mathfrak{A})\big)^{\mathrm{ar}(R)}$ for every $R \in \sigma$. The size of $\mathfrak{A}$ is $|\mathfrak{A}| \coloneqq |U(\mathfrak{A})|$.

Expansion, reduct

Let $\sigma' \supseteq \sigma$ be a signature. A σ'-structure $\mathfrak{A}'$ is a σ'*-expansion* of a σ-structure $\mathfrak{A}$ if $U(\mathfrak{A}') = U(\mathfrak{A})$ and $R(\mathfrak{A}') = R(\mathfrak{A})$ for all $R \in \sigma$. If $\mathfrak{A}'$ is a σ'-expansion of the σ-structure $\mathfrak{A}$, then $\mathfrak{A}$ is the σ*-reduct* of $\mathfrak{A}'$.

Substructure

A σ-structure $\mathfrak{B}$ is a *substructure* of a σ-structure $\mathfrak{A}$ if $U(\mathfrak{B}) \subseteq U(\mathfrak{A})$ and $R(\mathfrak{B}) \subseteq R(\mathfrak{A})$ for every $R \in \sigma$. For a set $X \subseteq U(\mathfrak{A})$, the *induced substructure of* $\mathfrak{A}$ *on* X is the σ-structure $\mathfrak{A}[X]$ with universe $U(\mathfrak{A}[X]) = X$ and $R(\mathfrak{A}[X]) = R(\mathfrak{A}) \cap X^{\mathrm{ar}(R)}$ for every relation symbol $R \in \sigma$.

Union, intersection

The *union* of two σ-structures $\mathfrak{A}$ and $\mathfrak{B}$ is the σ-structure $\mathfrak{A} \cup \mathfrak{B}$ with universe $U(\mathfrak{A} \cup \mathfrak{B}) = U(\mathfrak{A}) \cup U(\mathfrak{B})$ and relations $R(\mathfrak{A} \cup \mathfrak{B}) = R(\mathfrak{A}) \cup R(\mathfrak{B})$ for all $R \in \sigma$. If $U(\mathfrak{A}) \cap U(\mathfrak{B}) = \emptyset$, we call $\mathfrak{A} \cup \mathfrak{B}$ the *disjoint union* of $\mathfrak{A}$ and $\mathfrak{B}$ and denote it by $\mathfrak{A} \uplus \mathfrak{B}$. Let $\sigma' := \sigma \cup \{X, Y\}$ for two new unary relation symbols X and Y that do not belong to σ. If $U(\mathfrak{A}) \cap U(\mathfrak{B}) = \emptyset$, then the *disjoint sum* of $\mathfrak{A}$ and $\mathfrak{B}$ is the σ'-expansion $\mathfrak{A} \oplus \mathfrak{B}$ of the disjoint union $\mathfrak{A} \uplus \mathfrak{B}$ with $X(\mathfrak{A} \oplus \mathfrak{B}) = U(\mathfrak{A})$ and $Y(\mathfrak{A} \oplus \mathfrak{B}) = U(\mathfrak{B})$. The *intersection* of two σ-structures $\mathfrak{A}$ and $\mathfrak{B}$ is the σ-structure $\mathfrak{A} \cap \mathfrak{B}$ with universe $U(\mathfrak{A} \cap \mathfrak{B}) = U(\mathfrak{A}) \cap U(\mathfrak{B})$ and relations $R(\mathfrak{A} \cap \mathfrak{B}) = R(\mathfrak{A}) \cap R(\mathfrak{B})$ for all $R \in \sigma$.

Graphs

A *graph* is relational structure with signature $\{E\}$ where E is a binary relation symbol. The universe of a graph G is called the *vertex set* of G and is often denoted by $V(G)$; the relation $E(G)$ is called the *edge set* of G. The elements of the vertex set are called *vertices* and the elements of the edge set are called *edges*. All graphs in this thesis are undirected and do not contain self-loops, i. e. E is symmetric and irreflexive. A unary relation symbol is called a *colour*. A *(σ-)coloured graph* is a σ-expansion of a graph where σ is a signature with $E \in \sigma$ and all other relation symbols in σ are colours.

Let G be a (coloured) graph. If $(v, w) \in E(G)$, then we say that v and w are *neighbours*, v and w are *incident* to (v, w), and v and w are *adjacent*. The *degree* $\deg(v)$ of a vertex $v \in V(G)$ is the number of neighbours of v and the degree $\deg(G)$ of G is the maximum degree of its vertices.

For $n \in \mathbb{N}$, a *path of length* n in G is a sequence $v_0, \ldots, v_n$ of distinct vertices in $V(G)$ such that $(v_i, v_{i+1}) \in E(G)$ for all $i \in [0, n-1]$. We say that $v_0, \ldots, v_n$ is a *path from* v_0 *to* v_n *in* G. If G is non-empty and there is a path from v to w for all $v, w \in V(G)$, then we say that G is *connected*. A *connected component* of G is an inclusion-wise maximal connected induced substructure of G.

Distance, neighbourhood

The *distance* $\mathrm{dist}^G(v, w)$ between two vertices $v, w \in V(G)$ is the minimal length of a path from v to w in G; if no such path exists, we set $\mathrm{dist}^G(v, w) := \infty$. For a tuple $\bar{v} = (v_1, \ldots, v_k) \in \big(V(G)\big)^k$ and a vertex $w \in V(G)$, we let $\mathrm{dist}^G(\bar{v}, w) := \min_{i \in [k]} \mathrm{dist}^G(v_i, w)$. For a tuple $\bar{w} = (w_1, \ldots, w_\ell) \in \big(V(G)\big)^\ell$, we set $\mathrm{dist}^G(\bar{v}, \bar{w}) := \min_{j \in [\ell]} \mathrm{dist}^G(\bar{v}, w_j)$. For $r \in \mathbb{N}$ and a tuple $\bar{v} \in \big(V(G)\big)^k$ for some $k \in \mathbb{N}$, the *ball of radius* r (or r-*ball*) of $\bar{v}$ in G is the set $N_r^G(\bar{v}) := \{w \in V(G) \mid \mathrm{dist}^G(\bar{v}, w) \leqslant r\}$. The *neighbourhood of radius* r (or r-*neighbourhood*) of $\bar{v}$ in G is the induced substructure $\mathcal{N}_r^G(\bar{v}) := G[N_r^G(\bar{v})]$. Let $C_1, \ldots, C_k$ be new colours not used in G. The *sphere of radius* r (or r-*sphere*) of $\bar{v}$ in G is the structure $\mathcal{S}_r^G(\bar{v})$ that is the expansion of $\mathcal{N}_r^G(\bar{v})$ by the colours $C_1, \ldots, C_k$ with $C_i(\mathcal{S}_r^G(\bar{v})) = \{v_i\}$ for all $i \in [k]$.

Fact 2.1. *Let* G *be a graph,* $\bar{v} = (v_1, \ldots, v_k) \in \big(V(G)\big)^k$ *for some* $k \in \mathbb{N}$ *with* $k \geqslant 2$*, and let* $r \in \mathbb{N}$*. The neighbourhood* $\mathcal{N}_r^{\mathfrak{A}}(v_1, v_2)$ *is connected if and only if* $\mathrm{dist}^G(v_1, v_2) \leqslant 2r+1$*. Furthermore, if* $\mathcal{N}_r^G(\bar{v})$ *is connected, then* $N_r^G(\bar{v}) \subseteq N_{r+(k-1)(2r+1)}^G(v_i)$ *for every* $i \in [k]$.

Gaifman graph

The *Gaifman graph* $G_{\mathfrak{A}}$ of a σ-structure $\mathfrak{A}$ is the graph with vertex set $V(G_{\mathfrak{A}}) = U(\mathfrak{A})$ and edge set $E(G_{\mathfrak{A}})$ that contains exactly those pairs

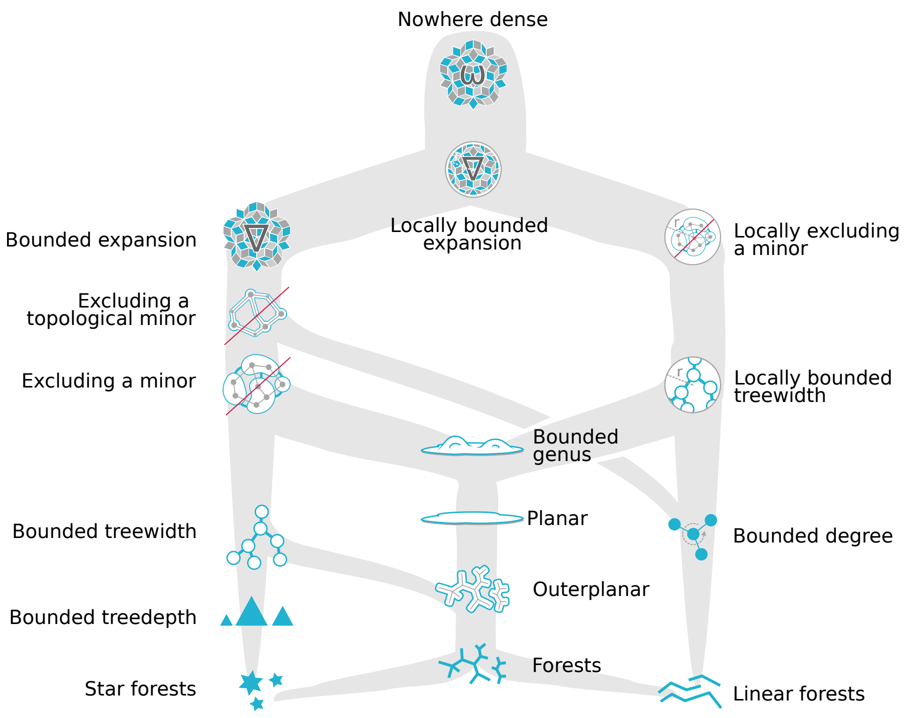


Figure 2.1: Illustration of classes of sparse graphs by Felix Reidl[1].

of distinct vertices $a, b \in U(\mathfrak{A})$ that appear in the same tuple of some relation of $\mathfrak{A}$, i. e. where $a, b \in \bar{v}$ for some $\bar{v} \in R(\mathfrak{A})$ and $R \in \sigma$.

We can generalise the graph-theoretic notions such as *degree*, *paths*, *connectivity*, *distance*, and *balls* from (coloured) graphs to general relational structures by applying the definitions to the corresponding Gaifman graphs. Using the generalised notion of balls, the notions of *neighbourhoods* and *spheres* also naturally generalise from (coloured) graphs to general relational structures.

2.2.1 *Nowhere Dense Classes*

In 2008, Nešetřil and Ossona de Mendez [78–80] introduced the notion of *nowhere dense graph classes*. This generalises different formalisations of classes of sparse graphs such as the class of planar graphs, classes of bounded degree, classes of bounded treewidth, or classes of graphs excluding a minor. See Figure 2.1 for an overview of these classes of structures. As we will see in Section 2.5, the notion of nowhere denseness has become a central criterion for tractability of several problems in parameterised complexity theory.

There are various, seemingly unrelated, equivalent characterisations of nowhere dense graph classes. In this thesis, we use a characterisation due to Grohe, Kreutzer, and Siebertz [52] via the so-called *splitter game*.

Splitter game

1 http://www-lti.informatik.rwth-aachen.de/~reidl/
(Date accessed: 2022-08-31)

Let G be a graph and $\ell, r \in \mathbb{N}_{\geqslant 1}$. The (ℓ, r)-*splitter game on* G is played by two players called *Connector* and *Splitter*. The game is played in a sequence of at most ℓ rounds. We let $G_0 \coloneqq G$. In round $i+1$ of the game, Connector chooses a vertex $v_{i+1} \in V(G_i)$. Then, Splitter chooses a vertex $w_{i+1} \in N_r^{G_i}(v_{i+1})$. We let $G_{i+1} \coloneqq G_i\left[N_r^{G_i}(v_{i+1}) \setminus \{w_{i+1}\}\right]$. If G_{i+1} is the empty graph, i.e. $V(G_{i+1}) = \emptyset$, then Splitter wins the game. Otherwise, the game continues. If Splitter has not won after ℓ rounds, Connector wins.

A *strategy* for Splitter in the (ℓ, r)-splitter game on G is a function f that associates a move $w_{i+1} \in N_r^{G_i}(v_{i+1})$ for Splitter to every partial play $(v_1, w_1, \dots, v_i, w_i)$ with associated graphs $G_0, \dots, G_i$ and move $v_{i+1} \in V(G_i)$ by Connector. A strategy is a *winning strategy* for Splitter in the (ℓ, r)-splitter game on G if Splitter wins every play in which they follow the strategy f. For a class $\mathcal{C}$ of graphs and a function $\lambda\colon \mathbb{N}_{\geqslant 1} \to \mathbb{N}_{\geqslant 1}$, we say that Splitter wins the λ-*splitter game on* $\mathcal{C}$ if for every $r \in \mathbb{N}_{\geqslant 1}$ and every graph $G \in \mathcal{C}$, Splitter has a winning strategy in the $(\lambda(r), r)$-splitter game on G.

Nowhere dense

Definition 2.2 (Nowhere dense class)**.** A class $\mathcal{C}$ of graphs is *nowhere dense* if there is a function $\lambda\colon \mathbb{N}_{\geqslant 1} \to \mathbb{N}_{\geqslant 1}$ such that Splitter wins the λ-splitter game on $\mathcal{C}$. The class $\mathcal{C}$ is *effectively nowhere dense* if λ is computable. A class $\mathcal{C}$ of relational structures is *(effectively) nowhere dense* if the class of Gaifman graphs of all structures in $\mathcal{C}$ is (effectively) nowhere dense.

2.3 LOGICS

In this section, we recapitulate the syntax and semantics of first-order logic as well as its extensions by counting and numerical predicates that we study in this thesis.

Throughout this section, let σ be a relational signature. Let vars and nvars be fixed, disjoint, and countably infinite sets of *structure variables* and *number variables*, respectively. In the logics described in this section, structure variables from vars denote elements from the structure and number variables from nvars denote integers.

Interpretation

A σ-*interpretation* $\mathcal{I} = (\mathfrak{A}, \beta)$ consists of a σ-structure $\mathfrak{A}$ and an *assignment* $\beta\colon \mathsf{vars} \cup \mathsf{nvars} \to U(\mathfrak{A}) \cup \mathbb{Z}$ with $\beta(x) \in U(\mathfrak{A})$ for every $x \in \mathsf{vars}$ and $\beta(\kappa) \in \mathbb{Z}$ for every $\kappa \in \mathsf{nvars}$. For $k, \ell \in \mathbb{N}$, k distinct structure variables $x_1, \dots, x_k \in \mathsf{vars}$, elements $v_1, \dots, v_k \in U(\mathfrak{A})$, ℓ distinct number variables $\kappa_1, \dots, \kappa_\ell \in \mathsf{nvars}$, and integers $n_1, \dots, n_\ell \in \mathbb{Z}$, we write $\mathcal{I}\frac{v_1,\dots,v_k}{x_1,\dots,x_k}\frac{n_1,\dots,n_\ell}{\kappa_1,\dots,\kappa_\ell}$ for the interpretation $(\mathfrak{A}, \beta\frac{v_1,\dots,v_k}{x_1,\dots,x_k}\frac{n_1,\dots,n_\ell}{\kappa_1,\dots,\kappa_\ell})$, where $\beta\frac{v_1,\dots,v_k}{x_1,\dots,x_k}\frac{n_1,\dots,n_\ell}{\kappa_1,\dots,\kappa_\ell}$ is the assignment β' with $\beta'(x_i) = v_i$ for every $i \in [k]$, $\beta'(\kappa_j) = n_j$ for every $j \in [\ell]$, and $\beta'(z) = \beta(z)$ for all $z \in (\mathsf{vars} \cup \mathsf{nvars}) \setminus \{x_1, \dots, x_k, \kappa_1, \dots, \kappa_\ell\}$.

First-order logic (FO)

Definition 2.3 (FO[σ])**.** The set of *formulas* for FO[σ] is built according to the following rules.

(1) $x_1{=}x_2$ and $R(x_1,\dots,x_k)$ are formulas for $x_1,x_2,\dots,x_k \in \mathsf{vars}$ and $R \in \sigma$ with $\mathrm{ar}(R) = k$.

(2) If φ and ψ are formulas, then $\neg\varphi$ and $(\varphi \vee \psi)$ are also formulas.

(3) If φ is a formula and $x \in \mathsf{vars}$, then $\exists x\, \varphi$ is a formula.

Let $\mathcal{I} = (\mathfrak{A}, \beta)$ be a σ-interpretation. For a formula φ from $\mathsf{FO}[\sigma]$, the semantics $[\![\varphi]\!]^{\mathcal{I}} \in \{0,1\}$ is defined as follows.

(1) $[\![x_1{=}x_2]\!]^{\mathcal{I}} = 1$ if $\beta(x_1) = \beta(x_2)$, and $[\![x_1{=}x_2]\!]^{\mathcal{I}} = 0$ otherwise; $[\![R(x_1,\dots,x_k)]\!]^{\mathcal{I}} = 1$ if $\big(\beta(x_1),\dots,\beta(x_k)\big) \in R(\mathfrak{A})$, and $[\![R(x_1,\dots,x_k)]\!]^{\mathcal{I}} = 0$ otherwise.

(2) $[\![\neg\varphi]\!]^{\mathcal{I}} = 1 - [\![\varphi]\!]^{\mathcal{I}}$ and $[\![(\varphi \vee \psi)]\!] = \max\{[\![\varphi]\!]^{\mathcal{I}}, [\![\psi]\!]^{\mathcal{I}}\}$.

(3) $[\![\exists x\, \varphi]\!]^{\mathcal{I}} = \max\{[\![\varphi]\!]^{\mathcal{I}\frac{v}{x}} \mid v \in U(\mathfrak{A})\}$.

The *quantifier rank* $\mathrm{qr}(\varphi)$ of an $\mathsf{FO}[\sigma]$-formula φ is the maximum nesting depth of constructs using rule (3) in order to construct φ. We write $(\varphi \wedge \psi)$ and $\forall x\, \varphi$ as shorthands for $\neg(\neg\varphi \vee \neg\psi)$ and $\neg\exists x \neg\varphi$.

Next, we consider the logic $\mathsf{FOC}(\mathbb{P})$ that Kuske and Schweikardt introduced in [69]. This logic allows building numerical statements based on counting terms as well as numerical predicates.

A *numerical predicate collection* is a triple $(\mathbb{P}, \mathrm{ar}, [\![.]\!])$ where $\mathbb{P}$ is a countable set of *predicate names*, and, to each $\mathsf{P} \in \mathbb{P}$, ar assigns an *arity* $\mathrm{ar}(\mathsf{P}) \in \mathbb{N}_{\geqslant 1}$ and $[\![.]\!]$ assigns a *semantics* $[\![\mathsf{P}]\!] \subseteq \mathbb{Z}^{\mathrm{ar}(\mathsf{P})}$. For the remainder of this section, fix a numerical predicate collection $(\mathbb{P}, \mathrm{ar}, [\![.]\!])$. FOC

Definition 2.4 ($\mathsf{FOC}(\mathbb{P})[\sigma]$)**.** The sets of *formulas* and *counting terms* for $\mathsf{FOC}(\mathbb{P})[\sigma]$ are built according to the rules (1)–(3) and the following rules.

(4) If φ is a formula and $\bar{x} = (x_1,\dots,x_k)$ is a tuple of k pairwise distinct variables, then $\#\bar{x}.\varphi$ is a counting term.

(5) Every integer $i \in \mathbb{Z}$ is a counting term.

(6) If t_1 and t_2 are counting terms, then $(t_1 + t_2)$ and $(t_1 \cdot t_2)$ are also counting terms.

(7) If $\mathsf{P} \in \mathbb{P}$, $m = \mathrm{ar}(\mathsf{P})$ and $t_1,\dots,t_m$ are counting terms, then $\mathsf{P}(t_1,\dots,t_m)$ is a formula.

Let $\mathcal{I} = (\mathfrak{A}, \beta)$ be a σ-interpretation. For a formula or counting term ξ from $\mathsf{FOC}(\mathbb{P})[\sigma]$, the semantics $[\![\xi]\!]^{\mathcal{I}}$ is defined by the rules (1)–(3) and the following rules.

(4) $[\![\#\bar{x}.\varphi]\!]^{\mathcal{I}} = \Big|\Big\{(v_1,\dots,v_k) \in \big(U(\mathfrak{A})\big)^k \mid [\![\varphi]\!]^{\mathcal{I}\frac{v_1,\dots,v_k}{x_1,\dots,x_k}} = 1\Big\}\Big|$, where $\bar{x} = (x_1,\dots,x_k)$.

(5) $[\![i]\!]^{\mathcal{I}} = i$ for $i \in \mathbb{Z}$.

(6) $[\![(t_1 + t_2)]\!]^{\mathcal{I}} = [\![t_1]\!]^{\mathcal{I}} + [\![t_2]\!]^{\mathcal{I}}$ and $[\![(t_1 \cdot t_2)]\!]^{\mathcal{I}} = [\![t_1]\!]^{\mathcal{I}} \cdot [\![t_2]\!]^{\mathcal{I}}$.

(7) $[\![\mathsf{P}(t_1, \dots, t_m)]\!]^{\mathcal{I}} = 1$ if $([\![t_1]\!]^{\mathcal{I}}, \dots, [\![t_m]\!]^{I}) \in [\![\mathsf{P}]\!]$, and $[\![\mathsf{P}(t_1, \dots, t_m)]\!]^{\mathcal{I}} = 0$ otherwise.

For counting terms t_1 and t_2, we write $(t_1 - t_2)$ as a shorthand for $\big(t_1 + ((-1) \cdot t_2)\big)$.

Finally, we describe the logic $\mathsf{FOCN}(\mathbb{P})$ introduced by Kuske and Schweikardt in [69] that includes number variables as well as quantification over numbers.

FOCN

Definition 2.5 ($\mathsf{FOCN}(\mathbb{P})[\sigma]$)**.** The sets of *formulas* and *counting terms* for $\mathsf{FOCN}(\mathbb{P})[\sigma]$ are built according to the rules (1)–(7) and the following rules.

(8) Every number variable $\kappa \in \mathsf{nvars}$ is a counting term.

(9) If φ is a formula and $\kappa \in \mathsf{nvars}$ is a number variable, then $\exists\kappa\,\varphi$ is a formula.

Let $\mathcal{I} = (\mathfrak{A}, \beta)$ be a σ-interpretation. For a formula or counting term ξ from $\mathsf{FOCN}(\mathbb{P})[\sigma]$, the semantics $[\![\xi]\!]^{\mathcal{I}}$ is defined by the rules (1)–(6) and the following rules.

(8) $[\![\kappa]\!]^{\mathcal{I}} = \beta(\kappa)$ for $\kappa \in \mathsf{nvars}$.

(9) $[\![\exists\kappa\,\varphi]\!]^{\mathcal{I}} = \max\big\{[\![\varphi]\!]^{\mathcal{I}\frac{n}{\kappa}} \bigm| n \in [0, |\mathfrak{A}|]\big\}$.

An *expression* is a formula or a counting term. Let ξ be an expression. The set $\mathsf{vars}(\xi)$ is the set of all variables in vars that occur in ξ. The set $\mathsf{nvars}(\xi)$ is defined analogously. The *free variables* $\mathrm{free}(\xi)$ of ξ are inductively defined as follows.

(1) $\mathrm{free}(x_1{=}x_2) = \{x_1, x_2\}$ and $\mathrm{free}\big(R(x_1, \dots, x_k)\big) = \{x_1, \dots, x_k\}$.

(2) $\mathrm{free}(\neg\varphi) = \mathrm{free}(\varphi)$ and $\mathrm{free}(\varphi \vee \psi) = \mathrm{free}(\varphi) \cup \mathrm{free}(\psi)$.

(3) $\mathrm{free}(\exists x\,\varphi) = \mathrm{free}(\varphi) \setminus \{x\}$ for $x \in \mathsf{vars}$.

(4) $\mathrm{free}\big(\#(x_1, \dots, x_k).\varphi\big) = \mathrm{free}\,\varphi \setminus \{x_1, \dots, x_k\}$.

(5) $\mathrm{free}(i) = \emptyset$ for $i \in \mathbb{Z}$.

(6) $\mathrm{free}\big((t_1 + t_2)\big) = \mathrm{free}\big((t_1 \cdot t_2)\big) = \mathrm{free}(t_1) \cup \mathrm{free}(t_2)$.

(7) $\mathrm{free}\big(\mathsf{P}(t_1, \dots, t_m)\big) = \bigcup_{i=1}^{m} \mathrm{free}(t_i)$.

(8) $\mathrm{free}(\kappa) = \{\kappa\}$ for $\kappa \in \mathsf{nvars}$.

(9) $\mathrm{free}(\exists\kappa\,\varphi) = \mathrm{free}(\varphi) \setminus \{\kappa\}$ for $\kappa \in \mathsf{nvars}$.

We write $\xi(z_1, \ldots, z_k)$ to indicate that $\text{free}(\xi) \subseteq \{z_1, \ldots, z_k\}$. A *sentence* is a formula without free variables and a *ground term* is a counting term without free variables.

Binding rank, binding width

The *binding rank* $\text{br}(\xi)$ of ξ is the maximal nesting depth of constructs using rules (3) and (4), i. e. constructs of the form $\exists x$ or $\#\bar{x}$, to construct ξ. The *binding width* $\text{bw}(\xi)$ of ξ is the maximal arity of an $\bar{x}$ of a term $\#\bar{x}.\psi$ in ξ. If ξ contains no such term, then $\text{bw}(\xi) = 1$ if ξ contains a quantifier $\exists x$ with $x \in \text{vars}$, and $\text{bw}(\xi) = 0$ otherwise.

For a formula φ and a σ-interpretation $\mathcal{I}$, we write $\mathcal{I} \models \varphi$ to indicate that $[\![\varphi]\!]^{\mathcal{I}} = 1$. Likewise, $\mathcal{I} \not\models \varphi$ indicates that $[\![\varphi]\!]^{\mathcal{I}} = 0$. For a formula φ, a σ-structure $\mathfrak{A}$, and a tuple $\bar{v} = (v_1, \ldots, v_k) \in \big(U(\mathfrak{A})\big)^k$, we write $\mathfrak{A} \models \varphi[\bar{v}]$ or $(\mathfrak{A}, \bar{v}) \models \varphi$ to indicate that $(\mathfrak{A}, \beta) \models \varphi$ for all assignments β with $\beta(x_i) = v_i$ for all $i \in [k]$. Furthermore, we set $[\![\varphi(\bar{v})]\!]^{\mathfrak{A}} := 1$ if $\mathfrak{A} \models \varphi[\bar{v}]$, and $[\![\varphi(\bar{v})]\!]^{\mathfrak{A}} := 0$ otherwise. Two expressions ξ, ξ' are *equivalent* if $[\![\xi]\!]^{\mathcal{I}} = [\![\xi']\!]^{\mathcal{I}}$ for all σ-interpretations $\mathcal{I}$. For $d \in \mathbb{N}$, the expressions are called d-*equivalent* if $[\![\xi]\!]^{\mathcal{I}} = [\![\xi']\!]^{\mathcal{I}}$ for all σ-interpretations $\mathcal{I} = (\mathfrak{A}, \beta)$ for all structures $\mathfrak{A}$ of degree at most d. The *length* $|\xi|$ of an expression ξ is the length of its encoding.

Example 2.6. Let G be a graph and let $\sigma = \{E\}$ and $\mathbb{P} = \{P_=\}$, where $P_=$ is the numerical predicate with $[\![P_=]\!] = \{(k, k) \mid k \in \mathbb{Z}\}$. We consider the $\mathsf{FOCN}(\mathbb{P})[\sigma]$-sentence

$$\varphi = \exists\kappa\, \forall x\, P_=\big(\#(y).E(x,y), \kappa\big).$$

The sentence has binding rank 2 and binding width 1. The sentence holds in G (i. e. $G \models \varphi$ holds) if and only if G is a regular graph, i. e., if there is some $k \in \mathbb{N}$ such that every vertex in G has degree k. The same statement can be expressed via the equivalent $\mathsf{FOC}(\mathbb{P})[\sigma]$-sentence

$$\psi = \forall x\, \forall y\, P_=\big(\#(z).E(x,z), \#(z).E(y,z)\big).$$

The sentence ψ has binding rank 3 and binding width 1.

Types

Let $\mathfrak{A}$ be a σ-structure and let $\bar{v} = (v_1, \ldots, v_k) \in \big(U(\mathfrak{A})\big)^k$ for some $k \in \mathbb{N}_{\geqslant 1}$. For a set of formulas Φ over the signature σ, the Φ-*type of* $\bar{v}$ *(in* $\mathfrak{A}$*)* is the set $\text{tp}^{\mathfrak{A}}_{\Phi}(\bar{v}) := \big\{\varphi(x_1, \ldots, x_k) \in \Phi \;\big|\; \mathfrak{A} \models \varphi[\bar{v}]\big\}$. For $q \in \mathbb{N}$, let $\mathsf{FO}[\sigma, q]$ denote the set of all formulas in $\mathsf{FO}[\sigma]$ with quantifier rank at most q. The q-*type of* $\bar{v}$ *(in* $\mathfrak{A}$*)* is the set $\text{tp}^{\mathfrak{A}}_{q}(\bar{v}) := \text{tp}^{\mathfrak{A}}_{\mathsf{FO}[\sigma,q]}(\bar{v})$. A k-*variable* q-*type (of signature* σ*)* is a set Ψ of $\mathsf{FO}[\sigma, q]$-formulas whose free variables are among $x_1, \ldots, x_k$.

Fact 2.7. *Up to equivalence, there are only finitely many* $\mathsf{FO}[\sigma, q]$*-formulas* φ *with* $\text{free}(\varphi) \subseteq \{x_1, \ldots, x_k\}$.

Formally, we can syntactically define a normal form for all σ, k, q such that there are only finitely many $\mathsf{FO}[\sigma, q]$-formulas in this normal form with free variables among $x_1, \ldots, x_k$. Furthermore, there is an algorithm that transforms every formula into an equivalent formula in normal form without increasing the quantifier rank. This allows us to view k-variable q-types as finite sets of formulas in normal form. We denote the set of all k-variable q-types of signature σ by $\mathrm{Tp}[\sigma, k, q]$.

Fact 2.8. *For every* $\mathsf{FO}[\sigma, q]$*-formula* $\varphi(x_1, \dots, x_k)$*, there is a set* $\Phi \subseteq \mathrm{Tp}[\sigma, k, q]$ *such that for all* σ*-structures* $\mathfrak{A}$ *and all* $\bar{v} \in \big(U(\mathfrak{A})\big)^k$*,*

$$\mathfrak{A} \models \varphi[\bar{v}] \iff \mathrm{tp}_q^{\mathfrak{A}}(\bar{v}) \in \Phi.$$

Before we discuss some of the locality properties of first-order logic in Section 2.4, we lastly consider the fragment $\mathsf{FOC}_1(\mathbb{P})$ of $\mathsf{FOC}(\mathbb{P})$ introduced by Grohe and Schweikardt in [56]. In Section 2.5, we will see that this fragment, while being relatively expressive, still provides sufficient locality properties to allow efficient query evaluation on nowhere dense structures.

FOC_1

Definition 2.9 ($\mathsf{FOC}_1(\mathbb{P})[\sigma]$)**.** The sets of *formulas* and *counting terms* for $\mathsf{FOC}_1(\mathbb{P})[\sigma]$ are built according to the rules (1)–(6) and the following restricted version of rule (7).

(7)$_1$ If $P \in \mathbb{P}$, $m = \mathrm{ar}(P)$ and $t_1, \dots, t_m$ are counting terms with $\big|\bigcup_{i=1}^m \mathrm{free}(t_i)\big| \leqslant 1$, then $P(t_1, \dots, t_m)$ is a formula.

By $\mathsf{FOCN}(\mathbb{P})$, we denote the union of all $\mathsf{FOCN}(\mathbb{P})[\sigma]$ for arbitrary relational signatures σ. This applies analogously to FO, $\mathsf{FOC}(\mathbb{P})$, and $\mathsf{FOC}_1(\mathbb{P})$.

Example 2.10. Let $\sigma = \{E\}$ and $\mathbb{P} = \{P_=\}$ as in Example 2.6. In the sentence $\psi = \forall x\, \forall y\, P_=\big(\#(z).E(x,z), \#(z).E(y,z)\big)$, expressing that a graph is regular, we have $\mathrm{free}\Big(P_=\big(\#(z).E(x,z), \#(z).E(y,z)\big)\Big) = \{x, y\}$. Hence, ψ is not contained in $\mathsf{FOC}_1(\mathbb{P})$. For every fixed $k \in \mathbb{N}$, however, we can express that a graph is k-regular via the $\mathsf{FOC}_1(\mathbb{P})$-sentence $\psi_k = \forall x\, P_=\big(\#(y).E(x,y), k\big)$.

2.4 LOCALITY OF FIRST-ORDER LOGIC

Local formulas

In this section, we study Gaifman's Locality Theorem and its implications on the locality of formulas from first-order logic. First, we introduce the notion of *local* formulas. Intuitively, the evaluation of a local formula only depends on the neighbourhood around the free variables up to a certain radius. The following definitions are based on [49]. Let σ be a relational signature and let $r \in \mathbb{N}$. An $\mathsf{FOCN}(\mathbb{P})[\sigma]$-formula $\varphi(\bar{x})$ with free variables $\bar{x} = (x_1, \dots, x_k)$ is r*-local (around* $\bar{x}$*)* if for every σ-structure $\mathfrak{A}$ and every tuple $\bar{v} = (v_1, \dots, v_k) \in \big(U(\mathfrak{A})\big)^k$, we have $\mathfrak{A} \models \varphi[\bar{v}] \iff \mathcal{N}_r^{\mathfrak{A}}(\bar{v}) \models \varphi[\bar{v}]$. A formula is *local* if it is r-local for some $r \in \mathbb{N}$.

Distance formulas

Let $\mathrm{dist}^{\sigma}_{\leqslant r}(x, y)$ be an $\mathsf{FO}[\sigma]$-formula such that for every σ-structure $\mathfrak{A}$ and all $v, w \in U(\mathfrak{A})$, we have $\mathfrak{A} \models \mathrm{dist}^{\sigma}_{\leqslant r}[v, w]$ if and only if

$\mathrm{dist}^{\mathfrak{A}}(v, w) \leqslant r$. Such a formula can be constructed recursively with quantifier rank at most $\lceil \log r \rceil$. To improve readability, we write $\mathrm{dist}^{\sigma}(x, y) \leqslant r$ instead of $\mathrm{dist}^{\sigma}_{\leqslant r}(x, y)$, and $\mathrm{dist}^{\sigma}(x, y) > r$ instead of $\neg\,\mathrm{dist}^{\sigma}_{\leqslant r}(x, y)$. We omit the superscript σ when it is clear from the context. For a tuple $\bar{x} = (x_1, \dots, x_k)$ of variables, $\mathrm{dist}(\bar{x}; y) > r$ is a shorthand for $\bigwedge_{i=1}^{k} \mathrm{dist}(x_i, y) > r$, and $\mathrm{dist}(\bar{x}; y) \leqslant r$ is a shorthand for $\bigvee_{i=1}^{k} \mathrm{dist}(x_i, y) \leqslant r$. For a tuple $\bar{y} = (y_1, \dots, y_\ell)$, we use $\mathrm{dist}(\bar{x}; \bar{y}) > r$ as a shorthand for $\bigwedge_{j=1}^{\ell} \mathrm{dist}(\bar{x}; y_j) > r$, and $\mathrm{dist}(\bar{x}; \bar{y}) \leqslant r$ as a shorthand for $\bigvee_{j=1}^{\ell} \mathrm{dist}(\bar{x}; y_j) \leqslant r$.

Basic local sentence

For $r \in \mathbb{N}$, a *basic local sentence (of radius r)* in $\mathsf{FO}[\sigma]$ is a sentence of the form

$$\exists x_1 \dots \exists x_k \Big(\bigwedge_{1 \leqslant i < j \leqslant k} \mathrm{dist}(x_i, x_j) > 2r \ \wedge \ \bigwedge_{i=1}^{k} \varphi(x_i) \Big),$$

where $k \in \mathbb{N}_{\geqslant 1}$, $x_1, \dots, x_k$ are k pairwise distinct variables, and $\varphi(x)$ is an r-local $\mathsf{FO}[\sigma]$-formula.

Gaifman normal form

Definition 2.11 (Gaifman normal form)**.** An $\mathsf{FO}[\sigma]$-formula is in *Gaifman normal form* if it is a Boolean combination of basic local sentences and local formulas. The *locality radius* of a formula φ in Gaifman normal form is the least r such that all basic local sentences in φ have radius at most r and every local formula in φ is r'-local for some $r' \leqslant r$.

Theorem 2.12 (Gaifman's Locality Theorem [41])**.** *Every $\mathsf{FO}[\sigma]$-formula is equivalent to a formula in Gaifman normal form. Furthermore, there is an algorithm that computes a formula in Gaifman normal form that is equivalent to a given $\mathsf{FO}[\sigma]$-formula.*

From the proof of Gaifman's Locality Theorem (cf. [41] and [49, Sect. 4.1]), it follows that, for a given formula with quantifier rank q, the locality radius of the equivalent formula in Gaifman normal form can be chosen to be at most $r(q) \in 2^{\mathcal{O}(q)}$, independent of the signature σ and the number of free variables of the given formula.

Local types

Let $\mathfrak{A}$ be a σ-structure and let $\bar{v} \in \big(U(\mathfrak{A})\big)^k$ for some $k \in \mathbb{N}_{\geqslant 1}$. For $q, r \in \mathbb{N}$, the *local (q, r)-type of $\bar{v}$ in $\mathfrak{A}$* is the set $\mathrm{ltp}^{\mathfrak{A}}_{q,r}(\bar{v}) := \mathrm{tp}^{\mathcal{N}^{\mathfrak{A}}_r(\bar{v})}_q(\bar{v})$. The following result is a consequence of Gaifman's Locality Theorem.

Fact 2.13. *For all $q \in \mathbb{N}$, there is an $r := r(q) \in 2^{\mathcal{O}(q)}$ such that for all $k \in \mathbb{N}_{\geqslant 1}$, all signatures σ, all σ-structures $\mathfrak{A}$, and all $\bar{v}, \bar{v}' \in \big(U(\mathfrak{A})\big)^k$, if $\mathrm{ltp}^{\mathfrak{A}}_{q,r}(\bar{v}) = \mathrm{ltp}^{\mathfrak{A}}_{q,r}(\bar{v}')$, then $\mathrm{tp}^{\mathfrak{A}}_q(\bar{v}) = \mathrm{tp}^{\mathfrak{A}}_q(\bar{v}')$.*

By combining Fact 2.8 and Fact 2.13, we obtain the following corollary.

Corollary 2.14. *Let $\varphi(x_1, \dots, x_k)$ be an $\mathsf{FO}[\sigma, q]$-formula, and let $r := r(q)$ be chosen according to Fact 2.13. Then, for every σ-structure $\mathfrak{A}$, there is a set Φ of k-variable q-types such that for all $\bar{v} \in \big(U(\mathfrak{A})\big)^k$,*

$$\mathfrak{A} \models \varphi[\bar{v}] \iff \mathrm{ltp}^{\mathfrak{A}}_{q,r}(\bar{v}) \in \Phi.$$

One of the main ingredients in the proof of Gaifman's Locality Theorem in [49] is the following "Feferman-Vaught style" composition lemma.

Lemma 2.15 (Composition Lemma [49]). *Let $\mathfrak{A}, \mathfrak{B}$ be σ-structures, $k \in \mathbb{N}$, $\ell, m \in \mathbb{N}_{\geqslant 1}$, $\bar{u} \in \big(U(\mathfrak{A})\big)^k$, $\bar{v} \in \big(U(\mathfrak{A})\big)^\ell$, and $\bar{w} \in \big(U(\mathfrak{B})\big)^m$ such that $\bar{u} = (u_1, \dots, u_k)$ and $U(\mathfrak{A}) \cap U(\mathfrak{B}) = \{u_1, \dots, u_k\}$. Then, for all $q \in \mathbb{N}$, $\mathrm{tp}_q^{\mathfrak{A} \cup \mathfrak{B}}(\bar{u}\bar{v}\bar{w})$ is uniquely determined by $\mathrm{tp}_q^{\mathfrak{A}}(\bar{u}\bar{v})$ and $\mathrm{tp}_q^{\mathfrak{B}}(\bar{u}\bar{w})$.*

Furthermore, there is an algorithm that computes $\mathrm{tp}_q^{\mathfrak{A} \cup \mathfrak{B}}(\bar{u}\bar{v}\bar{w})$ from $\mathrm{tp}_q^{\mathfrak{A}}(\bar{u}\bar{v})$ and $\mathrm{tp}_q^{\mathfrak{B}}(\bar{u}\bar{w})$.

For this thesis, we need the following local variant of the lemma that is a stronger version of a result given in [55].

Lemma 2.16 (Local Composition Lemma). *Let $\mathfrak{A}$ be a σ-structure, let $k, q \in \mathbb{N}$, $\ell, m \in \mathbb{N}_{\geqslant 1}$, $r = r(q)$ according to Fact 2.13, and $\bar{u} \in \big(U(\mathfrak{A})\big)^k$, $\bar{v} \in \big(U(\mathfrak{A})\big)^\ell$, and $\bar{w} \in \big(U(\mathfrak{A})\big)^m$ such that $\mathrm{dist}(\bar{v}, \bar{w}) > 2r + 1$. Then, $\mathrm{ltp}_{q,r}^{\mathfrak{A}}(\bar{u}\bar{v}\bar{w})$ is uniquely determined by $\mathrm{ltp}_{q,r}^{\mathfrak{A}}(\bar{u}\bar{v})$ and $\mathrm{ltp}_{q,r}^{\mathfrak{A}}(\bar{u}\bar{w})$.*

Furthermore, there is an algorithm that computes $\mathrm{ltp}_{q,r}^{\mathfrak{A}}(\bar{u}\bar{v}\bar{w})$ from the local types $\mathrm{ltp}_{q,r}^{\mathfrak{A}}(\bar{u}\bar{v})$ and $\mathrm{ltp}_{q,r}^{\mathfrak{A}}(\bar{u}\bar{w})$.

Proof. We prove the result by induction on $k \in \mathbb{N}$. For $k = 0$, i. e. for $\bar{u}$ being the empty tuple, the result follows directly from Lemma 2.15, since $\mathrm{ltp}_{q,r}^{\mathfrak{A}}(\bar{v}\bar{w}) = \mathrm{tp}_q^{\mathcal{N}_r^{\mathfrak{A}}(\bar{v}\bar{w})}(\bar{v}\bar{w})$ and $\mathcal{N}_r^{\mathfrak{A}}(\bar{v}\bar{w}) = \mathcal{N}_r^{\mathfrak{A}}(\bar{v}) \uplus \mathcal{N}_r^{\mathfrak{A}}(\bar{w})$.

Now let $k > 0$, $\bar{u} = (u_1, \dots, u_k)$, and let $\varphi(\bar{x}, \bar{y}, \bar{z})$ be a formula of quantifier rank at most q with $\bar{x} = (x_1, \dots, x_k)$, $|\bar{y}| = \ell$, and $|\bar{z}| = m$. We define a new structure $\mathfrak{A}'$ over an extended signature σ' by deleting the element u_k from $\mathfrak{A}$ and adding, for every relation symbol R in σ of arity $p \geqslant 2$, new relation symbols $R_1, \dots, R_p$ of arity $p - 1$ with

$$R_i(\mathfrak{A}') = \big\{(a_1, \dots, a_{i-1}, a_{i+1}, \dots, a_p) \;\big|\; (a_1, \dots, a_{i-1}, u_k, a_{i+1}, \dots, a_p) \in R(\mathfrak{A})\big\}.$$

Let $\bar{u}' := (u_1, \dots, u_{k-1})$ and $\bar{x}' := (x_1, \dots, x_{k-1})$. Using the new relation symbols, we can transform the formula $\varphi(\bar{x}, \bar{y}, \bar{z})$ into a new $\mathsf{FO}[\sigma', q]$-formula $\varphi'(\bar{x}', \bar{y}, \bar{z})$ such that $\varphi \in \mathrm{ltp}_{q,r}^{\mathfrak{A}}(\bar{u}\bar{v}\bar{w})$ if and only if $\varphi' \in \mathrm{ltp}_{q,r}^{\mathfrak{A}'}(\bar{u}'\bar{v}\bar{w})$. By the induction hypothesis, the local type $\mathrm{ltp}_{q,r}^{\mathfrak{A}'}(\bar{u}'\bar{v}\bar{w})$ can be computed from $\mathrm{ltp}_{q,r}^{\mathfrak{A}'}(\bar{u}'\bar{v})$ and $\mathrm{ltp}_{q,r}^{\mathfrak{A}'}(\bar{u}'\bar{w})$, which can be computed from $\mathrm{ltp}_{q,r}^{\mathfrak{A}}(\bar{u}\bar{v})$ and $\mathrm{ltp}_{q,r}^{\mathfrak{A}}(\bar{u}\bar{w})$. Thus, all in all, $\mathrm{ltp}_{q,r}^{\mathfrak{A}}(\bar{u}\bar{v}\bar{w})$ can be computed from $\mathrm{ltp}_{q,r}^{\mathfrak{A}}(\bar{u}\bar{v})$ and $\mathrm{ltp}_{q,r}^{\mathfrak{A}}(\bar{u}\bar{w})$. $\square$

In Chapters 4 and 6, we see analogous local composition results for $\mathsf{FOCN}(\mathbb{P})$ as well as for first-order logic with weight aggregation. For first-order logic with weight aggregation, in Chapter 5, we also provide a Gaifman normal form, while the results for $\mathsf{FOCN}(\mathbb{P})$ are based on so-called Hanf locality and Hanf normal forms.

2.5 PARAMETERISED COMPLEXITY

In this section, we introduce the central notions and results of parameterised complexity theory that are relevant for this thesis. We follow the formalism of Flum and Grohe [39]. In case of parameterised function problems, the definitions are based on the work of Fuhlbrück [40]. Let Σ be a finite alphabet. A *parameterised (decision) problem* over Σ is a pair (L, κ), where $L \subseteq \Sigma^*$ and $\kappa\colon \Sigma^* \to \mathbb{N}$ is a polynomial-time computable function, called *parameterisation* of Σ^*. If the parameterisation κ is clear from the context, we can omit it. A *parameterised function problem* over Σ is a pair (R, κ), where $R \subseteq \Sigma^* \times \Gamma^*$ for a finite alphabet Γ is a set of input-output pairs and κ is a parameterisation of Σ^*.

Fpt algorithm

Let κ be a parameterisation of Σ^*. An algorithm $\mathcal{A}$ with input alphabet Σ is an *fpt algorithm* with respect to κ if there is a computable function $f\colon \mathbb{N} \to \mathbb{N}$ and a polynomial p such that for every $x \in \Sigma^*$, the running time of $\mathcal{A}$ on input x is at most $f\big(\kappa(x)\big) \cdot p(|x|)$.

Fixed-parameter tractable, FPT

Definition 2.17 (Fixed-parameter tractable, FPT)**.** A parameterised decision problem (L, κ) is *fixed-parameter tractable* if there is an fpt algorithm with respect to κ that decides L. The class of all fixed-parameter tractable decision problems is denoted by FPT. A parameterised function problem (R, κ) is *fixed-parameter tractable* if there is an fpt algorithm with respect to κ that solves R, that is, for every input $x \in \Sigma^*$, if there exists an output $y \in \Sigma^*$ with $(x, y) \in R$, then the algorithm outputs one such y.

Let (L, κ) and (L', κ') be parameterised decision problems over the alphabets Σ and Σ', respectively. An *fpt (many-one) reduction* from (L, κ) to (L', κ') is a mapping $\alpha\colon \Sigma^* \to (\Sigma')^*$ such that for all $x \in \Sigma^*$, we have $x \in L$ if and only if $\alpha(x) \in L'$, α is computable by an fpt algorithm with respect to κ, and there is a computable function $g\colon \mathbb{N} \to \mathbb{N}$ such that $\kappa'\big(\alpha(x)\big) \leqslant g\big(\kappa(x)\big)$ for all $x \in \Sigma^*$.

Fpt reduction

Fpt Turing reduction

An *fpt Turing reduction* from (L, κ) to (L', κ') is an algorithm $\mathcal{A}$ with oracle access to L' such that $\mathcal{A}$ decides L, $\mathcal{A}$ is an fpt algorithm with respect to κ, and there is a computable function $g\colon \mathbb{N} \to \mathbb{N}$ such that $\kappa'\big(\alpha(x')\big) \leqslant g\big(\kappa(x)\big)$ for all oracle queries "$x' \in L'$?" on input x.

Lemma 2.18 ([39])**.** FPT *is closed under both types of fpt reductions, i. e., for two parameterised decision problems (L, κ) and (L', κ'), if there is an fpt (many-one or Turing) reduction from (L, κ) to (L', κ') and $(L', \kappa') \in$* FPT*, then $(L, \kappa) \in$* FPT.

For two parameterised function problems (R, κ) and (R', κ'), an *fpt Turing reduction* from (R, κ) to (R', κ') is an algorithm $\mathcal{A}$ with oracle access to R' such that $\mathcal{A}$ solves R, $\mathcal{A}$ is an fpt algorithm with respect to κ, and there is a computable function $g\colon \mathbb{N} \to \mathbb{N}$ such that $\kappa'\big(\alpha(x')\big) \leqslant g\big(\kappa(x)\big)$ for all oracle queries "Return y' such that $(x', y') \in R'$." on input x.

2.5.1 *Model Checking*

Model-checking problem

For a class Φ of formulas and a class $\mathcal{C}$ of relational structures, the *parameterised* Φ *model-checking problem on* $\mathcal{C}$, also called p-Φ-Mc($\mathcal{C}$) or p-Φ-Mc *on* $\mathcal{C}$, is defined as follows.

p-Φ-Mc($\mathcal{C}$)

Input:	structure $\mathfrak{A} \in \mathcal{C}$, sentence $\varphi \in \Phi$
Parameter:	$\lvert\varphi\rvert$
Problem:	Decide whether $\mathfrak{A} \models \varphi$ holds.

If $\mathcal{C}$ is the class of all relational structures or clear from the context, we can omit it. Let us briefly recapitulate some results on the complexity of the parameterised model-checking problem that are relevant for this thesis.

Theorem 2.19 ([30]). *The parameterised* FO *model-checking problem (on the class of all relational structures) is* AW[$*$]*-complete.*

Under the standard complexity theoretic assumption FPT $\neq$ W[1], this result by Downey, Fellows, and Taylor implies that in general, first-order model-checking is not fixed-parameter tractable. For more details on the parameterised complexity classes W[1] and AW[$*$], we refer to [39].

While Theorem 2.19 gives a negative result for the class of all relational structures, in 2017, Grohe, Kreutzer, and Siebertz [52] were able to show that the parameterised first-order model-checking problem is fixed-parameter tractable on nowhere dense classes.

Theorem 2.20 ([52]). *For every effectively nowhere dense class* $\mathcal{C}$ *of relational structures,* p-FO-Mc($\mathcal{C}$) *is fixed-parameter tractable.*

More precisely, for every effectively nowhere dense class $\mathcal{C}$ *of relational structures, there is a computable function* f *and an algorithm that, given* $\varepsilon > 0$*, an* FO*-sentence* φ*, and a structure* $\mathfrak{A} \in \mathcal{C}$*, decides whether* $\mathfrak{A} \models \varphi$ *holds in time* $f(\lvert\varphi\rvert, \varepsilon) \cdot \lvert\mathfrak{A}\rvert^{1+\varepsilon}$.

For classes of graphs that are closed under taking subgraphs, this result is optimal. That is, under the assumption FPT $\neq$ AW[$*$], if a class $\mathcal{C}$ that is closed under taking subgraphs is not nowhere dense, then the parameterised first-order model-checking problem on $\mathcal{C}$ is not fixed-parameter tractable [32, 68].

Moreover, Theorem 2.20 does not generalise to FOC($\mathbb{P}$). In [56], Grohe and Schweikardt gave fpt reductions from the first-order model-checking problem on the class of all graphs to the parameterised model-checking problem for FOC($\mathbb{P}$) on simple classes of structures such as the class of all strings or the class of all trees. In their reductions, they only require an equality predicate $P_= \in \mathbb{P}$.

Theorem 2.21 ([56]). *If $\mathcal{C}$ is the class of all strings (over the alphabet $\Sigma = \{a, b, c\}$) or the class of all trees, then the parameterised $\mathsf{FOC}(\{P_=\})$ model-checking problem on $\mathcal{C}$ is $\mathsf{AW}[*]$-complete.*

These findings resulted in the definition of the fragment $\mathsf{FOC}_1(\mathbb{P})$, which has a fixed-parameter tractable model-checking problem.

Theorem 2.22 ([56]). *For every effectively nowhere dense class $\mathcal{C}$ of relational structures, p-$\mathsf{FOC}_1(\mathbb{P})$-$\textsc{Mc}(\mathcal{C})$ is fixed-parameter tractable.*

More precisely, for every effectively nowhere dense class $\mathcal{C}$ of relational structures, there is a computable function f and an algorithm with a $\mathbb{P}$-oracle that, given $\varepsilon > 0$, an $\mathsf{FOC}_1(\mathbb{P})$-sentence φ, and a structure $\mathfrak{A} \in \mathcal{C}$, decides whether $\mathfrak{A} \models \varphi$ holds in time $f(|\varphi|, \varepsilon) \cdot |\mathfrak{A}|^{1+\varepsilon}$.

In the proof, Grohe and Schweikardt describe a decomposition of $\mathsf{FOC}_1(\mathbb{P})$-formulas into local FO-formulas and 0-ary relation symbols over an extended signature. The decomposed formula is then evaluated on an enriched structure that includes the new relation symbols. We present a similar decomposition for first-order logic with weight aggregation in Chapter 5.

3 LEARNING FIRST-ORDER LOGIC

Here, we introduce the learning framework that we consider in this thesis. We study Boolean classification problems. The input elements for the classification task come from a set X, the *instance space*. A *classifier* on X is a function $c\colon X \to \{0,1\}$. Given a *training sequence* T of labelled examples $(x_i, \lambda_i) \in X \times \{0,1\}$, we want to find a classifier, called a *hypothesis*, that explains the labels given in T and that can also be used to predict the labels of elements from X not given as examples. Examples (x_i, λ_i) with $\lambda_i = 1$ are called *positive examples*, while those with $\lambda_i = 0$ are called *negative examples*.

In this thesis, we study learning problems in the framework that was introduced by Grohe and Turán [57] and further studied in [11–13, 47, 53, 55]. There, the instance space X is a set of tuples from a (relational) structure, called the *background structure*, and classifiers are described using parametric models based on logics. Formally, for a background structure $\mathfrak{A}$ and the instance space $X = \big(U(\mathfrak{A})\big)^k$ for some $k \geqslant 1$, a hypothesis is defined via a formula φ and a parameter tuple $\bar{w} \in \big(U(\mathfrak{A})\big)^\ell$ as the mapping $h^{\mathfrak{A}}_{\varphi,\bar{w}}(\bar{x})\colon \big(U(\mathfrak{A})\big)^k \to \{0,1\}$ which maps a tuple $\bar{v} \in \big(U(\mathfrak{A})\big)^k$ to $[\![\varphi(\bar{v},\bar{w})]\!]^{\mathfrak{A}}$. We write $h_{\varphi,\bar{w}}(\bar{x})$ instead of $h^{\mathfrak{A}}_{\varphi,\bar{w}}(\bar{x})$ when the structure is clear from the context. If the hypothesis does not use any parameters, we write $h^{\mathfrak{A}}_{\varphi}(\bar{x})$ instead of $h^{\mathfrak{A}}_{\varphi,()}(\bar{x})$. The parameters can be seen as elements used as constants in the formula. Listing them explicitly as parameters allows a cleaner analysis of the computational complexity of the learning problems we introduce later in this chapter.

$h^{\mathfrak{A}}_{\varphi,\bar{w}}(\bar{x})$

Example 3.1. Let $G = \big(V(G), E(G)\big)$ be the structure shown in Figure 3.1.

(a) Let the instance space $X = V(G)$ be the set of all vertices in G and let the training sequence T contain the examples $(\text{Bob}, 1)$, $(\text{Carol}, 1)$, and $(\text{Emma}, 0)$, i. e. Bob and Carol are given as positive examples and Emma is given as a negative example. A plausible hypothesis would be $h\colon X \to \{0,1\}$ with $h(v) = 1$ if and only if v is a friend of Alice. We can also describe h via the first-order formula $\varphi(x,y) = E(x,y)$ and the parameter $w = \text{Alice}$ as $h = h_{\varphi,w}$. Then, $h(v) = [\![E(v, \text{Alice})]\!]^G$ for all $v \in V(G)$.

(b) We consider the instance space $X = \big(V(G)\big)^2$. Let the training sequence T contain $(\text{Alice}, \text{Emma})$, (Bob, Dan), and $(\text{Carol}, \text{Emma})$ as positive examples, and $(\text{Alice}, \text{Dan})$ and $(\text{Bob}, \text{Emma})$ as negative examples. The examples are also depicted in Figure 3.1. One hypothesis that is consistent with the labelled examples is

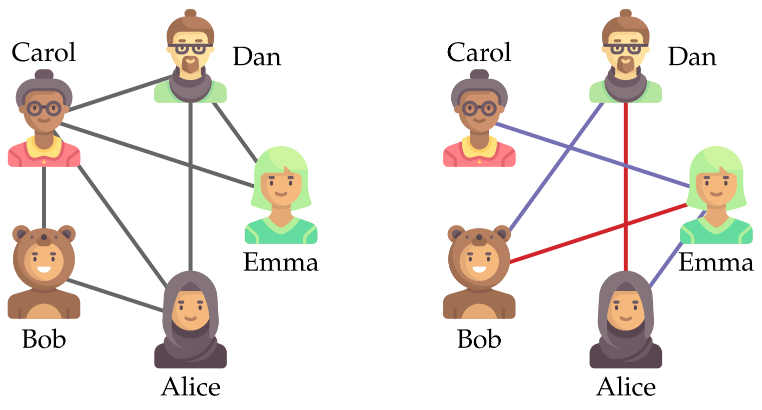


Figure 3.1: (Left) An excerpt of a social network[1]. An edge between two users indicates that they are friends. (Right) The training sequence from Example 3.1 (b). Positive examples are shown as purple edges while negative examples are shown in red.

$h\colon X \to \{0,1\}$ with $h(v_1,v_2) = 1$ if and only if v_1 and v_2 have a common friend who is not Carol. This hypothesis can be defined as $h = h_{\varphi,w}$ with $\varphi(x_1,x_2,y) = \exists z\,\big(E(x_1,z) \wedge E(x_2,z) \wedge \neg(z{=}y)\big)$ and $w = \mathrm{Carol}$, so $h(v_1,v_2) = [\![\varphi(v_1,v_2,\text{Carol})]\!]^G$ for all $v_1,v_2 \in V(G)$.

Regarding the requirements we impose on the hypotheses we are looking for, we consider the following well-known frameworks from computational learning theory.

Consistent learning

In *consistent learning*, the examples are assumed to be generated using an unknown classifier, the *target concept*, from a known *concept class*. The task is to find a hypothesis that is *consistent* with the training sequence T, i.e. a function $h\colon X \to \{0,1\}$ such that $h(x_i) = \lambda_i$ for all $(x_i,\lambda_i) \in T$. This is what we have done in Example 3.1. In Haussler's model of *agnostic probably approximately correct (PAC) learning* [62], a generalisation of Valiant's *PAC-learning* model [89], an (unknown) probability distribution $\mathcal{D}$ on $X \times \{0,1\}$ is assumed and training examples are drawn independently from this distribution. The goal is to find a hypothesis that generalises well, i.e. one is interested in algorithms that return with high probability a hypothesis with a small expected error on new instances drawn from the same distribution. We revisit both frameworks in Sections 3.3 and 3.4.

PAC learning

In the next sections, we discuss the main properties of the learning problems we consider in this thesis. We do this based on the results that Grohe and Ritzert obtained in [55] for concepts that can be described using first-order logic on structures of small degree.

1 Avatars designed by Freepik from Flaticon

3.1 LOCAL ACCESS AND COMPLEXITY MEASURES

Whenever we analyse learning problems in the context of classical complexity theory, we usually think of the background structures as being very large relational databases or huge graphs such as the web graph.

Hence, in case of relational databases, we would like to learn concepts from examples even if the database is too large to fit into the main memory. In case of the web graph, ideally our algorithms should also be able to explore only the regions of the web needed for learning, without having to rely on a previously gathered snapshot of the whole web graph saved to a hard disk.

Local access

Thus, within the subject of classical complexity theory, the learning algorithms we consider do not obtain the full representation of a background structure as input. Instead, we provide algorithms *local access* to the background structures, i. e., instead of having random access, algorithms may only retrieve the neighbours of vertices they already hold in memory, initially starting with the vertices given in the training examples. Formally, we give algorithms access to an oracle answering queries of the form "Is $\bar{v} \in R(\mathfrak{A})$?" and "Return the ith neighbour of v in $\mathfrak{A}$" in constant time. Often, instead of explicitly asking for neighbours of a vertex one after another, it will be convenient to use an oracle answering queries of the form "Return a list of all neighbours of v in $\mathfrak{A}$" in time linear in the number of neighbours of v. Similar access models have also been studied in property testing for structures of bounded degree [3, 4, 45] and, more broadly, in the subject of local algorithms [33, 71, 72, 82]. In addition to granting only local access, we want to learn concepts even without looking at the entire structure. Hence, we are mainly interested in learning problems that can be solved in sublinear time.

Uniform-cost measure

As our machine model, we use a random-access machine (RAM) model. Usually, we consider running times under the uniform-cost measure. This allows us to store an element of the background structure in a single memory cell and access it in a single computation step. The uniform-cost RAM model is commonly used in the database theory literature as well as in the analysis of algorithmic meta-theorems [8, 19, 31, 38, 48]. For further details on this model, we refer to [39]. In case of an analysis towards sublinear running times, we additionally consider the logarithmic-cost measure, where storing an element of a structure $\mathfrak{A}$ requires space $\mathcal{O}(\log|\mathfrak{A}|)$, so accessing and storing takes $\mathcal{O}(\log|\mathfrak{A}|)$ many steps. In this thesis, whenever we study the parameterised complexity of a problem, an additional log factor would have no impact on the results. Hence, regarding the parameterised complexity, we only consider the uniform-cost measure.

Data complexity

In contrast to the large background structures, we usually consider formulas as being human-written and hence, rather short. This justifies that in our complexity analyses within classical complexity theory, we focus on the *data complexity* of a problem, that is, we consider formulas as fixed and measure running times in terms of the size of the background structure, i. e. the number of its elements. This approach is also common in database theory when analysing the complexity of the query-evaluation problem [92]. When analysing the parameterised complexity of a problem, we use the length of the formula as a parameter.

3.2 CONSISTENT PARAMETER LEARNING

Before we introduce the model-learning problems that we study in the rest of this thesis, we start with the conceptually simpler parameter-learning problem. This has been introduced in [55]. Recall that a hypothesis $h^{\mathfrak{A}}_{\varphi,\bar{w}}$ is defined via a formula φ and a parameter tuple $\bar{w} \in \big(U(\mathfrak{A})\big)^{\ell}$. In parameter learning, we assume that the formula is already given and we only have to find the right parameters. While this sounds like an easier problem, we will see that having a fixed formula may force us to do more time-consuming computations than in cases where we can choose a formula for the hypothesis. In some cases, it even makes it impossible to solve the problem. The problem is formally defined as follows.

FO-Learn-Parameter

Input: structure $\mathfrak{A}$, FO-formula $\varphi(\bar{x}, \bar{y})$ with $|\bar{x}| = k$ and $|\bar{y}| = \ell$, training sequence $T \in \Big(\big(U(\mathfrak{A})\big)^{k} \times \{0,1\}\Big)^{m}$

Problem: Return a tuple $\bar{w} \in \big(U(\mathfrak{A})\big)^{\ell}$ such that the hypothesis $h^{\mathfrak{A}}_{\varphi,\bar{w}}$ is consistent with T, or reject the input if there is no such tuple.

To better understand the complexity of this problem, we use (a variation of) the example that Grohe and Ritzert gave in [55].

Example 3.2. Let P be a colour (i. e. a unary relation symbol) and let $G = \big(V(G), E(G), P(G)\big)$ be a coloured graph. Let $P(G) := \{w^*\}$ for some $w^* \in V(G)$. We consider the formula $\varphi(x, y) = P(y)$. Then, $h^{G}_{\varphi,w}$ is constant 1 if $w = w^*$, and constant 0 else. Suppose that the target concept is h^{G}_{φ,w^*}. In this example, an algorithm solving FO-Learn-Parameter would only receive positive training examples and the only way to solve the problem would be to find the parameter w^*. If the background structure is disconnected and there is no training example in the connected component of w^*, then the algorithm, having only local access to the background structure, is actually unable to find the correct parameter. Hence, the problem is not solvable with only local access.

Even if we require the background structure to be connected, algorithms are still unable to solve this problem in sublinear time. For instance, consider a coloured path with the training examples being at one end and the correct parameter being at the other end of the path. With only local access, an algorithm needs linear time to find the parameter.

Without the requirement to use φ as the formula in our hypothesis, we could have just used $\varphi'(x, y) = \mathrm{True}$ together with an arbitrary parameter.

This example shows that even on very simple structures, we are unable to solve the problem in sublinear time. Next, we analyse parameter learning in terms of its parameterised complexity. The following is a parameterised version of the problem.

p-FO-Learn-Parameter

Input:	structure $\mathfrak{A}$, FO-formula $\varphi(\bar{x}, \bar{y})$ with $\lvert\bar{x}\rvert = k$ and $\lvert\bar{y}\rvert = \ell$, training sequence $T \in \Big(\big(U(\mathfrak{A})\big)^k \times \{0, 1\}\Big)^m$
Parameter:	$\lvert\varphi\rvert$
Problem:	Return a tuple $\bar{w} \in \big(U(\mathfrak{A})\big)^\ell$ such that the hypothesis $h^{\mathfrak{A}}_{\varphi,\bar{w}}$ is consistent with T, or reject the input if there is no such tuple.

In [11], we gave a reduction from the parameterised clique problem to p-FO-Learn-Parameter. Due to Downey and Fellows [29], the parameterised clique problem is known to be complete for the parameterised complexity class W[1]. Although we did not explicitly state the reduction in [11] as an fpt reduction, one can easily see that the reduction implies that p-FO-Learn-Parameter is W[1]-hard.

Here, we present a stronger and yet much simpler result. Formally, in the complexity analysis of this problem, we consider its decision variant where the problem is to decide whether there is a tuple $\bar{w}$ that yields a consistent hypothesis. The following is an easy consequence of the AW[$*$] hardness of the parameterised FO model-checking problem (Theorem 2.19).

Corollary 3.3. p-FO-Learn-Parameter *is hard for the parameterised complexity class* AW[$*$] *under fpt reductions.*

Proof. We give an fpt many-one reduction from the parameterised FO model-checking problem to p-FO-Learn-Parameter. Then, the statement follows from Theorem 2.19.

Let $\mathfrak{A}$ be the structure and $\varphi \in \mathrm{FO}$ be the formula given as input in the model-checking problem. Let $\varphi'(x, y) = \varphi$ for two variables x, y that do not occur in φ. Moreover, let $T = \big((v, 1)\big)$ for some $v \in U(\mathfrak{A})$. Note that, since the evaluation of φ' is independent of the assignment to x and y, for every $w \in U(\mathfrak{A})$, the hypothesis $h^{\mathfrak{A}}_{\varphi,w}$ is consistent with T if and only if $\mathfrak{A} \models \varphi$ holds. Hence, $(\mathfrak{A}, \varphi) \in$ p-FO-Mc if and only if $(\mathfrak{A}, \varphi', T) \in$ p-FO-Learn-Parameter. □

Under common assumptions (cf. Section 2.5), this corollary implies that p-FO-Learn-Parameter is not fixed-parameter tractable.

Remark 3.4. Corollary 3.3 can even be extended to a weaker (promise) version of p-FO-Learn-Parameter, where we view the problem as a function problem and assert the existence of a tuple that yields a consistent hypothesis. The result can be shown via an fpt Turing reduction.

For that, let $(\mathfrak{A}, \varphi)$ be the input of the model-checking problem. We assume without loss of generality that $\mathfrak{A}$ has at least two distinct vertices. Now, let $\mathfrak{A}'$ be the expansion of $\mathfrak{A}$ with a fresh colour P, where we set $P(\mathfrak{A}') = \{w^*\}$ for some vertex $w^* \in U(\mathfrak{A}')$. As the input formula for the learning problem, we use

$$\varphi'(x, y) = \big(P(y) \land \varphi\big) \lor \big(\neg P(y) \land \neg\varphi\big).$$

Let $T = \big((v, 1)\big)$ for some $v \in U(\mathfrak{A}')$. Then, for every vertex $w \in U(\mathfrak{A})$, the hypothesis $h^{\mathfrak{A}'}_{\varphi',w}$ is consistent with T if and only if $w = w^*$ and $\mathfrak{A} \models \varphi$, or $w \neq w^*$ and $\mathfrak{A} \not\models \varphi$.

All in all, we can decide whether $\mathfrak{A} \models \varphi$ holds by computing $\mathfrak{A}'$, φ', and T, using an p-FO-Learn-Parameter oracle on input $(\mathfrak{A}', \varphi', T)$, and returning "yes" if and only if the returned parameter is w^*. The inputs $\mathfrak{A}'$, φ', and T can be computed in time linear in the size of $\mathfrak{A}$ and φ. Furthermore, $|\varphi'| \leqslant 2 \cdot |\varphi| + c$ for some constant c. Hence, the described reduction is an fpt Turing reduction.

3.3 CONSISTENT MODEL LEARNING

In the following, we introduce the model-learning problems that we will consider for the rest of this thesis. We start within the framework of consistent learning. That is, as described in the beginning of this chapter, we are given a sequence of training examples and we assume that the examples have been generated using an unknown target concept from a known concept class. Our task is to find a hypothesis that is consistent with the training sequence. In contrast to parameter learning, we have to find both the formula and the parameters of the hypothesis.

To make this problem feasible at all, we only consider concept classes of limited complexity. Concepts should be definable, like the hypotheses that we learn, via formulas and tuples of parameters. We limit the complexity of the formulas, and we also bound the numbers of parameters. Formally, for the learning problem on a background structure $\mathfrak{A}$ with k-tuples of elements given as examples, we require that the concept class can be defined as

$$\mathcal{H}_{\Phi^*,k,\ell}(\mathfrak{A}) \coloneqq \left\{h^{\mathfrak{A}}_{\varphi,\bar{w}} \;\middle|\; \varphi \in \Phi^*, \bar{w} \in \big(U(\mathfrak{A})\big)^{\ell}\right\}$$

for a set Φ^* of formulas $\varphi(\bar{x},\bar{y})$ with $|\bar{x}| = k$ and $|\bar{y}| = \ell$. To limit the complexity of the formulas in Φ^*, in case of first-order logic, the set will only contain formulas up a certain quantifier rank.

Since we would like to use the learned hypothesis to predict the label of tuples we have not seen yet, we also limit our choice of hypotheses and require that the hypothesis comes from a *hypothesis class* of limited complexity. We do this mainly for two reasons. First, we want to make sure that we are able to evaluate the hypothesis on new tuples efficiently. Second, we want to avoid *overfitting*, where the hypothesis perfectly fits the training examples, but it does so by simply memorising the examples instead of learning an underlying rule. As we will see in Section 3.4, limiting the complexity of a hypothesis class is a key ingredient to finding hypotheses that generalise well. In the results of this thesis, we usually allow algorithms to return hypotheses that are more complex than the concepts contained in the concept class. Hence, we use a hypothesis class $\mathcal{H}_{\Phi,k,\ell}(\mathfrak{A})$ with a set Φ of formulas that can be more complex than Φ^*.

To formally introduce the learning problem, let Φ^* and Φ be sets of formulas with $k+\ell$ free variables. The consistent model-learning problem for target concepts on k-tuples using only formulas from Φ^* and learned hypotheses using only formulas from Φ is defined as follows.

Learn-Consistent(k, Φ^*, Φ)

Input: structure $\mathfrak{A}$, training sequence $T \in \Big(\big(U(\mathfrak{A})\big)^{k} \times \{0,1\}\Big)^{m}$

Problem: Return a formula $\varphi \in \Phi$ and a tuple $\bar{w} \in \big(U(\mathfrak{A})\big)^{\ell}$ such that the hypothesis $h^{\mathfrak{A}}_{\varphi,\bar{w}}$ is consistent with T. The algorithm may reject if there is no formula $\varphi^* \in \Phi^*$ and tuple $\bar{w}^* \in \big(U(\mathfrak{A})\big)^{\ell}$ such that the hypothesis $h^{\mathfrak{A}}_{\varphi^*,\bar{w}^*}$ is consistent with T.

Now, we consider the model-learning problem for first-order logic that Grohe and Ritzert introduced in [55]. There, for fixed $k, \ell, q^* \in \mathbb{N}$ and a fixed signature σ, they considered concept classes based on first-order formulas of quantifier rank at most q^* with $k+\ell$ free variables, so

$$\Phi^* = \left\{\varphi(\bar{x},\bar{y}) \in \mathrm{FO}[\sigma, q^*] \;\middle|\; |\bar{x}| = k, |\bar{y}| = \ell\right\}.$$

FO-Learn-Consistent

By Fact 2.7, up to equivalence, there are only finitely many formulas in Φ^*. By Gaifman's Locality Theorem (Theorem 2.12), every single of those formulas is equivalent to a formula in Gaifman normal form. This shows that there is some $q \in \mathbb{N}$ such that every formula in Φ^* is equivalent to a formula in Gaifman normal form of quantifier rank at most q. Grohe and Ritzert use this q as the bound on the quantifier rank for Φ, i. e. they use $\Phi = \{\varphi(\bar{x}, \bar{y}) \in \mathrm{FO}[\sigma, q] \mid |\bar{x}| = k, |\bar{y}| = \ell\}$. This allows them in their algorithms to only look for formulas in Gaifman normal form. In the following, we will denote the resulting problem $\textsc{Learn-Consistent}(k, \Phi^*, \Phi)$ by $\textsc{FO-Learn-Consistent}(\sigma, k, \ell, q^*)$.

In general, this problem cannot be solved in sublinear time with only local access to the background structure.

Theorem 3.5. *Let σ be a signature that contains at least one relation symbol of arity at least 2. Then, for all $k, \ell \in \mathbb{N}_{\geqslant 1}$ and $q^* \geqslant 2$, there is no algorithm with only local access to the background structure that solves* $\textsc{FO-Learn-Consistent}(\sigma, k, \ell, q^*)$ *in time sublinear in the size of the background structure.*

Proof. First, we prove the statement for $k = \ell = 1$, $q^* = 2$, and $\sigma = \{E\}$ for a binary relation symbol E. This is based on a proof we presented in [11]. Afterwards, we generalise this result.

We prove the statement by contradiction. Assume that there is an algorithm solving $\textsc{FO-Learn-Consistent}(\sigma, k, \ell, q^*)$ in sublinear time. Choose $n \in \mathbb{N}$ such that for all $n' \geqslant n$, the algorithm uses at most $\frac{n'}{16}$ many steps on background structures of size n'.

Now, we construct two background structures $\mathfrak{A}_1$ and $\mathfrak{A}_2$ and corresponding training sequences T_1 and T_2 such that the algorithm is unable to distinguish the two inputs in sublinear time. Hence, the algorithm has to return the same formula and the same parameter on both inputs. As we will see, the resulting hypothesis has to be inconsistent with at least one of the two inputs, which then contradicts our assumption that the algorithm solves the problem.

The background structure $\mathfrak{A}_1$ is depicted in Figure 3.2. It is formally defined as the $\{E\}$-structure with

$$\begin{aligned}
U_{i,j} &= \{z_{i,j,p} \mid p \in [n]\} \quad \text{for } i \in [2] \text{ and } j \in [4],\\
U(\mathfrak{A}_1) &= \{x_1, x_2, x_3, x_4, y_1, y_2\} \cup \bigcup_{i \in [2],\, j \in [4]} U_{i,j},\\
R &= \big\{\{y_i, z_{i,j,p}\} \bigm| i \in [2], j \in [4], p \in [n]\big\}, && \text{(rows)}\\
C &= \big\{\{x_j, z_{i,j,p}\} \bigm| i \in [2], j \in [4], p \in [n]\big\}, && \text{(columns)}\\
E_1 &= \big\{\{z_{1,1,n-1}, z_{1,1,n}\}, \{z_{1,3,n-1}, z_{1,3,n}\},\\
&\qquad \{z_{2,1,n-1}, z_{2,1,n}\}, \{z_{2,4,n-1}, z_{2,4,n}\}\big\}, \text{ and}\\
E(\mathfrak{A}_1) &= R \cup C \cup E_1,
\end{aligned}$$

where $\{u, v\} \in E(\mathfrak{A}_1)$ means that both (u, v) and (v, u) are contained in $E(\mathfrak{A}_1)$. Intuitively, we can view the structure as eight sets of vertices $U_{i,j}$ being arranged in a table with two rows and four columns, and six additional vertices. The vertices y_1 and y_2 are used to indicate the first

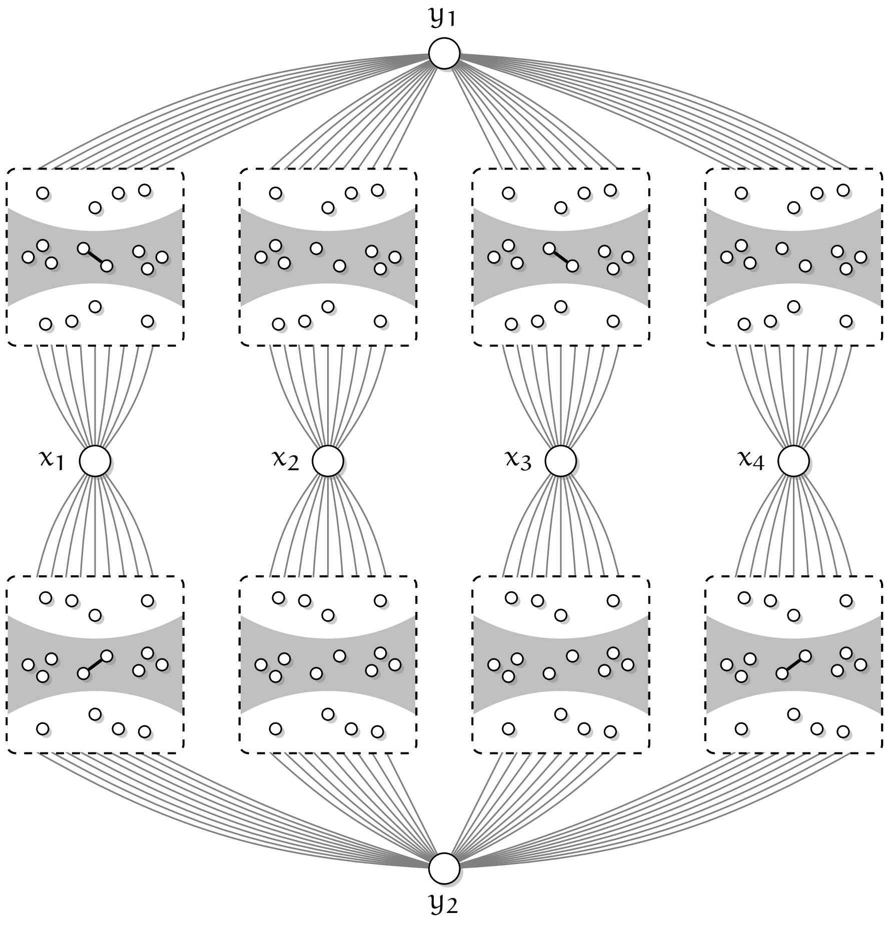


Figure 3.2: Background structure $\mathfrak{A}_1$ from the proof of Theorem 3.5. Eight sets of vertices are placed in a table with two rows and four columns. The y_i vertices are connected to all vertices in the sets in the ith row and the x_j vertices are connected to all vertices in the sets in the jth column. The vertices on the grey background are those parts of the background structure that the algorithm is unable to explore in sublinear time.

and second row. All vertices in a set in the ith row are connected to y_i via an edge. The vertices x_1 to x_4 are used to indicate the columns and the vertices in the jth column are connected to x_j via an edge. Finally, there are four additional edges within the table. In the first row, there is one edge connecting two vertices in the first column and one edge connecting two vertices in the third column. In the second row, there is an edge in the first and fourth column.

The structure $\mathfrak{A}_2$ is almost identical to $\mathfrak{A}_1$; only the additional edges differ. There, we have

$$\begin{aligned} E_2 = \big\{&\{z_{1,1,n-1}, z_{1,1,n}\}, \{z_{1,4,n-1}, z_{1,4,n}\}, \\ &\{z_{2,1,n-1}, z_{2,1,n}\}, \{z_{2,3,n-1}, z_{2,3,n}\}\big\} \text{ and} \\ E(\mathfrak{A}_2) = {}& R \cup C \cup E_2, \end{aligned}$$

i. e., the second edge in the first row is now in the fourth instead of the third column and in the second row the edge is in the third instead of the fourth column.

For the target concept in both background structures, we use

$$\varphi^*(x, y) = \exists z_1 \exists z_2 \left(E(x, z_1) \wedge E(x, z_2) \wedge E(y, z_1) \wedge E(y, z_2) \wedge E(z_1, z_2)\right).$$

For both training sequences, we use the labelled vertices x_1 to x_4 and we use y_1 or y_2 as a parameter. Hence, for the examples, the formula is satisfied if and only if there is an edge in the column indicated by x and the row indicated by y. For the first structure, we use y_1 as a parameter, so we select the first row. There, the first and third column contain an edge, so the resulting training sequence is

$$T_1 = \big((x_1, 1), (x_2, 0), (x_3, 1), (x_4, 0)\big).$$

For the second structure, we select y_2 as a parameter and hence the second row of $\mathfrak{A}_2$. There, again the first and third column contain an edge, so $T_2 = T_1$.

Because of our choice of n, we can make sure that, for a suitable ordering of the vertices in the background structures, the algorithm is unable to find any additional edge from E_1 or E_2. Hence, the algorithm is unable to distinguish the two inputs and will thus return the same formula φ and the same parameter w. Because the first and third column as well as the second and fourth column are indistinguishable for the algorithm, again, by choosing a suitable order on the vertices of the background structures, we can assume that the algorithm returns x_1, x_2, y_1, y_2, or some vertex from the first or second column as the parameter w.

We consider the isomorphism between $\mathfrak{A}_1$ and $\mathfrak{A}_2$ that keeps y_1, y_2 as well as the first and second column identical but swaps the third and fourth column (including x_3 and x_4). Note that the isomorphism also maps the parameter w to itself. The existence of such an isomorphism implies that the returned formula φ behaves in $\mathfrak{A}_1$ on x_3 like it does in $\mathfrak{A}_2$ on x_4, so $[\![\varphi(x_3, w)]\!]^{\mathfrak{A}_1} = [\![\varphi(x_4, w)]\!]^{\mathfrak{A}_2}$. However, in the training sequence $T_1 = T_2$, the vertices x_3 and x_4 have different labels. Hence, the algorithm cannot return on both $\mathfrak{A}_1$ and $\mathfrak{A}_2$ a consistent hypothesis, so it has to fail on at least one of them. This contradicts our assumption, so there is no algorithm solving FO-LEARN-CONSISTENT(σ, k, ℓ, q^*) in sublinear time for $\sigma = \{E\}$, $k = \ell = 1$ and $q^* = 2$.

Now, we generalise this result. Note that we did not use any bounds on the quantifier rank for the returned formula φ. Hence, our proof also works for larger values of q^*. If E is a relation symbol of higher arity, we can set the first two entries of the tuples like described above and then repeat the second entry to fill the rest of the tuple. Additional relation symbols have no influence on the argumentation presented above. Similarly, for $k > 1$, we can provide the same vertices as examples, but instead of using single vertices, we use tuples filled with the same vertex.

For $\ell > 1$, we use the disjoint union of ℓ copies of $\mathfrak{A}_1$ as the first background structure and proceed analogously for the second background structure. The training sequence consists of the vertices x_1 to x_4 with their corresponding labels from every single of those ℓ copies. Then, the algorithm either puts exactly one parameter in each of the copies, or there is at least one copy without any parameters. Thus, in both cases, there is at least one copy with at most one parameter. Hence, the argumentation from above still applies for this copy, showing that the algorithm is unable to provide a consistent hypothesis for at least one of the inputs. ☐

Grohe, Löding, and Ritzert [53] proved a similar result for learning on strings, although with a more restrictive local-access model.

To be able to learn first-order definable concepts in sublinear time, we need to restrict the class of background structures that we allow as input to our learning problem. Grohe and Ritzert [55] showed that the problem is solvable in time polynomial in the degree of the background structure and the number of examples in the training sequence.

Theorem 3.6 ([55]). *Let σ be a relational signature and let $k, \ell, q^* \in \mathbb{N}$. There is an algorithm that solves* FO-LEARN-CONSISTENT(σ, k, ℓ, q^*) *in time $(\log n + d + m)^{\mathcal{O}(1)}$ under the logarithmic-cost measure and in time $(d + m)^{\mathcal{O}(1)}$ under the uniform-cost measure, where n is the size and d is the degree of the background structure, and m is the length of the training sequence.*

On classes of structures of polylogarithmic degree, that is, classes $\mathcal{C}$ for which there is some $c \in \mathbb{N}$ such that $\deg(\mathfrak{A}) \in \mathcal{O}\big((\log |\mathfrak{A}|)^c\big)$ for all structures $\mathfrak{A}$ in $\mathcal{C}$, Theorem 3.6 implies that consistent model-learning is possible in sublinear time.

Corollary 3.7. *Let σ be a relational signature, let $k, \ell, q^* \in \mathbb{N}$, and let $\mathcal{C}$ be a class of structures of polylogarithmic degree. There is an algorithm that* *solves* FO-LEARN-CONSISTENT(σ, k, ℓ, q^*) *on* $\mathcal{C}$ *in time sublinear in the size of the background structure and polynomial in the length of the training sequence, under the logarithmic-cost as well as the uniform-cost measure.*

In the proof of Theorem 3.6, Grohe and Ritzert [55] provide a brute-force algorithm that tests all combinations of certain formulas and parameters. Since the assumed target concept uses a formula of quantifier rank at most q^*, by Gaifman's Locality Theorem (Theorem 2.12), there is an $r^* = r(q^*) \in \mathbb{N}$ such that the used formula is r^*-local. Furthermore, there is an equivalent formula in Gaifman normal form of quantifier rank at most q. Hence, Grohe and Ritzert test all formulas from the (up to equivalence) finite set of r^*-local formulas in Gaifman normal form of quantifier rank at most q. Using the locality of the considered formulas, they show that it suffices to limit the search for suitable parameters to a neighbourhood of a certain radius around the examples given in the training sequence. The size of the neighbourhood, and thus also the number of parameter tuples to test, is polynomial in the degree of the structure and the number of training examples. Finally, again due to the locality of the considered formulas, a single test of a hypothesis can be performed in time polynomial in the degree of the structure. All in all, this yields an algorithm with the desired running time bounds. For the detailed proof, we refer to [55].

In this thesis, we prove a similar result for the extension FOCN of first-order logic with counting quantifiers in Chapter 4. There, instead of using Gaifman locality and Gaifman normal forms, we use so-called Hanf locality and Hanf normal forms. In Chapter 6, we generalise Theorem 3.6 by describing properties of the sets Φ^* and Φ that guarantee that we can solve LEARN-CONSISTENT(k, Φ^*, Φ) via Gaifman locality and Gaifman normal forms in sublinear time, possibly with a precomputation step to build an index structure. Then, as an application of this general result, we show that concepts definable in first-order logic with weight aggregation can be learned in sublinear time.

3.4 PAC LEARNING

In this section, we introduce Haussler's model of *agnostic probably approximately correct (PAC) learning* [62], a generalisation of Valiant's *PAC-learning* model [89]. Moreover, to get familiar with this model within our logic learning framework, we discuss the agnostic PAC-learning results from [55] and study techniques to prove these results based on the consistent-learning results from the last section.

Intuitively, in (agnostic) PAC learning, we are interested in hypotheses that generalise well, i. e. hypotheses that not only work well on the examples from the training sequence but also on tuples not given as examples.

In PAC learning, we assume an (unknown) probability distribution $\mathcal{D}$ on the instance space X and, as in consistent learning, a consistent target concept $c\colon X \to \{0,1\}$. The learner's goal is to find a hypothesis $h\colon X \to \{0,1\}$, based on a sequence of training examples randomly drawn from $\mathcal{D}$, such that h minimises the *generalisation error*

$$\mathrm{err}_{\mathcal{D},c}(h) \coloneqq \Pr_{x\sim\mathcal{D}}\big(h(x) \neq c(x)\big),$$

i. e. the probability of being wrong on a random instance. In practice, we want to find a hypothesis with a generalisation error below a certain threshold ε.

In agnostic PAC learning, we drop the assumption of having a consistent target concept. Instead, we assume an (unknown) probability distribution $\mathcal{D}$ on $X \times \{0,1\}$. Again, a learning algorithm should find a hypothesis h that minimises the generalisation error, which is now defined as

$$\mathrm{err}_{\mathcal{D}}(h) \coloneqq \Pr_{(x,\lambda)\sim\mathcal{D}}\big(h(x) \neq \lambda\big).$$

Here, since a generalisation error of 0 might not be possible, we want to find a hypothesis with a generalisation error close to the best possible one.

Definition 3.8 (Agnostically PAC-learnable [85])**.** A hypothesis class $\mathcal{H}$ of hypotheses $h\colon X \to \{0,1\}$ is *agnostically PAC-learnable* if there is a function $m_{\mathcal{H}}\colon (0,1)^2 \to \mathbb{N}$ and a learning algorithm $\mathcal{L}$ with the following property: For all $\varepsilon, \delta \in (0,1)$ and for every distribution $\mathcal{D}$ over $X \times \{0,1\}$, when running $\mathcal{L}$ on a sequence T of m examples drawn i.i.d. from $\mathcal{D}$ with $m \geqslant m_{\mathcal{H}}(\varepsilon,\delta)$, it outputs a hypothesis $h \in \mathcal{H}$ such that, with probability of at least $1-\delta$ over the choice of training examples, it holds that

$$\mathrm{err}_{\mathcal{D}}(h) \leqslant \inf_{h'\in\mathcal{H}} \mathrm{err}_{\mathcal{D}}(h') + \varepsilon.$$

We call such an algorithm $\mathcal{L}$ an (agnostic) PAC-learning algorithm.

In this definition, we find two parameters, ε and δ. The first parameter ε, also called the accuracy parameter ("approximately correct"), describes how far the hypothesis returned by the algorithm is allowed to be from an optimal hypothesis. This allows the returned hypothesis to make a few mistakes, e. g. in case of outliers that are manually handled by an optimal solution but that we do not see in the limited number of training examples. The second parameter δ, also called the confidence parameter ("probably"), describes how confident we are to return a good hypothesis on a randomly chosen sequence of training examples. This refers to cases where the randomly chosen training sequence is not representative for $\mathcal{D}$, e. g. it consists only of positive examples or the same example is repeated over and over again. The function $m_{\mathcal{H}}$ determines, given the parameters ε and δ, the sample complexity of the problem, i. e. the number of examples needed to probably find an approximately correct hypothesis. For a more detailed discussion of (agnostic) PAC learning, we refer to [85].

Analogously to the results in the consistent-learning case, in [55], Grohe and Ritzert analysed a relaxed version of agnostic PAC learning. There, we want to approximately learn concepts from a concept class, but we allow the algorithms to return hypotheses from a slightly more complex hypothesis class. To define the agnostic PAC-learning problem within our logic learning framework, let Φ^* and Φ be sets of formulas with $k+\ell$ free variables. In addition to the previously defined membership and neighbourhood oracles for the background structure $\mathfrak{A}$, we allow algorithms to query the size $|\mathfrak{A}|$ of the structure. This information is needed to compute the sufficient length $m_{\mathcal{H}}(\varepsilon, \delta)$ of the training sequence. Furthermore, we give algorithms oracle access to the probability distribution $\mathcal{D}$ on $\big(U(\mathfrak{A})\big)^k \times \{0,1\}$. That is, whenever an algorithm queries the oracle, it receives a labelled example from $\big(U(\mathfrak{A})\big)^k \times \{0,1\}$ drawn from $\mathcal{D}$. The labelled examples are drawn independently of each other.

The agnostic PAC k-ary model-learning problem for Φ^*, with returned hypotheses using only formulas from Φ, is defined as follows.

Learn-PAC(k, Φ^*, Φ)

Input: structure $\mathfrak{A}$, rational numbers $\varepsilon, \delta > 0$, probability distribution $\mathcal{D}$ on $\big(U(\mathfrak{A})\big)^k \times \{0,1\}$

Problem: Return a formula $\varphi \in \Phi$ and a tuple $\bar{w} \in \big(U(\mathfrak{A})\big)^\ell$ such that, with probability of at least $1-\delta$ over the choice of examples drawn i.i.d. from $\mathcal{D}$, it holds that

$$\operatorname{err}_{\mathcal{D}}\big(h^{\mathfrak{A}}_{\varphi,\bar{w}}\big) \leqslant \varepsilon^* + \varepsilon,$$

where

$$\varepsilon^* := \min_{\substack{\varphi^* \in \Phi^*, \\ \bar{w}^* \in (U(\mathfrak{A}))^\ell}} \operatorname{err}_{\mathcal{D}}\big(h^{\mathfrak{A}}_{\varphi^*,\bar{w}^*}\big).$$

There is a strong connection between agnostic PAC learnability and the so-called VC dimension of a hypothesis class. It is known that, information theoretically, a hypothesis class is agnostically PAC-learnable if and only if it has finite VC dimension [14, 85, 91]. In particular, every finite hypothesis class is agnostically PAC-learnable. We briefly discuss this result in Section 7.3.

To solve the problem algorithmically, we can follow the *Empirical Risk Minimisation (ERM)* rule [85, 90], that is, our algorithm should return a hypothesis h that minimises the *training error* (or *empirical risk*)

$$\mathrm{err}_T(h) := \frac{1}{|T|} \cdot \left|\left\{(\bar{v}, \lambda) \in T \mid h(\bar{v}) \neq \lambda\right\}\right|$$

on the training sequence T of queried examples. Thus, in order to solve Learn-PAC(k, Φ^*, Φ), we first consider the following problem.

Learn-ERM(k, Φ^*, Φ)

Input: structure $\mathfrak{A}$, training sequence $T \in \left(\left(U(\mathfrak{A})\right)^k \times \{0,1\}\right)^m$

Problem: Return a formula $\varphi \in \Phi$ and a tuple $\bar{w} \in \left(U(\mathfrak{A})\right)^\ell$ such that

$$\mathrm{err}_T\left(h^{\mathfrak{A}}_{\varphi,\bar{w}}\right) \leqslant \min_{\substack{\varphi^* \in \Phi^*,\\ \bar{w}^* \in (U(\mathfrak{A}))^\ell}} \mathrm{err}_T(h^{\mathfrak{A}}_{\varphi^*,\bar{w}^*}).$$

This problem is very similar to the consistent model-learning problem. The only difference is that, instead of asking for a consistent hypothesis, we want to find a hypothesis that is at least as consistent as the best one from the concept class.

Now, we revisit the sets of first-order formulas Φ^* and Φ from the definition of FO-Learn-Consistent(k, Φ^*, Φ). Recall that Φ^* contains formulas of quantifier rank at most q^* and Φ contains formulas of quantifier rank at most q. The problems that Grohe and Ritzert studied are FO-Learn-ERM$(\sigma, k, \ell, q^*) :=$ Learn-ERM(k, Φ^*, Φ) and FO-Learn-PAC$(\sigma, k, \ell, q^*) :=$ Learn-PAC(k, Φ^*, Φ).

FO-Learn-ERM,
FO-Learn-PAC

To solve FO-Learn-ERM, they use a brute-force algorithm similar to the one they present for the problem FO-Learn-Consistent. However, instead of checking whether a hypothesis is consistent, they count the number of errors the hypotheses make on the training sequence and return the hypothesis that minimises this number.

To solve FO-Learn-PAC, the remaining missing ingredient is the following result that gives us a bound on the needed queried examples as well as a bound on the difference between the training and the generalisation error.

Lemma 3.9 (Uniform Convergence [85]). *Let $\mathcal{H}$ be a finite hypothesis class over the instance space X and let*

$$m^{\mathrm{UC}}_{\mathcal{H}}(\varepsilon, \delta) := \left\lceil \frac{\log(2\,|\mathcal{H}|\,/\delta)}{2\varepsilon^2} \right\rceil.$$

Then, for all $\varepsilon, \delta > 0$ and for every distribution $\mathcal{D}$ over $X \times \{0,1\}$, if a training sequence T of length at least $m^{\mathrm{UC}}_{\mathcal{H}}(\varepsilon, \delta)$ is drawn i.i.d. from $\mathcal{D}$, then, with probability at least $1 - \delta$, the training sequence is ε-representative, that is, for all $h \in \mathcal{H}$,

$$\left|\,\mathrm{err}_T(h) - \mathrm{err}_{\mathcal{D}}(h)\right| \leqslant \varepsilon.$$

Finally, we explain how Grohe and Ritzert combine the algorithm for FO-Learn-ERM with the Uniform Convergence Lemma to solve FO-Learn-PAC. Let $\mathfrak{A}$ be a background structure. We consider the concept class $\mathcal{C} = \mathcal{H}_{\Phi^*,k,\ell}(\mathfrak{A})$ and the hypothesis class $\mathcal{H} = \mathcal{H}_{\Phi,k,\ell}(\mathfrak{A})$. Since, up to equivalence, both Φ and Φ^* only contain finitely many formulas, the number of hypotheses in $\mathcal{C}$ and $\mathcal{H}$ is bounded by $s \cdot |\mathfrak{A}|^\ell$ for some constant s. We choose the number m of queried examples as $m(|\mathfrak{A}|, \varepsilon, \delta) := m_{\mathcal{H}}^{\mathrm{UC}}(\varepsilon/2, \delta)$. More specifically, we set

$$m(n, \varepsilon, \delta) := \left\lceil \frac{2 \log(2s \cdot n^\ell/\delta)}{\varepsilon^2} \right\rceil.$$

Now, let $\mathcal{D}$ be a distribution over $X \times \{0, 1\}$ and let $h^* \in \mathcal{C}$ be a hypothesis that minimises the generalisation error, that is, $\mathrm{err}_{\mathcal{D}}(h^*) = \min_{h' \in \mathcal{C}} \mathrm{err}_{\mathcal{D}}(h')$. Let T be a training sequence drawn i.i.d. from $\mathcal{D}$ of length at least $m(|\mathfrak{A}|, \varepsilon, \delta)$, and let $h \in \mathcal{H}$ be the hypothesis returned by an algorithm solving FO-Learn-ERM(σ, k, ℓ, q^*) on input T.

Recall that for every formula in Φ^*, there is an equivalent formula in Gaifman normal form in Φ. Thus, for the training error, we have $\mathrm{err}_T(h) \leqslant \mathrm{err}_T(h^*)$, since $h^* \in \mathcal{C} \subseteq \mathcal{H}$ and the algorithm returns a hypothesis h that minimises the training error. Moreover, by the Uniform Convergence Lemma, with probability at least $1 - \delta$, it holds that $\left| \mathrm{err}_T(h') - \mathrm{err}_{\mathcal{D}}(h') \right| \leqslant \frac{\varepsilon}{2}$ for all $h' \in \mathcal{H}$. This especially holds for h as well as for h^*. Hence,

$$\mathrm{err}_{\mathcal{D}}(h) \leqslant \mathrm{err}_T(h) + \frac{\varepsilon}{2} \leqslant \mathrm{err}_T(h^*) + \frac{\varepsilon}{2} \leqslant \mathrm{err}_D(h^*) + \frac{\varepsilon}{2} + \frac{\varepsilon}{2}$$

with probability at least $1 - \delta$. Note that this is the requirement we have in FO-Learn-PAC for the returned hypothesis. Thus, an algorithm solving the problem FO-Learn-ERM(σ, k, ℓ, q^*) also solves FO-Learn-PAC(σ, k, ℓ, q^*) when running it on $m = m(|\mathfrak{A}|, \varepsilon, \delta)$ queried examples. This number can be bounded by $\mathcal{O}\left(\frac{\log(|\mathfrak{A}|/\delta)}{\varepsilon^2}\right)$. All in all, this yields the following result.

Theorem 3.10 ([55])**.** *Let σ be a relational signature and let $k, \ell, q^* \in \mathbb{N}$. There is an algorithm that solves* FO-Learn-PAC(σ, k, ℓ, q^*) *in time* $(\log n + d + 1/\varepsilon + 1/\delta)^{\mathcal{O}(1)}$ *under the logarithmic-cost and the uniform-cost measure, where n is the size and d is the degree of the background structure.*

Analogously to the consistent model-learning problem, on classes of structures of polylogarithmic degree, Theorem 3.10 implies that probably approximately correct model-learning is possible in sublinear time.

Corollary 3.11. *Let σ be a relational signature, let $k, \ell, q^* \in \mathbb{N}$, and let $\mathcal{C}$ be a class of structures of polylogarithmic degree. There is an algorithm that solves* FO-Learn-PAC(σ, k, ℓ, q^*) *on $\mathcal{C}$ in time sublinear in the size of the background structure, under the logarithmic-cost as well as the uniform-cost measure.*

In Chapter 4, we prove PAC-learning results for logics with counting. In Chapter 6, we generalise the results to weighted structures and logics with weight aggregation.

Theorem 3.10 and Corollary 3.11 show a strong connection between consistent and PAC model learning. Only slight modifications are needed to turn the consistent-learning algorithm into an algorithm performing Empirical Risk Minimisation that can then be used within a PAC-learning algorithm. To conclude this section, we show that the strong connection also holds in the other direction. That is, analogously to a proof by Grohe, Löding, and Ritzert in [53], we transform Theorem 3.5, our negative result for the consistent model-learning problem, into a negative result for the PAC model-learning problem.

Theorem 3.12. *Let σ be a signature that contains at least one relation symbol of arity at least 2. Then, for all $k, \ell \in \mathbb{N}_{\geqslant 1}$ and $q^* \geqslant 2$, there is no algorithm with only local access to the background structure that solves* FO-LEARN-PAC(σ, k, ℓ, q^*) *in time sublinear in the size of the background structure.*

Proof. This proof is based on the proof of Theorem 3.5. We only consider the case $k = \ell = 1$, $q^* = 2$ and $\sigma = \{E\}$ for a binary relation symbol E. The generalisation can be done analogously to the original proof. Let $\mathfrak{A}_1$ and $\mathfrak{A}_2$ be the background structures and $T := T_1 = T_2$ be the training sequences from the proof. Let $\mathcal{D}$ be the uniform distribution over the examples from T, that is, $(x_1, 1)$, $(x_2, 0)$, $(x_3, 1)$, and $(x_4, 0)$ have probability $\frac{1}{4}$; all other $(v, \lambda) \in U(\mathfrak{A}_1) \times \{0, 1\} = U(\mathfrak{A}_2) \times \{0, 1\}$ have probability 0. By the choice of $\mathcal{D}$, if a hypothesis misclassifies at least one of the x_i, it has a generalisation error of at least $\frac{1}{4}$.

Assume that $\mathcal{L}$ is an algorithm that solves FO-LEARN-PAC(σ, k, ℓ, q^*) in sublinear time. As we argued in the proof of Theorem 3.5, $\mathcal{L}$ is unable to distinguish $\mathfrak{A}_1$ and $\mathfrak{A}_2$ from each other (by choosing a suitable ordering on the vertices). Furthermore, we argued that such an algorithm would also be unable to distinguish the first and third column as well as the second and fourth column of the background structures. In the proof of Theorem 3.5, we chose an ordering on the vertices such that the parameter returned by the algorithm is x_1, x_2, y_1, y_2, or some vertex from the first or second column. Here, the vertex returned by $\mathcal{L}$ may depend on the training sequence drawn from $\mathcal{D}$. However, by choosing a sufficient ordering on the vertices, we can still make sure that the returned parameter is among the mentioned ones (i. e. among x_1, x_2, y_1, y_2, or some vertex from the first or second column) with probability at least $\frac{1}{2}$ over the choice of examples drawn from $\mathcal{D}$.

Now, we only consider those cases where the parameter is among the mentioned ones. For every fixed choice of examples, analogously to the proof of the consistent-learning case, the algorithm $\mathcal{L}$ has to return the same hypothesis on both background structures. Thus, the hypothesis returned by the algorithm has to misclassify at least one of the x_i on at least one of the two background structures. Hence, on one of the two background structures, it makes at least one error in at least half of the cases where the parameter is among the mentioned ones, so with (conditional) probability at least $\frac{1}{2}$.

Overall, including the probability that the chosen parameter is among the mentioned vertices, on at least one of the two background structures, $\mathcal{L}$ has to make at least one error on the x_is with probability at least $\frac{1}{2} \cdot \frac{1}{2} = \frac{1}{4}$. Combined with our observation above, this means that, on one of two background structures, the algorithm has a generalisation error of at least $\frac{1}{4}$ with probability at least $\frac{1}{4}$ over the choice of examples drawn from $\mathcal{D}$. We choose $\varepsilon = \delta = \frac{1}{8}$. Then $\mathcal{L}$ does not meet the requirements of FO-Learn-PAC, which contradicts our assumption.

All in all, this shows that there is no algorithm that solves the problem FO-Learn-PAC(σ, k, ℓ, q^*) in sublinear time. □

3.5 RELATED WORK

As described in the beginning of this chapter, the learning framework we consider in this thesis has been introduced by Grohe and Turán in [57]. There, the authors proved information-theoretic learnability results for concepts that can be described using first-order and monadic second-order logic on restricted classes of background structures such as the class of planar graphs and classes of graphs of bounded degree. Algorithmic aspects of the framework, including the running time of a learning algorithm, have been first studied by Grohe and Ritzert in [55]. They showed, as we have seen in Sections 3.3 and 3.4, that first-order definable concepts are both consistent- and PAC-learnable in sublinear time, measured in the size of the background structure, over structures of at most polylogarithmic degree. In [53], Grohe, Löding, and Ritzert examined the learnability of concepts definable in first-order and monadic second-order logic over simple structures of unbounded degree, namely ordered strings. Even in the unary case, i. e. for $X = V(\mathfrak{A})$, they were able to show that there is no consistent-learning algorithm for first-order definable concepts running in sublinear time. However, by introducing a linear-time pre-processing phase to build an index for the background structure, they were able to show that concepts definable in monadic second-order logic can be learned in sublinear learning time. The results have been extended by Grienenberger and Ritzert [47] from strings to trees.

Closely related to the framework we consider is the framework of *inductive logic programming (ILP)* [25, 28, 66, 76, 77]. In both frameworks, we are in a passive supervised learning setting where the learning algorithms are given labelled examples. These examples are labelled according to some target concept and the algorithms should return a hypothesis that approximately matches this target concept. One of the main differences between both frameworks is that we encode the background knowledge in a relational structure, whereas in ILP, it is represented in a background theory, i. e. a set of formulas. PAC-learning results for ILP have often been obtained by syntactically restricting the hypothesis classes (see, e. g., [25, 66]), while we use restricted classes of background structures such as classes of small degree or nowhere dense classes.

In the database literature, various approaches to learning queries from examples have been studied, both in passive (such as ours) and active learning settings. In passive learning settings, results often focus on conjunctive queries [9, 10, 61, 63, 67] or consider queries outside the relational database model [15, 87], while we focus on (extensions of) full first-order logic. In the *active learning* setting, as introduced by Angluin in [7], learning algorithms are allowed to actively query an oracle. This includes membership queries that allow the learning algorithm to actively choose examples and obtain their corresponding labels. Results in this setting [1, 5, 15, 86] again consider different types of queries including, quite recently, conjunctive queries [20]. Another related subject in the database literature is the problem of learning schema mappings from examples [6, 17, 21, 22, 46].

In formal verification, related logical learning frameworks [23, 34, 44, 73, 94] have been studied as well.

4 LEARNING LOGICS WITH COUNTING

In this chapter, we generalise the results of Grohe and Ritzert for first-order logic to the extension $\mathsf{FOCN}(\mathbb{P})$ of first-order logic with counting quantifiers. Since there is no analogue of Gaifman's Theorem for $\mathsf{FOCN}(\mathbb{P})$, the proofs in this chapter are based on Hanf's Theorem, which we discuss in Section 4.1.

Based on the definition of a Hanf normal form, in Section 4.2, we introduce the problems for consistent learning as well as for PAC learning that we study in this chapter.

In Section 4.3, we prove that concepts definable in $\mathsf{FOCN}(\mathbb{P})$ can be learned in sublinear time on classes of structures of bounded degree. As in the previous chapter, we start with an algorithm for the consistent-learning problem. Then, we modify the algorithm to follow the Empirical Risk Minimisation rule and explain why this yields a PAC-learning algorithm.

In Section 4.4, we consider classes of structures where the degree is not bounded by a fixed number, but the degree of the structures is still small, that is, polylogarithmic in their size. Due to the differences between Gaifman and Hanf locality, in contrast to the learning results for first-order logic from Chapter 3, the results for bounded-degree structures cannot be easily extended to structures of unbounded but small degree. By modifying the approach, we can still provide an algorithm for the consistent-learning problem as well as an algorithm that follows the Empirical Risk Minimisation rule. Since our approach is based on Hanf's Theorem, we have to deal with isomorphism types of local neighbourhoods in our structures. To handle these within the desired time bounds, we apply a recent isomorphism test running in time $n^{\mathrm{polylog}(d)}$ for n-vertex graphs of maximum degree d [54]. By further restricting the degree of the considered structures in terms of their size, we also obtain a PAC-learning result at the end of the section.

Except for the last PAC-learning result, all results in this chapter (if not stated otherwise) have been published in [11].

4.1 HANF LOCALITY

For the learnability results for first-order logic with counting, we rely on normal forms based on Hanf's locality theorem for first-order logic [60]. This theorem implies that, to determine whether a finite structure satisfies a first-order sentence of quantifier rank at most q, it suffices to determine the number of realisations of neighbourhoods up to a certain radius within the structure. The version of the theorem provided by Fagin, Stockmeyer, and Vardi [36] implies that on structures of degree at most d, it even suffices to determine the number of these realisations up to a certain threshold. Since, in structures of degree at most d, there are only finitely many types of neighbourhoods of radius at most r, this condition can be expressed as a first-order sentence in so-called *Hanf normal form*.

In this thesis, we use the Hanf normal form for $\mathsf{FOCN}(\mathbb{P})$ provided by Kuske and Schweikardt [69]. Before stating the exact result, we first introduce the basic building blocks.

Sphere formula

Let $r \in \mathbb{N}$, $k \in \mathbb{N}_{\geqslant 1}$, let $\mathfrak{A}$ be a relational structure, and let $\bar{v} = (v_1, \ldots, v_k) \in \big(U(\mathfrak{A})\big)^k$. A *sphere formula with* k *centres of locality radius* r is a first-order formula $\mathrm{sph}^{\mathfrak{A}}_{r,\bar{v}}(x_1, \ldots, x_k)$ such that for every structure $\mathfrak{A}'$ and every tuple $\bar{v}' = (v'_1, \ldots, v'_k) \in \big(U(\mathfrak{A}')\big)^k$, it holds that $\mathfrak{A}' \models \mathrm{sph}^{\mathfrak{A}}_{r,\bar{v}}[\bar{v}']$ if and only if there is an isomorphism between the two neighbourhoods that maps the centres upon each other, i. e., there is an isomorphism π between $\mathcal{N}^{\mathfrak{A}}_r(\bar{v})$ and $\mathcal{N}^{\mathfrak{A}'}_r(\bar{v}')$ with $\pi(v_i) = v'_i$ for all $i \in [k]$, or, equivalently, there is an isomorphism between $\mathcal{S}^{\mathfrak{A}}_r(\bar{v})$ and $\mathcal{S}^{\mathfrak{A}'}_r(\bar{v}')$. For a fixed signature σ, given a tuple $\bar{v}$, a radius r, and local access to a σ-structure $\mathfrak{A}$, the time needed to construct the sphere formula $\mathrm{sph}^{\mathfrak{A}}_{r,\bar{v}}(x_1, \ldots, x_k)$ is polynomial in the size of the r-neighbourhood of $\bar{v}$ [69]. Note that sphere formulas of locality radius at most r are r-local.

A *basic counting term* is a counting term of the form $\#(x).\varphi(x)$ in $\mathsf{FOCN}(\mathbb{P})$, where x is a structure variable in vars and φ is a sphere formula with a single centre. The *locality radius* of the basic counting term is the locality radius of the sphere formula.

A *numerical condition on occurrences of types with one centre* (or *numerical oc-type condition*) is an $\mathsf{FOCN}(\mathbb{P})$-formula that is built from basic counting terms and rules (2) and (5)–(9) from Definitions 2.3 to 2.5, i. e., using number variables and integers, and combining them by addition, multiplication, numerical predicates from $\mathbb{P} \cup \{\mathsf{P}_\exists\}$, Boolean combinations, and quantification of number variables. Its locality radius is the maximal locality radius of the involved basic counting terms. Note that numerical oc-type conditions do not have any free structure variables.

Hanf normal form, hnf-formula

A formula is in *Hanf normal form for* $\mathsf{FOCN}(\mathbb{P})$ or an *hnf-formula for* $\mathsf{FOCN}(\mathbb{P})$ if it is a Boolean combination of numerical oc-type conditions and sphere formulas. The locality radius of an hnf-formula is the maximal locality radius of the involved conditions and formulas.

The following result is due to Kuske and Schweikardt [69].

Theorem 4.1 ([69]). *Let* $(\mathbb{P}, \mathrm{ar}, [\![.]\!])$ *be a numerical predicate collection. For any relational signature* σ*, any degree bound* $d \in \mathbb{N}$*, and any* $\mathsf{FOCN}(\mathbb{P})[\sigma]$*-formula* φ*, there exists a* d*-equivalent hnf-formula* ψ *for* $\mathsf{FOCN}(\mathbb{P})[\sigma]$ *of locality radius smaller than* $\big(2 \cdot \mathrm{bw}(\varphi) + 1\big)^{\mathrm{br}(\varphi)}$ *with* $\mathrm{free}(\psi) = \mathrm{free}(\varphi)$.

Next, analogously to the local types we introduced in Section 2.4, we introduce local hnf-formulas, and we also provide similar locality results for them.

Local hnf-formulas

Let $\mathfrak{A}$ be a relational structure, $k \in \mathbb{N}_{\geqslant 1}$, $r \in \mathbb{N}$, and $\bar{v} \in \big(U(\mathfrak{A})\big)^k$. Then the *local hnf-formulas for* $\mathsf{FOCN}(\mathbb{P})$ *of* $\bar{v}$ *with locality radius at most* r *in* $\mathfrak{A}$ are

$$\begin{aligned}\mathrm{lhf}_r^{\mathfrak{A}}(\bar{v}) \coloneqq \big\{\varphi(\bar{x}) \text{ hnf-formula } \mid\ & \mathfrak{A} \models \varphi[\bar{v}],\\ & \text{locality radius of } \varphi \text{ is at most } r\big\}.\end{aligned}$$

We use Kuske's and Schweikardt's result to show that $\mathsf{FOCN}(\mathbb{P})$-formulas are unable to distinguish tuples that have the same local hnf-formulas (of a certain locality radius).

Lemma 4.2. *Let* $\mathfrak{A}$ *be a relational structure, let* $\bar{x} = (x_1, \dots, x_k)$ *be a tuple of structure variables, let* $\bar{\kappa} = (\kappa_1, \dots, \kappa_\ell)$ *be a tuple of number variables, and let* $\varphi(\bar{x}, \bar{\kappa})$ *be an* $\mathsf{FOCN}(\mathbb{P})$*-formula. Then, for all* $\bar{v}, \bar{v}' \in \big(U(\mathfrak{A})\big)^k$ *and* $\bar{n} = (n_1, \dots, n_\ell) \in [0, |\mathfrak{A}|]^\ell$*, if*

$$\mathrm{lhf}_r^{\mathfrak{A}}(\bar{v}) = \mathrm{lhf}_r^{\mathfrak{A}}(\bar{v}') \qquad \textit{for} \quad r = (2 \cdot \mathrm{bw}(\varphi) + 1)^{\mathrm{br}(\varphi)},$$

then

$$\mathfrak{A} \models \varphi[\bar{v}, \bar{n}] \iff \mathfrak{A} \models \varphi[\bar{v}', \bar{n}].$$

Proof. Let $\varphi'(\bar{x}) \coloneqq \varphi(\bar{x}, \bar{n})$, i. e. we replace every occurrence of the number variable κ_i in φ with the integer n_i for all i. Note that $\mathrm{br}(\varphi) = \mathrm{br}(\varphi')$ and $\mathrm{bw}(\varphi) = \mathrm{bw}(\varphi')$. Using Theorem 4.1, we obtain an hnf-formula $\psi(\bar{x})$ of locality radius smaller than $r = \big(2 \cdot \mathrm{bw}(\varphi) + 1\big)^{\mathrm{br}(\varphi)}$ that is $\deg(\mathfrak{A})$-equivalent to φ'. Let $\bar{v}$ and $\bar{v}'$ be k-tuples from $\mathfrak{A}$ with $\mathrm{lhf}_r^{\mathfrak{A}}(\bar{v}) = \mathrm{lhf}_r^{\mathfrak{A}}(\bar{v}')$. We show $\mathfrak{A} \models \varphi[\bar{v}, \bar{n}] \implies \mathfrak{A} \models \varphi[\bar{v}', \bar{n}]$, then the other direction follows by symmetry.

Assume that $\mathfrak{A} \models \varphi[\bar{v}, \bar{n}]$ holds. This implies that $\mathfrak{A} \models \varphi'[\bar{v}]$ and $\mathfrak{A} \models \psi[\bar{v}]$ hold as well. Thus, since ψ is an hnf-formula of locality radius smaller than r, we have $\psi \in \mathrm{lhf}_r^{\mathfrak{A}}(\bar{v}) = \mathrm{lhf}_r^{\mathfrak{A}}(\bar{v}')$, which implies that $\mathfrak{A} \models \psi[\bar{v}']$. By the $\deg(\mathfrak{A})$-equivalence between ψ and φ', this shows that $\mathfrak{A} \models \varphi'[\bar{v}']$, which finally implies that $\mathfrak{A} \models \varphi[\bar{v}', \bar{n}]$. □

The following results help us to reduce the formula and parameter spaces we have to consider to find consistent hypotheses. The first lemma states that two tuples satisfy the same local hnf-formulas if and only if their spheres are isomorphic.

Lemma 4.3. *Let* $\mathfrak{A}$ *be a relational structure,* $k \in \mathbb{N}_{\geqslant 1}$*,* $r \in \mathbb{N}$*, and* $\bar{v}, \bar{v}' \in \big(U(\mathfrak{A})\big)^k$*. Then,* $\mathrm{lhf}_r^{\mathfrak{A}}(\bar{v}) = \mathrm{lhf}_r^{\mathfrak{A}}(\bar{v}')$ *if and only if* $\mathcal{S}_r^{\mathfrak{A}}(\bar{v}) \cong \mathcal{S}_r^{\mathfrak{A}}(\bar{v}')$.

Proof. For the forward direction, assume $\mathrm{lhf}_r^{\mathfrak{A}}(\bar{v}) = \mathrm{lhf}_r^{\mathfrak{A}}(\bar{v}')$. We have $\mathrm{sph}_{r,\bar{v}}^{\mathfrak{A}} \in \mathrm{lhf}_r^{\mathfrak{A}}(\bar{v})$ and hence, $\mathrm{sph}_{r,\bar{v}}^{\mathfrak{A}} \in \mathrm{lhf}_r^{\mathfrak{A}}(\bar{v}')$. Thus, $\mathfrak{A} \models \mathrm{sph}_{r,\bar{v}}^{\mathfrak{A}}[\bar{v}']$, which is equivalent to $\mathcal{S}_r^{\mathfrak{A}}(\bar{v})$ and $\mathcal{S}_r^{\mathfrak{A}}(\bar{v}')$ being isomorphic.

For the backward direction, assume the spheres $\mathcal{S}_r^{\mathfrak{A}}(\bar{v})$ and $\mathcal{S}_r^{\mathfrak{A}}(\bar{v}')$ are isomorphic. Let $\bar{x} = (x_1, \dots, x_k)$ and let $\varphi(\bar{x})$ be an hnf-formula of locality radius at most r. Then, φ is a Boolean combination of numerical oc-type conditions and sphere formulas with locality radius at most r. We show that $\mathfrak{A} \models \varphi[\bar{v}]$ if and only if $\mathfrak{A} \models \varphi[\bar{v}']$.

The numerical oc-type conditions in φ do not have any free structure variables. Hence, their evaluation only depends on the structure and is independent of the assignment.

The free variables of the sphere formulas used in φ are a subset of $\text{free}(\varphi)$. Let $\text{sph}_{r',\bar{w}}^{\mathfrak{A}'}(x_{i_1}, \dots, x_{i_\ell})$ be such a sphere formula used in φ for some relational structure $\mathfrak{A}'$, an ℓ-tuple $\bar{w}$ from $\mathfrak{A}'$, and some locality radius $r' \leqslant r$. It follows from our assumption that $\mathcal{S}_{r'}^{\mathfrak{A}}(v_{i_1}, \dots, v_{i_\ell}) \cong \mathcal{S}_{r'}^{\mathfrak{A}}(v'_{i_1}, \dots, v'_{i_\ell})$. Thus,

$$\begin{aligned} & \mathfrak{A} \models \text{sph}_{r',\bar{w}}^{\mathfrak{A}}[v_{i_1}, \dots, v_{i_\ell}] \\ \iff & \mathcal{S}_{r'}^{\mathfrak{A}}(v_{i_1}, \dots, v_{i_\ell}) \cong \mathcal{S}_{r'}^{\mathfrak{A}'}(w_1, \dots, w_\ell) \\ \iff & \mathcal{S}_{r'}^{\mathfrak{A}}(v'_{i_1}, \dots, v'_{i_\ell}) \cong \mathcal{S}_{r'}^{\mathfrak{A}'}(w_1, \dots, w_\ell) \\ \iff & \mathfrak{A} \models \text{sph}_{r',\bar{w}}^{\mathfrak{A}}[v'_{i_1}, \dots, v'_{i_\ell}]. \end{aligned}$$

This holds for all sphere formulas in φ. Thus, we have $\mathfrak{A} \models \varphi[\bar{v}]$ if and only if $\mathfrak{A} \models \varphi[\bar{v}']$. $\square$

The following result is a variant of the Local Composition Lemma for first-order logic from [55], translated to first-order logic with counting and local hnf-formulas. It allows us to analyse the parameters we choose by splitting them into two parts with disjoint neighbourhoods.

Lemma 4.4 (Local Composition Lemma for $\mathsf{FOCN}(\mathbb{P})$)**.** *Let $\mathfrak{A}$ be a relational structure, $k, \ell, r \in \mathbb{N}$, $\bar{v}, \bar{v}' \in (U(\mathfrak{A}))^k$, and $\bar{w}, \bar{w}' \in (U(\mathfrak{A}))^\ell$, such that $\text{dist}(\bar{v}, \bar{w}) > 2r+1$, $\text{dist}(\bar{v}', \bar{w}') > 2r+1$, $\text{lhf}_r^{\mathfrak{A}}(\bar{v}) = \text{lhf}_r^{\mathfrak{A}}(\bar{v}')$, and $\text{lhf}_r^{\mathfrak{A}}(\bar{w}) = \text{lhf}_r^{\mathfrak{A}}(\bar{w}')$. Then, $\text{lhf}_r^{\mathfrak{A}}(\bar{v}\bar{w}) = \text{lhf}_r^{\mathfrak{A}}(\bar{v}'\bar{w}')$.*

Proof. From $\text{lhf}_r^{\mathfrak{A}}(\bar{v}) = \text{lhf}_r^{\mathfrak{A}}(\bar{v}')$, using Lemma 4.3, it follows that $\mathcal{S}_r^{\mathfrak{A}}(\bar{v})$ and $\mathcal{S}_r^{\mathfrak{A}}(\bar{v}')$ are isomorphic. Similarly, we obtain that $\mathcal{S}_r^{\mathfrak{A}}(\bar{w})$ and $\mathcal{S}_r^{\mathfrak{A}}(\bar{w}')$ are isomorphic from $\text{lhf}_r^{\mathfrak{A}}(\bar{w}) = \text{lhf}_r^{\mathfrak{A}}(\bar{w}')$. Because of the lower bounds for the distances, we have $\mathcal{N}_r^{\mathfrak{A}}(\bar{v}) \cup \mathcal{N}_r^{\mathfrak{A}}(\bar{w}) = \mathcal{N}_r^{\mathfrak{A}}(\bar{v}\bar{w})$ and $\mathcal{N}_r^{\mathfrak{A}}(\bar{v}') \cup \mathcal{N}_r^{\mathfrak{A}}(\bar{w}') = \mathcal{N}_r^{\mathfrak{A}}(\bar{v}'\bar{w}')$. Hence, by combining the above-mentioned isomorphisms, we can deduce that $\mathcal{S}_r^{\mathfrak{A}}(\bar{v}\bar{w}) \cong \mathcal{S}_r^{\mathfrak{A}}(\bar{v}'\bar{w}')$. With Lemma 4.3, it follows that $\text{lhf}_r^{\mathfrak{A}}(\bar{v}\bar{w}) = \text{lhf}_r^{\mathfrak{A}}(\bar{v}'\bar{w}')$. $\square$

4.2 LEARNING PROBLEMS FOR FOCN

With the definition of the Hanf normal form at hand, we can now introduce the learning problems that we consider in this chapter.

Recall the problem LEARN-CONSISTENT(k, Φ^*, Φ), where target concepts only use formulas from Φ^*, and algorithms are only allowed to return hypotheses using formulas from Φ. Similarly as for the problems for first-order logic in Chapter 3, we want to solve a variation of the problem Learn-Consistent(k, Φ^*, Φ) for certain sets of formulas Φ^* and Φ. For first-order logic, we used sets of bounded quantifier rank. In this chapter, for the logic $\mathsf{FOCN}(\mathbb{P})$, we bound the binding rank and the binding width of the formulas by constants c_r and c_w. For $c_r, c_w \in \mathbb{N}$, let $\mathsf{FOCN}(\mathbb{P})[\sigma, c_r, c_w]$ denote the set of all formulas in $\mathsf{FOCN}(\mathbb{P})[\sigma]$ of binding rank at most c_r and binding width at most c_w. Since formulas from $\mathsf{FOCN}(\mathbb{P})$ may have free number variables, we allow concepts to use number parameters in addition to the parameters from the structure.

Let $k, \ell, c_r, c_w \in \mathbb{N}$ and fix a signature σ. We consider concepts that can be defined using a formula from

$$\Phi^* = \big\{\varphi(\bar{x}, \bar{y}, \bar{\kappa}) \in \mathsf{FOCN}(\mathbb{P})[\sigma, c_r, c_w] \;\mid\; |\bar{x}| = k,\; |\bar{y}| = \ell\big\},$$

combined with parameters that are elements from the structure as well as number parameters. For a σ-structure $\mathfrak{A}$, a formula $\varphi(\bar{x}, \bar{y}, \bar{\kappa}) \in \Phi^*$, and tuples $\bar{w} \in \big(U(\mathfrak{A})\big)^\ell$ and $\bar{n} \in [0, |\mathfrak{A}|]^{|\bar{\kappa}|}$, the resulting hypothesis is the mapping $h^{\mathfrak{A}}_{\varphi, \bar{w}, \bar{n}}(\bar{x})\colon \big(U(\mathfrak{A})\big)^k \to \{0, 1\}$ which maps a tuple $\bar{v} \in \big(U(\mathfrak{A})\big)^k$ to $[\![\varphi(\bar{v}, \bar{w}, \bar{n})]\!]^{\mathfrak{A}}$.

As it turns out, to describe these concepts on a fixed structure, it actually suffices to use a Boolean combination of sphere formulas up to a certain locality radius without any number variables or number parameters. Hence, the formulas that our algorithms return come from the set

$$\begin{aligned}\Phi = \big\{\varphi(\bar{x}, \bar{y}) \in \mathsf{FO}[\sigma] \;\mid\; & |\bar{x}| = k,\; |\bar{y}| = \ell,\\ & \varphi \text{ is a Boolean combination of sphere formulas}\\ & \text{of locality radius at most } (2 \cdot c_w + 1)^{c_r}\big\}.\end{aligned}$$

As we will see, with different techniques in Sections 4.3 and 4.4, the restriction of the locality radius of the returned formula still allows us to evaluate the hypothesis on new tuples efficiently.

The consistent-learning problem for $\mathsf{FOCN}(\mathbb{P})$ is formally defined as follows.

FOCN($\mathbb{P}$)-Learn-Consistent$(\sigma, k, \ell, c_r, c_w)$

Input: σ-structure $\mathfrak{A}$, training sequence $T \in \Big(\big(U(\mathfrak{A})\big)^k \times \{0, 1\}\Big)^m$

Problem: Return a first-order formula φ and a tuple $\bar{w} \in \big(U(\mathfrak{A})\big)^\ell$, where φ is a Boolean combination of sphere formulas of locality radius at most $(2 \cdot c_w + 1)^{c_r}$, such that the hypothesis $h^{\mathfrak{A}}_{\varphi, \bar{w}}$ is consistent with T.
The algorithm may reject if there is no combination of a formula $\varphi^*(\bar{x}, \bar{y}, \bar{\kappa}) \in \mathsf{FOCN}(\mathbb{P})[\sigma, c_r, c_w]$ and tuples $\bar{w}^* \in \big(U(\mathfrak{A})\big)^\ell$, $\bar{n}^* \in [0, |\mathfrak{A}|]^{|\bar{\kappa}|}$ such that the hypothesis $h^{\mathfrak{A}}_{\varphi^*, \bar{w}^*, \bar{n}^*}$ is consistent with T.

In Section 4.3, we show that this problem is solvable in sublinear time on classes of structures of bounded degree. In Section 4.4, with a different approach, we extend this result to classes of structures of polylogarithmic degree.

The ERM- and PAC-learning problems for $\mathsf{FOCN}(\mathbb{P})$ that we study in this chapter are defined as follows.

FOCN(P)-LEARN-ERM$(\sigma, k, \ell, c_r, c_w)$

Input: structure $\mathfrak{A}$, training sequence $T \in \Big(\big(U(\mathfrak{A})\big)^k \times \{0,1\}\Big)^m$

Problem: Return a first-order formula φ and a tuple $\bar{w} \in \big(U(\mathfrak{A})\big)^\ell$, where φ is a Boolean combination of sphere formulas of locality radius at most $(2 \cdot c_w + 1)^{c_r}$, such that

$$\begin{aligned} \mathrm{err}_T\big(h^{\mathfrak{A}}_{\varphi,\bar{w}}\big) \leqslant \min\big\{ & \mathrm{err}_T(h^{\mathfrak{A}}_{\varphi^*,\bar{w}^*,\bar{n}^*}) \ \big| \\ & \varphi^*(\bar{x},\bar{y},\bar{\kappa}) \in \mathsf{FOCN}(\mathbb{P})[\sigma, c_r, c_w], \\ & \bar{w}^* \in (U(\mathfrak{A}))^\ell,\ \bar{n}^* \in [0, |\mathfrak{A}|]^{|\bar{\kappa}|}\big\}. \end{aligned}$$

FOCN(P)-LEARN-PAC$(\sigma, k, \ell, c_r, c_w)$

Input: structure $\mathfrak{A}$, rational numbers $\varepsilon, \delta > 0$, probability distribution $\mathcal{D}$ on $\big(U(\mathfrak{A})\big)^k \times \{0,1\}$

Problem: Return a first-order formula φ and a tuple $\bar{w} \in \big(U(\mathfrak{A})\big)^\ell$, where φ is a Boolean combination of sphere formulas of locality radius at most $(2 \cdot c_w + 1)^{c_r}$, such that, with probability of at least $1 - \delta$ over the choice of examples drawn i.i.d. from $\mathcal{D}$, it holds that

$$\mathrm{err}_{\mathcal{D}}\big(h^{\mathfrak{A}}_{\varphi,\bar{w}}\big) \leqslant \varepsilon^* + \varepsilon,$$

where

$$\begin{aligned} \varepsilon^* := \min\big\{ & \mathrm{err}_{\mathcal{D}}(h^{\mathfrak{A}}_{\varphi^*,\bar{w}^*,\bar{n}^*}) \ \big| \\ & \varphi^*(\bar{x},\bar{y},\bar{\kappa}) \in \mathsf{FOCN}(\mathbb{P})[\sigma, c_r, c_w], \\ & \bar{w}^* \in (U(\mathfrak{A}))^\ell,\ \bar{n}^* \in [0, |\mathfrak{A}|]^{|\bar{\kappa}|}\big\}. \end{aligned}$$

In Section 4.3, we modify the algorithm we give for the consistent-learning problem to show that FOCN(P)-LEARN-ERM is solvable in sublinear time on classes of structures of bounded degree. Afterwards, we use this result to show that also PAC learning is possible in sublinear time on these classes of structures. In Section 4.4, we extend the consistent-learning result on classes of structures of polylogarithmic degree to ERM learning. Furthermore, we provide a PAC-learning algorithm that runs in sublinear time on classes of structures with a stricter (but still not constant) degree bound.

4.3 STRUCTURES OF BOUNDED DEGREE

In this section, we present learning results for $\mathsf{FOCN}(\mathbb{P})$ on classes of structures of bounded degree. We start with the consistent-learning problem.

Consistent learning

Theorem 4.5. *Let σ be a relational signature, let $k, \ell, c_r, c_w \in \mathbb{N}$, and let $\mathcal{C}$ be a class of structures of degree at most d for some $d \in \mathbb{N}$. There is an algorithm that solves* $\mathsf{FOCN}(\mathbb{P})$-LEARN-CONSISTENT$(\sigma, k, \ell, c_r, c_w)$ *on $\mathcal{C}$ in time $(\log n + m)^{\mathcal{O}(1)}$ under the logarithmic-cost measure and in time $m^{\mathcal{O}(1)}$ under the uniform-cost measure, where n is the size of the background structure and m is the length of the training sequence.*

Furthermore, the hypotheses returned by the algorithm can be evaluated in time $(\log n)^{\mathcal{O}(1)}$ under the logarithmic-cost measure and in constant time under the uniform-cost measure.

The high-level proof idea is very similar to the one Grohe and Ritzert [55] presented for the consistent-learning problem for first-order logic. We use a brute-force algorithm that checks all combinations of certain choices of formulas and certain choices of parameters.

As we show in Lemma 4.7, for fixed σ, d, k, ℓ, c_r, and c_w, the number of formulas we need to check is constant.

To bound the number of parameters to check, we show that it suffices to consider only parameters in a certain neighbourhood around the training examples. As shown in Figure 4.1, intuitively, this holds because parameters that are far away from the training examples do not help to distinguish positive from negative examples. The formal result is given in Lemma 4.6.

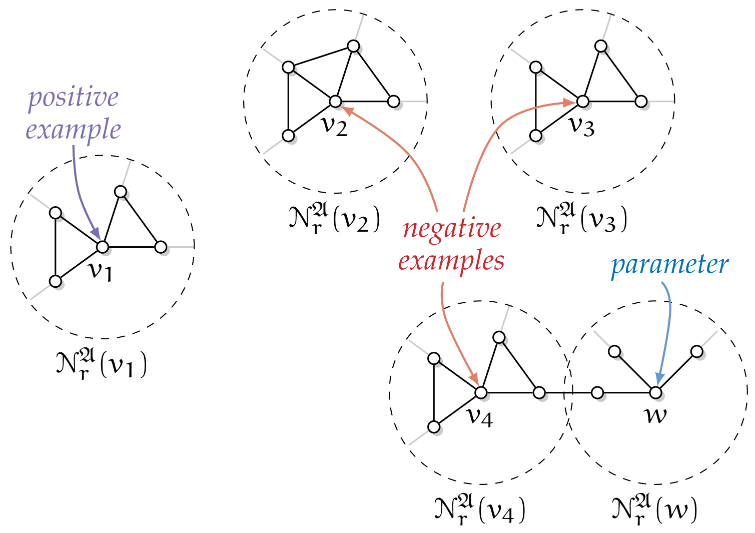
For the rest of this section, let σ be a fixed relational signature, $d, k, \ell, c_r, c_w \in \mathbb{N}$, let $r := (2 \cdot c_w + 1)^{c_r}$, let $\mathcal{C}$ be a class of structures of degree at most d, and let $\mathfrak{A}$ be a structure from $\mathcal{C}$. Let Φ^*, i. e. the set of formulas that our target concepts are based upon, be defined as in the last section, that is,

$$\Phi^* = \{\varphi(\bar{x}, \bar{y}, \bar{\kappa}) \in \mathsf{FOCN}(\mathbb{P})[\sigma, c_r, c_w] \mid |\bar{x}| = k,\ |\bar{y}| = \ell\}.$$

For Φ, that is, the set of formulas our algorithms are allowed to return in a hypothesis, we can even use a restriction of the set from the last section and set

$$\begin{aligned}\Phi_d := \{\varphi(\bar{x}, \bar{y}) \in \mathsf{FO}[\sigma] \mid\ & |\bar{x}| = k,\ |\bar{y}| = \ell,\\ & \varphi \text{ is a Boolean combination of sphere formulas}\\ & \text{of locality radius at most } r\\ & \text{based on spheres of degree at most } d\}.\end{aligned}$$

For a training sequence $T = \big((\bar{v}_1, \lambda_1), \dots, (\bar{v}_m, \lambda_m)\big)$ and a radius $r' \in \mathbb{N}$, let $N_{r'}^{\mathfrak{A}}(T) := \bigcup_{i \in [m]} N_{r'}^{\mathfrak{A}}(\bar{v}_i)$.



positive example
v_1
$\mathcal{N}_r^{\mathfrak{A}}(v_1)$
v_2
$\mathcal{N}_r^{\mathfrak{A}}(v_2)$
negative examples
v_3
$\mathcal{N}_r^{\mathfrak{A}}(v_3)$
parameter
v_4
w
$\mathcal{N}_r^{\mathfrak{A}}(v_4)$
$\mathcal{N}_r^{\mathfrak{A}}(w)$

Figure 4.1: One positive and three negative examples from a training sequence as well as a parameter with their local neighbourhoods. The vertices v_1 and v_2 can easily be distinguished by a formula since they have different local types. The vertices v_1 and v_3 have the same local types and even if we take the parameter w into consideration, the local types of the tuples (v_1, w) and (v_3, w) are still the same since the parameter is too far away from both vertices v_1 and v_3. Thus, there is no way to distinguish v_1 and v_3 using w and a formula with locality radius at most r. The only way to distinguish vertices of the same local type is to have a parameter close to one of the vertices, as shown for v_4. This argumentation is formalised in the proof of Lemma 4.6.

Lemma 4.6. *Let* $T \in \Big(\big(U(\mathfrak{A})\big)^k \times \{0,1\}\Big)^m$ *be a training sequence and let* $\varphi^* \in \Phi^*$, $\bar{w}^* \in \big(U(\mathfrak{A})\big)^\ell$, *and* $\bar{n}^* \in [0, |\mathfrak{A}|]^{|\bar{\kappa}|}$ *be such that the hypothesis* $h^{\mathfrak{A}}_{\varphi^*,\bar{w}^*,\bar{n}^*}$ *is consistent with* T. *Then, there is a formula* $\varphi \in \Phi_d$ *and a tuple* $\bar{w} \in \big(N^{\mathfrak{A}}_{(2r+1)\ell}(T)\big)^\ell$ *such that the hypothesis* $h^{\mathfrak{A}}_{\varphi,\bar{w}}$ *is consistent with* T.

Proof. Let $T = \big((\bar{v}_1, \lambda_1), \dots, (\bar{v}_m, \lambda_m)\big)$, φ^*, $\bar{w}^* = (w_1^*, \dots, w_\ell^*)$, and $\bar{n}^*$ be as given in the lemma. We iteratively select vertices $w^{(i)}$ from the parameters $w_1^*, \dots, w_\ell^*$ that have distance at most $2r+1$ from the examples or the already selected vertices. This process is repeated for s steps until all remaining parameters are too far away (or all parameters have already been selected). For the tuple $\bar{w}$ that we are looking for in this proof, we use these selected parameters and omit the others.

Formally, to select the parameters, we start with the neighbourhood $N^{(0)} := N^{\mathfrak{A}}_{2r+1}(T)$ of radius $2r+1$ around the examples and select a vertex $w \in \{w_1^*, \dots, w_\ell^*\} \cap N^{(0)}$. If there is no such vertex, we set $s := 0$ and stop this process. Otherwise, we set $w^{(1)} := w$, $N^{(1)} := N^{(0)} \cup N^{\mathfrak{A}}_{2r+1}(w)$, and continue. For $i \geqslant 2$, we select a vertex $w \in \{w_1^*, \dots, w_\ell^*\} \setminus \{w^{(1)}, \dots, w^{(i-1)}\}$ that is contained in the neighbourhood $N^{(i-1)}$. If there is no such vertex, we set $s := i-1$ and stop. Otherwise, we set $w^{(i)} := w$, $N^{(i)} := N^{(i-1)} \cup N^{\mathfrak{A}}_{2r+1}(w)$, and continue. W.l.o.g. let $w^{(i)} = w^*_i$ for $i \in [s]$. Let $\bar{w}^{\text{in}} := (w^*_1, \dots, w^*_s)$ and $\bar{w}^{\text{out}} := (w^*_{s+1}, \dots, w^*_\ell)$. We let $\bar{y}^{\text{in}} := (y_1, \dots, y_s)$ and choose

$$\varphi(\bar{x}, \bar{y}) := \bigvee_{i \in [m],\ \lambda_i = 1} \text{sph}^{\mathfrak{A}}_{r, \bar{v}_i \bar{w}^{\text{in}}}(\bar{x}, \bar{y}^{\text{in}}).$$

The formula φ is a Boolean combination of sphere formulas of locality radius at most r based on spheres of degree at most d and thus, $\varphi \in \Phi_d$. We turn $\bar{w}^{\text{in}} = (\bar{w}^*_1, \dots, \bar{w}^*_s)$ into a tuple $\bar{w} \in \left(N^{\mathfrak{A}}_{(2r+1)\ell}(T)\right)^\ell$ by choosing an arbitrary $w \in N^{\mathfrak{A}}_{(2r+1)\ell}(T)$ and filling the missing $(\ell - s)$ positions with the vertex w.

It remains to show that the hypothesis $h^{\mathfrak{A}}_{\varphi, \bar{w}}$ is consistent with T. If $\lambda_i = 1$, then by the construction of $h^{\mathfrak{A}}_{\varphi, \bar{w}}$ (especially the construction of φ), it holds that $h^{\mathfrak{A}}_{\varphi, \bar{w}}(\bar{v}_i) = 1$. For the other direction, we use the following claim.

Claim. Let $i, j \in [m]$ such that $\mathfrak{A} \models \text{sph}^{\mathfrak{A}}_{r, \bar{v}_i \bar{w}^{\text{in}}}[\bar{v}_j \bar{w}^{\text{in}}]$. Then $\lambda_i = \lambda_j$.

Proof. First, from $\mathfrak{A} \models \text{sph}^{\mathfrak{A}}_{r, \bar{v}_i \bar{w}^{\text{in}}}(\bar{v}_j \bar{w}^{\text{in}})$, it follows that $\mathcal{S}^{\mathfrak{A}}_r(\bar{v}_i \bar{w}^{\text{in}}) \cong \mathcal{S}^{\mathfrak{A}}_r(\bar{v}_j \bar{w}^{\text{in}})$. Using Lemma 4.3, we obtain $\text{lhf}^{\mathfrak{A}}_r(\bar{v}_i \bar{w}^{\text{in}}) = \text{lhf}^{\mathfrak{A}}_r(\bar{v}_j \bar{w}^{\text{in}})$.

Second, from the construction of $w^{(i)}$ and $N^{(i)}$, it follows that $N^{\mathfrak{A}}_{2r+1}(\bar{v}_p) \subseteq N^{(0)} \subseteq N^{(s)}$ for all $p \in [m]$, $N^{\mathfrak{A}}_{2r+1}(\bar{w}^*_p) \subseteq N^{(p)} \subseteq N^{(s)}$ for all $p \in [s]$, and $\bar{w}^*_p \not\in N^{(s)}$ for all $p \in [s+1, \ell]$. Thus, $\text{dist}^{\mathfrak{A}}(\bar{v}_p \bar{w}^{\text{in}}, \bar{w}^{\text{out}}) > 2r+1$ for every $p \in [m]$.

Using Lemma 4.4 and $\bar{w}^* = \bar{w}^{\text{in}} \bar{w}^{\text{out}}$, we obtain $\text{lhf}^{\mathfrak{A}}_r(\bar{v}_i \bar{w}^*) = \text{lhf}^{\mathfrak{A}}_r(\bar{v}_j \bar{w}^*)$. With Lemma 4.2 and our choice of the radius r, it then follows that

$$\mathfrak{A} \models \varphi^*[\bar{v}_i, \bar{w}^*, \bar{n}^*] \iff \mathfrak{A} \models \varphi^*[\bar{v}_j, \bar{w}^*, \bar{n}^*].$$

Since $h^{\mathfrak{A}}_{\varphi^*, \bar{w}^*, \bar{n}^*}$ is assumed to be consistent with T, this implies $\lambda_i = \lambda_j$. ⌟

If $h^{\mathfrak{A}}_{\varphi, \bar{w}}(\bar{v}_i) = 1$, then there is some $p \in [m]$ such that $\lambda_p = 1$ and $\mathfrak{A} \models \text{sph}^{\mathfrak{A}}_{r, \bar{v}_p \bar{w}^{\text{in}}}[\bar{v}_i \bar{w}^{\text{in}}]$. Using the claim, we obtain $\lambda_i = \lambda_p = 1$. Thus, all in all, $h^{\mathfrak{A}}_{\varphi, \bar{w}}$ is consistent with T. □

This result shows that we only have to look for parameters in a local neighbourhood around the examples. In structures of bounded degree, this drastically reduces the number of parameters we have to check. Next, we bound the number of formulas we have to consider.

Lemma 4.7. *For fixed σ, d, k, ℓ, c_r, and c_w, up to equivalence, the number of formulas in Φ_d is constant.*

Proof. In σ-structures of degree at most d, for $r = (2 \cdot c_w + 1)^{c_r}$, the number of elements in an r-sphere with $(k+\ell)$ centres can be bounded by $(k+\ell) \cdot \mu_d(r)$ with $\mu_0(r) := 1$, $\mu_1(r) := 2$, and $\mu_d(r) := 1 + d \cdot$

```
Require: local access to background structure $\mathfrak{A}$,
    training sequence $T = \big((\bar{v}_1,\lambda_1),\dots,(\bar{v}_m,\lambda_m)\big)$
 1: $N \leftarrow N^{\mathfrak{A}}_{(2r+1)\ell}(T)$
 2: for all $\bar{w} \in N^{\ell}$ do
 3:   for all $\varphi \in \Phi_d$ do
 4:     consistent $\leftarrow$ true
 5:     for all $i \in [m]$ do
 6:       if $[\![\varphi(\bar{v}_i,\bar{w})]\!]^{\mathcal{N}^{\mathfrak{A}}_r(\bar{v}_i\bar{w})} \neq \lambda_i$ then
 7:         consistent $\leftarrow$ false
 8:         break
 9:     if consistent then
10:       return $(\varphi,\bar{w})$
11: reject
```

Figure 4.2: Learning algorithm $\mathcal{A}^{d}_{\text{con}}$ for Theorem 4.5

$\sum_{i=0}^{r}(d-1)^i$ for $d \geqslant 2$. We have $\mu_2(r) = 2r+1$ and, for $d > 2$, one can show that $\mu_d(r) \leqslant (d-1)^{r+1}$. Hence, since σ is fixed, there is a constant number of non-isomorphic spheres of radius at most r in such σ-structures. Thus, the number of sphere formulas based on those spheres, up to equivalence, is also constant. Since Φ_d consists of all Boolean combinations of these sphere formulas, the number of non-equivalent formulas in Φ_d is constant as well. □

With a bound on the number of parameters and a constant number of formulas, it remains to show that we can check every single hypothesis efficiently. For this, we use the following result due to Seese [84].

Theorem 4.8 ([84])**.** *Let* $d \in \mathbb{N}$. *On the class* $\mathcal{C}$ *of relational structures of degree at most* d, *the problem* p-FO-Mc($\mathcal{C}$) *is fixed-parameter linear, that is, there is an algorithm* $\mathcal{A}_{MC}$ *and a function* $f\colon \mathbb{N} \to \mathbb{N}$ *such that* $\mathcal{A}_{MC}$ *decides* p-FO-Mc($\mathcal{C}$) *and, on input* $(\mathfrak{A},\varphi)$, *the algorithm runs in time* $f(|\varphi|)\cdot|\mathfrak{A}|$. *under the uniform-cost measure. Under the logarithmic-cost measure, the algorithm runs in time* $f(|\varphi|)\cdot|\mathfrak{A}|\log|\mathfrak{A}|$.

We can now prove the consistent-learning result.

Proof of Theorem 4.5. We show that the algorithm given in Figure 4.2 fulfils the requirements of the theorem. The algorithm goes through all tuples $\bar{w} \in \big(N^{\mathfrak{A}}_{(2r+1)\ell}(T)\big)^{\ell}$ and all non-equivalent formulas $\varphi \in \Phi_d$. A hypothesis $h^{\mathfrak{A}}_{\varphi,\bar{w}} = [\![\varphi(\bar{x},\bar{y})]\!]^{\mathfrak{A}}(\bar{x},\bar{w})$ is consistent with the training sequence T if and only if $[\![\varphi(\bar{v}_i,\bar{w})]\!]^{\mathfrak{A}} = \lambda_i$ for all $i \in [m]$. Since Φ_d only contains Boolean combinations of sphere formulas of locality radius at most r, all formulas in Φ_d are r-local. Thus, $h^{\mathfrak{A}}_{\varphi,\bar{w}}$ is consistent with T if and only if $[\![\varphi(\bar{v}_i,\bar{w})]\!]^{\mathcal{N}^{\mathfrak{A}}_r(\bar{v}_i\bar{w})} = \lambda_i$ for every $i \in [m]$. Hence, if the algorithm returns a hypothesis, then it is consistent. Furthermore, if there is a consistent hypothesis $h^{\mathfrak{A}}_{\varphi^*,\bar{w}^*,\bar{n}^*}$ using a formula $\varphi^*(\bar{x},\bar{y},\bar{\kappa}) \in$

Φ^* and tuples $\bar{w}^* \in \big(U(\mathfrak{A})\big)^\ell$, and $\bar{n}^* \in [0, |\mathfrak{A}|]^{|\bar{\kappa}|}$, then, by Lemma 4.6, there is a consistent hypothesis among the ones we check, so the algorithm returns a hypothesis.

It remains to show that the algorithm satisfies the running time requirements while only using local access to the structure $\mathfrak{A}$. For all $\bar{v} \in \big(U(\mathfrak{A})\big)^k$ and $\bar{w} \in \big(U(\mathfrak{A})\big)^\ell$, as discussed in the proof of Lemma 4.7, the size of the neighbourhood $N_r^{\mathfrak{A}}(\bar{v}\bar{w})$ can be bounded by $(k+\ell)\cdot\mu_d(r)$, so it is constant for fixed d, k, ℓ, r. Hence, under the logarithmic-cost measure, the neighbourhood can be computed in time $\mathcal{O}(\log n)$ using only local access. Under the uniform-cost measure, it takes constant time to compute the neighbourhood. By Theorem 4.8, on an already computed constant-size neighbourhood, the evaluation of the hypothesis in line 6 runs in constant time. The algorithm checks up to $|N|^\ell \cdot |\Phi_d| \in \mathcal{O}\big((m \cdot k \cdot d^{(2r+1)\ell+1})^\ell \cdot |\Phi_d|\big)$ hypotheses on m examples with $N = N_{(2r+1)\ell}^{\mathfrak{A}}(T)$ and where $|\Phi_d|$ only considers non-equivalent formulas. All in all, since d, k, ℓ, r are considered constant, the running time of the algorithm is in $(m + \log n)^{\mathcal{O}(1)}$ under the logarithmic-cost measure, in $m^{\mathcal{O}(1)}$ under the uniform-cost measure, and it only uses local access to the structure $\mathfrak{A}$.

To evaluate the hypothesis returned by the algorithm on a tuple $\bar{v}$, we only have to evaluate it within the neighbourhood $\mathcal{N}_r^{\mathfrak{A}}(\bar{v}\bar{w})$, using only local access to $\mathfrak{A}$. Analogously to the consistency check, the hypothesis can be evaluated in time $(\log n)^{\mathcal{O}(1)}$ under the logarithmic-cost measure and in constant time under the uniform-cost measure. □

In the proof, we rely on Φ_d being a constant-sized set of formulas which is expressive enough to describe every concept that can be described using a formula from Φ^*. We obtain the expressiveness via formulas in Hanf normal form. However, to bound the number of these formulas, we need to bound the degree of the structures we consider in Lemma 4.7. Without this bound on the degree, even in structures of only logarithmic degree, the bound on the number of formulas in Φ_d would be superlinear in the size of the structure, so this would not yield a sublinear-time learning algorithm any more. Thus, in Section 4.4, we use a different technique to prove consistent learnability on structures of polylogarithmic degree.

Next, we extend Theorem 4.5 to the ERM problem.

Empirical Risk Minimisation

Theorem 4.9. *Let σ be a relational signature, let $k, \ell, c_r, c_w \in \mathbb{N}$, and let $\mathcal{C}$ be a class of structures of degree at most d for some $d \in \mathbb{N}$. There is an algorithm that solves* FOCN(P)-Learn-ERM$(\sigma, k, \ell, c_r, c_w)$ *on $\mathcal{C}$ in time $(\log n + m)^{\mathcal{O}(1)}$ under the logarithmic-cost measure and in time $m^{\mathcal{O}(1)}$ under the uniform-cost measure, where n is the size of the background structure and m is the length of the training sequence.*

```
Require: local access to background structure $\mathfrak{A}$,
    training sequence $T = ((\bar{v}_1, \lambda_1), \dots, (\bar{v}_m, \lambda_m))$
1: $N \leftarrow N^{\mathfrak{A}}_{(2r+1)\ell}(T)$
2: $err_{min} \leftarrow |T| + 1$
3: for all $\bar{w} \in N^{\ell}$ do
4:   for all $\varphi \in \Phi_d$ do
5:     $err \leftarrow 0$
6:     for all $i \in [m]$ do
7:       if $[\![\varphi(\bar{v}_i, \bar{w})]\!]^{\mathcal{N}^{\mathfrak{A}}_r(\bar{v}_i \bar{w})} \neq \lambda_i$ then
8:         $err \leftarrow err + 1$
9:     if $err < err_{min}$ then
10:      $err_{min} \leftarrow err$
11:      $(\varphi_{min}, \bar{w}_{min}) \leftarrow (\varphi, \bar{w})$
12: return $(\varphi_{min}, \bar{w}_{min})$
```

Figure 4.3: Learning algorithm $\mathcal{A}^{d}_{\mathrm{ERM}}$ for Theorem 4.9

Furthermore, the hypotheses returned by the algorithm can be evaluated in time $(\log n)^{O(1)}$ *under the logarithmic-cost measure and in constant time under the uniform-cost measure.*

To prove this result, we use the following corollary of Lemma 4.6.

Corollary 4.10. *Let* $T \in \left((U(\mathfrak{A}))^k \times \{0,1\}\right)^m$ *be a training sequence and let* $\varphi^* \in \Phi^*$, $\bar{w}^* \in \left(U(\mathfrak{A})\right)^{\ell}$, *and* $\bar{n}^* \in [0, |\mathfrak{A}|]^{|\bar{\kappa}|}$. *Then, there is a formula* $\varphi \in \Phi_d$ *and a tuple* $\bar{w} \in \left(N^{\mathfrak{A}}_{(2r+1)\ell}(T)\right)^{\ell}$ *such that* $\mathrm{err}_T\left(h^{\mathfrak{A}}_{\varphi,\bar{w}}\right) \leqslant \mathrm{err}_T\left(h^{\mathfrak{A}}_{\varphi^*,\bar{w}^*,\bar{n}^*}\right)$.

Proof. Let $\varepsilon := \mathrm{err}_T\left(h^{\mathfrak{A}}_{\varphi^*,\bar{w}^*,\bar{n}^*}\right)$. Then, there is a sequence S that is a subsequence of T of length $(1-\varepsilon)\cdot|T|$ such that $h^{\mathfrak{A}}_{\varphi^*,\bar{w}^*,\bar{n}^*}$ is consistent with S. By Lemma 4.6, there is also a formula $\varphi \in \Phi_d$ and a tuple $\bar{w} \in \left(N^{\mathfrak{A}}_{(2r+1)\ell}(T)\right)^{\ell}$ such that $h^{\mathfrak{A}}_{\varphi,\bar{w}}$ is consistent with S. Thus, we have $\mathrm{err}_T\left(h^{\mathfrak{A}}_{\varphi,\bar{w}}\right) \leqslant \varepsilon = \mathrm{err}_T\left(h^{\mathfrak{A}}_{\varphi^*,\bar{w}^*,\bar{n}^*}\right)$. □

Using this corollary, we can now prove Theorem 4.9.

Proof of Theorem 4.9. We show that the algorithm given in Figure 4.3 fulfils the requirements of the theorem. The algorithm goes through all tuples $\bar{w} \in \left(N^{\mathfrak{A}}_{(2r+1)\ell}(T)\right)^{\ell}$ and all non-equivalent formulas $\varphi \in \Phi_d$ and counts the number of errors that $h^{\mathfrak{A}}_{\varphi,\bar{w}} = [\![\varphi(\bar{x},\bar{y})]\!]^{\mathfrak{A}}(\bar{x},\bar{w})$ makes on T. Then, it returns a hypothesis with minimal training error. By Corollary 4.10, the hypothesis returned by the algorithm fulfils the requirements of the problem FOCN(P)-Learn-ERM. The running time analysis is analogous to the one presented in the proof of Theorem 4.5. □

Agnostic PAC learning

Finally, we obtain agnostic PAC learnability of $\mathsf{FOCN}(\mathbb{P})$ via the ERM algorithm.

Theorem 4.11. *Let σ be a relational signature, let $k, \ell, c_r, c_w \in \mathbb{N}$, and let $\mathcal{C}$ be a class of structures of degree at most d for some $d \in \mathbb{N}$. There is an algorithm that solves* FOCN(P)-LEARN-PAC$(\sigma, k, \ell, c_r, c_w)$ *on $\mathcal{C}$ in time $\big(\log|\mathfrak{A}| + \log\frac{1}{\delta} + \frac{1}{\varepsilon}\big)^{\mathcal{O}(1)}$, under the logarithmic-cost as well as the uniform-cost measure, where n is the size of the background structure.*

Furthermore, the hypotheses returned by the algorithm can be evaluated in time $(\log n)^{\mathcal{O}(1)}$ under the logarithmic-cost measure and in constant time under the uniform-cost measure.

Proof. Let $\mathfrak{A} \in \mathcal{C}$ be a background structure of degree at most d.

We consider the concept class

$$\mathcal{H}^* = \left\{h^{\mathfrak{A}}_{\varphi,\bar{w},\bar{n}} \;\middle|\; \varphi(\bar{x},\bar{y},\bar{\kappa}) \in \Phi^*,\ \bar{w} \in \big(U(\mathfrak{A})\big)^{\ell},\ \bar{n} \in [0,|\mathfrak{A}|]^{|\bar{\kappa}|}\right\}$$

and the hypothesis class

$$\mathcal{H} = \left\{h^{\mathfrak{A}}_{\varphi,\bar{w}} \;\middle|\; \varphi(\bar{x},\bar{y}) \in \Phi_d,\ \bar{w} \in \big(U(\mathfrak{A})\big)^{\ell}\right\}.$$

Since, by Lemma 4.7, Φ_d contains (up to equivalence) only finitely many formulas, the number of hypotheses in $\mathcal{H}$ is bounded by $s \cdot |\mathfrak{A}|^{\ell}$ for some constant s.

Claim. It holds that $\mathcal{H}^* \subseteq \mathcal{H}$.

Proof. Let $h^* := h^{\mathfrak{A}}_{\varphi^*,\bar{w}^*,\bar{n}^*} \in \mathcal{H}^*$. We consider the training sequence T that contains an example $\big(\bar{v}, h^*(\bar{v})\big)$ for every k-tuple $\bar{v}$ from $\mathfrak{A}$. Then, by Lemma 4.6, there is a formula $\varphi \in \Phi_d$ and a tuple $\bar{w} \in \big(U(\mathfrak{A})\big)^{\ell}$ such that the hypothesis $h^{\mathfrak{A}}_{\varphi,\bar{w}} \in \mathcal{H}$ is consistent with T. By the definition of T, we have $h^* = h^{\mathfrak{A}}_{\varphi,\bar{w}}$, and thus, $h^* \in \mathcal{H}$. ⌟

By using the claim, we can also bound the number of hypotheses in $\mathcal{H}^*$ by $s \cdot |\mathfrak{A}|^{\ell}$. Our algorithm that solves FOCN(P)-LEARN-PAC works as follows.

Given local access to a background structure $\mathfrak{A}$, oracle access to the size $|\mathfrak{A}|$ of the structure, oracle access to a probability distribution $\mathcal{D}$ on $\big(U(\mathfrak{A})\big)^k \times \{0,1\}$, and given rational numbers $\varepsilon, \delta > 0$, our algorithm queries

$$m(|\mathfrak{A}|, \varepsilon, \delta) := \left\lceil \frac{2\log(2s \cdot |\mathfrak{A}|^{\ell}/\delta)}{\varepsilon^2} \right\rceil$$

many examples from $\mathcal{D}$. Then, it runs $\mathcal{A}^d_{\text{ERM}}$ on the resulting training sequence.

Next, we show that this algorithm indeed solves the problem FOCN(P)-LEARN-PAC. Let $\mathcal{D}$ be a distribution over $\big(U(\mathfrak{A})\big)^k \times \{0,1\}$ and let $h^* \in \mathcal{H}^*$ be a hypothesis that minimises the generalisation error, that is, $\operatorname{err}_{\mathcal{D}}(h^*) = \min_{h' \in \mathcal{H}^*} \operatorname{err}_{\mathcal{D}}(h')$. Let T be the training sequence of length $m(|\mathfrak{A}|, \varepsilon, \delta)$ drawn i.i.d. from $\mathcal{D}$ by our algorithm, and let $h \in \mathcal{H}$ be the hypothesis returned by $\mathcal{A}^d_{\text{ERM}}$ on input T. By Theorem 4.9, the hypothesis h fulfils $\operatorname{err}_T(h) \leqslant \operatorname{err}_T(h^*)$.

Furthermore, by the Uniform Convergence Lemma (Lemma 3.9), with probability at least $1-\delta$, it holds that $\left|\mathrm{err}_T(h') - \mathrm{err}_{\mathcal{D}}(h')\right| \leqslant \frac{\varepsilon}{2}$ for all $h' \in \mathcal{H}$. This especially holds for h as well as for h^*. Hence,

$$\mathrm{err}_{\mathcal{D}}(h) \leqslant \mathrm{err}_T(h) + \frac{\varepsilon}{2} \leqslant \mathrm{err}_T(h^*) + \frac{\varepsilon}{2} \leqslant \mathrm{err}_D(h^*) + \frac{\varepsilon}{2} + \frac{\varepsilon}{2}$$

with probability at least $1-\delta$. This is exactly the requirement we have in FOCN(P)-Learn-PAC for the returned hypothesis.

The number $m(|\mathfrak{A}|, \varepsilon, \delta)$ of queried examples can be bounded by $\mathcal{O}\left(\frac{\log(|\mathfrak{A}|/\delta)}{\varepsilon^2}\right)$. Thus, by Theorem 4.9, we can bound the running time of our algorithm by $\left(\log|\mathfrak{A}| + \log\frac{1}{\delta} + \frac{1}{\varepsilon}\right)^{\mathcal{O}(1)}$ under the logarithmic-cost as well as the uniform-cost measure. The evaluation time of the hypothesis given in the theorem follows directly from Theorem 4.9. □

4.4 STRUCTURES OF SMALL DEGREE

In this section, we extend the sublinear-time results for consistent learning and the ERM problem of the previous section to classes of structures of at most polylogarithmic degree. At the end of this section, we give a bound on the degree of a structure in terms of its size such that PAC-learning is still possible in sublinear time. We start with the extension of the consistent-learning result.

Consistent learning

Theorem 4.12. *Let σ be a relational signature and let $k, \ell, c_r, c_w \in \mathbb{N}$. There is an algorithm that solves* FOCN($\mathbb{P}$)-Learn-Consistent$(\sigma, k, \ell, c_r, c_w)$ *in time $(\log n + m)^{\mathcal{O}(1)} \cdot d^{\mathrm{polylog}\, d}$ under the logarithmic-cost measure and in time $m^{\mathcal{O}(1)} \cdot d^{\mathrm{polylog}\, d}$ under the uniform-cost measure, where n is the size of the background structure, d is the degree of the background structure, and m is the length of the training sequence. Furthermore, the hypotheses returned by the algorithm can be evaluated with the same time bound.*

On classes of structures of polylogarithmic degree, Theorem 4.12 implies that consistent model-learning is possible in sublinear time.

Corollary 4.13. *Let σ be a relational signature, let $k, \ell, c_r, c_w \in \mathbb{N}$, and let $\mathcal{C}$ be a class of structures of polylogarithmic degree. There is an algorithm that solves* FOCN($\mathbb{P}$)-Learn-Consistent$(\sigma, k, \ell, c_r, c_w)$ *on $\mathcal{C}$ in time sublinear in the size of the background structure and polynomial in the length of the training sequence, under the logarithmic-cost as well as the uniform-cost measure. The hypotheses returned by the algorithm can be evaluated with the same bound on the running time.*

In the proof of Theorem 4.12, to check the consistency of a hypothesis and to evaluate it on new tuples, we use the following result on isomorphism testing due to Grohe, Neuen, and Schweitzer [54].

Theorem 4.14 ([54]). *There is a constant c such that for all σ-structures $\mathfrak{A}_1$ and $\mathfrak{A}_2$, it can be decided in time $n^{\mathcal{O}(a \cdot (\log d)^c)}$ whether $\mathfrak{A}_1$ and $\mathfrak{A}_2$ are isomorphic, where $n := \max\{|\mathfrak{A}_1|, |\mathfrak{A}_2|\}$, $d := \max\{\deg(\mathfrak{A}_1), \deg(\mathfrak{A}_2)\}$, and $a := \max_{R \in \sigma} \mathrm{ar}(R)$.*

```
Require: local access to background structure $\mathfrak{A}$,
    training sequence $T = \big((\bar{v}_1, \lambda_1), \dots, (\bar{v}_m, \lambda_m)\big)$
 1: $N \leftarrow N^{\mathfrak{A}}_{(2r+1)\ell}(T)$
 2: for all $\bar{w} = (w_1, \dots, w_\ell) \in N^\ell$ do
 3:   for all $s \in [0, \ell]$ do
 4:     $\mathtt{consistent} \leftarrow$ true
 5:     $\bar{w}^{\text{in}} \leftarrow (w_1, \dots, w_s)$
 6:     for all $i \in [m]$ do
 7:       $\mathfrak{S}_i \leftarrow \mathcal{S}^{\mathfrak{A}}_r(\bar{v}_i \bar{w}^{\text{in}})$
 8:     for all $i, j \in [m]$ with $\lambda_i = 0$ and $\lambda_j = 1$ do
 9:       if $\mathfrak{S}_i \cong \mathfrak{S}_j$ then
10:         $\mathtt{consistent} \leftarrow$ false
11:         break
12:     if $\mathtt{consistent}$ then
13:       $\varphi(\bar{x}, \bar{y}) \leftarrow \bigvee_{i \in [m],\, \lambda_i = 1} \mathrm{sph}^{\mathfrak{A}}_{r, \bar{v}_i \bar{w}^{\text{in}}}(\bar{x}, y_1, \dots, y_s)$
14:       return $(\varphi, \bar{w})$
15: reject
```

Figure 4.4: Learning algorithm $\mathcal{A}_{\text{con}}$ for Theorem 4.12

We assume that we are not only given the formula for the hypothesis, but also a description of the spheres, i. e. the relational structures, that are the basis for the sphere formulas used in the hypothesis. Then, to evaluate the hypothesis, for every sphere formula used in the hypothesis, we determine whether the sphere of the sphere formula is isomorphic to the sphere around the elements given to the sphere formula. The label defined by the hypothesis is then simply a Boolean combination of the determined truth values. We analyse the running time of this procedure in the following proof of the consistent-learning result.

Proof of Theorem 4.12. The pseudocode for our algorithm is shown in Figure 4.4. As in the last section, let $r \coloneqq (2 \cdot c_w + 1)^{c_r}$. The algorithm is based on the proof of Lemma 4.6. It goes through all tuples $\bar{w} \in \big(N^{\mathfrak{A}}_{(2r+1)\ell}(T)\big)^\ell$, and, for all $s \in [0, \ell]$, it considers the tuple consisting of the first s entries of $\bar{w}$. For these values, it checks whether the hypothesis $(\varphi, \bar{w})$ is consistent with the training sequence, where φ is the formula given in Lemma 4.6, that is, the disjunction of sphere formulas around the positive examples and the s-tuple derived from $\bar{w}$.

First, we show that every hypothesis returned by the algorithm is consistent with the training sequence. Let $(\bar{v}_i, \lambda_i) \in T$. By the construction of φ, we have $\mathfrak{A} \models \varphi[\bar{v}_i, \bar{w}]$ (and thus $h^{\mathfrak{A}}_{\varphi,\bar{w}}(\bar{v}_i) = 1$) if and only if there is some j with $\lambda_j = 1$ such that $\mathfrak{A} \models \mathrm{sph}^{\mathfrak{A}}_{r, \bar{v}_j \bar{w}^{\text{in}}}[\bar{v}_i, \bar{w}^{\text{in}}]$, or, equivalently, $\mathcal{S}^{\mathfrak{A}}_r(\bar{v}_i) \cong \mathcal{S}^{\mathfrak{A}}_r(\bar{v}_j)$. If $\lambda_i = 1$, then this is trivially the case, so the hypothesis correctly classifies the tuple $\bar{v}_i$ as positive. If

$\lambda_i = 0$, then the checks in lines 8–11 of the algorithm guarantee that there is no positive example with an isomorphic sphere, and hence the hypothesis correctly classifies the tuple $\bar{v}_i$ as negative. All in all, this shows that every hypothesis returned by the algorithm is consistent.

For the other direction, we assume that there is a formula $\varphi^* \in \Phi^*$ and tuples $\bar{w}^* \in \big(U(\mathfrak{A})\big)^\ell$ and $\bar{n}^* \in [0, |\mathfrak{A}|]^{|\bar{\kappa}|}$ such that the hypothesis $h^{\mathfrak{A}}_{\varphi^*,\bar{w}^*,\bar{n}^*}$ is consistent with T. Then it follows from the proof of Lemma 4.6 that there is a tuple $\bar{w}$ among the ones we check such that the resulting hypothesis is consistent with the training sequence. Thus, our algorithm returns a hypothesis in these cases.

It remains to show that the algorithm satisfies the running time requirements while only using local access to the structure $\mathfrak{A}$. Analogously to the proof of Theorem 4.8, for fixed k, ℓ, c_r, and c_w, the size of the set N computed in line 1 is polynomial in m and d. It can be computed in time $(\log n + m + d)^{\mathcal{O}(1)}$ under the logarithmic-cost measure and in time $(m + d)^{\mathcal{O}(1)}$ under the uniform-cost measure, using only local access to the background structure. For every single choice of $\bar{w}$ and s, the size of a single sphere $\mathfrak{S}_i$ is polynomial in d and it can be computed in time polynomial in d and $\log n$ under the logarithmic-cost measure and polynomial in d under the uniform-cost measure. By Theorem 4.14, every single isomorphism test between the spheres runs in time $d^{\operatorname{polylog} d}$. All in all, the algorithm runs in time $(\log n + m)^{\mathcal{O}(1)} d^{\operatorname{polylog} d}$ under the logarithmic-cost measure and in time $m^{\mathcal{O}(1)} d^{\operatorname{polylog} d}$ under the uniform-cost measure, while only using local access.

To evaluate the hypothesis returned by the algorithm on a new tuple $\bar{v}$, we compute the r-sphere around $\bar{v}\bar{w}^{\text{in}}$ and check whether it is isomorphic to one of the spheres used in the returned formula φ. Thus, we obtain the same running time bounds as for the learning algorithm. □

Empirical Risk Minimisation

Next, we extend this result to the ERM problem.

Theorem 4.15. *Let σ be a relational signature and let $k, \ell, c_r, c_w \in \mathbb{N}$. There is an algorithm that solves* FOCN(P)-Learn-ERM$(\sigma, k, \ell, c_r, c_w)$ *in time $(\log n + m)^{\mathcal{O}(1)} \cdot d^{\operatorname{polylog} d}$ under the logarithmic-cost measure and in time $m^{\mathcal{O}(1)} \cdot d^{\operatorname{polylog} d}$ under the uniform-cost measure, where n is the size of the background structure, d is the degree of the background structure, and m is the length of the training sequence. Furthermore, the hypotheses returned by the algorithm can be evaluated with the same time bound.*

Proof. The pseudocode for our algorithm $\mathcal{A}_{\text{ERM}}$ is shown in Figure 4.5. Let $r := (2 \cdot c_w + 1)^{c_r}$. The algorithm goes through all tuples $\bar{w} \in \big(N^{\mathfrak{A}}_{(2r+1)\ell}(T)\big)^\ell$. For all $s \in [0, \ell]$, it considers the tuple $\bar{w}^{\text{in}}$ consisting of the first s entries of $\bar{w}$. For every sphere $\mathfrak{S}_i = \mathcal{S}^{\mathfrak{A}}_r(\bar{v}_i \bar{w}^{\text{in}})$, the algorithm counts the number err^+_i of errors the hypothesis would make on the training sequence if we would include the sphere formula for $\mathfrak{S}_i$ in the hypothesis. Additionally, it also counts the number err^-_i of errors

```
Require: local access to background structure $\mathfrak{A}$,
    training sequence $T = \big((\bar{v}_1, \lambda_1), \dots, (\bar{v}_m, \lambda_m)\big)$
 1: $N \leftarrow N^{\mathfrak{A}}_{(2r+1)\ell}(T)$
 2: $\mathrm{err}_{\min} \leftarrow |T| + 1$
 3: for all $\bar{w} = (w_1, \dots, w_\ell) \in N^\ell$ do
 4:     for all $s \in [0, \ell]$ do
 5:         $\mathrm{err} \leftarrow 0$
 6:         $\bar{w}^{\mathrm{in}} \leftarrow (w_1, \dots, w_s)$
 7:         for all $i \in [m]$ do
 8:             $\mathfrak{S}_i \leftarrow \mathcal{S}^{\mathfrak{A}}_r(\bar{v}_i \bar{w}^{\mathrm{in}})$
 9:         for all $i \in [m]$ do
10:             $\mathrm{err}_i^+ \leftarrow \big|\{j \in [m] \mid \mathfrak{S}_i \cong \mathfrak{S}_j \text{ and } \lambda_j = 0\}\big|$
11:             $\mathrm{err}_i^- \leftarrow \big|\{j \in [m] \mid \mathfrak{S}_i \cong \mathfrak{S}_j \text{ and } \lambda_j = 1\}\big|$
12:             $\mathrm{err} \leftarrow \mathrm{err} + \min\{\mathrm{err}_i^+, \mathrm{err}_i^-\}$
13:         if $\mathrm{err} < \mathrm{err}_{\min}$ then
14:             $\mathrm{err}_{\min} \leftarrow \mathrm{err}$
15:             $\bar{w}_{\min} \leftarrow \bar{w}$
16:             $\varphi_{\min}(\bar{x}, \bar{y}) \leftarrow \bigvee_{i \in [m],\ \mathrm{err}_i^+ \leqslant \mathrm{err}_i^-} \mathrm{sph}^{\mathfrak{A}}_{r, \bar{v}_i \bar{w}^{\mathrm{in}}}(\bar{x}, y_1, \dots, y_s)$
17: return $(\varphi_{\min}, \bar{w}_{\min})$
```

Figure 4.5: Learning algorithm $\mathcal{A}_{\mathrm{ERM}}$ for Theorem 4.15

the hypothesis would make on the training sequence if we would leave out the sphere formula for $\mathfrak{S}_i$. The sphere formula is included in the hypothesis if $\mathrm{err}_i^+ \leqslant \mathrm{err}_i^-$. For every combination of a tuple $\bar{w}$ and a number s, the algorithm sums up the number of errors the hypothesis would make on the training sequence and in the end, it returns the hypothesis with the minimum number of errors.

Claim. Let $(\varphi_{\min}, \bar{w}_{\min})$ be the hypothesis returned by the algorithm. Then, for all $\varphi^*(\bar{x}, \bar{y}, \bar{\kappa}) \in \mathsf{FOCN}(\mathbb{P})[\sigma, c_r, c_w]$, $\bar{w}^* \in (U(\mathfrak{A}))^\ell$, and $\bar{n}^* \in [0, |\mathfrak{A}|]^{|\bar{\kappa}|}$, it holds that $\mathrm{err}_T\big(h^{\mathfrak{A}}_{\varphi_{\min}, \bar{w}_{\min}}\big) \leqslant \mathrm{err}_T\big(h^{\mathfrak{A}}_{\varphi^*, \bar{w}^*, \bar{n}^*}\big)$.

Proof. Choose $\varphi^*(\bar{x}, \bar{y}, \bar{\kappa}) \in \mathsf{FOCN}(\mathbb{P})[\sigma, c_r, c_w]$, $\bar{w}^* \in (U(\mathfrak{A}))^\ell$, and $\bar{n}^* \in [0, |\mathfrak{A}|]^{|\bar{\kappa}|}$ such that $\mathrm{err}_T\big(h^{\mathfrak{A}}_{\varphi^*, \bar{w}^*, \bar{n}^*}\big)$ is minimal. Let T^* be the subsequence of T that contains exactly those examples that are correctly classified by $h^* := h^{\mathfrak{A}}_{\varphi^*, \bar{w}^*, \bar{n}^*}$. It suffices to show that, for all examples $(\bar{v}_i, \lambda_i)$ in T^*, we have that $\mathrm{err}_i^+ \leqslant \mathrm{err}_i^-$ if $\lambda_i = 1$ and $\mathrm{err}_i^+ \geqslant \mathrm{err}_i^-$ if $\lambda_i = 0$. If we would have $\mathrm{err}_i^+ > \mathrm{err}_i^-$ and $\lambda_i = 1$, then using $\varphi := \varphi^* \wedge \neg \mathrm{sph}^{\mathfrak{A}}_{r, \bar{v}_i \bar{w}^{\mathrm{in}}}$ would yield a hypothesis that is consistent with more examples than h^*, which contradicts the optimality of h^*. On the other hand, if we would have $\mathrm{err}_i^+ < \mathrm{err}_i^-$ and $\lambda_i = 0$, then we could use $\varphi := \varphi^* \vee \mathrm{sph}^{\mathfrak{A}}_{r, \bar{v}_i \bar{w}^{\mathrm{in}}}$. ⌟

The analysis of the running time of the algorithm $\mathcal{A}_{\mathrm{ERM}}$ is analogous to the analysis of the algorithm $\mathcal{A}_{\mathrm{con}}$ in the proof of Theorem 4.12 and it yields the same result. □

Analogously to the consistent-learning case, on classes of structures of polylogarithmic degree, Theorem 4.15 implies that the ERM problem is solvable in sublinear time.

Corollary 4.16. *Let σ be a relational signature, let $k, \ell, c_r, c_w \in \mathbb{N}$, and let $\mathcal{C}$ be a class of structures of polylogarithmic degree. There is an algorithm that solves* FOCN(P)-Learn-ERM$(\sigma, k, \ell, c_r, c_w)$ *on $\mathcal{C}$ in time sublinear in the size of the background structure and polynomial in the length of the training sequence, under the logarithmic-cost as well as the uniform-cost measure. The hypotheses returned by the algorithm can be evaluated with the same bound on the running time.*

To turn the algorithm $\mathcal{A}_{\mathrm{ERM}}$ into a sublinear-time PAC-learning algorithm, we want to find a sublinear bound on the number of examples needed to fulfil the probability bounds. In contrast to the approach in the last section, the formulas we use in the hypotheses do not come from a constant-sized set of formulas any more. Instead, the number of non-equivalent disjunctions of sphere formulas is exponential in the number of non-isomorphic spheres, which is again exponential in their size. This leads to the following result.

Agnostic PAC learning

Theorem 4.17. *Let σ be a relational signature, let $k, \ell, c_r, c_w \in \mathbb{N}$, let $a := \max_{R \in \sigma} \mathrm{ar}(R)$, $r := (2 \cdot c_w + 1)^{c_r}$, and let $\mathcal{C}$ be a class of structures $\mathfrak{A}$ of degree at most $\big(\log(\log|\mathfrak{A}|)\big)^{\frac{1}{(r+1)\cdot a}}$. There is an algorithm that solves* FOCN(P)-Learn-PAC$(\sigma, k, \ell, c_r, c_w)$ *on $\mathcal{C}$ in time sublinear in the size of the background structure and polynomial in $\log \frac{1}{\delta}$ and $\frac{1}{\varepsilon}$ under the logarithmic-cost as well as the uniform-cost measure.*

Furthermore, the hypotheses returned by the algorithm can be evaluated with the same bound on the running time.

Proof. Let $\mathfrak{A} \in \mathcal{C}$ be a background structure of degree d with $d \leqslant \big(\log(\log|\mathfrak{A}|)\big)^{\frac{1}{(r+1)\cdot a}}$. We consider the concept class

$$\mathcal{H}^* = \left\{ h^{\mathfrak{A}}_{\varphi,\bar{w},\bar{n}} \;\middle|\; \varphi(\bar{x},\bar{y},\bar{\kappa}) \in \Phi^*,\ \bar{w} \in \big(U(\mathfrak{A})\big)^{\ell},\ \bar{n} \in [0, |\mathfrak{A}|]^{|\bar{\kappa}|} \right\}.$$

Running on $\mathfrak{A}$, the algorithm $\mathcal{A}_{\mathrm{ERM}}$ only returns formulas from the set

$$\begin{aligned}\Phi_d := \big\{\varphi(\bar{x},\bar{y}) \in \mathsf{FO}[\sigma] \;\mid\; & |\bar{x}| = k,\ |\bar{y}| = \ell,\\ & \varphi \text{ is a disjunction of sphere formulas}\\ & \text{of locality radius at most } r\\ & \text{based on spheres of degree at most } d\big\}.\end{aligned}$$

Thus, we consider the hypothesis class

$$\mathcal{H} = \left\{ h^{\mathfrak{A}}_{\varphi,\bar{w}} \;\middle|\; \varphi(\bar{x},\bar{y}) \in \Phi_d,\ \bar{w} \in \big(U(\mathfrak{A})\big)^{\ell} \right\}.$$

As in the proof of Theorem 4.11, it holds that $\mathcal{H}^* \subseteq \mathcal{H}$.

Next, we bound number of non-equivalent hypotheses in $\mathcal{H}$ and thus also in $\mathcal{H}^*$. As discussed in Lemma 4.7, in a structure of degree at most d, a sphere of radius at most r with $(k+\ell)$ centres has size at most $s \coloneqq (k+\ell) \cdot \mu_d(r) \in \mathcal{O}\big(d^{r+1}\big) \subseteq \mathcal{O}\big((\log(\log(|\mathfrak{A}|)))^{\frac{1}{a}}\big)$. Thus, over a signature σ, the number of non-isomorphic spheres of radius at most r with $(k+\ell)$ centres can be bounded by $\prod_{R\in\sigma} 2^{s^{\mathrm{ar}(R)}} = 2^{\left(\sum_{R\in\sigma} s^{\mathrm{ar}(R)}\right)} \leqslant 2^{|\sigma|\cdot s^{a}}$. The number of non-equivalent disjunctions of sphere formulas based on such spheres is at most exponential in the number of non-isomorphic spheres. Hence, the set Φ_d contains at most $\mathcal{O}\left(|\mathfrak{A}|^{|\sigma|}\right)$ non-equivalent formulas, and the number of non-equivalent hypotheses in $\mathcal{H}$ and $\mathcal{H}^*$ is bounded by $c \cdot |\mathfrak{A}|^{\ell+|\sigma|}$ for some constant c.

The remainder of this proof is analogous to the proof of Theorem 4.11. We use Lemma 3.9 to bound the number of examples needed for a PAC-learning algorithm by

$$m(|\mathfrak{A}|, \varepsilon, \delta) \coloneqq \left\lceil \frac{2\log(2c \cdot |\mathfrak{A}|^{\ell+|\sigma|}/\delta)}{\varepsilon^2} \right\rceil .$$

Then, it suffices to query $m(|\mathfrak{A}|, \varepsilon, \delta)$ examples from the distribution $\mathcal{D}$ and run $\mathcal{A}_{\mathrm{ERM}}$ on the resulting training sequence. With the bound on the number of training examples, Theorem 4.15 yields the desired running time. □

5 WEIGHTED STRUCTURES AND LOGICS WITH WEIGHT AGGREGATION

In machine learning, input data is often given via numerical values which are contained in or extracted from a more complex structure, such as a relational database (cf., [51, 58, 81, 83]). The logic-based learning results obtained so far, however, only deal with pure relational structures. Thus, they are often too weak for describing meaningful classifiers for real-world machine-learning problems.

To overcome this issue and combine relational and numerical information, we are interested in hybrid structures, which extend relational ones by numerical values. Just as in commonly used relational database systems, to utilise the power of such hybrid structures, the classifiers we consider should be allowed to use different methods to aggregate the numerical values. Our main contribution in this chapter is the design of a logic that is capable of expressing meaningful machine-learning problems and, at the same time, well-behaved enough to have similar locality properties as first-order logic, which enable us to learn concepts in sublinear time on structures of small degree.

For that, in Section 5.1, we introduce a new logic, called *first-order logic with weight aggregation* (FOWA). It operates on *weighted structures*, which extend ordinary relational structures by assigning weights, i. e. elements from a particular abelian group or ring, to tuples present in the structure. Such weighted structures were recently considered by Toruńczyk [88], who studied the complexity of query evaluation problems for the related logic $\mathsf{FO}[\mathbb{C}]$ and its fragment $\mathsf{FO}_\mathsf{G}[\mathbb{C}]$. Our logic FOWA, however, is closer to the syntax and semantics of the extension FOC of first-order logic with counting quantifiers described in Section 2.3. This connection enables us to achieve locality results for the fragments FOW_1 and FOWA_1 of FOWA similar to those obtained in [56, 70]. Specifically, in Sections 5.2 and 5.3, we achieve Feferman-Vaught decompositions and a Gaifman normal form for FOW_1. In Section 5.4, we provide a localisation theorem for the more expressive logic FOWA_1. We give examples illustrating that FOWA_1 can express concepts relevant for various machine-learning scenarios. Using the locality properties, we provide learnability results for concepts definable in FOWA_1 in Chapter 6. This generalises the results described in Chapter 3 that Grohe and Ritzert [55] obtained for first-order logic to the substantially more expressive logic FOWA_1.

The results of this and the next chapter have been published in [13].

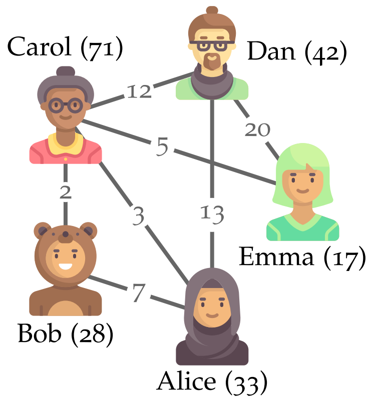


Figure 5.1: An excerpt of a social network[1], based on the network given in Example 3.1. An edge between two users indicates that they are friends. The weight for each user represents their age and the weight of the edges indicates the length of the users' friendship (in years).

5.1 FIRST-ORDER LOGIC WITH WEIGHT AGGREGATION

Weighted structures

In this section, we introduce weighted structures as well as the logic FOWA and its fragments FOW_1 and FOWA_1.

Let σ be a signature and let $\mathcal{S}$ be a collection of rings and/or abelian groups. Let $\mathbf{W}$ be a finite set of *weight symbols*, such that each $\mathsf{w} \in \mathbf{W}$ has an associated *arity* $\mathrm{ar}(\mathsf{w}) \in \mathbb{N}_{\geqslant 1}$ and a *type* $\mathrm{type}(\mathsf{w}) \in \mathcal{S}$. A $(\sigma, \mathbf{W})$*-structure* is a σ-structure $\mathfrak{A}$ that is enriched, for every $\mathsf{w} \in \mathbf{W}$, by an interpretation $\mathsf{w}^{\mathfrak{A}} \colon \big(U(\mathfrak{A})\big)^{\mathrm{ar}(\mathsf{w})} \to \mathrm{type}(\mathsf{w})$, which satisfies the following *locality condition*: if $\mathsf{w}^{\mathfrak{A}}(v_1, \ldots, v_k) \neq 0_S$ for $S := \mathrm{type}(\mathsf{w})$, $k := \mathrm{ar}(\mathsf{w})$, and $v_1, \ldots, v_k \in U(\mathfrak{A})$, then $v_1 = \cdots = v_k$ or all of the vertices $v_1, \ldots, v_k$ are contained in one tuple of a relation of $\mathfrak{A}$. More formally, in that case, there exists a relation symbol $R \in \sigma$ and a tuple $(w_1, \ldots, w_{\mathrm{ar}(R)}) \in R(\mathfrak{A})$ such that $\{v_1, \ldots, v_k\} \subseteq \{w_1, \ldots, w_{\mathrm{ar}(R)}\}$. Hence, if $\mathsf{w}^{\mathfrak{A}}(v_1, \ldots, v_k) \neq 0_S$, then the elements in $\{v_1, \ldots, v_k\}$ form a clique in the Gaifman graph of $\mathfrak{A}$.

All notions that were introduced in Section 2.3 for σ-structures carry over to $(\sigma, \mathbf{W})$-structures in the obvious way. Specifically, if $\mathfrak{A}$ is a $(\sigma, \mathbf{W})$-structure and σ' is a signature with $\sigma' \supseteq \sigma$, then a σ'*-expansion of* $\mathfrak{A}$ is a $(\sigma', \mathbf{W})$-structure $\mathfrak{A}'$ with $U(\mathfrak{A}') = U(\mathfrak{A})$, $R(\mathfrak{A}') = R(\mathfrak{A})$ for all $R \in \sigma$, and $\mathsf{w}^{\mathfrak{A}'} = \mathsf{w}^{\mathfrak{A}}$ for all $\mathsf{w} \in \mathbf{W}$.

We will use the following as running examples throughout this section.

Example 5.1. (a) Recall the network of friends from Example 3.1. Let $(\mathbb{Q}, +, \cdot)$ be the field of rationals and let $\mathbf{W}$ contain the unary weight symbol `age` and the binary weight symbol `length` of type $(\mathbb{Q}, +, \cdot)$ indicating the age of a user and the length of the friendship

1 Avatars designed by Freepik from Flaticon

between two users. For $\sigma = \{E\}$, let $\mathfrak{A}$ be the $(\sigma, \mathbf{W})$-structure that enriches the network of friends from Example 3.1 as shown in Figure 5.1.

(b) In a recent survey [81], Pan and Ding describe different approaches to represent social-media users via embeddings into a low-dimensional vector space, where the embeddings are based on the users' social-media posts[2]. We represent the available data by a weighted structure $\mathfrak{A}$ as follows. Consider the group $(\mathbb{R}^k, +)$, where $\mathbb{R}^k$ is the set of k-dimensional real vectors and $+$ is the usual vector addition; let $\mathbf{W}$ contain a unary weight symbol $\mathtt{embedding}$ of type $(\mathbb{R}^k, +)$. Let $\sigma = \{F\}$ and let $\mathfrak{A}$ be a $(\sigma, \mathbf{W})$-structure such that the universe $U(\mathfrak{A})$ consists of the users of a social network. Let $F(\mathfrak{A})$ contain all pairs of users (v, w) such that v is a follower of w. For every user $v \in U(\mathfrak{A})$, let $\mathtt{embedding}^{\mathfrak{A}}(v)$ be a k-dimensional vector representing v's social-media posts.

(c) Consider an online marketplace that allows retailers to sell their products to consumers. The database of the marketplace contains a table with transactions, and each entry consists of an identifier, a customer, a product, a retailer, the price per item, and the number of items sold. We can describe the database of the marketplace as a weighted structure as follows. Let $(\mathbb{Q}, +, \cdot)$ be the field of rationals, let $\mathbf{W}$ contain two unary weight symbols $\mathtt{price}$ and $\mathtt{quantity}$ of type $(\mathbb{Q}, +, \cdot)$, let $\sigma = \{T\}$, and let $\mathfrak{A}$ be a $(\sigma, \mathbf{W})$-structure such that the universe $U(\mathfrak{A})$ contains the identifiers for the transactions, customers, products, and retailers. For every transaction, let $T(\mathfrak{A})$ contain the 4-tuple (i, c, p, r) consisting of the identifier for the transaction, the customer, the product, and the retailer. For every transaction identifier i let $\mathtt{price}^{\mathfrak{A}}(i)$ be the price per item in the transaction and let $\mathtt{quantity}^{\mathfrak{A}}(i)$ be the number of items sold; for every other identifier v in $U(\mathfrak{A})$, let $\mathtt{price}^{\mathfrak{A}}(v) = \mathtt{quantity}^{\mathfrak{A}}(v) = 0$.

(d) Consider vertex-coloured edge-weighted graphs, where R, G, B are unary relations of red, green, and blue vertices, E is a binary relation of edges, and where every edge (v, w) has an associated *weight* that is a k-dimensional vector of reals (for some fixed number k). Such graphs can be viewed as $(\sigma, \mathbf{W})$-structures $\mathfrak{A}$, where $\sigma = \{E, R, G, B\}$, $\mathbf{W}$ contains a binary weight symbol $\mathtt{w}$ of type $(\mathbb{R}^k, +)$ and $\mathtt{w}^{\mathfrak{A}}(v, w) \in \mathbb{R}^k$ for all edges $(v, w) \in E(\mathfrak{A})$.

Based on these weighted structures, we can now introduce first-order logic with weight aggregation (FOWA). Let σ be a signature, $\mathbb{S}$ a collection of rings and/or abelian groups, and $\mathbf{W}$ a finite set of weight symbols. Fix a countably infinite set vars of variables. Analogously

2 Among other applications, such embeddings might be used to predict a user's personality or political leaning.

to σ-interpretations as defined in Section 2.3, a $(\sigma, \mathbf{W})$-*interpretation* $\mathcal{I} = (\mathfrak{A}, \beta)$ consists of a $(\sigma, \mathbf{W})$-structure $\mathfrak{A}$ and an *assignment* $\beta\colon \mathsf{vars} \to U(\mathfrak{A})$.

An $\mathbb{S}$-*predicate collection* is a 4-tuple $(\mathbb{P}, \mathrm{ar}, \mathrm{type}, [\![\cdot]\!])$ where $\mathbb{P}$ is a countable set of *predicate names* and, to each $P \in \mathbb{P}$, ar assigns an *arity* $\mathrm{ar}(P) \in \mathbb{N}_{\geqslant 1}$, type assigns a *type* $\mathrm{type}(P) \in \mathbb{S}^{\mathrm{ar}(P)}$, and $[\![\cdot]\!]$ assigns a *semantics* $[\![P]\!] \subseteq \mathrm{type}(P)$. For the remainder of this section, fix an $\mathbb{S}$-predicate collection $(\mathbb{P}, \mathrm{ar}, \mathrm{type}, [\![\cdot]\!])$.

For every $S \in \mathbb{S}$ that is not a ring but just an abelian group, a $\mathbf{W}$-*product of type* S is either an element $s \in S$ or an expression of the form $\mathtt{w}(x_1, \dots, x_k)$ where $\mathtt{w} \in \mathbf{W}$ is of type S, $k = \mathrm{ar}(\mathtt{w})$, and $x_1, \dots, x_k$ are k pairwise distinct variables in vars. For every ring $S \in \mathbb{S}$, a $\mathbf{W}$-*product of type* S is an expression of the form $t_1 \cdot \dots \cdot t_\ell$ where $\ell \in \mathbb{N}_{\geqslant 1}$ and for each $i \in [\ell]$ either $t_i \in S$ or there exists a $\mathtt{w} \in \mathbf{W}$ with $\mathrm{type}(\mathtt{w}) = S$ and there exist $k \coloneqq \mathrm{ar}(\mathtt{w})$ pairwise distinct variables $x_1, \dots, x_k$ in vars such that $t_i = \mathtt{w}(x_1, \dots, x_k)$. By $\mathrm{vars}(p)$ we denote the set of all variables that occur in a $\mathbf{W}$-product p.

Example 5.2. Recall Example 5.1(a)–(d), and let x and y be variables. Examples of $\mathbf{W}$-products are $\mathtt{age}(x) \cdot \mathtt{length}(x, y)$, $\mathtt{embedding}(x)$, $\mathtt{price}(x) \cdot \mathtt{quantity}(x)$, and $\mathtt{w}(x, y)$. Below, in Definition 5.3, we will provide the formal definition of a logic (including notions of *formulas* and so-called $\mathbb{S}$-*terms*) which is capable of expressing the following statements.

(a) For each user x, the sum of the ages of the friends, multiplied by the length of the friendship, can be expressed as the $\mathbb{S}$-*term*

$$t_{\mathrm{agg}}(x) \coloneqq \sum \mathtt{age}(y) \cdot \mathtt{length}(x', y)\ .\ x' = x \wedge E(x', y).$$

(b) For vectors $u, v \in \mathbb{R}^k$, let $d(u, v)$ denote the Euclidean distance between u and v. We might want to use a formula $\varphi_{\mathrm{similar}}(x, y)$ expressing that the two k-dimensional vectors associated with persons x and y have Euclidean distance at most 1. To express this in our logic, we can add the rational field $(\mathbb{Q}, +, \cdot)$ to the collection $\mathbb{S}$ and use a predicate name P_{ED} of arity 3 and type $(\mathbb{R}^k, +) \times (\mathbb{R}^k, +) \times (\mathbb{Q}, +, \cdot)$ with $[\![P_{\mathrm{ED}}]\!] = \{(u, v, q) \in \mathbb{R}^k \times \mathbb{R}^k \times \mathbb{Q} \mid d(u, v) \leqslant q\}$. Then,

$$\varphi_{\mathrm{similar}}(x, y) \coloneqq P_{\mathrm{ED}}(\mathtt{embedding}(x), \mathtt{embedding}(y), 1)$$

is a formula with the desired meaning.

(c) Given a first-order formula $\varphi_{\mathrm{group}}(p)$ that defines products of a certain product group based on the structure of their transactions, we can describe the amount of money a customer c paid on the specified product group via the $\mathbb{S}$-*term*

$$t_{\mathrm{spending}}(c) \coloneqq \sum \mathtt{price}(i) \cdot \mathtt{quantity}(i)\ .\ \exists p \exists r \big(\varphi_{\mathrm{group}}(p) \wedge T(i, c, p, r)\big).$$

This term associates with every customer c the sum of the product of $\texttt{price}(i)$ and $\texttt{quantity}(i)$ for all transaction identifiers i for which there exists a product p and a retailer r such that the tuple (i, c, p, r) belongs to the transaction table and the product p belongs to the specified product group. The $\mathbb{S}$-term

$$t_{\text{sales}} := \sum \texttt{price}(i) \cdot \texttt{quantity}(i)\ .\ \exists c \exists p \exists r\big(\varphi_{\text{group}}(p) \\ \wedge T(i, c, p, r)\big)$$

specifies the amount *all* customers have paid on products from the product group.

We might want to select the "heavy hitters", i. e. all customers c for whom $t_{\text{spending}}(c) > 0.01 \cdot t_{\text{sales}}$ holds. In our logic, this is expressed by the formula

$$P_{>}(t_{\text{spending}}(c), 0.01 \cdot t_{\text{sales}}),$$

where $P_{>}$ is a predicate name of type $(\mathbb{Q}, +, \cdot) \times (\mathbb{Q}, +, \cdot)$ with $[\![P_{>}]\!] = \{(r, s) \in \mathbb{Q}^2 \mid r > s\}$.

(d) For each vertex x, the sum of the edge weights between x and its blue neighbours is specified by the $\mathbb{S}$-term

$$t_B(x) := \sum \texttt{w}(x', y).(x'{=}x \wedge E(x', y) \wedge B(y)).$$

We have designed the definition of the syntax of our logic in a way particularly suitable for formulating and proving the locality results that are crucial for obtaining our learning results. To obtain a more user-friendly syntax, i. e. which allows reading and constructing formulas in a more intuitive way, one could of course introduce syntactic sugar that allows explicitly writing statements of the form

- $\sum_y \texttt{age}(y) \cdot \texttt{length}(x, y)\ .\ E(x, y)$ instead of $\sum \texttt{age}(y) \cdot \texttt{length}(x', y)\ .\ x'{=}x \wedge E(x', y)$,
- $d(\texttt{embedding}(x), \texttt{embedding}(y)) \leqslant 1$ instead of $P_{\text{ED}}(\texttt{embedding}(x), \texttt{embedding}(y), 1)$, and
- $t_{\text{spending}}(c) > 0.01 \cdot t_{\text{sales}}$ instead of $P_{>}(t_{\text{spending}}(c), 0.01 \cdot t_{\text{sales}})$.

We now define the precise syntax and semantics of our weight-aggregation logic. FOWA

Definition 5.3 (FOWA($\mathbb{P}$)[σ, $\mathbb{S}$, $\mathbf{W}$])**.** For FOWA($\mathbb{P}$)[σ, $\mathbb{S}$, $\mathbf{W}$], the set of *formulas* and $\mathbb{S}$-*terms* is built according to the following rules.

(1) $x_1{=}x_2$ and $R(x_1, \ldots, x_k)$ are formulas for $x_1, x_2, \ldots, x_k \in \textsf{vars}$ and $R \in \sigma$ with $\text{ar}(R) = k$.

(2) If $\texttt{w} \in \mathbf{W}$, $S = \text{type}(\texttt{w})$, $s \in S$, $k = \text{ar}(\texttt{w})$, and $\bar{x} = (x_1, \ldots, x_k)$ is a tuple of k pairwise distinct variables, then $\big(s = \texttt{w}(\bar{x})\big)$ is a formula.

(3) If φ and ψ are formulas, then $\neg\varphi$ and $(\varphi \vee \psi)$ are also formulas.

(4) If φ is a formula and $x \in \mathsf{vars}$, then $\exists x\, \varphi$ is a formula.

(5) If φ is a formula, $\mathsf{w} \in \mathbf{W}$, $S = \text{type}(\mathsf{w})$, $s \in S$, $k = \text{ar}(\mathsf{w})$, and $\bar{x} = (x_1, \dots, x_k)$ is a tuple of k pairwise distinct variables, then $\big(s = \sum \mathsf{w}(\bar{x}).\varphi\big)$ is a formula.

(6) If $P \in \mathbb{P}$, $m = \text{ar}(P)$, and $t_1, \dots, t_m$ are $\mathbb{S}$-terms with $\text{type}(P) = \big(\text{type}(t_1), \dots, \text{type}(t_m)\big)$, then $\mathsf{P}(t_1, \dots, t_m)$ is a formula.

(7) For every $S \in \mathbb{S}$ and every $s \in S$, s is an $\mathbb{S}$-term of type S.

(8) For every $S \in \mathbb{S}$, every $\mathsf{w} \in \mathbf{W}$ of type S, and every tuple $(x_1, \dots, x_k)$ of $k := \text{ar}(\mathsf{w})$ pairwise distinct variables in vars, $\mathsf{w}(x_1, \dots, x_k)$ is an $\mathbb{S}$-term of type S.

(9) If t_1 and t_2 are $\mathbb{S}$-terms of the same type S, then $(t_1 + t_2)$ and $(t_1 - t_2)$ are also $\mathbb{S}$-terms of type S; furthermore, if S is a ring (and not just an abelian group), then also $(t_1 \cdot t_2)$ is an $\mathbb{S}$-term of type S.

(10) If φ is a formula, $S \in \mathbb{S}$, and p is a $\mathbf{W}$-product of type S, then $\sum p.\varphi$ is an $\mathbb{S}$-term of type S.

Let $\mathcal{I} = (\mathfrak{A}, \beta)$ be a $(\sigma, \mathbf{W})$-interpretation. For a formula or $\mathbb{S}$-term ξ from $\mathsf{FOWA}(\mathbb{P})[\sigma, \mathbb{S}, \mathbf{W}]$, the semantics $[\![\xi]\!]^{\mathcal{I}}$ is defined as follows.

(1) $[\![x_1{=}x_2]\!]^{\mathcal{I}} = 1$ if $\beta(x_1) = \beta(x_2)$, and $[\![x_1{=}x_2]\!]^{\mathcal{I}} = 0$ otherwise; $[\![R(x_1, \dots, x_k)]\!]^{\mathcal{I}} = 1$ if $\big(\beta(x_1), \dots, \beta(x_k)\big) \in R(\mathfrak{A})$, and $[\![R(x_1, \dots, x_k)]\!]^{\mathcal{I}} = 0$ otherwise.

(2) $\Big[\!\!\Big[\big(s = \mathsf{w}(\bar{x})\big)\Big]\!\!\Big]^{\mathcal{I}} = 1$ if $s = \mathsf{w}^{\mathfrak{A}}\big(\beta(x_1), \dots, \beta(x_k)\big)$, and $\Big[\!\!\Big[\big(s = \mathsf{w}(\bar{x})\big)\Big]\!\!\Big]^{\mathcal{I}} = 0$ otherwise.

(3) $[\![\neg\varphi]\!]^{\mathcal{I}} = 1 - [\![\varphi]\!]^{\mathcal{I}}$ and $[\![(\varphi \vee \psi)]\!] = \max\big\{[\![\varphi]\!]^{\mathcal{I}}, [\![\psi]\!]^{\mathcal{I}}\big\}$.

(4) $[\![\exists x\, \varphi]\!]^{\mathcal{I}} = \max\big\{[\![\varphi]\!]^{\mathcal{I}\frac{v}{x}} \mid v \in U(\mathfrak{A})\big\}$.

(5) $\Big[\!\!\Big[\big(s = \sum \mathsf{w}(\bar{x}).\varphi\big)\Big]\!\!\Big]^{\mathcal{I}} = 1$ if $s = \sum_S \big\{\mathsf{w}^{\mathfrak{A}}(\bar{v}) \mid \bar{v} = (v_1, \dots, v_k) \in \big(U(\mathfrak{A})\big)^k$ with $[\![\varphi]\!]^{\mathcal{I}\frac{v_1, \dots, v_k}{x_1, \dots, x_k}} = 1\big\}$, and $\Big[\!\!\Big[\big(s = \sum \mathsf{w}(\bar{x}).\varphi\big)\Big]\!\!\Big]^{\mathcal{I}} = 0$ otherwise. As usual, by convention, we let $\sum_S X = 0_S$ if $X = \emptyset$.

(6) $[\![\mathsf{P}(t_1, \dots, t_m)]\!]^{\mathcal{I}} = 1$ if $\big([\![t_1]\!]^{\mathcal{I}}, \dots, [\![t_m]\!]^{\mathcal{I}}\big) \in [\![\mathsf{P}]\!]$, and $[\![\mathsf{P}(t_1, \dots, t_m)]\!]^{\mathcal{I}} = 0$ otherwise.

(7) $[\![s]\!]^{\mathcal{I}} = s$ for $s \in S$ for some $S \in \mathbb{S}$.

(8) $[\![\mathsf{w}(x_1, \dots, x_k)]\!]^{\mathcal{I}} = \mathsf{w}^{\mathfrak{A}}\big(\beta(x_1), \dots, \beta(x_k)\big)$.

(9) $[\![(t_1 * t_2)]\!]^{\mathcal{I}} = [\![t_1]\!]^{\mathcal{I}} *_S [\![t_2]\!]^{\mathcal{I}}$, for $* \in \{+, -, \cdot\}$.

(10) $[\![\sum p.\varphi]\!]^{\mathcal{I}} = \sum_S \{ [\![p]\!]^{\mathcal{I}\frac{v_1,\dots,v_k}{x_1,\dots,x_k}} \mid v_1,\dots,v_k \in U(\mathfrak{A}), [\![\varphi]\!]^{\mathcal{I}\frac{v_1,\dots,v_k}{x_1,\dots,x_k}} = 1 \}$, where $\text{vars}(p) = \{x_1,\dots,x_k\}$ and $[\![p]\!]^{\mathcal{I}} = [\![t_1]\!]^{\mathcal{I}} \cdot_S \cdots \cdot_S [\![t_\ell]\!]^{\mathcal{I}}$ if $p = t_1 \cdot \cdots \cdot t_\ell$ is of type S.

An *expression* is a formula or an $\mathbb{S}$-term. The set $\text{vars}(\xi)$ of an expression ξ is defined as the set of all variables in vars that occur in ξ. The *free variables* $\text{free}(\xi)$ of ξ are inductively defined as follows.

(1) $\text{free}(x_1{=}x_2) = \{x_1, x_2\}$ and $\text{free}\big(R(x_1,\dots,x_k)\big) = \{x_1,\dots,x_k\}$.

(2) $\text{free}\Big(\big(s = w(x_1,\dots,x_k)\big)\Big) = \{x_1,\dots,x_k\}$.

(3) $\text{free}(\neg\varphi) = \text{free}(\varphi)$ and $\text{free}(\varphi \vee \psi) = \text{free}(\varphi) \cup \text{free}(\psi)$.

(4) $\text{free}(\exists x\, \varphi) = \text{free}(\varphi) \setminus \{x\}$.

(5) $\text{free}\Big(\big(s = \sum w(x_1,\dots,x_k).\varphi\big)\Big) = \text{free}(\varphi) \setminus \{x_1,\dots,x_k\}$,

(6) $\text{free}\big(P(t_1,\dots,t_m)\big) = \bigcup_{i=1}^m \text{free}(t_i)$.

(7) $\text{free}(s) = \emptyset$ for $s \in S$ for some $S \in \mathbb{S}$.

(8) $\text{free}\big(w(x_1,\dots,x_k)\big) = \{x_1,\dots,x_k\}$.

(9) $\text{free}\big((t_1 * t_2)\big) = \text{free}(t_1) \cup \text{free}(t_2)$ for $* \in \{+,-,\cdot\}$.

(10) $\text{free}(\sum p.\varphi) = \text{free}(\varphi) \setminus \text{vars}(p)$.

As for the logics introduced in Section 2.3, we write $\xi(x_1,\dots,x_k)$ to indicate that $\text{free}(\xi) \subseteq \{x_1,\dots,x_k\}$. A *sentence* is a formula without free variables and a *ground* $\mathbb{S}$-*term* is an $\mathbb{S}$-term without free variables.

For a formula φ and a $(\sigma, \mathbf{W})$-interpretation $\mathcal{I}$, we write $\mathcal{I} \models \varphi$ to indicate that $[\![\varphi]\!]^{\mathcal{I}} = 1$. Likewise, $\mathcal{I} \not\models \varphi$ indicates that $[\![\varphi]\!]^{\mathcal{I}} = 0$. For a formula φ, a $(\sigma, \mathbf{W})$-structure $\mathfrak{A}$, and a tuple $\bar{v} = (v_1,\dots,v_k) \in \big(U(\mathfrak{A})\big)^k$, we write $\mathfrak{A} \models \varphi[\bar{v}]$ or $(\mathfrak{A}, \bar{v}) \models \varphi$ to indicate that $(\mathfrak{A}, \beta) \models \varphi$ for all assignments β with $\beta(x_i) = v_i$ for all $i \in [k]$. Furthermore, we set $[\![\varphi(\bar{v})]\!]^{\mathfrak{A}} := 1$ if $\mathfrak{A} \models \varphi[\bar{v}]$, and $[\![\varphi(\bar{v})]\!]^{\mathfrak{A}} := 0$ otherwise. Similarly, for an $\mathbb{S}$-term $t(\bar{x})$, we write $t^{\mathfrak{A}}[\bar{v}]$ to denote $[\![t]\!]^{\mathcal{I}}$.

Next, we introduce the fragments that we will use in Chapter 6 for our learning problems on weighted structures.

FOWA_1, FOW_1

Definition 5.4 ($\mathsf{FOWA}_1(\mathbb{P})[\sigma, \mathbb{S}, \mathbf{W}]$ and $\mathsf{FOW}_1(\mathbb{P})[\sigma, \mathbb{S}, \mathbf{W}]$)**.** The set of *formulas* and $\mathbb{S}$-*terms* of the logic $\mathsf{FOWA}_1(\mathbb{P})[\sigma, \mathbb{S}, \mathbf{W}]$ is built according to the same rules as for the logic $\mathsf{FOWA}(\mathbb{P})[\sigma, \mathbb{S}, \mathbf{W}]$, with the following restrictions:

$(5)_1$ rule (5) can only be applied if S is finite,

$(6)_1$ rule (6) can only be applied if $|\text{free}(t_1) \cup \cdots \cup \text{free}(t_m)| \leqslant 1$.

The logic $\mathsf{FOW}_1(\mathbb{P})[\sigma, \mathbb{S}, \mathbf{W}]$ is the restriction of $\mathsf{FOWA}_1(\mathbb{P})[\sigma, \mathbb{S}, \mathbf{W}]$ where rule (10) cannot be applied.

Note that FO is the restriction of FOW_1 where only rules (1), (3), and (4) can be applied. As usual, we write $(\varphi \wedge \psi)$ and $\forall x\, \varphi$ as shorthands for $\neg(\neg\varphi \vee \neg\psi)$ and $\neg\exists x \neg\varphi$. The *quantifier rank* $\mathrm{qr}(\xi)$ of an $\mathsf{FOWA}(\mathbb{P})[\sigma, \mathbb{S}, \mathbf{W}]$-expression ξ is defined as the maximum nesting depth of constructs using rules (4) and (5) in order to construct ξ. The *aggregation depth* $\mathrm{d_{ag}}(\xi)$ of ξ is defined as the maximum nesting depth of term constructions using rule (10) in order to construct ξ.

Remark 5.5. FOW_1 can be viewed as an extension of first-order logic with modulo-counting quantifiers. Let $\mathbb{S}$ contain the abelian group $(\mathbb{Z}/m\mathbb{Z}, +)$ for some $m \geqslant 2$, and let $\mathbf{W}$ contain a unary weight symbol one_m of type $\mathbb{Z}/m\mathbb{Z}$ such that $\mathsf{one}_m^{\mathfrak{A}}(v) = 1$ for all $v \in U(\mathfrak{A})$. Then the modulo-m-counting quantifier $\exists^{i \bmod m} x\, \varphi$, stating that the number of interpretations for x that satisfy φ is congruent to i modulo m, can be expressed in $\mathsf{FOW}_1(\mathbb{P})[\sigma, \mathbb{S}, \mathbf{W}]$ via $\big(i = \sum \mathsf{one}_m(x).\varphi\big)$.

FOWA_1 can be viewed as an extension of the logic FOC_1 that we described in Section 2.3. Let $\mathbb{S}$ contain the integer ring $(\mathbb{Z}, +, \cdot)$ and let $\mathbf{W}$ contain a unary weight symbol one of type $\mathbb{Z}$ such that $\mathsf{one}^{\mathfrak{A}}(v) = 1$ for all $v \in U(\mathfrak{A})$. Then the counting term $\#(x_1, \ldots, x_k).\varphi$ of FOC_1, which counts the number of interpretations for $(x_1, \ldots, x_k)$ that satisfy φ, can be expressed in $\mathsf{FOWA}_1(\mathbb{P})[\sigma, \mathbb{S}, \mathbf{W}]$ via the $\mathbb{S}$-term $\sum p.\varphi$ for $p := \mathsf{one}(x_1) \cdot \cdots \cdot \mathsf{one}(x_k)$.

Let us mention, again, that we have designed the precise definition of the syntax of our logic in a way particularly suitable for formulating and proving the locality results that are crucial for obtaining our learning results. To obtain a more user-friendly syntax, i. e. which allows reading and constructing formulas in a more intuitive way, it would of course make sense to introduce syntactic sugar that allows explicitly writing statements of the form $\#(x_1, \ldots, x_k).\varphi$ instead of $\sum p.\varphi$ for $p := \mathsf{one}(x_1) \cdot \cdots \cdot \mathsf{one}(x_k)$ and $\big(\#(x).\varphi \equiv i \bmod m\big)$ or $\exists^{i \bmod m} x\, \varphi$ instead of $\big(i = \sum \mathsf{one}_m(x).\varphi\big)$. For this, one would tacitly assume that $\mathbb{S}$ contains $(\mathbb{Z}, +, \cdot)$ and $(\mathbb{Z}/m\mathbb{Z}, +)$, and $\mathbf{W}$ contains the unary weight symbols one of type $\mathbb{Z}$ and one_m of type $\mathbb{Z}/m\mathbb{Z}$, where $\mathsf{one}^{\mathfrak{A}}(v) = \mathsf{one}_m^{\mathfrak{A}} = 1$ for every $v \in U(\mathfrak{A})$ and every considered $(\sigma, \mathbf{W})$-structure $\mathfrak{A}$.

To close this section, we return to the running examples from Examples 5.1 and 5.2.

Example 5.6. We use the syntactic sugar introduced at the end of Remark 5.5.

(a) Let $\mathsf{P}_{\geqslant}$ be a binary predicate in $\mathbb{P}$ of type $\mathbb{Q} \times \mathbb{Q}$ that is interpreted by the $\geqslant$-relation. The term

$$t_{\#\text{over}40}(x) := \#(y).\Big(E(x,y) \wedge \mathsf{P}_{\geqslant}\big(\mathsf{age}(y), 40\big)\Big)$$

specifies the number of friends of x that are at least 40 years old. Then, the $\mathsf{FOWA}_1(\mathbb{P})[\sigma, \mathbb{S}, \mathbf{W}]$-formula

$$\varphi(x) := \mathsf{P}_{\geqslant}\big(2 \cdot t_{\#\text{over}40}(x)\,,\, \#(y).E(x,y)\big)$$

specifies those users for whom at least half of their friends are at least 40 years old.

(b) The term $t_{\#\text{follows}}(x) := \#(y).F(x,y)$ specifies the number of users y followed by person x. The term $t_{\text{sum}}(x) := \sum \texttt{embedding}(y).F(x,y)$ specifies the sum of the vectors associated with all users y followed by x. To describe the users x whose embedding is δ-close (for some fixed $\delta > 0$) to the average of the embeddings of users they follow[3], we might want to use a formula $\varphi_{\text{close}}(x)$ of the form

$$d\left(\texttt{embedding}(x),\ \frac{1}{t_{\#\text{follows}}(x)} \cdot t_{\text{sum}}(x)\right) \ < \ \delta.$$

We can describe this in $\mathsf{FOWA}_1(\mathbb{P})[\sigma, \mathbb{S}, \mathbf{W}]$ by the formula

$$\varphi_{\text{close}}(x) \ := \ P_{\text{dist}<\delta}\big(\texttt{embedding}(x), t_{\#\text{follows}}(x), t_{\text{sum}}(x)\big),$$

where $P_{\text{dist}<\delta}$ is a ternary predicate in $\mathbb{P}$ of type $\mathbb{R}^k \times \mathbb{Z} \times \mathbb{R}^k$ consisting of all triples $(\bar{v}, \ell, \bar{w})$ with $\ell > 0$ and $d(\bar{v}, \frac{1}{\ell} \cdot \bar{w}) < \delta$.

(c) The number of consumers who bought products p from the product group defined by $\varphi_{\text{group}}(p)$ is specified by the $\mathbb{S}$-term

$$t_{\#\text{cons}} \ := \ \sum \texttt{one}(c) \ . \ \exists i \, \exists p \, \exists r \, \big(\varphi_{\text{group}}(p) \wedge T(i,c,p,r)\big).$$

Using the syntactic sugar described above, this $\mathbb{S}$-term can be expressed via $\#(c).\ \exists i \, \exists p \, \exists r \, \big(\varphi_{\text{group}}(p) \wedge T(i,c,p,r)\big)$. The consumers c who spent at least as much as the average consumer on the products p satisfying $\varphi_{\text{group}}(p)$ can be described by the formula

$$\varphi_{\text{spending}}(c) \ := \ P_{\geqslant}\Big(\big(t_{\text{spending}}(c) \ \cdot \ t_{\#\text{cons}}\big) \ , \ t_{\text{sales}}\Big),$$

where $P_{\geqslant}$ is the binary predicate from (a). To improve readability, one could introduce syntactic sugar that allows expressing this as $t_{\text{spending}}(c) \geqslant t_{\text{sales}}/t_{\#\text{cons}}$. The formula $\varphi_{\text{spending}}(c)$ belongs to the fragment $\mathsf{FOWA}_1(\mathbb{P})[\sigma, \mathbb{S}, \mathbf{W}]$.

(d) Recall the term $t_B(x)$ introduced in Example 5.2 (d) that specifies the sum of the weights of edges between x and its blue neighbours. Let $t_R(x)$ be a similar term summing up the weights of edges between x and its red neighbours. Using the syntactic sugar introduced at the end of Example 5.2, this can be described as $\sum_y w(x,y).\big(E(x,y) \wedge R(y)\big)$. To specify the vertices x that have exactly 5 red neighbours, we can use the formula $\varphi_{5\,\text{red}}(x) := \Big(5 = \#(y).\big(E(x,y) \wedge R(y)\big)\Big)$. Let us now assume we are given a particular set $H \subseteq \mathbb{R}^{2k}$ and we want to specify the vertices x that have exactly 5 red neighbours and for which, in

3 Depending on the target of the embeddings, this could mean that the user mostly follows users with a very similar personality or political leaning.

addition, the $2k$-ary vector obtained by concatenating the k-ary vectors computed by summing up the weights of edges between x and its blue neighbours and by summing up the weights of edges between x and its red neighbours belongs to H. To express this, we can use a binary predicate P of type $\mathbb{R}^k \times \mathbb{R}^k$ with $[\![\mathsf{P}]\!] = \big\{(\bar{u},\bar{v}) \in \mathbb{R}^k \times \mathbb{R}^k \mid (u_1,\dots,u_k,v_1,\dots,v_k) \in H\big\}$. Then, the $\mathsf{FOWA}_1(\mathbb{P})[\sigma,\mathbb{S},\mathbf{W}]$-formula $\psi(x) \coloneqq \varphi_{5\,\mathrm{red}}(x) \wedge \mathsf{P}\big(t_B(x), t_G(x)\big)$ specifies the vertices x we are interested in.

In the following sections, we provide locality properties of FOW_1 and FOWA_1 that are similar to well-known locality properties of FO and to locality properties of FOC_1 achieved in [56]. This includes *Feferman-Vaught decompositions* and a *Gaifman normal form* for FOW_1 in Sections 5.2 and 5.3, and a *localisation theorem* for the more expressive logic FOWA_1 in Section 5.4.

For the remainder of this chapter, let us fix a signature σ, a collection $\mathbb{S}$ of rings and/or abelian groups, a finite set $\mathbf{W}$ of weight symbols, and an $\mathbb{S}$-predicate collection $(\mathbb{P}, \mathrm{ar}, \mathrm{type}, [\![\cdot]\!])$.

The notion of *local formulas* for FOWA is defined analogously to local FOCN-formulas in Section 2.4. Let $r \in \mathbb{N}$. A $\mathsf{FOWA}(\mathbb{P})[\sigma,\mathbb{S},\mathbf{W}]$-formula $\varphi(\bar{x})$ with free variables $\bar{x} = (x_1,\dots,x_k)$ is r-*local (around $\bar{x}$)* if for every $(\sigma,\mathbf{W})$-structure $\mathfrak{A}$ and all $\bar{v} \in \big(U(\mathfrak{A})\big)^k$, we have $\mathfrak{A} \models \varphi[\bar{v}] \iff \mathcal{N}_r^{\mathfrak{A}}(\bar{v}) \models \varphi[\bar{v}]$. A formula is *local* if it is r-local for some $r \in \mathbb{N}$.

Localisation

The r-*localisation* $\varphi^{(r)}$ of an $\mathsf{FOWA}(\mathbb{P})[\sigma,\mathbb{S},\mathbf{W}]$-formula $\varphi(\bar{x})$ is the formula obtained from φ by replacing every subformula of the form $\exists y\, \varphi'$ with the formula $\exists y\, \big(\varphi' \wedge \mathrm{dist}(\bar{x};y) \leqslant r\big)$, replacing every subformula $\big(s = \sum \mathsf{w}(\bar{y}).\varphi'\big)$, for $\bar{y} = (y_1,\dots,y_k)$, with the formula $\big(s = \sum \mathsf{w}(\bar{y}).(\varphi' \wedge \bigwedge_{j=1}^{k} \mathrm{dist}(\bar{x};y_j) \leqslant r)\big)$, and replacing every $\mathbb{S}$-term of the form $\sum p.\varphi'$ with the $\mathbb{S}$-term $\sum p.\big(\varphi' \wedge \bigwedge_{j=1}^{k} \mathrm{dist}(\bar{x};y_j) \leqslant r\big)$, where $\{y_1,\dots,y_k\} = \mathrm{vars}(p)$. The resulting formula $\varphi^{(r)}(\bar{x})$ is r-local.

5.2 FEFERMAN-VAUGHT DECOMPOSITIONS FOR FOW_1

We start this section with an introduction of Feferman-Vaught decompositions. Let $X, Y \notin \sigma$ be the two unary relation symbols used in the disjoint-sum construction in Chapter 2.

Feferman-Vaught decomposition

Definition 5.7. Let L be a subset of $\mathsf{FOWA}(\mathbb{P})[\sigma,\mathbb{S},\mathbf{W}]$. Let $k,\ell \in \mathbb{N}$ and let $\bar{x} = (x_1,\dots,x_k)$, $\bar{y} = (y_1,\dots,y_\ell)$ be tuples of $k+\ell$ pairwise distinct variables. Let φ be an $\mathsf{FOWA}(\mathbb{P})[\sigma',\mathbb{S},\mathbf{W}]$-formula with $\sigma' \coloneqq \sigma \cup \{X,Y\}$ and $\mathrm{free}(\varphi) \subseteq \{x_1,\dots,x_k,y_1,\dots,y_\ell\}$. A *Feferman-Vaught decomposition* of φ in L w.r.t. $(\bar{x};\bar{y})$ is a finite, non-empty set Δ of tuples of the form (α,β) where $\alpha,\beta \in L$, $\mathrm{free}(\alpha) \subseteq \{x_1,\dots,x_k\}$, and $\mathrm{free}(\beta) \subseteq \{y_1,\dots,y_\ell\}$, such that the following is true for all $(\sigma,\mathbf{W})$-structures $\mathfrak{A},\mathfrak{B}$ with $U(\mathfrak{A}) \cap U(\mathfrak{B}) = \emptyset$ and all $\bar{v} \in \big(U(\mathfrak{A})\big)^k$, $\bar{w} \in \big(U(\mathfrak{B})\big)^\ell$:

$\mathfrak{A} \oplus \mathfrak{B} \models \varphi[\bar{v}, \bar{w}]$ if and only if there exists $(\alpha, \beta) \in \Delta$ such that $\mathfrak{A} \models \alpha[\bar{v}]$ and $\mathfrak{B} \models \beta[\bar{w}]$.

In our first result of this section, we provide Feferman-Vaught decompositions for FOW_1.

Theorem 5.8 (Feferman-Vaught decompositions for FOW_1).
Let $k, \ell \in \mathbb{N}$ and let $\bar{x} = (x_1, \dots, x_k)$, $\bar{y} = (y_1, \dots, y_\ell)$ be tuples of $k+\ell$ pairwise distinct variables. For every $\mathsf{FOW}_1(\mathbb{P})[\sigma', \mathbb{S}, \mathbf{W}]$-formula φ with $\text{free}(\varphi) \subseteq \{x_1, \dots, x_k, y_1, \dots, y_\ell\}$, there exists a Feferman-Vaught decomposition Δ in L of φ w.r.t. $(\bar{x}; \bar{y})$, where $\mathsf{L} := \mathsf{L}_\varphi$ is the class of all $\mathsf{FOW}_1(\mathbb{P})[\sigma, \mathbb{S}, \mathbf{W}]$-formulas of quantifier rank at most $\text{qr}(\varphi)$ which use only those $P \in \mathbb{P}$ and $S \in \mathbb{S}$ that occur in φ and only those $\mathbb{S}$-terms that occur in φ or that are of the form s for an $s \in S$ with $S \in \mathbb{S}$ where S is finite and occurs in φ.

Furthermore, there is an algorithm that computes Δ upon input of $\varphi, \bar{x}, \bar{y}$.

The proof proceeds similarly to the proof of the Feferman-Vaught decomposition for first-order logic with modulo-counting quantifiers in [70]. Before presenting the proof, let us formulate a straightforward corollary of Theorem 5.8.

Corollary 5.9. *Let $k, \ell \in \mathbb{N}$ and let $\bar{x} = (x_1, \dots, x_k)$, $\bar{y} = (y_1, \dots, y_\ell)$ be tuples of $k+\ell$ pairwise distinct variables. Upon input of an $r \in \mathbb{N}$ and an r-local $\mathsf{FOW}_1(\mathbb{P})[\sigma, \mathbb{S}, \mathbf{W}]$-formula $\varphi(\bar{x}, \bar{y})$, one can compute a finite, non-empty set Δ of pairs $\big(\alpha(\bar{x}), \beta(\bar{y})\big)$ of L-formulas, where L is the class of all r-localisations of formulas in the class L_φ of Theorem 5.8, such that the following two formulas are equivalent:*

- $\text{dist}(\bar{x}; \bar{y}) > 2r{+}1 \;\wedge\; \varphi(\bar{x}, \bar{y})$
- $\text{dist}(\bar{x}; \bar{y}) > 2r{+}1 \;\wedge\; \bigvee_{(\alpha,\beta)\in\Delta} \big(\alpha(\bar{x}) \wedge \beta(\bar{y})\big)$

The remainder of this section is devoted to the proofs of Theorem 5.8 and Corollary 5.9. In the proof of Theorem 5.8, we will use the following result that allows us to turn a Feferman-Vaught decomposition into one where the αs are mutually exclusive.

Lemma 5.10. *Let L be a subset of $\mathsf{FOWA}(\mathbb{P})[\sigma, \mathbb{S}, \mathbf{W}]$. Let $k, \ell \in \mathbb{N}$ and let $\bar{x} = (x_1, \dots, x_k)$, $\bar{y} = (y_1, \dots, y_\ell)$ be tuples of $k+\ell$ pairwise distinct variables. Let Δ be a finite, non-empty set of tuples of the form (α, β) where $\alpha, \beta \in \mathsf{L}$, $\text{free}(\alpha) \subseteq \{x_1, \dots, x_k\}$, and $\text{free}(\beta) \subseteq \{y_1, \dots, y_\ell\}$.*

Then there exists a finite, non-empty set $\hat{\Delta}$ of tuples of the form $(\hat{\alpha}, \hat{\beta})$, where $\hat{\alpha}, \hat{\beta} \in \mathsf{L}$, $\text{free}(\hat{\alpha}) \subseteq \{x_1, \dots, x_k\}$, and $\text{free}(\hat{\beta}) \subseteq \{y_1, \dots, y_\ell\}$, with the following two properties.

1. *The $\hat{\alpha}$s are mutually exclusive, i.e., for every two distinct $(\hat{\alpha}_1, \hat{\beta}_1)$ and $(\hat{\alpha}_2, \hat{\beta}_2)$ in $\hat{\Delta}$, the formula $(\hat{\alpha}_1 \wedge \hat{\alpha}_2)$ is unsatisfiable.*

2. *The sets Δ and $\hat{\Delta}$ are equivalent, i.e., for all $(\sigma, \mathbf{W})$-structures $\mathfrak{A}, \mathfrak{B}$ with $U(\mathfrak{A}) \cap U(\mathfrak{B}) = \emptyset$ and all $\bar{v} \in \big(U(\mathfrak{A})\big)^k$, $\bar{w} \in \big(U(\mathfrak{B})\big)^\ell$, there exists $(\alpha, \beta) \in \Delta$ such that $\mathfrak{A} \models \alpha[\bar{v}]$ and $\mathfrak{B} \models \beta[\bar{w}]$ if and only if there exists $(\hat{\alpha}, \hat{\beta}) \in \hat{\Delta}$ such that $\mathfrak{A} \models \hat{\alpha}[\bar{v}]$ and $\mathfrak{B} \models \hat{\beta}[\bar{w}]$.*

Furthermore, there is an algorithm that computes $\hat{\Delta}$ upon input of Δ.

Proof. Let $A := \{\alpha \mid \text{there exists } \beta \text{ such that } (\alpha, \beta) \in \Delta\}$ and, for every $\alpha \in A$, let $B(\alpha) := \{\beta \mid (\alpha, \beta) \in \Delta\}$. For every $I \subseteq A$, let $\alpha_I := \bigwedge_{\alpha \in I} \alpha \wedge \bigwedge_{\alpha \in A \setminus I} \neg\alpha$ and $\beta_I := \bigvee_{\alpha \in I} \bigvee_{\beta \in B(\alpha)} \beta$. We set

$$\hat{\Delta} := \big\{(\alpha_I, \beta_I) \mid \emptyset \neq I \subseteq A\big\}.$$

It is straightforward to verify that $\hat{\Delta}$ meets all requirements from Lemma 5.10. $\square$

With this result at hand, we can now prove Theorem 5.8.

Proof of Theorem 5.8. We proceed by induction on the construction of φ, and we use an arbitrary unsatisfiable formula $\bot$ (e. g., $\bot := \exists z \neg z{=}z$) and an arbitrary tautology $\top$ (e. g., $\top := \neg\bot$).

For the induction base, we consider formulas built according to the rules (1), (2), and $(6)_1$ of Definitions 5.3 and 5.4. Rule (1) can be handled in exactly the same way as in the traditional Feferman-Vaught construction for first-order logic (cf., e. g., [37, 49, 74]). That is, for an atomic formula φ, we proceed as follows.

- If $\varphi = X(x_i)$ for some $i \in [k]$ or $\varphi = Y(y_j)$ for some $j \in [\ell]$, then $\Delta := \big\{(\top, \top)\big\}$.
- If $\varphi = X(y_j)$ for some $j \in [\ell]$, $\varphi = Y(x_i)$ for some $i \in [k]$, or $\text{free}(\varphi)$ contains variables from both $\{x_1, \dots, x_k\}$ and $\{y_1, \dots, y_\ell\}$, then $\Delta := \big\{(\bot, \bot)\big\}$.
- If $\text{free}(\varphi) \subseteq \{x_1, \dots, x_k\}$ and X and Y do not occur in φ, then $\Delta := \big\{(\varphi, \top)\big\}$.
- If $\text{free}(\varphi) \subseteq \{y_1, \dots, y_\ell\}$ and X and Y do not occur in φ, then $\Delta := \big\{(\top, \varphi)\big\}$.

For rule (2), let φ be of the form $\big(s = w(z_1, \dots, z_m)\big)$. If $\{z_1, \dots, z_m\} \subseteq \{x_1, \dots, x_k\}$, we can choose $\Delta := \big\{(\varphi, \top)\big\}$. If $\{z_1, \dots, z_m\} \subseteq \{y_1, \dots, y_\ell\}$, we can choose $\Delta := \big\{(\top, \varphi)\big\}$. Otherwise, we know that $\{z_1, \dots, z_m\}$ contains variables from $\bar{x}$ and variables from $\bar{y}$. If $s = 0_S$, we can choose $\Delta := \big\{(\top, \top))\big\}$, and otherwise, we can choose $\Delta := \big\{(\bot, \bot)\big\}$. It is straightforward to verify that Δ is a Feferman-Vaught decomposition in $\mathsf{FOW}_1(\mathbb{P})[\sigma, S, \mathbf{W}]$ of φ w.r.t. $(\bar{x}; \bar{y})$. For rule $(6)_1$, let φ be of the form $P(t_1, \dots, t_m)$, where $P \in \mathbb{P}$ and $t_1, \dots, t_m$ are S-terms. We know that each t_i is built using the rules (7)–(9), and that there is one variable z such that $\text{vars}(t_i) \subseteq \{z\}$ for all $i \in [m]$. Thus, if $z \in \{x_1, \dots, x_k\}$, we can choose $\Delta \coloneqq \{(\varphi, \top)\}$; and if $z \in \{y_1, \dots, y_\ell\}$, we can choose $\Delta \coloneqq \{(\top, \varphi)\}$.

For the induction step, we consider formulas built according to the rules (3), (4), and $(5)_1$ of Definitions 5.3 and 5.4. Rules (3) and (4) can be handled in exactly the same way as for first-order logic (cf., e. g., [37, 49, 74]). If φ is of the form $\varphi_1 \vee \varphi_2$, then, by the induction hypothesis, there are Feferman-Vaught decompositions Δ_1 and Δ_2 in L of φ_1 and φ_2 w.r.t. $(\bar{x}; \bar{y})$, so we can choose $\Delta \coloneqq \Delta_1 \cup \Delta_2$. If φ is of the form $\neg\psi$ and $\Delta' = \{(\alpha_1, \beta_1), \dots, (\alpha_m, \beta_m)\}$ is a Feferman-Vaught decomposition in L of ψ w.r.t. $(\bar{x}; \bar{y})$, then we can set $\Delta \coloneqq \{(\alpha_I, \beta_I) \mid I \subseteq [m]\}$ with $\alpha_I \coloneqq \bigwedge_{i \in I} \neg\alpha_i$ and $\beta_I \coloneqq \bigwedge_{i \in [m] \setminus I} \neg\beta_i$. If φ is of the form $\exists z\, \psi$, then, by the induction hypothesis, we can compute Feferman-Vaught decompositions Δ_1 and Δ_2 in L of ψ w.r.t. $(\bar{x}z; \bar{y})$ and $(\bar{x}; \bar{y}z)$, so we can choose $\Delta \coloneqq \{(\exists z\, \alpha, \beta) \mid (\alpha, \beta) \in \Delta_1\} \cup \{(\alpha, \exists z\, \beta) \mid (\alpha, \beta) \in \Delta_2\}$. One can easily verify that Δ is a Feferman-Vaught decomposition in L of φ w.r.t. $(\bar{x}; \bar{y})$ in all the above-mentioned cases.

For rule $(5)_1$, we proceed similarly to the case of modulo-counting quantifiers in [70]. Let φ be of the form $(s = \sum \mathsf{w}(\bar{z}).\psi)$, for a tuple of variables $\bar{z} = (z_1, \dots, z_m)$ and a weight symbol $\mathsf{w} \in \mathbf{W}$ whose type $S \coloneqq \mathrm{type}(\mathsf{w})$ is finite. For every $i \in S$, let

$$\chi_i \coloneqq \Big(i = \sum \mathsf{w}(\bar{z}).\big(\psi \wedge \bigwedge_{j=1}^{m} X(z_j)\big)\Big)$$

and

$$\gamma_i \coloneqq \Big(i = \sum \mathsf{w}(\bar{z}).\big(\psi \wedge \bigwedge_{j=1}^{m} Y(z_j)\big)\Big).$$

Let $I \coloneqq \{(i_1, i_2) \in S \times S \mid i_1 +_S i_2 = s\}$. It is straightforward to see that for all $(\sigma, \mathbf{W})$-structures $\mathfrak{A}$ and $\mathfrak{B}$ with $U(\mathfrak{A}) \cap U(\mathfrak{B}) = \emptyset$ and all $\bar{v} \in (U(\mathfrak{A}))^k$ and $\bar{w} \in (U(\mathfrak{B}))^\ell$, we have $(\mathfrak{A} \oplus \mathfrak{B}, \bar{v}\bar{w}) \models (s = \sum \mathsf{w}(\bar{z}).\psi)$ if and only if $(\mathfrak{A} \oplus \mathfrak{B}, \bar{v}\bar{w}) \models \bigvee_{(i,j) \in I}(\chi_i \wedge \gamma_j)$. Since $(\chi_i \wedge \gamma_j)$ is equivalent to $\neg(\neg\chi_i \vee \neg\gamma_j)$, and we already know how to handle formulas built using rule (3), it remains to show the following.

Claim. For every $i \in S$, one can compute Feferman-Vaught decompositions Δ_{χ_i} and Δ_{γ_i} in $\mathsf{FOW}_1(\mathbb{P})[\sigma, S, \mathbf{W}]$ of χ_i and γ_i w.r.t. $(\bar{x}; \bar{y})$.

Proof. Let $i \in S$. We show how to construct Δ_{γ_i}; the construction of Δ_{χ_i} is analogous. By the induction hypothesis, we can construct a Feferman-Vaught decomposition in $\mathsf{FOW}_1(\mathbb{P})[\sigma, S, \mathbf{W}]$ of ψ w.r.t. $(\bar{x}; \bar{y}\bar{z})$. Using Lemma 5.10, we can turn this decomposition into a Feferman-Vaught decomposition Δ_ψ where the αs are mutually exclusive, i. e., for every two distinct $(\alpha, \beta), (\alpha', \beta') \in \Delta_\psi$, the formula $(\alpha \wedge \alpha')$ is unsatisfiable.

Let $\Delta'_{\gamma_i} \coloneqq \{(\alpha, (i = \sum \mathsf{w}(\bar{z}).\beta) \mid (\alpha, \beta) \in \Delta_\psi\}$. If $i \neq 0_S$, then we let $\Delta_{\gamma_i} \coloneqq \Delta'_{\gamma_i}$. Otherwise, if $i = 0_S$, we let $\Delta_{\gamma_i} \coloneqq \Delta'_{\gamma_i} \cup \{(\bigwedge_{\alpha \in A} \neg\alpha, \top)\}$, where $A \coloneqq \{\alpha \mid \text{there exists } \beta \text{ such that } (\alpha, \beta) \in \Delta_\psi\}$.

It remains to verify that Δ_{γ_i} is a Feferman-Vaught decomposition of γ_i. Consider arbitrary $(\sigma, \mathbf{W})$-structures $\mathfrak{A}$ and $\mathfrak{B}$ with $U(\mathfrak{A}) \cap U(\mathfrak{B}) = \emptyset$, and let $\bar{u} \in \big(U(\mathfrak{A})\big)^k$ and $\bar{v} \in \big(U(\mathfrak{B})\big)^\ell$. By definition, we have $\mathfrak{A} \oplus \mathfrak{B} \models \gamma_i[\bar{u}, \bar{v}]$ if and only if $i = \sum_S \{w^{\mathfrak{B}}(\bar{w}) \mid \bar{w} \in M\}$ for $M := \big\{\bar{w} \in \big(U(\mathfrak{B})\big)^m \bigm| \mathfrak{A} \oplus \mathfrak{B} \models \psi[\bar{u}, \bar{v}, \bar{w}]\big\}$. Since Δ_ψ is a Feferman-Vaught decomposition of ψ w.r.t. $(\bar{x}; \bar{y}\bar{z})$, we have $\mathfrak{A} \oplus \mathfrak{B} \models \psi[\bar{u}, \bar{v}, \bar{w}]$ if and only if there exists $(\alpha, \beta) \in \Delta_\psi$ such that $\mathfrak{A} \models \alpha[\bar{u}]$ and $\mathfrak{B} \models \beta[\bar{v}, \bar{w}]$. Furthermore, we know that the αs in Δ_ψ are mutually exclusive. Thus, there either is exactly one $\alpha \in A$ such that $\mathfrak{A} \models \alpha[\bar{u}]$ (we call this *Case 1*), or for all $\alpha \in A$, we have $\mathfrak{A} \not\models \alpha[\bar{u}]$ (we call this *Case 2*).

In *Case 1*, by our definition of the notion "the αs are mutually exclusive", there is exactly one formula β such that $(\alpha, \beta) \in \Delta_\psi$. Hence, $M = \big\{\bar{w} \in \big(U(\mathfrak{B})\big)^m \bigm| \mathfrak{B} \models \beta[\bar{v}, \bar{w}]\big\}$. Thus, we have $\mathfrak{A} \oplus \mathfrak{B} \models \gamma_i[\bar{u}, \bar{v}]$ $\iff$ $i = \sum_S \{w^{\mathfrak{B}}(\bar{w}) \mid \bar{w} \in M\}$ $\iff$ $\mathfrak{B} \models \big(i = \sum w(\bar{z}).\beta\big)[\bar{v}]$ $\iff$ there are $(\alpha', \beta') \in \Delta_{\gamma_i}$ such that $\mathfrak{A} \models \alpha'[\bar{u}]$ and $\mathfrak{B} \models \beta'[\bar{v}]$.

In *Case 2*, we have

$$\begin{aligned} M &= \big\{\bar{w} \in \big(U(\mathfrak{B})\big)^m \bigm| \mathfrak{A} \oplus \mathfrak{B} \models \psi[\bar{u}, \bar{v}, \bar{w}]\big\} \\ &= \big\{\bar{w} \in \big(U(\mathfrak{B})\big)^m \bigm| \text{there exists } (\alpha', \beta') \in \Delta_\psi \\ &\qquad\qquad \text{such that } \mathfrak{A} \models \alpha'[\bar{u}] \text{ and } \mathfrak{B} \models \beta'[\bar{v}]\big\} \\ &= \emptyset. \end{aligned}$$

Hence, $\mathfrak{A} \oplus \mathfrak{B} \models \gamma_i[\bar{u}, \bar{v}]$ $\iff$ $i = 0$ $\iff$ Δ_{γ_i} contains the tuple $\big(\bigwedge_{\alpha \in A} \neg\alpha, \top\big)$ $\iff$ there are $(\alpha', \beta') \in \Delta_{\gamma_i}$ such that $\mathfrak{A} \models \alpha'[\bar{u}]$ and $\mathfrak{B} \models \beta'[\bar{v}]$.

All in all, we obtain that Δ_{γ_i} is a Feferman-Vaught decomposition in $\mathsf{FOW}_1(\mathbb{P})[\sigma, \mathbb{S}, \mathbf{W}]$ of γ_i w.r.t. $(\bar{x}; \bar{y})$. ⌟

This completes the induction step and thus also the proof of Theorem 5.8. □

We conclude this section with the proof of Corollary 5.9.

Proof of Corollary 5.9. Let φ be an r-local $\mathsf{FOW}_1(\mathbb{P})[\sigma, \mathbb{S}, \mathbf{W}]$-formula. Using Theorem 5.8, we can compute a Feferman-Vaught decomposition Δ' in L_φ of φ w.r.t. $(\bar{x}; \bar{y})$. Let $\Delta := \big\{(\alpha^{(r)}, \beta^{(r)}) \bigm| (\alpha, \beta) \in \Delta'\big\}$, where $\alpha^{(r)}$ and $\beta^{(r)}$ are the r-localisations of α and β. We show that the two formulas

$$\psi_1(\bar{x}, \bar{y}) \;:=\; \mathrm{dist}(\bar{x}; \bar{y}) > 2r{+}1 \;\wedge\; \varphi(\bar{x}, \bar{y})$$

and

$$\psi_2(\bar{x}, \bar{y}) \;:=\; \mathrm{dist}(\bar{x}; \bar{y}) > 2r{+}1 \;\wedge \bigvee_{(\alpha^{(r)}, \beta^{(r)}) \in \Delta} \big(\alpha^{(r)}(\bar{x}) \wedge \beta^{(r)}(\bar{y})\big)$$

given in Corollary 5.9 are equivalent.

Let $\mathfrak{A}$ be a $(\sigma, \mathbf{W})$-structure, $\bar{v} \in \big(U(\mathfrak{A})\big)^k$, and $\bar{w} \in \big(U(\mathfrak{A})\big)^\ell$. If $\mathrm{dist}(\bar{v}, \bar{w}) \leqslant 2r{+}1$, then $\mathfrak{A} \not\models \psi_1[\bar{v}, \bar{w}]$ and $\mathfrak{A} \not\models \psi_2[\bar{v}, \bar{w}]$. Now let

$\text{dist}(\bar{v},\bar{w}) > 2r+1$. Then, since φ is r-local, we have that $\mathfrak{A} \models \psi_1[\bar{v},\bar{w}]$ if and only if $\mathcal{N}_r^{\mathfrak{A}}(\bar{v}) \uplus \mathcal{N}_r^{\mathfrak{A}}(\bar{w}) \models \varphi[\bar{v},\bar{w}]$. Thus, we obtain

$$\begin{aligned}
& \mathfrak{A} \models \psi_1[\bar{v},\bar{w}] \\
\iff & \mathcal{N}_r^{\mathfrak{A}}(\bar{v}) \uplus \mathcal{N}_r^{\mathfrak{A}}(\bar{w}) \models \varphi[\bar{v},\bar{w}] \\
\iff & \mathcal{N}_r^{\mathfrak{A}}(\bar{v}) \oplus \mathcal{N}_r^{\mathfrak{A}}(\bar{w}) \models \varphi[\bar{v},\bar{w}] \\
\iff & \exists(\alpha,\beta) \in \Delta' \colon \mathcal{N}_r^{\mathfrak{A}}(\bar{v}) \models \alpha[\bar{v}] \wedge \mathcal{N}_r^{\mathfrak{A}}(\bar{w}) \models \beta[\bar{w}] \\
\iff & \exists(\alpha,\beta) \in \Delta' \colon \mathcal{N}_r^{\mathfrak{A}}(\bar{v}) \models \alpha^{(r)}[\bar{v}] \wedge \mathcal{N}_r^{\mathfrak{A}}(\bar{w}) \models \beta^{(r)}[\bar{w}] \\
\iff & \exists(\alpha,\beta) \in \Delta' \colon \mathcal{N}_r^{\mathfrak{A}}(\bar{v}) \oplus \mathcal{N}_r^{\mathfrak{A}}(\bar{w}) \models \alpha^{(r)}[\bar{v}] \wedge \beta^{(r)}[\bar{w}] \\
\iff & \exists(\alpha,\beta) \in \Delta' \colon \mathcal{N}_r^{\mathfrak{A}}(\bar{v}) \uplus \mathcal{N}_r^{\mathfrak{A}}(\bar{w}) \models \alpha^{(r)}[\bar{v}] \wedge \beta^{(r)}[\bar{w}] \\
\iff & \exists(\alpha,\beta) \in \Delta' \colon \mathfrak{A} \models \alpha^{(r)}[\bar{v}] \wedge \beta^{(r)}[\bar{w}] \\
\iff & \mathfrak{A} \models \psi_2[\bar{v},\bar{w}].
\end{aligned}$$

We can switch between the disjoint sum and the disjoint union of structures because the considered formulas only use relations from the disjoint union. All in all, this shows that $\psi_1 \equiv \psi_2$. □

5.3 GAIFMAN NORMAL FORM FOR FOW1

Basic local sentence

We now turn to a notion of Gaifman normal form for FOW_1. For $r \in \mathbb{N}$, a *basic local sentence (of radius* r*)* in $\mathsf{FOW}_1(\mathbb{P})[\sigma,\mathbb{S},\mathbf{W}]]$ is a sentence of the form

$$\exists x_1 \ldots \exists x_k \Big(\bigwedge_{1 \leqslant i<j \leqslant k} \text{dist}^{\sigma}(x_i,x_j) > 2r \ \wedge \ \bigwedge_{i=1}^{k} \varphi(x_i)\Big),$$

where $k \in \mathbb{N}_{\geqslant 1}$, $x_1,\ldots,x_k$ are k pairwise distinct variables, and $\varphi(x)$ is an r-local $\mathsf{FOW}_1(\mathbb{P})[\sigma,\mathbb{S},\mathbf{W}]$-formula. A *local aggregation sentence (of radius* r*)* in $\mathsf{FOW}_1(\mathbb{P})[\sigma,\mathbb{S},\mathbf{W}]$ is a sentence of the form $\big(s = \sum \mathsf{w}(\bar{x}).\varphi(\bar{x})\big)$, where $\mathsf{w} \in \mathbf{W}$, $s \in S$ with $S \coloneqq \text{type}(\mathsf{w})$, $k = \text{ar}(\mathsf{w})$, $\bar{x} = (x_1,\ldots,x_k)$ is a tuple of k pairwise distinct variables, and $\varphi(\bar{x})$ is an r-local $\mathsf{FOW}_1(\mathbb{P})[\sigma,\mathbb{S},\mathbf{W}]$-formula.

Local aggregation sentence

Gaifman normal form

Definition 5.11. A $\mathsf{FOW}_1(\mathbb{P})[\sigma,\mathbb{S},\mathbf{W}]$-formula is in *Gaifman normal form* if it is a Boolean combination of local $\mathsf{FOW}_1(\mathbb{P})[\sigma,\mathbb{S},\mathbf{W}]$-formulas, basic local sentences in $\mathsf{FOW}_1(\mathbb{P})[\sigma,\mathbb{S},\mathbf{W}]$, and local aggregation sentences in $\mathsf{FOW}_1(\mathbb{P})[\sigma,\mathbb{S},\mathbf{W}]$.

The *locality radius* of an $\mathsf{FOW}_1(\mathbb{P})[\sigma,\mathbb{S},\mathbf{W}]$-formula φ in Gaifman normal form is the least r such that all basic local sentences and all local aggregation sentences in φ have radius at most r and every local formula in φ is r'-local for some $r' \leqslant r$.

In our next result, we provide a Gaifman normal form for FOW_1.

Theorem 5.12 (Gaifman normal form for FOW_1)**.**
Every $\mathsf{FOW}_1(\mathbb{P})[\sigma,\mathbb{S},\mathbf{W}]$*-formula* φ *is equivalent to an* $\mathsf{FOW}_1(\mathbb{P})[\sigma,\mathbb{S},\mathbf{W}]$*-formula* γ *in Gaifman normal form with* $\text{free}(\gamma) = \text{free}(\varphi)$. *Furthermore, there is an algorithm that computes* γ *upon input of* φ.

The proof proceeds similarly as Gaifman's original proof for first-order logic ([41], see also [49, Sect. 4.1]), but since subformulas are from FOW_1, we use Corollary 5.9 instead of Feferman-Vaught decompositions for FO (cf. [49, Lemma 2.3]). Furthermore, for formulas built according to rule $(5)_1$, we proceed similarly to the case of modulo-counting quantifiers in the Gaifman normal construction of [70].

The remainder of this section is devoted to the proof of Theorem 5.12.

Proof of Theorem 5.12. The proof proceeds by induction on the construction of φ. Formulas built according to rules (1) and (2) of Definition 5.3 are 0-local. The statement of Theorem 5.12 trivially extends to Boolean combinations of formulas, so formulas being built according to rule (3). A formula φ that is built according to rule $(6)_1$ is of the form $P(t_1, \dots, t_m)$, where $P \in \mathbb{P}$ and $t_1, \dots, t_m$ are S-terms built using the rules (7)–(9). Thus, φ is 0-local.

If φ is of the form $\exists x\, \psi$, we can argue in the same way as in Gaifman's original proof for first-order logic ([41], see also [49, Sect. 4.1]), but since ψ is from $\mathsf{FOW}_1(\mathbb{P})[\sigma, \mathbb{S}, \mathbf{W}]$, we use Corollary 5.9 instead of Feferman-Vaught decompositions for first-order logic.

For formulas built according to rule $(5)_1$ of Definition 5.4, we proceed similarly to the case of modulo-counting quantifiers in the Gaifman normal construction of [70]. Let φ be of the form $\big(s = \sum w(\bar{y}).\psi(\bar{x}, \bar{y})\big)$ for a tuple of variables $\bar{y} = (y_1, \dots, y_\ell)$, a weight symbol $w \in \mathbf{W}$ whose type $S \coloneqq \text{type}(w)$ is finite, and where $\bar{x} = (x_1, \dots, x_k)$ are the free variables of φ. Note that k might be 0.

By the induction hypothesis, we can transform ψ into an equivalent formula in Gaifman normal form. We can assume w.l.o.g. that this formula is of the form $\bigvee_{j=1}^{m} (\chi_j \wedge \gamma_j(\bar{x}, \bar{y}))$, where each χ_j is an $\mathsf{FOW}_1(\mathbb{P})[\sigma, \mathbb{S}, \mathbf{W}]$-sentence in Gaifman normal form and each $\gamma_j(\bar{x}, \bar{y})$ is r-local for some $r \in \mathbb{N}$. For every $J \subseteq [m]$, let

$$\chi_J \coloneqq \bigwedge_{j \in J} \chi_j \wedge \bigwedge_{j \in [m] \setminus J} \neg \chi_j \qquad \text{and} \qquad \gamma_J(\bar{x}, \bar{y}) \coloneqq \bigvee_{j \in J} \gamma_j(\bar{x}, \bar{y}).$$

Clearly, $\bigvee_{j=1}^{m} (\chi_j \wedge \gamma_j(\bar{x}, \bar{y}))$ is equivalent to $\bigvee_{\emptyset \neq J \subseteq [m]} (\chi_J \wedge \gamma_J(\bar{x}, \bar{y}))$, the $(\chi_J)_{J \subseteq [m]}$ are mutually exclusive sentences in Gaifman normal form, and $\gamma_J(\bar{x}, \bar{y})$ is r-local. Let

$$\tilde{\varphi} \coloneqq \bigvee_{\emptyset \neq J \subseteq [m]} \Big(\chi_J \wedge \big(s = \sum w(\bar{y}).\gamma_J(\bar{x}, \bar{y})\big) \Big).$$

Claim 1. If $s \neq 0_S$, then φ is equivalent to $\tilde{\varphi}$. Otherwise, if $s = 0_S$, then φ is equivalent to $(\tilde{\varphi} \vee \chi_\emptyset)$.

Proof. The formula φ is equivalent to

$$\varphi' \coloneqq \Big(s = \sum w(\bar{y}). \bigvee_{\emptyset \neq J \subseteq [m]} (\chi_J \wedge \gamma_J(\bar{x}, \bar{y})) \Big).$$

If $s \neq 0_S$, then, since the χ_Js are mutually exclusive, φ' is equivalent to $\tilde{\varphi}$. For $s = 0$, we need to consider the additional case that none of the χ_Js hold, so φ' is equivalent to $\tilde{\varphi} \vee \chi_\emptyset$. ⌟

To complete the proof of Theorem 5.12, it suffices to consider an arbitrary non-empty $J \subseteq [m]$ and the r-local formula $\gamma(\bar{x}, \bar{y}) := \gamma_J(\bar{x}, \bar{y})$ and show how to transform the formula $\psi'(\bar{x}) := \big(s = \sum w(\bar{y}).\gamma(\bar{x}, \bar{y})\big)$ into an equivalent formula in Gaifman normal form. If $k = 0$, then we are done since ψ' is a local aggregation sentence in $\mathsf{FOW}_1(\mathbb{P})[\sigma, \mathcal{S}, \mathbf{W}]$. If $k > 0$, then we split the sum into two parts where one only considers tuples $\bar{y}$ where one of the elements is in a certain neighbourhood around $\bar{x}$ and the other one considers all other tuples. Formally, we proceed as follows. Let $I_s := \big\{(i_1, i_2) \in S \times S \;\big|\; i_1 +_S i_2 = s\big\}$. Then, $\psi'(\bar{x})$ is equivalent to $\bigvee_{(i_1,i_2)\in I}\big(\psi^{\text{in}}_{i_1}(\bar{x}) \wedge \psi^{\text{out}}_{i_2}(\bar{x})\big)$, where

$$\psi^{\text{in}}_{i_1}(\bar{x}) := \Big(i_1 = \sum w(\bar{y})\ .\ \big(\gamma(\bar{x}, \bar{y}) \wedge \text{dist}(\bar{x}; \bar{y}) \leqslant 2r{+}1\big)\Big)$$

and

$$\psi^{\text{out}}_{i_2}(\bar{x}) := \Big(i_2 = \sum w(\bar{y})\ .\ \big(\gamma(\bar{x}, \bar{y}) \wedge \text{dist}(\bar{x}; \bar{y}) > 2r{+}1\big)\Big).$$

For a $(\sigma, \mathbf{W})$-structure $\mathfrak{A}$ and a tuple $\bar{v} \in \big(U(\mathfrak{A})\big)^k$, to check whether $\mathfrak{A} \models \psi^{\text{in}}_{i_1}[\bar{v}]$ holds, it suffices to consider only those assignments $\bar{w}$ to $\bar{y}$ where at least one of the w_i is in the $(2r{+}1)$-neighbourhood of $\bar{v}$. Furthermore, we have $w^{\mathfrak{A}}(\bar{w}) \neq 0_S$ only if the vertices in $\bar{w}$ form a clique in the Gaifman graph of $\mathfrak{A}$, so they have distance at most 1 from each other. Finally, since γ is r-local, we can deduce that $\psi^{\text{in}}_{i_1}$ is $(3r{+}2)$-local.

It remains to transform $\psi^{\text{out}}_{i_2}$ into an equivalent formula in Gaifman normal form. To achieve this, we use Corollary 5.9 to obtain a finite, non-empty set Δ of pairs $\big(\alpha(\bar{x}), \beta(\bar{y})\big)$ of r-local $\mathsf{FOW}_1(\mathbb{P})[\sigma, \mathcal{S}, \mathbf{W}]$-formulas such that the formula $\big(\gamma(\bar{x}, \bar{y}) \wedge \text{dist}(\bar{x}; \bar{y}) > 2r{+}1\big)$ is equivalent to $\Big(\bigvee_{(\alpha,\beta)\in\Delta}\big(\alpha(\bar{x}) \wedge \beta(\bar{y})\big)\ \wedge\ \text{dist}(\bar{x}; \bar{y}) > 2r{+}1\Big)$.

By Lemma 5.10, we can assume that the αs in Δ are mutually exclusive. Let

$$\tilde{\psi}^{\text{out}}_{i_2} := \bigvee_{(\alpha,\beta)\in\Delta}\Big(\alpha(\bar{x}) \wedge \Big(i_2 = \sum w(\bar{y})\ .\ \big(\beta(\bar{y}) \wedge \text{dist}(\bar{x}; \bar{y}) > 2r{+}1\big)\Big)\Big).$$

Analogously to Claim 1, we obtain the following.

Claim 2. If $i_2 \neq 0_S$, then $\psi^{\text{out}}_{i_2}(\bar{x})$ is equivalent to $\tilde{\psi}^{\text{out}}_{i_2}(\bar{x})$. Otherwise, if $i_2 = 0_S$, then $\psi^{\text{out}}_{i_2}(\bar{x})$ is equivalent to $\tilde{\psi}^{\text{out}}_{i_2}(\bar{x}) \vee \bigwedge_{\alpha\in A} \neg\alpha(\bar{x})$ for $A := \big\{\alpha \;\big|\; \text{there exists } \beta \text{ such that } (\alpha, \beta) \in \Delta\big\}$.

To complete the proof of Theorem 5.12, we show how to transform a formula $\lambda(\bar{x}) := \Big(i_2 = \sum w(\bar{y})\ .\ \big(\beta(\bar{y}) \wedge \text{dist}(\bar{x}; \bar{y}) > 2r{+}1\big)\Big)$ for an arbitrary r-local formula $\beta(\bar{y})$ into an equivalent $\mathsf{FOW}_1(\mathbb{P})[\sigma, \mathcal{S}, \mathbf{W}]$-formula in Gaifman normal form. Let $J := \big\{(j_1, j_2) \in S \times S \;\big|\; j_1 -_S j_2 =$

$i_2\}$. Then, $\lambda(\bar{x})$ is equivalent to the formula $\bigvee_{(j_1,j_2)\in J}\big(\lambda^{\text{all}}_{j_1} \wedge \lambda^{\text{in}}_{j_2}(\bar{x})\big)$, where

$$\begin{aligned}\lambda^{\text{all}}_{j_1} &:= \big(j_1 = \sum \mathsf{w}(\bar{y}) \,.\, \beta(\bar{y})\big),\\ \lambda^{\text{in}}_{j_2}(\bar{x}) &:= \Big(j_2 = \sum \mathsf{w}(\bar{y}) \,.\, \big(\gamma(\bar{x},\bar{y}) \wedge \operatorname{dist}(\bar{x};\bar{y}) \leqslant 2r{+}1\big)\Big).\end{aligned}$$

Now, $\lambda^{\text{all}}_{j_1}$ is a local aggregation sentence in $\mathsf{FOW}_1(\mathbb{P})[\sigma,\mathbb{S},\mathbf{W}]$. Furthermore, with the same argumentation as for the formula $\psi^{\text{in}}_{i_1}$ above, the formula $\lambda^{\text{in}}_{j_2}$ is $(3r+2)$-local. This completes the proof of Theorem 5.12. □

5.4 LOCALISATION OF FOWA1

In this section, we provide a locality result for the logic FOWA_1, which is a logic substantially more expressive than FOW_1.

Theorem 5.13 (Localisation Theorem for FOWA_1)**.** *Let $k \in \mathbb{N}$. For every $\mathsf{FOWA}_1(\mathbb{P})[\sigma,\mathbb{S},\mathbf{W}]$-formula $\varphi(x_1,\dots,x_k)$, there is an extension σ_φ of σ with relation symbols of arity $\leqslant 1$, and an $\mathsf{FOW}_1(\mathbb{P})[\sigma_\varphi,\mathbb{S},\mathbf{W}]$-formula $\varphi'(x_1,\dots,x_k)$ that is a Boolean combination of local formulas and statements of the form $R()$ where $R \in \sigma_\varphi$ has arity 0, for which the following holds. There is an algorithm*[4] *that, upon input of a $(\sigma,\mathbf{W})$-structure $\mathfrak{A}$, computes in time $|\mathfrak{A}|\log|\mathfrak{A}|\cdot d^{\mathcal{O}(1)}$ under the logarithmic-cost measure, and in time $|\mathfrak{A}|\cdot d^{\mathcal{O}(1)}$ under the uniform-cost measure, where d is the degree of $\mathfrak{A}$, a σ_φ-expansion $\mathfrak{A}_\varphi$ of $\mathfrak{A}$ such that for all $\bar{v} \in \big(U(\mathfrak{A})\big)^k$ it holds that $\mathfrak{A}_\varphi \models \varphi'[\bar{v}]$ if and only if $\mathfrak{A} \models \varphi[\bar{v}]$.*

We prove this by decomposing FOWA_1-expressions into simpler expressions that can be evaluated in a structure $\mathfrak{A}$ by exploring for each element $v \in U(\mathfrak{A})$ only a local neighbourhood around v. This is achieved by proving a decomposition theorem (Theorem 5.20) that is a generalisation of the decomposition for $\mathsf{FOC}_1(\mathbb{P})$ provided in [56, Theorem 6.6], and it builds upon the Gaifman normal form result of Theorem 5.12. The remainder of this section is devoted to the proof of Theorem 5.13.

5.4.1 *Connected local terms*

For every $k \in \mathbb{N}_{\geqslant 1}$, let $\mathcal{G}_k$ be the set of all graphs G with vertex set $[k]$. For a graph $G \in \mathcal{G}_k$, a radius $r \in \mathbb{N}$, and a tuple $\bar{x} = (x_1,\dots,x_k)$ of k pairwise distinct variables, we consider the formula

$$\delta^\sigma_{G,r}(\bar{x}) := \bigwedge_{\{i,j\}\in E(G)} \operatorname{dist}^\sigma(x_i,x_j) \leqslant r \quad\wedge \bigwedge_{\{i,j\}\notin E(G)} \operatorname{dist}^\sigma(x_i,x_j) > r.$$

4 with $\mathbb{P}$- and $\mathbb{S}$-oracles, i.e., the operations $+_S, \cdot_S$ for $S \in \mathbb{S}$ and checking if a tuple belongs to $[\![P]\!]$ for $P \in \mathbb{P}$ can be done in constant time by referring to an oracle that provides us with the answers

Note that $\mathfrak{A} \models \delta^\sigma_{G,r}[\bar{v}]$ means that the connected components of the r-neighbourhood $\mathcal{N}^{\mathfrak{A}}_r(\bar{v})$ correspond to the connected components of G. Moreover, $\delta^\sigma_{G,2r+1}(\bar{x})$ is r-local around its free variables $\bar{x}$.

The main ingredient of our decomposition of $\mathsf{FOWA}_1(\mathbb{P})[\sigma, \mathcal{S}, \mathbf{W}]$-expressions are the connected local terms (*cl terms*, for short), defined as follows. *cl Term*

Definition 5.14. Let $r \in \mathbb{N}$ and $k \in \mathbb{N}_{\geqslant 1}$. A *basic cl term (of radius* r *and width* k*)* is an $\mathcal{S}$-term of the form

$$\sum p.\big(\psi(x_1, \ldots, x_k) \wedge \delta^\sigma_{G,2r+1}(x_1, \ldots, x_k)\big),$$

where $x_1, \ldots, x_k$ are k pairwise distinct variables, we have $\text{vars}(p) \subseteq \{x_1, \ldots, x_k\}$, $\psi(x_1, \ldots, x_k)$ is an $\mathsf{FOW}_1(\mathbb{P})[\sigma, \mathcal{S}, \mathbf{W}]$-formula that is r-local around $(x_1, \ldots, x_k)$, and $G \in \mathcal{G}_k$ is connected. A *cl term (of radius* $\leqslant r$ *and width* $\leqslant k$*)* is built from basic cl terms (of radius $\leqslant r$ and width $\leqslant k$) by using rules (7)–(9) of Definition 5.3.

Note that cl terms are "easy" with respect to query evaluation in the following sense.

Lemma 5.15. *For every fixed cl term* $t(y_1, \ldots, y_\ell)$ *(with* $\ell \geqslant 0$*), there is an algorithm such that the following holds. Upon input of a* $(\sigma, \mathbf{W})$*-structure* $\mathfrak{A}$*, within precomputation time* $|\mathfrak{A}| \log |\mathfrak{A}| \cdot d^{\mathcal{O}(1)}$ *under the logarithmic-cost measure and within precomputation time* $|\mathfrak{A}| \cdot d^{\mathcal{O}(1)}$ *under the uniform-cost measure, where* d *is the degree of* $\mathfrak{A}$*, the algorithm computes a data structure that, whenever given a tuple* $(v_1, \ldots, v_\ell) \in \big(U(\mathfrak{A})\big)^\ell$*, returns the value* $t^{\mathfrak{A}}[v_1, \ldots, v_\ell]$ *in time* $\mathcal{O}\big(\log |\mathfrak{A}|\big)$ *under the logarithmic-cost measure and in constant time under the uniform-cost measure.*

Proof. It suffices to prove the lemma for basic cl terms. The statement for general cl terms then follows by induction. Consider a basic cl term $t(y_1, \ldots, y_\ell)$ of the form $\sum p.\big(\psi(x_1, \ldots, x_k) \wedge \delta^\sigma_{G,2r+1}(x_1, \ldots, x_k)\big)$. Recall from Definition 5.14 that $\text{vars}(p) \subseteq \{x_1, \ldots, x_k\}$ and G is a connected graph. We can assume w.l.o.g. that $(y_1, \ldots, y_\ell) = (x_1, \ldots, x_\ell)$ and $\text{vars}(p) = \{x_{\ell+1}, \ldots, x_k\}$. Let $S \in \mathcal{S}$ be the type of the $\mathbf{W}$-product p.

Given a $(\sigma, \mathbf{W})$-structure $\mathfrak{A}$ and an element $u \in U(\mathfrak{A})$, we can explore the R-neighbourhood of u for $R := r + (k-1)(2r+1)$ (cf. Fact 2.1) and thereby compute the set M_u of all $\bar{v} = (v_1, \ldots, v_k) \in \big(U(\mathfrak{A})\big)^k$ with $v_1 = u$ such that $(\mathfrak{A}, \bar{v}) \models (\psi \wedge \delta^\sigma_{G,2r+1})$. For each such tuple $\bar{v}$, we compute and store the value $s_{\bar{v}} := p^{\mathfrak{A}}[v_{\ell+1}, \ldots, v_k] \in S$. Then, we group the tuples in M_u by their prefix $(v_1, \ldots, v_\ell)$ of length ℓ, and for each group, we compute the $+_S$-sum $s_{u,(v_1,\ldots,v_\ell)}$ of the values $s_{\bar{v}}$ of all tuples $\bar{v} \in M_u$ that have the same prefix $(v_1, \ldots, v_\ell)$.

In case that $\ell = 0$, t is a ground term, and we have $t^{\mathfrak{A}} = \sum_S \big\{ s_{u,()} \bigm| u \in U(\mathfrak{A}) \big\}$. In case that $\ell \geqslant 1$, whenever given an arbitrary tuple $(v_1, \ldots, v_\ell) \in \big(U(\mathfrak{A})\big)^\ell$, we can determine $t^{\mathfrak{A}}[v_1, \ldots, v_\ell]$ as follows.

Let $u \coloneqq v_1$. If M_u contains a tuple with prefix $(v_1,\dots,v_\ell)$, then $t^{\mathfrak{A}}[v_1,\dots,v_\ell] = s_{u,(v_1,\dots,v_\ell)}$, and otherwise $t^{\mathfrak{A}}[v_1,\dots,v_\ell] = 0_{\mathbb{S}}$.

Thus, upon input of a $(\sigma,\mathbf{W})$-structure $\mathfrak{A}$, we can compute a data structure which, on input $(v_1,\dots,v_\ell) \in \big(U(\mathfrak{A})\big)^\ell$, returns the value $t^{\mathfrak{A}}[v_1,\dots,v_\ell]$.

The data structure can be computed within precomputation time $|\mathfrak{A}|\log|\mathfrak{A}| \cdot d^{\mathcal{O}(1)}$ under the logarithmic-cost measure and within precomputation time $|\mathfrak{A}| \cdot d^{\mathcal{O}(1)}$ under the uniform-cost measure, where d is the degree of $\mathfrak{A}$. Once the data structure has been computed, the values $t^{\mathfrak{A}}[v_1,\dots,v_\ell]$ can be returned in time $\mathcal{O}\big(\log|\mathfrak{A}|\big)$ under the logarithmic-cost measure and in constant time under the uniform-cost measure. $\square$

Our decomposition of $\mathsf{FOWA}_1(\mathbb{P})[\sigma,\mathbb{S},\mathbf{W}]$-expressions proceeds by induction on the construction of the input expression. The main technical tool for the construction is the following lemma.

Lemma 5.16. *Let $r \in \mathbb{N}$, $k \in \mathbb{N}_{\geqslant 1}$, let $\bar{x} = (x_1,\dots,x_k)$ be a tuple of k pairwise distinct variables, let $\psi(\bar{x})$ be an r-local $\mathsf{FOW}_1(\mathbb{P})[\sigma,\mathbb{S},\mathbf{W}]$-formula, and consider an $\mathbb{S}$-term $t(y_1,\dots,y_\ell)$ of the form $\sum p.\psi(x_1,\dots,x_k)$, where p is a $\mathbf{W}$-product, $\ell \in \mathbb{N}$, and $\{y_1 \dots, y_\ell\} \subseteq \{x_1,\dots,x_k\}$. Then, there exists a cl term $\hat{t}(y_1,\dots,y_\ell)$ of radius $\leqslant r$ and width $\leqslant k$, such that $\hat{t}^{\mathfrak{A}}[\bar{v}] = t^{\mathfrak{A}}[\bar{v}]$ holds for every $(\sigma,\mathbf{W})$-structure $\mathfrak{A}$ and every $\bar{v} \in \big(U(\mathfrak{A})\big)^\ell$. Furthermore, there is an algorithm which, upon input of r and t, constructs the cl term $\hat{t}$.*

Proof. For a $(\sigma,\mathbf{W})$-structure $\mathfrak{A}$ and a formula $\varphi(\bar{x})$, we consider the set $M^{\mathfrak{A}}_{\varphi} \coloneqq \big\{\bar{v} \in \big(U(\mathfrak{A})\big)^k \mid \mathfrak{A} \models \varphi[\bar{v}]\big\}$. For every graph $G \in \mathcal{G}_k$, let $\psi_G(\bar{x}) \coloneqq \psi(\bar{x}) \wedge \delta^{\sigma}_{G,2r+1}(\bar{x})$. Note that $\psi_G(\bar{x})$ is r-local around $\bar{x}$. Furthermore, for every $(\sigma,\mathbf{W})$-structure $\mathfrak{A}$, the set $M^{\mathfrak{A}}_{\psi}$ is the disjoint union of the sets $M^{\mathfrak{A}}_{\psi_G}$ for all $G \in \mathcal{G}_k$. Therefore, t is equivalent to the $+$-sum, over all $G \in \mathcal{G}_k$, of the $\mathbb{S}$-terms $t^{\psi}_{G} \coloneqq \sum p.\psi_G(x_1,\dots,x_k)$. To complete the proof of Lemma 5.16, it therefore suffices to show that, for every $G \in \mathcal{G}_k$, the $\mathbb{S}$-term t^{ψ}_{G} is equivalent to a cl term of radius r. We prove this by induction on the number of connected components of G. That is, we show that the following statement $(*)_c$ is true for every number of components $c \in \mathbb{N}_{\geqslant 1}$.

$(*)_c$: For every $k \geqslant c$, for every tuple $\bar{x} = (x_1,\dots,x_k)$ of k pairwise distinct variables, for every radius $r \in \mathbb{N}$, for every r-local $\mathsf{FOW}_1(\mathbb{P})[\sigma,\mathbb{S},\mathbf{W}]$-formula $\psi(\bar{x})$, for every $\mathbf{W}$-product p with $\operatorname{vars}(p) \subseteq \{x_1,\dots,x_k\}$, and for every graph $G \in \mathcal{G}_k$ that has at most c connected components, the $\mathbb{S}$-term t^{ψ}_{G} is equivalent to a cl term of radius r.

In the induction base, for $c = 1$, we only consider connected graphs G. Thus, by Definition 5.14, t^{ψ}_{G} is a basic cl term.

For the induction step from c to $c+1$, consider a $k \geqslant c+1$ and a graph $G = (V,E) \in \mathcal{G}_k$ that has $c+1$ connected components. Let

V' be the set of all vertices of V that are connected to the vertex 1, and let $V'' := V \setminus V'$ contain all other vertices. Let $G' := G[V']$ and $G'' := G[V'']$ be the induced subgraphs of G on V' and V'', respectively. Clearly, G is the disjoint union of G' and G'', G' is connected, and G'' has c connected components. W.l.o.g., let $V' = \{1,\dots,\ell\}$ and $V'' = \{\ell+1,\dots,k\}$ for an ℓ with $1 \leqslant \ell < k$. For a tuple $\bar{v} = (v_1,\dots,v_k)$, we let $\bar{v}' := (v_1,\dots,v_\ell)$ and $\bar{v}'' := (v_{\ell+1},\dots,v_k)$.

Now consider a radius $r \in \mathbb{N}$ and the formula $\delta^\sigma_{G,2r+1}(\bar{x})$ for $\bar{x} = (x_1,\dots,x_k)$. For every σ-structure $\mathfrak{A}$ and every tuple $\bar{v} \in \big(U(\mathfrak{A})\big)^k$ with $\mathfrak{A} \models \delta^\sigma_{G,2r+1}[\bar{v}]$, the r-neighbourhood $\mathcal{N}^{\mathfrak{A}}_r(\bar{v})$ is the disjoint union of the r-neighbourhoods $\mathcal{N}^{\mathfrak{A}}_r(\bar{v}')$ and $\mathcal{N}^{\mathfrak{A}}_r(\bar{v}'')$.

Let $\psi(\bar{x})$ be an $\mathsf{FOW}_1(\mathbb{P})[\sigma,\mathcal{S},\mathbf{W}]$-formula that is r-local. By using Corollary 5.9, we can compute a decomposition of $\psi(\bar{x})$ into a formula of the form

$$\bigvee_{i\in I}\Big(\psi'_i(\bar{x}') \wedge \psi''_i(\bar{x}'')\Big),$$

where I is a finite non-empty set, $\psi'_i(\bar{x}')$ and $\psi''_i(\bar{x}'')$ are r-local $\mathsf{FOW}_1(\mathbb{P})[\sigma,\mathcal{S},\mathbf{W}]$-formulas, and the $\psi'_i(\bar{x}')$ are mutually exclusive. This implies that, for every $(\sigma,\mathbf{W})$-structure $\mathfrak{A}$, the set $M^{\mathfrak{A}}_{\psi_G}$ is the disjoint union of the sets $M^{\mathfrak{A}}_{(\psi'_i\wedge\psi''_i\wedge\delta^\sigma_{G,2r+1})}$ over all $i \in I$.

Now let p be an arbitrary $\mathbf{W}$-product with $\mathrm{vars}(p) \subseteq \{x_1,\dots,x_k\}$, and consider the $\mathcal{S}$-term $t^\psi_G = \sum p.\psi_G(x_1,\dots,x_k)$. From the above reasoning, it follows that t^ψ_G is equivalent to the $+$-sum, over all $i \in I$, of the $\mathcal{S}$-terms $t^{\psi,i}_G := \sum p.\big(\psi'_i(\bar{x}') \wedge \psi''_i(\bar{x}'') \wedge \delta^\sigma_{G,2r+1}(\bar{x})\big)$. To complete the proof, it suffices to show that $t^{\psi,i}_G$ is equivalent to a cl term of radius r.

By the definition of the formula $\delta^\sigma_{G,2r+1}(\bar{x})$, we obtain that the formula $\psi'_i(\bar{x}') \wedge \psi''_i(\bar{x}'') \wedge \delta^\sigma_{G,2r+1}(\bar{x})$ is equivalent to the formula

$$\vartheta'_i(\bar{x}') \;\wedge\; \vartheta''_i(\bar{x}'') \;\wedge\; \mathrm{dist}^\sigma(\bar{x}';\bar{x}'') > 2r+1, \quad \text{where}$$

$$\vartheta'_i(\bar{x}') := \psi'_i(\bar{x}') \wedge \delta^\sigma_{G',2r+1}(\bar{x}') \text{ and } \vartheta''_i(\bar{x}'') := \psi''_i(\bar{x}'') \wedge \delta^\sigma_{G'',2r+1}(\bar{x}'').$$

Therefore, for every $(\sigma,\mathbf{W})$-structure $\mathfrak{A}$, we have

$$M^{\mathfrak{A}}_{(\psi'_i\wedge\psi''_i\wedge\delta^\sigma_{G,2r+1})} = \big(M^{\mathfrak{A}}_{\vartheta'_i} \times M^{\mathfrak{A}}_{\vartheta''_i}\big) \;\setminus\; X^{\mathfrak{A}}_i,$$

for

$$X^{\mathfrak{A}}_i := \Big\{\bar{v} \in \big(U(\mathfrak{A})\big)^k \;\Big|\; \mathfrak{A} \models \vartheta'_i[\bar{v}'],\ \mathfrak{A} \models \vartheta''_i[\bar{v}''],\ \mathrm{dist}^{\mathfrak{A}}(\bar{v}',\bar{v}'') \leqslant 2r+1\Big\}.$$

Let $\mathcal{H}_k$ be the set of all graphs $H \in \mathcal{G}_k$ with $H \neq G$, but $H[V'] = G'$ and $H[V''] = G''$. Then every $H \in \mathcal{H}_k$ has at most c connected components. Furthermore, for every $(\sigma,\mathbf{W})$-structure $\mathfrak{A}$, the set $X^{\mathfrak{A}}_i$ is the disjoint union over all $H \in \mathcal{H}_k$ of the sets

$$X^{\mathfrak{A}}_{i,H} := \Big\{\bar{v} \in \big(U(\mathfrak{A})\big)^k \;\Big|\; \mathfrak{A} \models \vartheta'_i[\bar{v}'],\ \mathfrak{A} \models \vartheta''_i[\bar{v}''],\ \mathfrak{A} \models \delta^\sigma_{H,2r+1}[\bar{v}]\Big\}.$$

Next, we take a closer look at the $\mathbf{W}$-product p of type $S \in \mathbb{S}$ used in the $\mathbb{S}$-term $t_G^{\psi,i}$. If p contains a factor $w(\bar{y})$ for some $w \in \mathbf{W}$ and a tuple $\bar{y}$ that contains variables from both $\bar{x}' = (x_1, \dots, x_\ell)$ and $\bar{x}'' = (x_{\ell+1}, \dots, x_k)$, then $p^{\mathfrak{A}}[\bar{v}] = 0_S$ for every $(\sigma, \mathbf{W})$-structure $\mathfrak{A}$ and every tuple $\bar{v} \in M^{\mathfrak{A}}_{(\psi'_i \wedge \psi''_i \wedge \delta^{\sigma}_{G,2r+1})}$. Thus, $t_G^{\psi,i}$ would be equivalent to the $\mathbb{S}$-term 0_S, and we would be done. Hence, in the following, let p be of the form $p'_1 \cdot p''_1 \cdot \dots \cdot p'_m \cdot p''_m$ for some $m \in \mathbb{N}_{\geqslant 1}$, where $\text{vars}(p'_i) \subseteq \{x_1, \dots, x_\ell\}$ and $\text{vars}(p''_i) \subseteq \{x_{\ell+1}, \dots, x_k\}$ for all $i \in [m]$.

Let $\mathfrak{A}$ be a $(\sigma, \mathbf{W})$-structure and fix an assignment $\beta\colon \text{free}(t_G^{\psi,i}) \to U(\mathfrak{A})$. Evaluating $t_G^{\psi,i}$ in $(\mathfrak{A}, \beta)$ means computing the value

$$\left[\!\left[t_G^{\psi,i}\right]\!\right]^{(\mathfrak{A},\beta)} = \sum_S \{p^{\mathfrak{A}}[\bar{v}] \mid \bar{v} \in M^{\mathfrak{A}}_{(\psi'_i \wedge \psi''_i \wedge \delta^{\sigma}_{G,2r+1})}, \\ \bar{v} \text{ agrees with } \beta \text{ on } \text{free}(t_G^{\psi,i})\}.$$

Since $M_{(\psi'_i \wedge \psi''_i \wedge \delta^{\sigma}_{G,2r+1})} = (M^{\mathfrak{A}}_{\vartheta'_i} \times M^{\mathfrak{A}}_{\vartheta''_i}) \setminus \big(\bigcup_{H \in \mathcal{H}_k} X^{\mathfrak{A}}_{i,H}\big)$, where the sets $X^{\mathfrak{A}}_{i,H}$ for $H \in \mathcal{H}_k$ are pairwise disjoint and contained in $M^{\mathfrak{A}}_{\vartheta'_i} \times M^{\mathfrak{A}}_{\vartheta''_i}$, the value $\left[\!\left[t_G^{\psi,i}\right]\!\right]^{(\mathfrak{A},\beta)}$ is equal to

$$\sum_S \{p^{\mathfrak{A}}[\bar{v}] \mid \bar{v} \in M^{\mathfrak{A}}_{\vartheta'_i} \times M^{\mathfrak{A}}_{\vartheta''_i}, \bar{v} \text{ agrees with } \beta \text{ on } \text{free}(t_G^{\psi,i})\} \\ -_S \Big(\sum_{H \in \mathcal{H}_k} \sum_S \{p^{\mathfrak{A}}[\bar{v}] \mid \bar{v} \in X^{\mathfrak{A}}_{i,H}, \bar{v} \text{ agrees with } \beta \text{ on } \text{free}(t_G^{\psi,i})\}\Big).$$

Moreover, since $p = p'_1 \cdot p''_1 \cdot \dots \cdot p'_m \cdot p''_m$, we can decompose the first term for computing $\left[\!\left[t_G^{\psi,i}\right]\!\right]^{(\mathfrak{A},\beta)}$ even further into factors of the form

$$\sum_S \{p'^{\mathfrak{A}}_j[\bar{v}'] \mid \bar{v}' \in M^{\mathfrak{A}}_{\vartheta'_i}, \bar{v}' \text{ agrees with } \beta \text{ on } \text{free}(\vartheta'_i) \setminus \text{vars}(p'_j)\}$$

and

$$\sum_S \{p''^{\mathfrak{A}}_j[\bar{v}''] \mid \bar{v}'' \in M^{\mathfrak{A}}_{\vartheta''_i}, \bar{v}'' \text{ agrees with } \beta \text{ on } \text{free}(\vartheta''_i) \setminus \text{vars}(p''_j)\}$$

for $j \in [m]$. Therefore, $t_G^{\psi,i}(\bar{x})$ is equivalent to

$$\prod_{j=1}^{m} \big(t'_j(\bar{x}') \cdot t''_j(\bar{x}'')\big) - \sum_{H \in \mathcal{H}_k} t_H(\bar{x})$$

for

$$\begin{aligned} t'_j(\bar{x}') &\coloneqq \sum p'_j . \vartheta'_i(\bar{x}'), \\ t''_j(\bar{x}'') &\coloneqq \sum p''_j . \vartheta''_i(\bar{x}''), \text{ and} \\ t_H(\bar{x}) &\coloneqq \sum p . \big(\vartheta'_i(\bar{x}') \wedge \vartheta''_i(\bar{x}'') \wedge \delta^{\sigma}_{H,2r+1}(\bar{x})\big). \end{aligned}$$

By the induction hypothesis $(*)_c$, each of the terms t'_i, t''_i, and t_H is equivalent to a cl term of radius r. Hence, also $t_G^{\psi,i}$ is equivalent to a cl term of radius r. This completes the proof of Lemma 5.16. $\square$

As a consequence of Lemma 5.16, we obtain the following result.

Lemma 5.17. *Let $s \in \mathbb{N}$ and let $\chi_1, \ldots, \chi_s$ be arbitrary sentences that can be evaluated in $(\sigma, \mathbf{W})$-structures.*[5] *Let $r \in \mathbb{N}$, $k \in \mathbb{N}_{\geqslant 1}$, and let $\bar{x} = (x_1, \ldots, x_k)$ be a tuple of k pairwise distinct variables. Let $\varphi(\bar{x})$ be a Boolean combination of the sentences $\chi_1, \ldots, \chi_s$ and of r-local $\mathsf{FOW}_1(\mathbb{P})[\sigma, \mathbb{S}, \mathbf{W}]$-formulas. Consider an $\mathbb{S}$-term $t(y_1, \ldots, y_\ell)$ of the form $\sum p.\varphi(x_1, \ldots, x_k)$, where p is a $\mathbf{W}$-product, $\ell \in \mathbb{N}$, and $\{y_1, \ldots, y_\ell\} \subseteq \{x_1, \ldots, x_k\}$. For every $I \subseteq [s]$, there is a cl term $\hat{t}_I$ (of radius $\leqslant r$ and width $\leqslant k$) such that the following holds. For every $(\sigma, \mathbf{W})$-structure $\mathfrak{A}$, there is exactly one set $J \subseteq [s]$ such that $\mathfrak{A} \models \chi_J$ for*

$$\chi_J \coloneqq \bigwedge_{j \in J} \chi_j \wedge \bigwedge_{j \in [s] \setminus J} \neg \chi_j,$$

and, for this set J, we have $\hat{t}_J^{\mathfrak{A}}[\bar{v}] = t^{\mathfrak{A}}[\bar{v}]$ for every $\bar{v} \in \big(U(\mathfrak{A})\big)^\ell$.

Furthermore, there is an algorithm which, upon input of r, t, and J, constructs $\hat{t}_J$.

Proof. We can assume w.l.o.g. that $\varphi(\bar{x})$ is of the form

$$\bigvee_{J \subseteq [s]} (\chi_J \wedge \psi_J(\bar{x})),$$

where, for each $J \subseteq [s]$, $\psi_J(\bar{x})$ is an r-local $\mathsf{FOW}_1(\mathbb{P})[\sigma, \mathbb{S}, \mathbf{W}]$-formula. For every $J \subseteq [s]$ let $\hat{t}_J$ be the cl term obtained by Lemma 5.16 for the term $t_J \coloneqq \sum p.\psi_J(\bar{x})$. Now consider an arbitrary $J \subseteq [s]$ and a σ-structure $\mathfrak{A}$ with $\mathfrak{A} \models \chi_J$. Then, for every tuple $\bar{v} \in \big(U(\mathfrak{A})\big)^\ell$, we have $t^{\mathfrak{A}}[\bar{v}] = \big(\sum p.\psi(\bar{x})\big)^{\mathfrak{A}}[\bar{v}] = \big(\sum p.\psi_J(\bar{x})\big)^{\mathfrak{A}}[\bar{v}] = \hat{t}_J^{\mathfrak{A}}[\bar{v}]$. □

5.4.2 *A connected local normal form for FOW$_1$*

From now on, we assume that whenever $\mathbb{S}$ contains the integer ring $(\mathbb{Z}, +, \cdot)$, there is a weight symbol $\mathtt{one} \in \mathbf{W}$ of arity 1 and type $\mathbb{Z}$ such that, in every $(\sigma, \mathbf{W})$-structure $\mathfrak{A}$ that we consider, we have $\mathtt{one}(v) = 1 \in \mathbb{Z}$ for all $v \in U(\mathfrak{A})$. By combining Lemma 5.16 with the Gaifman normal form for FOW_1 (Theorem 5.12), we obtain the following normal form, which may be of independent interest. *cl Normal form*

Theorem 5.18 (cl Normal form)**.** *Let $\mathbb{S}$ contain the integer ring $(\mathbb{Z}, +, \cdot)$. Every formula $\varphi(\bar{x})$ of $\mathsf{FOW}_1(\mathbb{P})[\sigma, \mathbb{S}, \mathbf{W}]$ is equivalent to a Boolean combination $\varphi'(\bar{x})$ of*

(a) local $\mathsf{FOW}_1(\mathbb{P})[\sigma, \mathbb{S}, \mathbf{W}]$-formulas $\psi(\bar{x})$,

(b) local aggregation sentences in $\mathsf{FOW}_1(\mathbb{P})[\sigma, \mathbb{S}, \mathbf{W}]$, and

5 We do not restrict attention to $\mathsf{FOW}_1(\mathbb{P})[\sigma, \mathbb{S}, \mathbf{W}]$-sentences here—the χ_js may be sentences of any logic, e.g., $\mathsf{FOWA}(\mathbb{P})[\sigma, \mathbb{S}, \mathbf{W}]$.

(c) statements of the form "$g \geqslant 1$", for a ground cl term g of type $\mathbb{Z}$.

Furthermore, there is an algorithm which, given an $\mathsf{FOW}_1(\mathbb{P})[\sigma, \mathbb{S}, \mathbf{W}]$-formula $\varphi(\bar{x})$, transforms it into an equivalent such formula $\varphi'(\bar{x})$ and outputs the radius of each ground cl term in φ' as well as a number r such that every local formula in φ' is r-local.

Proof. By Theorem 5.12, it suffices to translate a basic local sentence into a statement of the form "$g \geqslant 1$" for a ground cl term g of type $\mathbb{Z}$. For a basic local sentence $\chi := \exists y_1 \cdots \exists y_k\, \vartheta(y_1, \ldots, y_k)$ with

$$\vartheta(y_1, \ldots, y_k) := \bigwedge_{1 \leqslant i < j \leqslant k} \mathrm{dist}(y_i, y_j) > 2r \wedge \bigwedge_{i=1}^{k} \psi(y_i),$$

we use the ground term $g_\chi := \sum p.\vartheta(y_1, \ldots, y_k)$ for the $\mathbf{W}$-product $p := \mathsf{one}(y_1) \cdot \cdots \cdot \mathsf{one}(y_k)$. Note that $\vartheta(y_1, \ldots, y_k)$ is r-local around its free variables. Hence, by Lemma 5.16, we obtain a ground cl term $\hat{g}_\chi$ such that $\hat{g}_\chi^{\mathfrak{A}} = g_\chi^{\mathfrak{A}}$ for every $(\sigma, \mathbf{W})$-structure $\mathfrak{A}$. Furthermore, we have $\mathfrak{A} \models \chi \iff g_\chi^{\mathfrak{A}} \geqslant 1 \iff \hat{g}_\chi^{\mathfrak{A}} \geqslant 1$. This completes the proof of Theorem 5.18. $\square$

We use the notion *cl normal form* to denote the formulas $\varphi'(\bar{x})$ provided by Theorem 5.18. Note that cl normal forms do not necessarily belong to $\mathsf{FOW}_1(\mathbb{P})[\sigma, \mathbb{S}, \mathbf{W}]$, but can be viewed as formulas in $\mathsf{FOWA}(\mathbb{P})[\sigma, \mathbb{S}, \mathbf{W}]$, where $\mathbb{P}$ contains a unary predicate $\mathsf{P}_{\geqslant 1}$ of type $\mathbb{Z}$ with $[\![\mathsf{P}_{\geqslant 1}]\!] := \mathbb{N}_{\geqslant 1}$. Then, statements of the form "$g \geqslant 1$" can be expressed via $\mathsf{P}_{\geqslant 1}(g)$.

5.4.3 *A decomposition of FOWA$_1$-expressions*

Our decomposition of $\mathsf{FOWA}_1(\mathbb{P})[\sigma, \mathbb{S}, \mathbf{W}]$ utilises the cl normal form from Theorem 5.18 and is based on an induction on the aggregation depth $d_{\mathrm{ag}}(\xi)$ of a formula or term ξ, that is, the maximal nesting depth of term constructions of the form $\sum p.\psi$ (i.e., constructions by rule (10) of Definition 5.3). The base case of our decomposition of $\mathsf{FOWA}_1(\mathbb{P})[\sigma, \mathbb{S}, \mathbf{W}]$ is provided by the following lemma. The proof uses Theorem 5.18.

Lemma 5.19. *Let $\mathbb{S}$ contain the integer ring $(\mathbb{Z}, +, \cdot)$. Let φ be a formula in $\mathsf{FOWA}_1(\mathbb{P})[\sigma, \mathbb{S}, \mathbf{W}]$ of the form $\mathsf{P}(t_1, \ldots, t_m)$ with $\mathsf{P} \in \mathbb{P}$, $m = \mathrm{ar}(\mathsf{P})$, and where $t_1, \ldots, t_m$ are $\mathbb{S}$-terms of aggregation depth at most 1. Then, φ is equivalent to a Boolean combination of*

(a) formulas of the form $\mathsf{P}(t'_1, \ldots, t'_m)$, for cl terms $t'_1, \ldots, t'_m$ with free variables $\mathrm{free}(t'_i) = \mathrm{free}(t_i)$ for all $i \in [m]$,

(b) local aggregation sentences in $\mathsf{FOW}_1(\mathbb{P})[\sigma, \mathbb{S}, \mathbf{W}]$, and

(c) statements of the form "$g \geqslant 1$" for ground cl terms g of type $\mathbb{Z}$.

Also, there is an algorithm which transforms an input formula φ into such a Boolean combination φ', and which outputs the radius of each cl term and each local formula in φ'.

Proof. From the definition of FOWA$_1$ (Definition 5.4), we know that the free variables of φ are either $\mathrm{free}(\varphi) = \emptyset$ or $\mathrm{free}(\varphi) = \{x\}$ for some variable x. Furthermore, we know that for every $i \in [m]$, the $\mathbb{S}$-term t_i is built by using rules (7)–(9) and $\mathbb{S}$-terms ϑ of the form $\sum p.\vartheta'$ for a $\mathbf{W}$-product p such that $\mathrm{free}(\vartheta') \setminus \mathrm{vars}(p) \subseteq \{x\}$. Let Θ be the set of all these $\mathbb{S}$-terms ϑ and let Θ' be the set of all the according formulas ϑ'.

By assumption, we have $d_{\mathrm{ag}}(\varphi) \leqslant 1$. Therefore, every $\vartheta' \in \Theta'$ has aggregation depth 0. Thus, every $\vartheta' \in \Theta'$ is an $\mathrm{FOW}_1(\mathbb{P})[\sigma, \mathbb{S}, \mathbf{W}]$-formula. By Theorem 5.18, for each ϑ' in Θ', we obtain an equivalent formula $\varphi_{\vartheta'}$ in cl normal form. Let Φ be the set of all these $\varphi_{\vartheta'}$.

For each ϑ' in Θ', the formula $\varphi_{\vartheta'}$ is a Boolean combination of (a) $\mathrm{FOW}_1(\mathbb{P})[\sigma, \mathbb{S}, \mathbf{W}]$-formulas that are local around the free variables of ϑ', (b) local aggregation sentences in $\mathrm{FOW}_1(\mathbb{P})[\sigma, \mathbb{S}, \mathbf{W}]$, and (c) statements of the form "$g \geqslant 1$" for a ground cl term g of type $\mathbb{Z}$.

Let $\chi_1, \dots, \chi_s$ be a list of all statements of the forms (b) or (c) such that each formula in Φ is a Boolean combination of statements in $\{\chi_1, \dots, \chi_s\}$ and of $\mathrm{FOW}_1(\mathbb{P})[\sigma, \mathbb{S}, \mathbf{W}]$-formulas that are local around their free variables. For every $J \subseteq [s]$ let $\chi_J \coloneqq \bigwedge_{j \in J} \chi_j \wedge \bigwedge_{j \in [s] \setminus J} \neg\chi_j$. Let $r \in \mathbb{N}$ be such that each of the local $\mathrm{FOW}_1(\mathbb{P})[\sigma, \mathbb{S}, \mathbf{W}]$-formulas that occur in a formula in Φ is r-local around its free variables. For each ϑ in Θ of the form $\sum p.\vartheta'$, we apply Lemma 5.17 to the term

$$t_{\vartheta'} \coloneqq \sum p.\varphi_{\vartheta'}$$

and obtain for every $J \subseteq [s]$ a cl term $\hat{t}_{\vartheta',J}$ for which the following is true. If $\mathrm{free}(\vartheta') = \emptyset$, then $(\vartheta')^{\mathfrak{A}} = (\hat{t}_{\vartheta',J})^{\mathfrak{A}}$ for every $(\sigma, \mathbf{W})$-structure $\mathfrak{A}$ with $\mathfrak{A} \models \chi_J$. Otherwise, if $\mathrm{free}(\vartheta') = \{x\}$, then $(\vartheta')^{\mathfrak{A}}[v] = (\hat{t}_{\vartheta',J})^{\mathfrak{A}}[v]$ for every $(\sigma, \mathbf{W})$-structure $\mathfrak{A}$ with $\mathfrak{A} \models \chi_J$ and for every $v \in U(\mathfrak{A})$.

Thus, for each $J \subseteq [s]$, we have

$$(\chi_J \ \wedge\ P(t_1, \dots, t_m)) \quad \equiv \quad (\chi_J \ \wedge\ P(t_{1,J}, \dots, t_{m,J})),$$

where, for every $i \in [m]$, we let $t_{i,J}$ be the cl term obtained from t_i by replacing each occurrence of a term $\vartheta' \in \Theta'$ with the term $\hat{t}_{\vartheta',J}$. In summary, we obtain the following:

$$\begin{aligned} \varphi = P(t_1, \dots, t_m) \quad &\equiv \quad \bigvee_{J \subseteq [s]} (\chi_J \ \wedge\ P(t_1, \dots, t_m)) \\ &\equiv \quad \bigvee_{J \subseteq [s]} (\chi_J \ \wedge\ P(t_{1,J}, \dots, t_{m,J})) \eqqcolon \varphi'. \end{aligned}$$

The formula χ_J is a Boolean combination of local aggregation sentences in $\mathrm{FOW}_1(\mathbb{P})[\sigma, \mathbb{S}, \mathbf{W}]$ and of statements of the form "$g \geqslant 1$" for ground cl terms g of type $\mathbb{Z}$. Furthermore, all terms $t_{i,J}$ are cl terms with $\mathrm{free}(t_{i,J}) \subseteq \mathrm{free}(t_i)$, and we can easily modify them to achieve that $\mathrm{free}(t_{i,J}) = \mathrm{free}(t_i)$. This completes the proof of Lemma 5.19. □

We are now ready for the decomposition theorem for FOWA_1, which can be viewed as a generalisation of the decomposition theorem for FOC_1 provided in [56].

Theorem 5.20 (Decomposition of FOWA_1). *Let $\mathbb{S}$ contain the integer ring $(\mathbb{Z}, +, \cdot)$. Let z be a fixed variable in* vars. *For every $d \in \mathbb{N}$ and for every $\mathsf{FOWA}_1(\mathbb{P})[\sigma, \mathbb{S}, \mathbf{W}]$-formula $\varphi(\bar{x})$ of aggregation depth $d_{ag}(\varphi) = d$, there exists a sequence $(L_1, \ldots, L_{d+1}, \varphi')$ with the following properties.*

(1) For every $i \in [d+1]$, we have $L_i = (\tau_i, \iota_i)$, where τ_i is a finite set of relation symbols of arity $\leqslant 1$ that do not belong to $\sigma_{i-1} := \sigma \cup \bigcup_{j=1}^{i-1} \tau_j$. Furthermore, ι_i is a mapping that associates with every relation symbol $R \in \tau_i$ a formula $\iota_i(R)$

(a) of the form $P(t_1, \ldots, t_m)$, where $P \in \mathbb{P}$, $m = \mathrm{ar}(P)$, and $t_1, \ldots, t_m$ are cl terms of signature σ_{i-1} with $\mathrm{free}(t_j) \subseteq \{z\}$ for each $j \in [m]$, or

(b) that is a local aggregation sentence in $\mathsf{FOW}_1(\mathbb{P})[\sigma_{i-1}, \mathbb{S}, \mathbf{W}]$ or a statement of the form "$g \geqslant 1$" for a ground cl term g of signature σ_{i-1} and of type $\mathbb{Z}$.

If R has arity 0, then $\iota_i(R)$ has no free variable. If R has arity 1, then z is the unique free variable of $\iota_i(R)$ (thus, $\iota_i(R)$ is of the form (a)).

(2) $\varphi'(\bar{x})$ is a Boolean combination of (a) $\mathsf{FOW}_1(\mathbb{P})[\sigma_{d+1}, \mathbb{S}, \mathbf{W}]$-formulas $\psi(\bar{x})$ that are local around their free variables $\bar{x}$, where $\sigma_{d+1} := \sigma \cup \bigcup_{i=1}^{d+1} \tau_i$, and (b) statements of the form $R()$, where R is a 0-ary relation symbol in σ_{d+1}. In case that $\mathrm{free}(\varphi) = \emptyset$, φ' only contains statements of the latter form.

(3) For every $(\sigma, \mathbf{W})$-interpretation $\mathcal{I} = (\mathfrak{A}, \beta)$, we have $\mathcal{I} \models \varphi$ if and only if $\mathcal{I}_{d+1} \models \varphi'$, where $\mathcal{I}_{d+1} = (\mathfrak{A}_{d+1}, \beta)$, and $\mathfrak{A}_{d+1}$ is the σ_{d+1}-expansion of $\mathfrak{A}$ defined as follows. We set $\mathfrak{A}_0 := \mathfrak{A}$. Furthermore, for every $i \in [d+1]$, $\mathfrak{A}_i$ is the σ_i-expansion of $\mathfrak{A}_{i-1}$, where for every unary $R \in \tau_i$, we have $R^{\mathfrak{A}_i} := \{ v \in U(\mathfrak{A}) \mid (\mathfrak{A}_{i-1}, v) \models \iota_i(R) \}$ and for every 0-ary $R \in \tau_i$ we have $R^{\mathfrak{A}_i} := \{()\}$ if $\mathfrak{A}_{i-1} \models \iota_i(R)$, and $R^{\mathfrak{A}_i} := \emptyset$ if $\mathfrak{A}_{i-1} \not\models \iota_i(R)$.

Moreover, there is an algorithm which constructs such a sequence $D = (L_1, \ldots, L_{d+1}, \varphi')$ for an input formula φ and outputs the radius of each cl term in D as well as a number r such that every local formula in φ' is r-local around its free variables.

Proof. We proceed by induction on i to construct for all $i \in [0, d]$ a tuple $L_i = (\tau_i, \iota_i)$ and a $\mathsf{FOWA}_1(\mathbb{P})[\sigma_i, \mathbb{S}, \mathbf{W}]$-formula $\varphi_i(\bar{x})$ of aggregation depth $(d-i)$, such that for every $(\sigma, \mathbf{W})$-interpretation $\mathcal{I} = (\mathfrak{A}, \beta)$ and the interpretation $\mathcal{I}_i := (\mathfrak{A}_i, \beta)$, we have $\mathcal{I} \models \varphi \iff \mathcal{I}_i \models \varphi_i$. The last step $i = d+1$ will be handled separately.

For $i = 0$, we are done by letting $\tau_0 := \emptyset$, $\sigma_0 := \sigma$, $\varphi_0 := \varphi$, and ι_0 be the mapping with empty domain.

Now assume that for some $i < d$, we have already constructed $L_i = (\tau_i, \iota_i)$ and φ_i. To construct $L_{i+1} = (\tau_{i+1}, \iota_{i+1})$ and φ_{i+1}, we proceed as follows. Let Π be the set of all $\mathsf{FOWA}_1(\mathbb{P})[\sigma_i, \mathbb{S}, \mathbf{W}]$-formulas of aggregation depth $\leqslant 1$ of the form $\mathsf{P}(t_1, \ldots, t_m)$, for $\mathsf{P} \in \mathbb{P}$, that occur in φ_i. Consider an arbitrary formula π in Π of the form $\mathsf{P}(t_1, \ldots, t_m)$. From Definition 5.4, we know that there is a variable x such that $\text{free}(t_j) \subseteq \{x\}$ for every $j \in [m]$. By Lemma 5.19, π is equivalent to a Boolean combination π' of

(a) formulas of the form $\mathsf{P}(t'_1, \ldots, t'_m)$, for cl terms $t'_1, \ldots, t'_m$ of signature σ_i, where $\text{free}(t'_j) = \text{free}(t_j) \subseteq \{x\}$ for each $j \in [m]$,

(b) local aggregation sentences in $\mathsf{FOW}_1(\mathbb{P})[\sigma_i, \mathbb{S}, \mathbf{W}]$, and

(c) statements of the form "$g \geqslant 1$" for ground cl terms g of signature σ_i.

For each statement χ of the form (b) or (c), and for statements χ of the form (a) with $\text{free}(\chi) = \emptyset$, we include into τ_{i+1} a 0-ary relation symbol R_χ, we replace each occurrence of χ in π' with the new atomic formula $R_\chi()$, and we let $\iota_{i+1}(R_\chi) := \chi$. For each statement χ in π of the form (a) with $\text{free}(\chi) = \{x\}$, we include into τ_{i+1} a unary relation symbol R_χ, we replace each occurrence of χ in π' with the new atomic formula $R_\chi(x)$, and we let $\iota_{i+1}(R_\chi)$ be the formula obtained from χ by consistently replacing every free occurrence of the variable x with the variable z. We write π'' for the resulting formula. Clearly, π'' is of signature $\sigma_{i+1} := \sigma_i \cup \tau_i$, it has aggregation depth 0, and for every σ-interpretation $\mathcal{I} = (\mathfrak{A}, \beta)$ and $\mathcal{I}_i := (\mathfrak{A}_i, \beta)$ and $\mathcal{I}_{i+1} := (\mathfrak{A}_{i+1}, \beta)$, we have $\mathcal{I}_i \models \pi$ if and only if $\mathcal{I}_{i+1} \models \pi''$.

The induction step is completed by letting φ_{i+1} be the formula obtained from φ_i by replacing every occurrence of a formula $\pi \in \Pi$ with the formula π''. It can easily be verified that φ_{i+1} is an $\mathsf{FOWA}_1(\mathbb{P})[\sigma_{i+1}, \mathbb{S}, \mathbf{W}]$-formula of aggregation depth $d_{ag}(\varphi_i) - 1 = \big((d-i)-1\big) = \big(d-(i+1)\big)$ and that $\mathcal{I}_i \models \varphi_i \iff \mathcal{I}_{i+1} \models \varphi_{i+1}$.

By the above induction, we have constructed $L_1, \ldots, L_d$ and an $\mathsf{FOWA}_1(\mathbb{P})[\sigma_d, \mathbb{S}, \mathbf{W}]$-formula φ_d of aggregation depth 0. Hence, φ_d is an $\mathsf{FOW}_1(\mathbb{P})[\sigma_d, \mathbb{S}, \mathbf{W}]$-formula. Theorem 5.18 yields an equivalent formula $\tilde{\varphi}$ of signature σ_d in cl normal form. That is, $\tilde{\varphi}$ is a Boolean combination of

(a) local $\mathsf{FOW}_1(\mathbb{P})[\sigma_d, \mathbb{S}, \mathbf{W}]$-formulas

(b) local aggregation sentences in $\mathsf{FOW}_1(\mathbb{P})[\sigma_d, \mathbb{S}, \mathbf{W}]$, and

(c) statements of the form "$g \geqslant 1$", for a ground cl term g of type $\mathbb{Z}$ and of signature σ_d.

For each statement χ of the form (b) or (c), we include into τ_{d+1} a new relation symbol R_χ of arity 0, we replace each occurrence of χ in $\tilde{\varphi}$ with the new atomic formula $R_\chi()$, and we let $\iota_{d+1}(R_\chi) := \chi$. Letting φ' be the resulting formula $\tilde{\varphi}$ completes the proof. $\square$

cl Decomposition

We call the sequence $D = (L_1, \dots, L_{d_{ag}(\varphi)+1}, \varphi')$ that Theorem 5.20 provides for a formula φ in $\mathsf{FOWA}_1(\mathbb{P})[\sigma, S, \mathbf{W}]$ a *cl decomposition* of φ. By combining Theorem 5.20 with Lemmas 5.15 and 5.16, we can now prove Theorem 5.13, which provides for every FOWA_1-formula φ an FOW_1-formula φ' over an enriched signature σ_φ such that φ holds in a σ-structure $\mathfrak{A}$ if and only if φ' holds in a certain σ_φ-expansion $\mathfrak{A}_\varphi$ of $\mathfrak{A}$. Furthermore, it provides an algorithm to compute $\mathfrak{A}_\varphi$ from $\mathfrak{A}$ and φ.

Proof of Theorem 5.13. First, we use Theorem 5.20 to compute a cl decomposition $D = (L_1, \dots, L_{d+1}, \varphi')$ of φ, for $d \coloneqq d_{ag}(\varphi)$. This formula φ' is the desired formula. We let $\sigma_\varphi \coloneqq \sigma_{d+1} \coloneqq \sigma \cup \bigcup_{i=1}^{d+1} \tau_i$ and $\mathfrak{A}_\varphi \coloneqq \mathfrak{A}_{d+1}$. To compute $\mathfrak{A}_\varphi$, we proceed as follows.

Let $\mathfrak{A}_0 \coloneqq \mathfrak{A}$. For each $i \in [d+1]$, compute the σ_i-expansion of $\mathfrak{A}_{i-1}$. To achieve this, consider for each $R \in \tau_i$ the formula $\iota_i(R)$. This formula is of signature σ_{i-1} and

(a) of the form $P(t_1, \dots, t_m)$ for a $P \in \mathbb{P}$ and cl terms $t_1, \dots, t_m$,

(b) a local aggregation sentence of the form $\big(s = \sum w(\bar{y}).\lambda(\bar{y})\big)$ for a local $\mathsf{FOW}_1(\mathbb{P})[\sigma_{i-1}, S, \mathbf{W}]$-formula λ, or

(c) a statement of the form "$g \geqslant 1$" for a ground cl term g of type $\mathbb{Z}$.

By Lemma 5.16, the term $\sum w(\bar{y}).\lambda(\bar{y})$ from a formula of form (b) is equivalent to a ground cl term. Thus, in all three cases, $\iota_i(R)$ is a simple statement that concerns one or several cl terms and that involves at most one free variable. By using Lemma 5.15, we can compute, in time $|\mathfrak{A}| \log |\mathfrak{A}| \cdot d^{\mathcal{O}(1)}$ under the logarithmic-cost measure and in time $|\mathfrak{A}| \cdot d^{\mathcal{O}(1)}$ under the uniform-cost measure, for each such cl term t the values $t^{\mathfrak{A}}[v]$ for all $v \in U(\mathfrak{A})$ (respectively the value $t^{\mathfrak{A}}$, if t is a ground term). Then, we combine the values and use a $\mathbb{P}$-oracle to check for each $v \in U(\mathfrak{A})$ whether $\iota_i(R)$ is satisfied by $(\mathfrak{A}_{i-1}, v)$, and we store the new relation $R(\mathfrak{A}_i)$ accordingly.

This completes the computation of $\mathfrak{A}_\varphi$ and thus the proof of Theorem 5.13. □

6 LEARNING LOGICS WITH WEIGHT AGGREGATION

In this chapter, we generalise the results of Grohe and Ritzert [55] for first-order logic on relational structures, which we described in Chapter 3, to logics with weight aggregation on weighted structures. To do so, in Section 6.1, we describe a framework to learn concepts on weighted structures, without restricting ourselves to a specific logic. As in Chapters 3 and 4, we are interested in concepts that can be learned in sublinear time. Hence, we describe properties of the formulas (or rather of the sets of formulas) the hypotheses are based upon that are sufficient to yield learning results with suitable running times. We prove learning results for consistent learning in Section 6.2 and for PAC learning in Section 6.3. Finally, in Section 6.4, we use the introduced framework and the locality properties from Chapter 5 to obtain learning results for concepts that can be described in the weight-aggregation logic FOWA_1. More specifically, we show that concepts definable in FOWA_1 over weighted structures of at most polylogarithmic degree are agnostically PAC-learnable in polylogarithmic time after pseudo-linear time preprocessing.

6.1 LEARNING WITH PRECOMPUTATION

Throughout this chapter, fix a collection $\mathbb{S}$ of rings and/or abelian groups, an $\mathbb{S}$-predicate collection $(\mathbb{P}, \mathrm{ar}, \mathrm{type}, [\![\cdot]\!])$, and a finite set $\mathbf{W}$ of weight symbols. Furthermore, fix numbers $k, \ell \in \mathbb{N}$. Let L be a logic (e.g. FO, $\mathsf{FOW}_1(\mathbb{P})$, $\mathsf{FOWA}_1(\mathbb{P})$, $\mathsf{FOWA}(\mathbb{P})$), let σ be a signature, and let $\Phi \subseteq \mathsf{L}[\sigma, \mathbb{S}, \mathbf{W}]$ be a set of formulas $\varphi(\bar{x}, \bar{y})$ with $|\bar{x}| = k$ and $|\bar{y}| = \ell$. Analogously to the learning problems from the previous chapters, for a $(\sigma, \mathbf{W})$-structure $\mathfrak{A}$, we consider the instance space $X = \big(U(\mathfrak{A})\big)^k$ and concepts from the concept class

$$\mathcal{H}_{\Phi,k,\ell}(\mathfrak{A}) = \left\{ h^{\mathfrak{A}}_{\varphi,\bar{w}} \;\middle|\; \varphi \in \Phi, \bar{w} \in \big(U(\mathfrak{A})\big)^{\ell} \right\},$$

where $h^{\mathfrak{A}}_{\varphi,\bar{w}}(\bar{x})\colon \big(U(\mathfrak{A})\big)^k \to \{0,1\}$ is the mapping that maps a tuple $\bar{v} \in \big(U(\mathfrak{A})\big)^k$ to $[\![\varphi(\bar{v}, \bar{w})]\!]^{\mathfrak{A}}$.

As in Chapters 3 and 4, instead of allowing random access to the background structure, we limit our algorithms to have only *local access*.

In many applications, the same background structure is used multiple times to learn different concepts. Hence, similarly to the approaches in [47, 53], we allow a precomputation step to enrich the background structure with additional information. That is, instead of learning on a $(\sigma, \mathbf{W})$-structure $\mathfrak{A}$, we use an enriched $(\sigma', \mathbf{W})$-structure $\mathfrak{A}'$, which has the same universe as $\mathfrak{A}$, but $\sigma' \supseteq \sigma$ contains additional relation symbols. The hypotheses we compute may make use of this additional information and thus, instead of representing them via formulas from the fixed set Φ, we consider a set Φ' of formulas of signature σ'. These formulas may even belong to a logic L' different from L.

For the sets of formulas $\Phi \subseteq L[\sigma, \mathbb{S}, \mathbf{W}]$ and $\Phi' \subseteq L'[\sigma', \mathbb{S}, \mathbf{W}]$, we require that for every $(\sigma, \mathbf{W})$-structure $\mathfrak{A}$, there is a $(\sigma', \mathbf{W})$-structure $\mathfrak{A}'$ with $U(\mathfrak{A}') = U(\mathfrak{A})$ such that $\mathcal{H}_{\Phi,k,\ell}(\mathfrak{A}) \subseteq \mathcal{H}_{\Phi',k,\ell}(\mathfrak{A}')$, so every concept that can be defined on $\mathfrak{A}$ using Φ can also be defined on $\mathfrak{A}'$ using Φ'. In our algorithms, we assume that we are given local access to the precomputed structure $\mathfrak{A}'$.

6.2 CONSISTENT LEARNING

The consistent-learning problem is formally defined as follows.

LEARN-CONSISTENT-PRECOMP(k, Φ, Φ')

Input: $(\sigma', \mathbf{W})$-structure $\mathfrak{A}'$ that has been computed from the input $(\sigma, \mathbf{W})$-structure $\mathfrak{A}$ with $U(\mathfrak{A}') = U(\mathfrak{A})$, training sequence $T \in \Big(\big(U(\mathfrak{A})\big)^k \times \{0,1\}\Big)^m$

Problem: Return a formula $\varphi' \in \Phi'$ and a tuple $\bar{w}' \in \big(U(\mathfrak{A})\big)^\ell$ such that the hypothesis $h^{\mathfrak{A}'}_{\varphi',\bar{w}'}$ is consistent with T. The algorithm may reject if there is no formula $\varphi \in \Phi$ and tuple $\bar{w} \in \big(U(\mathfrak{A})\big)^\ell$ such that the hypothesis $h^{\mathfrak{A}}_{\varphi,\bar{w}}$ is consistent with T.

Next, we examine requirements for Φ and Φ' that help us solve LEARN-CONSISTENT-PRECOMP efficiently. Following the approach by Grohe and Ritzert presented in Section 3.3, to obtain algorithms that run in sublinear time, we study concepts that can be represented via a set of *local* formulas Φ with a finite set Φ' of normal forms. Using Feferman-Vaught decompositions and the locality of the formulas, we can then limit the search space for the parameters to those that are in a certain neighbourhood of the training sequence. Recall that Φ is a set of formulas $\varphi(\bar{x}, \bar{y})$ in $L[\sigma, \mathbb{S}, \mathbf{W}]$ with $|\bar{x}| = k$ and $|\bar{y}| = \ell$. In the following, we require Φ to have the following property.

Property 6.1. There are a signature σ', a logic L', an $r \in \mathbb{N}$, and a finite set of r-local formulas $\Phi' \subseteq L'[\sigma', \mathbb{S}, \mathbf{W}]$ such that the following hold.

(1) For every $(\sigma, \mathbf{W})$-structure $\mathfrak{A}$, there is a $(\sigma', \mathbf{W})$-structure $\mathfrak{A}'$ with $U(\mathfrak{A}') = U(\mathfrak{A})$ such that, for every formula $\varphi(\bar{x}, \bar{y}) \in \Phi$, there is a formula $\varphi'(\bar{x}, \bar{y}) \in \Phi'$ with $\mathfrak{A} \models \varphi[\bar{v}, \bar{w}] \iff \mathfrak{A}' \models \varphi'[\bar{v}, \bar{w}]$ for all $\bar{v} \in \big(U(\mathfrak{A})\big)^k$, $\bar{w} \in \big(U(\mathfrak{A})\big)^\ell$.

(2) Every $\varphi' \in \Phi'$ has, for every partition $(\bar{z}_1; \bar{z}_2)$ of the free variables of φ', a Feferman-Vaught decomposition in Φ' w.r.t. $(\bar{z}_1; \bar{z}_2)$.

(3) The set Φ' is, up to equivalence, closed under Boolean combinations. That is, for all $\varphi'_1, \varphi'_2 \in \Phi'$, the set Φ' contains formulas equivalent to $\neg\varphi'_1$ and to $(\varphi'_1 \vee \varphi'_2)$.

In the following, let Φ' be the set of formulas mentioned in Property 6.1. Moreover, for a $(\sigma, \mathbf{W})$-structure $\mathfrak{A}$, we call the structure $\mathfrak{A}'$ from Property 6.1 the *associated structure*, and we say that the structure $\mathfrak{A}'$ is *associated with* the structure $\mathfrak{A}$. Our first main result for this chapter is that Property 6.1 suffices to solve the problem LEARN-CONSISTENT-PRECOMP.

Theorem 6.2 (Consistent learning with precomputation)**.** *There is an algorithm that solves* LEARN-CONSISTENT-PRECOMP(k, Φ, Φ') *with local access to a structure* $\mathfrak{A}'$ *that is associated with the input structure* $\mathfrak{A}$ *in time* $f_{\Phi'}(\mathfrak{A}') \cdot \big(\log n + d + m\big)^{\mathcal{O}(1)}$ *under the logarithmic-cost measure and in time* $f_{\Phi'}(\mathfrak{A}') \cdot \big(d + m\big)^{\mathcal{O}(1)}$ *under the uniform-cost measure, where* $\mathfrak{A}$, $\mathfrak{A}'$, Φ, *and* Φ' *are as described in Property 6.1,* m *is the number of training examples,* n *and* d *are the size and the degree of* $\mathfrak{A}'$*, and* $f_{\Phi'}(\mathfrak{A}')$ *is an upper bound on the time complexity of model checking for formulas in* Φ' *on* $\mathfrak{A}'$*.*

Let $\mathfrak{A}$ be a $(\sigma, \mathbf{W})$-structure and let $\mathfrak{A}'$ and Φ' be as in Property 6.1. To prove Theorem 6.2, we present an algorithm that follows similar ideas as the algorithms described in Section 3.3 and Chapter 4. While the set of possible formulas Φ' already has constant size, we have to reduce the parameter space to obtain an algorithm that runs in sublinear time. Since the formulas in Φ' are r-local, we show that it suffices to consider parameters in a neighbourhood of the training sequence with a fixed radius.

For a training sequence $T = \big((\bar{v}_1, \lambda_1), \ldots, (\bar{v}_m, \lambda_m)\big)$ and a radius $r' \in \mathbb{N}$, let $N^{\mathfrak{A}}_{r'}(T) := \bigcup_{i \in [m]} N^{\mathfrak{A}}_{r'}(\bar{v}_i)$.

Lemma 6.3. *Let* $T \in \Big(\big(U(\mathfrak{A})\big)^k \times \{0,1\}\Big)^m$ *be a training sequence that is consistent with some classifier in* $\mathcal{H}_{\Phi',k,\ell}(\mathfrak{A}')$*. Then there are a formula* $\varphi'(\bar{x}, \bar{y}) \in \Phi'$ *and a tuple* $\bar{w}' \in \big(N^{\mathfrak{A}'}_{(2r+1)\ell}(T)\big)^\ell$ *such that the hypothesis* $h^{\mathfrak{A}'}_{\varphi',\bar{w}'}$ *is consistent with* T*.*

The proof is similar to the proof of the analogous statement in [55] for the special case of FO as well as the proof of Lemma 4.6, but it relies on Property 6.1. The main ingredient for the proof is the following variant of the Local Composition Lemma (Lemma 2.16) for the r-local formulas from Φ' on weighted structures.

Lemma 6.4. *For numbers* $k', \ell' \in \mathbb{N}$*, let* $\bar{v}_1, \bar{v}_2 \in \big(U(\mathfrak{A})\big)^{k'}$ *and* $\bar{w}_1, \bar{w}_2 \in \big(U(\mathfrak{A})\big)^{\ell'}$ *be tuples with* $\mathrm{dist}^{\mathfrak{A}'}(\bar{v}_1, \bar{w}_1) > 2r{+}1$, $\mathrm{dist}^{\mathfrak{A}'}(\bar{v}_2, \bar{w}_2) > 2r{+}1$, $\mathrm{tp}^{\mathfrak{A}'}_{\Phi'}(\bar{v}_1) = \mathrm{tp}^{\mathfrak{A}'}_{\Phi'}(\bar{v}_2)$*, and* $\mathrm{tp}^{\mathfrak{A}'}_{\Phi'}(\bar{w}_1) = \mathrm{tp}^{\mathfrak{A}'}_{\Phi'}(\bar{w}_2)$*. Then* $\mathrm{tp}^{\mathfrak{A}'}_{\Phi'}(\bar{v}_1\bar{w}_1) = \mathrm{tp}^{\mathfrak{A}'}_{\Phi'}(\bar{v}_2\bar{w}_2)$*.*

Proof. Let $\varphi(\bar{x}, \bar{y}) \in \mathrm{tp}^{\mathfrak{A}'}_{\Phi'}(\bar{v}_1\bar{w}_1)$. Then, with Property 6.1 (2), φ has a Feferman-Vaught decomposition Δ in Φ' w.r.t. $(\bar{x}; \bar{y})$, and thus,

$\mathcal{N}_r^{\mathfrak{A}'}(\bar{v}_1) \oplus \mathcal{N}_r^{\mathfrak{A}'}(\bar{w}_1) \models \varphi[\bar{v}_1, \bar{w}_1]$ if and only if there exists $(\alpha, \beta) \in \Delta$ such that $\mathcal{N}_r^{\mathfrak{A}'}(\bar{v}_1) \models \alpha[\bar{v}_1]$ and $\mathcal{N}_r^{\mathfrak{A}'}(\bar{w}_1) \models \beta[\bar{w}_1]$. Since $\mathfrak{A}' \models \varphi[\bar{v}_1, \bar{w}_1]$ and φ, α, β are r-local, it follows that $\mathfrak{A}' \models \alpha[\bar{v}_1]$ and $\mathfrak{A}' \models \beta[\bar{w}_1]$. Hence, $\alpha \in \mathrm{tp}_{\Phi'}^{\mathfrak{A}'}(\bar{v}_1) = \mathrm{tp}_{\Phi'}^{\mathfrak{A}'}(\bar{v}_2)$ and $\beta \in \mathrm{tp}_{\Phi'}^{\mathfrak{A}'}(\bar{w}_1) = \mathrm{tp}_{\Phi'}^{\mathfrak{A}'}(\bar{w}_2)$. We obtain $\mathfrak{A}' \models \bigvee_{(\alpha,\beta)\in\Delta} \alpha[\bar{v}_2] \wedge \beta[\bar{w}_2]$ and thus $\mathfrak{A}' \models \varphi[\bar{v}_2, \bar{w}_2]$. □

We are now ready to prove Lemma 6.3.

Proof of Lemma 6.3. Let $T = \big((\bar{v}_1, \lambda_1), \dots, (\bar{v}_m, \lambda_m)\big)$. Furthermore, let $\varphi(\bar{x}, \bar{y}) \in \Phi'$ and $\bar{w} = (w_1, \dots, w_\ell) \in \big(U(\mathfrak{A})\big)^\ell$ be such that the hypothesis $h_{\varphi,\bar{w}}^{\mathfrak{A}'} \in \mathcal{H}_{\Phi',k,\ell}(\mathfrak{A}')$ is consistent with T. Analogously to the proof of Lemma 4.6, we iteratively select vertices $w^{(i)}$ from the parameters $w_1, \dots, w_\ell$ that have distance at most $2r+1$ from the examples or the already selected vertices. This process is repeated for s steps until all remaining parameters are too far away (or all parameters have already been selected). For the tuple $\bar{w}'$ that we are looking for in this proof, we use these selected parameters and omit the others.

Formally, to select the parameters, we start with the neighbourhood $N^{(0)} := N_{2r+1}^{\mathfrak{A}'}(T)$ of radius $2r+1$ around the examples and select a vertex $w \in \{w_1, \dots, w_\ell\} \cap N^{(0)}$. If there is no such vertex, we set $s := 0$ and stop this process. Otherwise, we set $w^{(1)} := w$, $N^{(1)} := N^{(0)} \cup N_{2r+1}^{\mathfrak{A}'}(w)$, and continue. For $i \geqslant 2$, we select a vertex $w \in \{w_1, \dots, w_\ell\} \setminus \{w^{(1)}, \dots, w^{(i-1)}\}$ that is contained in the neighbourhood $N^{(i-1)}$. If there is no such vertex, we set $s := i-1$ and stop. Otherwise, we set $w^{(i)} := w$, $N^{(i)} := N^{(i-1)} \cup N_{2r+1}^{\mathfrak{A}'}(w)$, and continue. W.l.o.g. let $w^{(i)} = w_i$ for $i \in [s]$. Let $\bar{w}^{\text{in}} := (w_1, \dots, w_s)$ and $\bar{w}^{\text{out}} := (w_{s+1}, \dots, w_\ell)$.

Claim. Let $i, j \in [m]$ such that $\mathrm{tp}_{\Phi'}^{\mathfrak{A}'}(\bar{v}_i \bar{w}^{\text{in}}) = \mathrm{tp}_{\Phi'}^{\mathfrak{A}'}(\bar{v}_j \bar{w}^{\text{in}})$. Then $\lambda_i = \lambda_j$.

Proof. From the construction, it follows that $N_{2r+1}^{\mathfrak{A}'}(w_p) \subseteq N^{(p)} \subseteq N^{(s)}$ for every $p \in [s]$, $w_p \not\in N^{(s)}$ for every $p \in [s+1, \ell]$, and $N_{2r+1}^{\mathfrak{A}'}(\bar{v}_p) \subseteq N^{(0)} \subseteq N^{(s)}$ for every $p \in [m]$. Hence, for every $p \in [m]$, we obtain $\mathrm{dist}^{\mathfrak{A}'}(\bar{w}^{\text{out}}, \bar{v}_p \bar{w}^{\text{in}}) > 2r{+}1$. With Lemma 6.4, it follows that $\mathrm{tp}_{\Phi'}^{\mathfrak{A}'}(\bar{v}_i \bar{w}) = \mathrm{tp}_{\Phi'}^{\mathfrak{A}'}(\bar{v}_i \bar{w}^{\text{in}} \bar{w}^{\text{out}}) = \mathrm{tp}_{\Phi'}^{\mathfrak{A}'}(\bar{v}_j \bar{w}^{\text{in}} \bar{w}^{\text{out}}) = \mathrm{tp}_{\Phi'}^{\mathfrak{A}'}(\bar{v}_j \bar{w})$. Thus, in particular, $\varphi \in \mathrm{tp}_{\Phi'}^{\mathfrak{A}'}(\bar{v}_i \bar{w}) \iff \varphi \in \mathrm{tp}_{\Phi'}^{\mathfrak{A}'}(\bar{v}_j \bar{v})$. Since $h_{\varphi,\bar{w}}^{\mathfrak{A}'}$ is consistent with T, this implies that $\lambda_i = \lambda_j$. ⌟

We let $\bar{y}^{\text{in}} := (y_1, \dots, y_s)$ and choose

$$\varphi^{\text{in}}(\bar{x}, \bar{y}^{\text{in}}) := \bigvee_{i \in [m], \lambda_i = 1} \psi_i(\bar{x}, \bar{y}^{\text{in}})$$

for

$$\psi_i(\bar{x}, \bar{y}^{\text{in}}) := \bigwedge_{\gamma(\bar{x}, \bar{y}^{\text{in}}) \in \mathrm{tp}_{\Phi'}^{\mathfrak{A}'}(\bar{v}_i \bar{w}^{\text{in}})} \gamma(\bar{x}, \bar{y}^{\text{in}}).$$

The formula φ^{in} is a Boolean combination of formulas in Φ' and thus, according to Property 6.1, there is a formula $\varphi' \in \Phi'$ that is equivalent

```
Require: local access to background structure $\mathfrak{A}'$,
    training sequence $T = ((\bar{v}_1, \lambda_1), \dots, (\bar{v}_m, \lambda_m))$
 1: $N \leftarrow N^{\mathfrak{A}'}_{(2r+1)\ell}(T)$
 2: for all $\bar{w}' \in N^{\ell}$ do
 3:   for all $\varphi' \in \Phi'$ do
 4:     $\mathtt{consistent} \leftarrow$ true
 5:     for all $i \in [m]$ do
 6:       $\mathfrak{N} \leftarrow \mathcal{N}^{\mathfrak{A}'}_r(\bar{v}_i\bar{w}')$
 7:       if $[\![\varphi'(\bar{v}_i, \bar{w}')]\!]^{\mathfrak{N}} \neq \lambda_i$ then
 8:         $\mathtt{consistent} \leftarrow$ false
 9:     if $\mathtt{consistent}$ then
10:       return $(\varphi', \bar{w}')$
11: reject
```

Figure 6.1: Learning algorithm for Theorem 6.2

to φ^{in}. The free variables of φ' are among $\bar{x}$ and $\bar{y}^{\text{in}}$, and since $\bar{y}^{\text{in}}$ is a prefix of $\bar{y}$, we can safely write $\varphi'(\bar{x}, \bar{y})$.

We turn $\bar{w}^{\text{in}} = (\bar{w}_1, \dots, \bar{w}_s)$ into a tuple $\bar{w}' \in \left(N^{\mathfrak{A}}_{(2r+1)\ell}(T)\right)^{\ell}$ by choosing an arbitrary $w \in N^{\mathfrak{A}}_{(2r+1)\ell}(T)$ and filling the missing $(\ell - s)$ positions with the vertex w.

By the choice of φ^{in}, the following is true for all $j \in [m]$. If $\mathfrak{A}' \models \varphi'[\bar{v}_j, \bar{w}^{\text{in}}]$, then there exists a positive example $\bar{v}_i$ with $\mathfrak{A}' \models \psi_j[\bar{v}_i, \bar{w}^{\text{in}}]$. Thus $\text{tp}^{\mathfrak{A}'}_{\Phi'}(\bar{v}_i\bar{w}^{\text{in}}) = \text{tp}^{\mathfrak{A}'}_{\Phi'}(\bar{v}_j\bar{w}^{\text{in}})$ for some positive example $\bar{v}_i$; with the claim, we can conclude that $\lambda_j = 1$. On the other hand, if $\lambda_j = 1$, then $\mathfrak{A}' \models \psi_j[\bar{v}_j, \bar{w}^{\text{in}}]$ and hence $\mathfrak{A}' \models \varphi'[\bar{v}_j, \bar{w}']$. Thus, $h^{\mathfrak{A}'}_{\varphi', \bar{w}'}$ is consistent with T. □

Using Lemma 6.3, we can now prove Theorem 6.2, the main result of this section.

Proof of Theorem 6.2. We show that the algorithm depicted in Figure 6.1 fulfils the requirements given in Theorem 6.2. The algorithm goes through all tuples $\bar{w}' \in \left(N^{\mathfrak{A}'}_{(2r+1)\ell}(T)\right)^{\ell}$ and all formulas $\varphi'(\bar{x}, \bar{y}) \in \Phi'$. A hypothesis $h^{\mathfrak{A}'}_{\varphi', \bar{w}'}$ is consistent with the training sequence T if and only if $[\![\varphi'(\bar{v}_i, \bar{w}')]\!]^{\mathfrak{A}'} = \lambda_i$ for all $i \in [m]$. Since Φ' only contains r-local formulas, this holds if and only if $[\![\varphi'(\bar{v}_i, \bar{w}')]\!]^{\mathcal{N}^{\mathfrak{A}'}_r(\bar{v}_i\bar{w}')} = \lambda_i$ for every $i \in [m]$. Hence, the algorithm only returns a hypothesis if it is consistent. Furthermore, if there is a consistent hypothesis in $\mathcal{H}_{\Phi,k,\ell}(\mathfrak{A})$, then by Property 6.1 (1), there is also a consistent hypothesis in $\mathcal{H}_{\Phi',k,\ell}(\mathfrak{A}')$, and Lemma 6.3 ensures that the algorithm then returns a hypothesis.

It remains to show that the algorithm satisfies the running-time requirements while only using local access to the structure $\mathfrak{A}'$. For all

$\bar{v} \in \big(U(\mathfrak{A})\big)^k$ and $\bar{w}' \in \big(U(\mathfrak{A})\big)^\ell$, based on the proof of Lemma 4.7, we can bound the size of their neighbourhood by

$$\left|N_r^{\mathfrak{A}'}(\bar{v}\bar{w}')\right| \leqslant (k+\ell) \cdot \sum_{i=0}^{r} d^i \leqslant (k+\ell) \cdot (1+d^{r+1}).$$

Therefore, the representation size of the substructure $\mathcal{N}_r^{\mathfrak{A}'}(\bar{v}\bar{w}')$ is in $\mathcal{O}\big((k+\ell) \cdot d^{r+1} \cdot \log n\big)$. Thus, the consistency check in lines 4–8 runs in time $f_{\Phi'}(\mathfrak{A}') \cdot m \cdot \mathcal{O}\big((k+\ell) \cdot d^{r+1} \cdot \log n\big)$ under the logarithmic-cost measure and in time $f_{\Phi'}(\mathfrak{A}') \cdot m \cdot \mathcal{O}\big((k+\ell) \cdot d^{r+1}\big)$ under the uniform-cost measure. Let $N = N_{(2r+1)\ell}^{\mathfrak{A}'}(T)$. The algorithm checks up to $|N|^\ell \cdot |\Phi'| \in \mathcal{O}\Big(\big(m \cdot k \cdot d^{(2r+1)\ell+1}\big)^\ell \cdot |\Phi'|\Big)$ hypotheses.

All in all, since k, ℓ, r are considered constant, the running time of the algorithm is in $f_{\Phi'}(\mathfrak{A}') \cdot (\log n + d + m)^{\mathcal{O}(1)}$ under the logarithmic-cost measure and in $f_{\Phi'}(\mathfrak{A}') \cdot (d+m)^{\mathcal{O}(1)}$ under the uniform-cost measure. Furthermore, the algorithm only uses local access to the structure $\mathfrak{A}'$. □

6.3 AGNOSTIC PAC LEARNING

Apart from consistent learning with precomputation, we also study agnostic PAC learnability.

LEARN-PAC-PRECOMP(k, Φ, Φ')

Input: structure $\mathfrak{A}'$ (computed from the input structure $\mathfrak{A}$ with $U(\mathfrak{A}') = U(\mathfrak{A})$), rational numbers $\varepsilon, \delta > 0$, probability distribution $\mathcal{D}$ on $\big(U(\mathfrak{A})\big)^k \times \{0,1\}$

Problem: Return a formula $\varphi' \in \Phi'$ and a tuple $\bar{w}' \in \big(U(\mathfrak{A})\big)^\ell$ such that, with probability of at least $1-\delta$ over the choice of examples drawn i.i.d. from $\mathcal{D}$, it holds that

$$\operatorname{err}_{\mathcal{D}}\big(h_{\varphi',\bar{w}'}^{\mathfrak{A}'}\big) \leqslant \varepsilon^* + \varepsilon,$$

where

$$\varepsilon^* := \min_{\substack{\varphi \in \Phi, \\ \bar{w} \in (U(\mathfrak{A}))^\ell}} \operatorname{err}_{\mathcal{D}}\big(h_{\varphi,\bar{w}}^{\mathfrak{A}}\big).$$

The following theorem provides an agnostic PAC-learning algorithm.

Theorem 6.5 (Agnostic PAC learning with precomputation). *There is an algorithm that solves* LEARN-PAC-PRECOMP(k, Φ, Φ') *with local access to a structure $\mathfrak{A}'$ that is associated with the input structure $\mathfrak{A}$ in time $f_{\Phi^*}(\mathfrak{A}^*) \cdot \big(\log n + d + \frac{1}{\varepsilon} + \log \frac{1}{\delta}\big)^{\mathcal{O}(1)}$ under both the logarithmic-cost and the uniform-cost measure, where $\mathfrak{A}$, $\mathfrak{A}'$, Φ, and Φ' are as described in Property 6.1, n and d are the size and the degree of $\mathfrak{A}'$, and $f_{\Phi'}(\mathfrak{A}')$ is an*

```
Require: local access to background structure $\mathfrak{A}'$,
    training sequence $T = \big((\bar v_1, \lambda_1), \dots, (\bar v_m, \lambda_m)\big)$
 1: $N \leftarrow N^{\mathfrak{A}'}_{(2r+1)\ell}(T)$
 2: $\mathrm{err}_{\min} \leftarrow |T| + 1$
 3: for all $\bar w' \in N^\ell$ do
 4:   for all $\varphi' \in \Phi'$ do
 5:     $\mathrm{err} \leftarrow 0$
 6:     for all $i \in [m]$ do
 7:       $\mathfrak{N} \leftarrow \mathcal{N}^{\mathfrak{A}'}_r(\bar v_i \bar w')$
 8:       if $[\![\varphi'(\bar v_i, \bar w')]\!]^{\mathfrak{N}} \neq \lambda_i$ then
 9:         $\mathrm{err} \leftarrow \mathrm{err} + 1$
10:     if $\mathrm{err} < \mathrm{err}_{\min}$ then
11:       $\mathrm{err}_{\min} \leftarrow \mathrm{err}$
12:       $(\varphi'_{\min}, \bar w'_{\min}) \leftarrow (\varphi', \bar w')$
13: return $(\varphi'_{\min}, \bar w'_{\min})$
```

Figure 6.2: ERM algorithm used in Theorem 6.5.

upper bound on the time complexity of model checking for formulas in Φ' on $\mathfrak{A}'$.

We prove this result in a similar fashion as the agnostic PAC-learning result for first-order logic with counting on structures of bounded degree.

Proof of Theorem 6.5. Let $\mathfrak{A}$ be a $(\sigma, \mathbf{W})$-structure and let $\mathfrak{A}'$ be the associated $(\sigma', \mathbf{W})$-structure. We consider the concept class

$$\mathcal{H} = \left\{ h^{\mathfrak{A}}_{\varphi, \bar w} \;\middle|\; \varphi(\bar x, \bar y) \in \Phi,\ \bar w \in \big(U(\mathfrak{A})\big)^\ell \right\}$$

and the hypothesis class

$$\mathcal{H}' = \left\{ h^{\mathfrak{A}'}_{\varphi', \bar w'} \;\middle|\; \varphi'(\bar x, \bar y) \in \Phi',\ \bar w' \in \big(U(\mathfrak{A})\big)^\ell \right\}.$$

Since, by Property 6.1, Φ' contains only finitely many formulas, the number of hypotheses in $\mathcal{H}'$ is bounded by $s \cdot |\mathfrak{A}|^\ell$ for some constant s. Furthermore, by Property 6.1 (1), it holds that $\mathcal{H} \subseteq \mathcal{H}'$. Thus, we can also bound the number of hypotheses in $\mathcal{H}$ by $s \cdot |\mathfrak{A}|^\ell$. Our algorithm that solves Learn-PAC-Precomp works as follows.

Given local access to a associated structure $\mathfrak{A}'$, oracle access to the size $|\mathfrak{A}| = |\mathfrak{A}'|$ of the structure, oracle access to a probability distribution $\mathcal{D}$ on $\big(U(\mathfrak{A})\big)^k \times \{0,1\}$, and given rational numbers $\varepsilon, \delta > 0$, our algorithm queries

$$m(|\mathfrak{A}|, \varepsilon, \delta) := \left\lceil \frac{2\log(2s \cdot |\mathfrak{A}|^\ell / \delta)}{\varepsilon^2} \right\rceil$$

many examples from $\mathcal{D}$. Then, it runs the ERM algorithm depicted in Figure 6.2 on the resulting training sequence.

Next, we show that this algorithm indeed solves the problem Learn-PAC-Precomp. Let $\mathcal{D}$ be a distribution over $\big(U(\mathfrak{A})\big)^k \times \{0,1\}$ and let $h \in \mathcal{H}$ be a hypothesis that minimises the generalisation error, that is, $\operatorname{err}_{\mathcal{D}}(h) = \min_{h'' \in \mathcal{H}} \operatorname{err}_{\mathcal{D}}(h'')$. Let T be the training sequence of length $m(|\mathfrak{A}|, \varepsilon, \delta)$ drawn i.i.d. from $\mathcal{D}$ by our algorithm, and let $h' \in \mathcal{H}'$ be the hypothesis returned by the ERM algorithm on input T. Analogously to the ERM algorithm for first-order logic with counting (Theorem 4.9), the returned hypothesis h' fulfils $\operatorname{err}_T(h') \leqslant \operatorname{err}_T(h)$, since the ERM algorithm returns a hypothesis from $\mathcal{H}'$ that minimises the training error and $h \in \mathcal{H} \subseteq \mathcal{H}'$.

Furthermore, by the Uniform Convergence Lemma (Lemma 3.9), with probability at least $1-\delta$, it holds that $\big|\operatorname{err}_T(h'') - \operatorname{err}_{\mathcal{D}}(h'')\big| \leqslant \frac{\varepsilon}{2}$ for all $h'' \in \mathcal{H}'$. This especially holds for h' as well as for h. Hence,

$$\operatorname{err}_{\mathcal{D}}(h') \leqslant \operatorname{err}_T(h') + \frac{\varepsilon}{2} \leqslant \operatorname{err}_T(h) + \frac{\varepsilon}{2} \leqslant \operatorname{err}_D(h) + \frac{\varepsilon}{2} + \frac{\varepsilon}{2}$$

with probability at least $1-\delta$. This is exactly the requirement we have in Learn-PAC-Precomp for the returned hypothesis.

The number $m(|\mathfrak{A}|, \varepsilon, \delta)$ of queried examples can be bounded by $\mathcal{O}\left(\frac{\log(|\mathfrak{A}|/\delta)}{\varepsilon^2}\right)$. Thus, based on the running-time analysis in the proof of Theorem 6.2, we can bound the running time of our algorithm by $\big(\log|\mathfrak{A}| + d + \log\frac{1}{\delta} + \frac{1}{\varepsilon}\big)^{\mathcal{O}(1)}$ under the logarithmic-cost as well as the uniform-cost measure. ☐

6.4 LEARNING FOWA1

In this section, we combine the learning results of Sections 6.2 and 6.3 with the locality results of Chapter 5 to provide learning results for the logic FOWA_1. Before we do so, we first revisit the case of plain first-order logic within our framework.

Remark 6.6. Fix a quantifier rank $q \in \mathbb{N}$ as well as $k, \ell \in \mathbb{N}$ and a signature σ. Let $\Phi = \big\{\varphi(\bar{x}, \bar{y}) \in \mathsf{FO}[\sigma, q] \;\big|\; |\bar{x}| = k, |\bar{y}| = \ell\big\}$. By the well-known properties of first-order logic (i. e., the existence of Feferman-Vaught decompositions as well as the locality properties we described in Section 2.4), the set Φ has Property 6.1, e. g. via $L' := L = \mathsf{FO}$, $\sigma' := \sigma$, and $\mathfrak{A}' := \mathfrak{A}$. This is exactly the setting considered by Grohe and Ritzert [55] that we described in Chapter 3. Using the framework described in Section 6.1 (and skipping the precomputation step since $\mathfrak{A}' = \mathfrak{A}$), the results from Chapter 3 follow from Theorem 6.2 and Theorem 6.5.

Next, we establish the crucial link between the learning results of this chapter and the locality results of Chapter 5. For that, we show that suitably chosen sets $\Phi \subseteq \mathsf{FOWA}_1(\mathbb{P})[\sigma, \mathbb{S}, \mathbf{W}]$ indeed have Property 6.1. By using the locality properties of FOW_1 and FOWA_1, we can apply a similar reasoning to $\mathsf{FOWA}_1(\mathbb{P})[\sigma, \mathbb{S}, \mathbf{W}]$ as to $\mathsf{FO}[\sigma]$. Fix numbers $k, \ell, q \in \mathbb{N}$ and a signature σ. Let the collections $\mathbb{P}$ and $\mathbb{S}$ be finite (but $\mathbb{S}$ may contain some infinite rings or abelian groups), and fix a finite set $\mathcal{S}$ of elements $s \in S \in \mathbb{S}$.

Let $\Phi := \Phi_{q,k+\ell,\mathcal{S}}$ be the set of all $\mathsf{FOWA}_1(\mathbb{P})[\sigma, \mathbb{S}, \mathbf{W}]$-formulas φ of quantifier rank and aggregation depth at most q and with free variables among $\{x_1, \dots, x_k, y_1, \dots, y_\ell\}$ that have the following additional property; all symbols $s \in S$ for some $S \in \mathbb{S}$ that are present in φ belong to $\mathcal{S}$, all $\mathbf{W}$-products present in φ have length at most q, and the maximum nesting depth of term constructions using rule (9) from Definition 5.3 in order to construct terms present in φ is at most q.

Lemma 6.7. *$\Phi = \Phi_{q,k+\ell,\mathcal{S}}$ has Property 6.1.*

Proof. In the following, we describe a procedure to compute a signature σ', a finite set of formulas $\Phi' \subseteq \mathsf{FOW}_1(\mathbb{P})[\sigma', \mathbb{S}, \mathbf{W}]$, and, for every $(\sigma, \mathbf{W})$-structure $\mathfrak{A}$, a $(\sigma', \mathbf{W})$-structure $\mathfrak{A}'$ that witness that Φ has Property 6.1.

Claim 1. Up to logical equivalence, Φ only contains a finite number of formulas.

Proof. Let $a := \max_{w \in \mathbf{W}} \mathrm{ar}(w)$. The maximum nesting depth of constructs using rules (4) and (5) as well as the maximum nesting depth of constructs using rule (10) from Definition 5.3 is bounded by q and every construct using rule (4) adds one new variable, rule (5) adds at most a new variables, and rule (10) adds at most $a \cdot q$ new variables. Thus, every subformula of a formula in Φ has at most $k + \ell + a \cdot q + a \cdot q^2$ free variables. With finitely many free variables and σ, $\mathcal{S}$, and $\mathbf{W}$ being finite as well, rules (1) and (2) only produce a finite number of formulas. With the same argument, rules (7) and (8) only produce a finite number of $\mathbb{S}$-terms. We show by induction on the nesting depth of constructs using rules (4), (5), and (10) that there are, up to logical equivalence, only finitely many (sub-)formulas and $\mathbb{S}$-terms used in Φ, which implies the claim.

If there are only finitely many $\mathbb{S}$-terms, then, with a bounded nesting depth, rule (9) only yields a finite number of new $\mathbb{S}$-terms. Thus, since $\mathbb{P}$ is also finite, rule (6) only produces a finite number of formulas of the form $P(t_1, \dots, t_m)$. Hence, rule (3) only creates a finite number of formulas up to logical equivalence. (Consider them being in a normal form analogous to CNF.)

Applying rule (4) or rule (5) to a set of finitely many formulas only creates finitely many new formulas. Then, rule (10) only yields finitely many $\mathbb{S}$-terms. This completes the proof of Claim 1. ⌟

For each of these finitely many formulas $\varphi \in \Phi$, we apply Theorem 5.13 to obtain an extension σ_φ of σ, a σ_φ-expansion $\mathfrak{A}_\varphi$ of $\mathfrak{A}$, and a local $\mathsf{FOW}_1(\mathbb{P})[\sigma_\varphi, \mathbb{S}, \mathbf{W}]$-formula φ^*. Then, we let σ^* be the union of all the σ_φ, we let $\mathfrak{A}^*$ be the σ^*-expansion of $\mathfrak{A}$ whose σ_φ-reduct coincides with $\mathfrak{A}_\varphi$ for every φ, and we let Φ^* be the set of all the formulas φ^*. Choose a number $r \in \mathbb{N}$ such that each of the $\varphi^* \in \Phi^*$ is r-local.

We can repeatedly apply Theorem 5.8 for every partition of the free variables, take the r-localisations $\alpha^{(r)}, \beta^{(r)}$ of the resulting formulas α, β, and take Boolean combinations to obtain an extension Φ' of Φ^* such that Φ' satisfies statements (2) and (3) of Property 6.1 and contains only r-local formulas. We stop the process once, up to equivalence, no new formulas have been added.

Claim 2. The process stops after finitely many steps. Thus, we obtain a finite extension Φ'.

Proof. Let $\Phi^{(0)} = \Phi^*, \Phi^{(1)}, \Phi^{(2)}, \dots$ be sets of formulas, where $\Phi^{(i+1)}$ is computed from $\Phi^{(i)}$ by applying Theorem 5.8 for every partition of the free variables and taking the r-localisations $\alpha^{(r)}, \beta^{(r)}$ of the resulting formulas α, β. This is the above described procedure without adding all Boolean combinations. When applying Theorem 5.8 to a formula $\varphi \in \Phi^{(i)}$ w.r.t. a partition of the free variables $(\bar{z}_1; \bar{z}_2)$, the Feferman-Vaught decomposition only contains new formulas not already contained in $\Phi^{(i)}$ if the partition is not trivial, that is, if neither $\bar{z}_1$ nor $\bar{z}_2$ contain all free variables of φ. Thus, the application of Theorem 5.8 can only produce new formulas not equivalent to one of the already computed ones if the new formula has less free variables than the original formula. Hence, if one only applies Theorem 5.8 and takes the r-localisations of the resulting formulas, then one can stop the process after finitely many, say m, steps.

Recall that, for two formulas $\varphi_1, \varphi_2 \in \mathsf{FOW}_1(\mathbb{P})[\sigma^*, \mathbb{S}, \mathbf{W}]$ and for the Feferman-Vaught decompositions $\Delta_{\varphi_1}, \Delta_{\varphi_2}, \Delta_{\varphi_1 \vee \varphi_2}$, and $\Delta_{\neg\varphi_1}$ of φ_1, φ_2, $\varphi_1 \vee \varphi_2$, and $\neg\varphi_1$ w.r.t. $\bar{z}_1, \bar{z}_2$, we have $\Delta_{\varphi_1 \vee \varphi_2} = \Delta_{\varphi_1} \cup \Delta_{\varphi_2}$ and, for $\Delta_{\varphi_1} = \{(\alpha_1, \beta_1), \dots, (\alpha_s, \beta_2)\}$, we have $\Delta_{\neg\varphi_1} = \{(\alpha_A, \beta_A) \mid A \subseteq [s]\}$ with $\alpha_A = \bigwedge_{i \in A} \neg\alpha_i$ and $\beta_A = \bigwedge_{i \in [s] \setminus A} \neg\beta_i$. Thus, for any of the sets $\Phi^{(i)}$, the Feferman-Vaught decompositions of Boolean combinations of formulas from $\Phi^{(i)}$ only contain Boolean combinations of the formulas that occur in the Feferman-Vaught decompositions of $\Phi^{(i)}$. This shows that it suffices to compute the Boolean combinations only in the last iteration of our procedure.

Hence, the result of the overall process is the set of Boolean combinations of formulas in $\Phi^{(m)}$. Since the set of Boolean combinations of finitely many formulas is, up to logical equivalence, again finite, the process stops after finitely many steps with a finite extension Φ'. This completes the proof of Claim 2. ⌟

Let $\mathfrak{A}' := \mathfrak{A}^*$ and $\sigma' := \sigma^*$. Then Φ' witnesses that Φ has Property 6.1. □

For the remainder of this section, let $\Phi = \Phi_{q,k+\ell,\mathbb{S}}$, let σ', Φ' be as described in the proof of Lemma 6.7, and, for every $(\sigma, \mathbf{W})$-structure $\mathfrak{A}$, let $\mathfrak{A}'$ be the $(\sigma', \mathbf{W})$-structure described in the proof of Lemma 6.7.

By Theorem 5.13, $\mathfrak{A}'$ can be computed from $\mathfrak{A}$ in time $|\mathfrak{A}| \log |\mathfrak{A}| \cdot d^{\mathcal{O}(1)}$ under the logarithmic-cost measure and in time $|\mathfrak{A}| \cdot d^{\mathcal{O}(1)}$ under the uniform-cost measure, where d is the degree of $\mathfrak{A}$. Since Φ' witnesses that Φ has Property 6.1, the formulas in Φ' are r-local for a fixed number r. This implies that the model-checking problem for formulas in Φ' can be solved in time polynomial in the degree of the structure. Combining this with Theorems 6.2 and 6.5 yields the following learnability result for FOWA_1, where we assume all mentioned algorithms to have $\mathbb{P}$- and $\mathbb{S}$-oracles, so that operations $+_S, \cdot_S$ for $S \in \mathbb{S}$ and checking if a tuple is in $[\![P]\!]$ for $P \in \mathbb{P}$ takes time $\mathcal{O}(1)$.

Theorem 6.8. *For an input structure $\mathfrak{A}$ and an input training sequence T, let n and d denote the size and the degree of $\mathfrak{A}$ and let m denote the number of training examples in T.*

(1) There is an algorithm that solves the consistent-learning with precomputation problem Learn-Consistent-Precomp(k, Φ, Φ') *with local access to a structure $\mathfrak{A}'$ that is associated with the input structure $\mathfrak{A}$ in time $\left(\log n + d + m\right)^{\mathcal{O}(1)}$ under the logarithmic-cost measure and in time $\left(d + m\right)^{\mathcal{O}(1)}$ under the uniform-cost measure.*

(2) There is an algorithm that solves the PAC-learning with precomputation problem Learn-PAC-Precomp(k, Φ, Φ') *with local access to a structure $\mathfrak{A}'$ that is associated with the input structure $\mathfrak{A}$ in time $\left(\log n + d + \frac{1}{\varepsilon} + \log \frac{1}{\delta}\right)^{\mathcal{O}(1)}$ under both the logarithmic-cost and the uniform-cost measure.*

Additionally, the algorithms can be chosen such that the returned hypotheses can be evaluated in time $(\log n + d)^{\mathcal{O}(1)}$ under the logarithmic-cost measure and in time $d^{\mathcal{O}(1)}$ under the uniform-cost measure. Moreover, the precomputation step for both algorithms runs in time $|\mathfrak{A}| \log |\mathfrak{A}| \cdot d^{\mathcal{O}(1)}$ under the logarithmic-cost measure and in time $|\mathfrak{A}| \cdot d^{\mathcal{O}(1)}$ under the uniform-cost measure.

On classes of structures of polylogarithmic degree, Theorem 6.8 implies that consistent learning and PAC learning are possible in sublinear time.

Corollary 6.9. *Let $\mathcal{C}$ be a class of structures of polylogarithmic degree.*

(1) There is an algorithm that solves the consistent-learning with precomputation problem Learn-Consistent-Precomp(k, Φ, Φ') *on $\mathcal{C}$ with local access in time sublinear in the size of the background structure and polynomial in the length of the training sequence, under the logarithmic-cost as well as the uniform-cost measure.*

(2) There is an algorithm that solves the PAC-learning with precomputation problem Learn-PAC-Precomp(k, Φ, Φ') *with local access in time sublinear in the size of the background structure under the logarithmic-cost as well as the uniform-cost measure.*

The hypotheses returned by the algorithms can be evaluated in sublinear time and the precomputation step for both algorithms runs in pseudo-linear time, measured in the size of the background structure under both the logarithmic-cost and the uniform-cost measure.

We conclude with an example that illustrates an application scenario for Theorem 6.8.

Example 6.10. Recall the $(\sigma, \mathbf{W})$-structure $\mathfrak{A}$ for the online marketplace from Examples 5.1, 5.2, and 5.6. Retailers can pay the marketplace to advertise their products to consumers. Since the marketplace demands a fee for every single view of the advertisement, retailers want the marketplace to only show the advertisement to those consumers that are likely to buy the product. One possible way to choose suitable consumers is to consider only those who buy a variety of products from the same or a similar product group as the advertised product and who are thus more likely to try new products that are similar to the advertised one. At the same time, the money spent by the chosen consumers on the product group should be above average.

In Example 5.6, we have already seen a formula $\varphi_{\text{spending}}(c)$ that defines consumers who have spent at least as much as the average consumer on the product group. The formula depends on a formula $\varphi_{\text{group}}(p)$ that defines a certain group of products based on the structure of their transactions. Due to the connection between graph neural networks and the Weisfeiler-Leman algorithm described in [75], we may assume that there is a formula in first-order logic that at least roughly approximates such a product group. Likewise, we might assume that there is a formula $\varphi_{\text{variety}}(c)$ in first-order logic that defines consumers with a wide variety of products bought from a specific product group. However, it is a non-trivial task to design such formulas by hand. It is even not clear whether there exist better rules for finding suitable consumers. Meanwhile, we can easily show the advertisement to consumers and then check whether they buy the product. Thus, we can generate a list with positive and negative examples of consumers. Since the proposed rule can be defined in $\mathsf{FOWA}_1(\mathbb{P})[\sigma, \mathbb{S}, \mathbf{W}]$ as $\varphi_{\text{advertise}}(c) \coloneqq (\varphi_{\text{variety}}(c) \wedge \varphi_{\text{spending}}(c))$, we can use one of the learning algorithms from Theorem 6.8 to find good definitions for $\varphi_{\text{variety}}(c)$ and $\varphi_{\text{group}}(p)$ or to learn an even better definition for $\varphi_{\text{advertise}}(c)$ in $\mathsf{FOWA}_1(\mathbb{P})[\sigma, \mathbb{S}, \mathbf{W}]$ from examples.

7 PARAMETERISED COMPLEXITY OF LEARNING

As we have seen in the last chapters, to solve a PAC-learning problem, we can combine an algorithm that performs Empirical Risk Minimisation with a sufficient upper bound on the number of examples needed to guarantee the desired probability bounds. The learning algorithms studied in the previous chapters use this technique and all run in sublinear time on classes of structures of bounded or polylogarithmic degree.

In the present chapter, we loosen the requirements we impose on the classes of structures and look for classes with a tractable (i. e., polynomial-time-solvable) ERM problem. In the previous chapters, we considered k, the length of the input tuples, and ℓ, the number of parameters, as fixed. When looking for classes with a tractable learning problem, this approach would not lead to significant results, since, for example, going through all $|\mathfrak{A}|^{\ell}$ many possible choices of parameters would still be allowed in a polynomial-time algorithm. On the other hand, compared to the size of the background structure, k and ℓ are typically small, so considering them as part of the input and requiring algorithms to run in time polynomial in k and ℓ also seems to be too restrictive. Thus, we analyse the parameterised complexity of learning problems and consider k and ℓ as parameters. Specifically, we study the ERM problem for concepts definable in first-order logic, and we look for classes with a fixed-parameter tractable ERM problem.

Inspired by the hypothesis classes $\mathcal{H}_{\Phi,k,\ell}$ in Chapter 3, for a σ-structure $\mathfrak{A}$ and for $k, \ell, q \in \mathbb{N}$, let

$$\mathcal{H}_{q,k,\ell}(\mathfrak{A}) := \left\{ h^{\mathfrak{A}}_{\varphi,\bar{w}} \,\middle|\, \varphi(\bar{x},\bar{y}) \in \mathsf{FO}[\sigma,q],\ |\bar{x}| = k,\ |\bar{y}| = \ell,\ \bar{w} \in \left(U(\mathfrak{A})\right)^{\ell} \right\}.$$

Then, for functions $L, Q\colon \mathbb{N}^3 \to \mathbb{N}$ which satisfy $L(k,\ell,q) \geqslant \ell$ and $Q(k,\ell,q) \geqslant q$ for all $k,\ell,q \in \mathbb{N}$, we consider the following parameterised problem.

p-FO-Learn-ERM(L, Q)

Input: σ-structure $\mathfrak{A}$, training sequence $T \in \Big(\big(U(\mathfrak{A})\big)^k \times \{0,1\}\Big)^m$, $k, \ell^*, q^* \in \mathbb{N}$, $\varepsilon > 0$

Parameter: $k + \ell^* + q^* + \frac{1}{\varepsilon} + |\sigma|$

Problem: Return a formula $\varphi(\bar{x}, \bar{y}) \in \mathsf{FO}[\sigma, q]$ for some $q \leqslant Q(k, \ell^*, q^*)$ with $|\bar{x}| = k$ and $|\bar{y}| = \ell \leqslant L(k, \ell^*, q^*)$, and return a tuple $\bar{w} \in \big(U(\mathfrak{A})\big)^\ell$ such that

$$\mathrm{err}_T\big(h^{\mathfrak{A}}_{\varphi,\bar{w}}\big) \leqslant \varepsilon^* + \varepsilon,$$

where

$$\varepsilon^* := \min\big\{\mathrm{err}_T\big(h^{\mathfrak{A}}_{\varphi^*,\bar{w}^*}\big) \bigm| h^{\mathfrak{A}}_{\varphi^*,\bar{w}^*} \in \mathcal{H}_{q^*,k,\ell^*}(\mathfrak{A})\big\}.$$

In Section 7.1, we describe an fpt Turing reduction from the parameterised model-checking problem p-FO-Mc to p-FO-Learn-ERM(L, Q). Since, in general, the parameterised FO model-checking problem is $\mathsf{AW}[*]$-hard, this implies that p-FO-Learn-ERM(L, Q), without any restrictions on the structures, is $\mathsf{AW}[*]$-hard as well (for all L, Q). Thus, under common assumptions (cf. Section 2.5), the general parameterised ERM problem is not fixed-parameter tractable.

In Section 7.2, we study first-order learning problems that are fixed-parameter tractable. First, we consider a variant of the ERM problem where the parameter ℓ^* is considered as a constant. Second, we study the parameterised consistent-learning problem for the 1-dimensional case, that is, for $k = 1$. For both scenarios, we show that the learning problems are fixed-parameter tractable on classes of structures that have a fixed-parameter tractable model-checking problem. More precisely, in both scenarios, we give fpt Turing reductions from the respective learning problems to the first-order model-checking problem. After this, we come back to the problem p-FO-Learn-ERM(L, Q) and look for restricted classes of structures with a tractable learning problem. Since the reduction in Section 7.1 from the FO model-checking problem applies to *all* graph classes satisfying mild closure conditions, we limit our search to sparse classes with a tractable model-checking problem. Here, we consider the most general type of such classes, namely nowhere dense classes. We prove that for every effectively nowhere dense class $\mathcal{C}$, there are functions L and Q such that p-FO-Learn-ERM(L, Q) is fixed-parameter tractable on $\mathcal{C}$.

Finally, in Section 7.3, we extend the result for the ERM problem on nowhere dense classes to PAC learning.

The restrictions of the results of this chapter from arbitrary relational structures to coloured graphs have been published in [12].

7.1 HARDNESS OF LEARNING

The main result of this section is the following.

Theorem 7.1. *For all functions* $L, Q \colon \mathbb{N}^3 \to \mathbb{N}$ *with* $L(k, \ell, q) \geqslant \ell$ *and* $Q(k, \ell, q) \geqslant q$ *for all* $k, \ell, q \in \mathbb{N}$, p-FO-Learn-ERM$(L, Q)$ *is hard for the parameterised complexity class* $\mathsf{AW}[*]$ *under fpt Turing reductions.*

This theorem is a direct consequence of the following reduction.

Lemma 7.2. *Let* $L, Q \colon \mathbb{N}^3 \to \mathbb{N}$ *be functions with* $L(k, \ell, q) \geqslant \ell$ *and* $Q(k, \ell, q) \geqslant q$ *for all* $k, \ell, q \in \mathbb{N}$*. Then,* p-FO-Mc *is fpt Turing-reducible to* p-FO-Learn-ERM(L, Q)*.*

The overall idea of the reduction is to solve p-FO-Mc recursively by decomposing the input formula. While handling negation and Boolean connectives is easy, the crucial part of the computation is handling quantification. In a naive approach, for a formula $\exists x\, \psi(x)$, one could go through all possible assignments to x and then check whether the remaining formula is satisfied in the given structure. This, however, would not lead to an fpt algorithm.

For a formula ψ of quantifier rank q and vertices v and w of the same q-type in the input structure, it does not matter whether we assign v or w to x. Thus, our goal is to find a set of representatives for the vertices in the structure such that the number of representatives only depends on the formula ψ, and for every vertex we have a representative of the same q-type. Recursively checking ψ for each of the representatives then leads to an fpt algorithm that solves p-FO-Mc.

To find the set of representatives, we start with the set of all vertices. Then, we iteratively remove vertices that are already represented by other vertices in the set until the number of remaining representatives is below a certain threshold that only depends on the input formula. In this process, we use an oracle for the problem p-FO-Learn-ERM(L, Q) on every pair (v, w) of vertices, where we give v as a positive and w as a negative example. The oracle is then supposed to return a formula of quantifier rank at most $Q(k, \ell, q)$ that distinguishes the two vertices. Since there are, up to equivalence, only finitely many of such formulas, the oracle has to return the same formula on some inputs. Using Ramsey's Theorem [18], we can show that for every set of representatives of a certain size, there are triples of vertices v_1, v_2, and v_3 such that the oracle returns the same formula on input (v_1, v_2), (v_2, v_3), and (v_3, v_1). Furthermore, we can show that for such triples, the vertex v_3 is already represented by v_1 or v_2, so we can remove it from the set of representatives. By repeating this process, we end up with a set of representatives whose size only depends on the input formula and not on the input structure. For the full proof of Lemma 7.2, we refer to [12].

7.2 TRACTABILITY OF EMPIRICAL RISK MINIMISATION

Before we study the complexity of p-FO-Learn-ERM on nowhere dense classes, we consider two different learning problems that can be solved using the first-order model-checking problem.

For a class of structures $\mathcal{C}$, we say that $\mathcal{C}$ is *closed under colour expansions*, if for every σ-structure $\mathfrak{A}$ in $\mathcal{C}$ and every signature $\sigma' \supseteq \sigma$, where all relation symbols in $\sigma' \setminus \sigma$ are unary, all σ'-expansions of $\mathfrak{A}$ are also contained in $\mathcal{C}$.

For our first result, we consider the variant p-FO-Learn-ERM where we view the parameter ℓ^* as a constant.

Proposition 7.3. *Let $\mathcal{C}$ be a class of structures that is closed under colour expansions such that* p-FO-Mc *is fixed-parameter tractable on $\mathcal{C}$. Then, for every constant $\ell \in \mathbb{N}$, the restriction of* p-FO-Learn-ERM(L, Q) *to inputs with $\ell^* = \ell$ is fixed-parameter tractable on $\mathcal{C}$ as well (for all* L *and* Q*).*

In case of a constant ℓ, a simple brute-force algorithm suffices to solve the problem. Our task is to return a hypothesis that is at least as good as the best hypothesis that uses ℓ parameters and a formula of quantifier rank at most q. Thus, we can go through all possible combinations of ℓ parameters and formulas of quantifier rank at most q (up to equivalence) and check the resulting hypotheses using an oracle to the model-checking problem. Since the number of formulas to check is independent of the size of the structure and, for a constant ℓ, the number $|\mathfrak{A}|^{\ell}$ of parameter tuples to check is polynomial in the size of the structure, all in all, the described brute-force algorithm is an fpt algorithm. Although an algorithm solving p-FO-Learn-ERM(L, Q) would be allowed to return up to $L(k, \ell, q)$ parameters and a formula of quantifier rank at most $Q(k, \ell, q)$, we limit ourselves to ℓ parameters and quantifier rank q. Hence, this result holds for all functions L and Q (with $L(k, \ell, q) \geqslant \ell$ and $Q(k, \ell, q) \geqslant q$, as given in the definition of the problem). Since the model-checking problem only allows to check sentences, we encode the inputs of the formula using colours. This is the reason why we require the class $\mathcal{C}$ to be closed under colour expansions. A formal proof of this result can be found in [12].

For our second result, we consider the parameterised 1-dimensional consistent-learning problem.

Proposition 7.4. *Let $\mathcal{C}$ be a class of structures that is closed under colour expansions such that* p-FO-Mc *is fixed-parameter tractable on $\mathcal{C}$. Then, the following learning problem is fixed-parameter tractable on $\mathcal{C}$ as well.*

Input:	*σ-structure $\mathfrak{A}$, training sequence* $T \in \Big(\big(U(\mathfrak{A})\big)^k \times \{0,1\}\Big)^m$, $\ell, q \in \mathbb{N}$
Parameter:	$\ell + q + \lvert\sigma\rvert$
Problem:	*Return a formula* $\varphi(x, \bar{y}) \in \mathrm{FO}[\sigma, q]$ *with* $\lvert\bar{y}\rvert = \ell$ *and return a tuple* $\bar{w} \in \big(U(\mathfrak{A})\big)^\ell$ *such that the hypothesis* $h^{\mathfrak{A}}_{\varphi,\bar{w}}$ *is consistent with* T, *or reject if there is no such hypothesis.*

To solve this problem, we first encode the positive and negative examples in the structure using fresh colours P_+ and P_-. To check whether there is a consistent hypothesis, we can go through all non-equivalent formulas $\varphi \in \mathrm{FO}[\sigma, q]$ and check whether the sentence

$$\exists y_1 \ldots \exists y_\ell \forall x \Big(\big(P_+x \implies \varphi(x, y_1, \ldots, y_\ell)\big) \\ \wedge \big(P_-x \implies \neg\varphi(x, y_1, \ldots, y_\ell)\big)\Big)$$

holds in $\mathfrak{A}$. Once we have found a suitable formula, we can iteratively search for the parameters w_1 to w_ℓ. For that, in the ith iteration, we can encode the already found parameters w_1 to w_{i-1} as well as our current choice for the parameter w_i in the structure and hard code this partial assignment in the above sentence. Then, using an oracle for the model-checking problem, we can ask whether there are choices for the remaining parameters such that the resulting hypothesis is consistent. If there are such choices, we fix the current value for w_i and continue with the next parameter. Again, the formal proof can be found in [12].

Now, we come to the main result of this section.

Theorem 7.5. *For every effectively nowhere dense class $\mathcal{C}$ of structures, there are functions $L, Q\colon \mathbb{N}^3 \to \mathbb{N}$ such that* p-FO-Learn-ERM(L, Q) *is fixed-parameter tractable on $\mathcal{C}$.*

The remainder of this section is devoted to the proof and the consequences of this theorem. In the proofs we have seen before, we typically go through all (finitely many) formulas and then look for the right parameters for a suitable hypothesis. For the proof of Theorem 7.5, we do it the other way round. That is, our goal is to find a parameter tuple $\bar{w}$ such that there is some formula φ so that the hypothesis $h^{\mathfrak{A}}_{\varphi,\bar{w}}$ is (almost) consistent. Once we have found the right parameter tuple, we can go through all formulas, compute the training error using a model-checking algorithm, and then return the hypothesis that minimises this error.

To choose the right parameters, we first group the examples from the training sequence by their local types. If vertices of a specific type appear only as positive or only as negative examples, this type can be hard coded as positive or negative in the final formula, without the use of any parameters. Thus, it suffices to consider only the remaining ('critical') examples that cannot be classified by their local types alone. In our algorithm, we identify the regions in the structure where most of the critical examples are present. Since the problem p-FO-Learn-ERM allows an additional error ε, we can ignore regions in the structure with a very low density of critical examples. For the other regions, our algorithm identifies the vertices that Splitter would choose in the splitter game on the structure and use those vertices as parameters. A formula can then use these parameters to follow the splitter game and individualise almost all the critical examples such that they can be correctly classified.

For the remainder of this section, let $r\colon \mathbb{N} \to \mathbb{N}$ be the function from Fact 2.13 that gives us the bound on the locality radius $r(q)$ for formulas of quantifier rank at most q.

The following lemma helps us to limit the search space for the parameters to the neighbourhood of a small set. The size of the set depends on the additive error that we allow.

Lemma 7.6. *Let σ be a relational signature, let $\mathfrak{A}$ be a σ-structure, $k, \ell, q \in \mathbb{N}$, $r := r(q)$, $\varepsilon > 0$, and let $T \in \left(\left(U(\mathfrak{A})\right)^k \times \{0,1\}\right)^m$ be a training sequence. Then there exists a set $X \subseteq U(\mathfrak{A})$ of size $|X| \leqslant k\ell/\varepsilon$ such that the following holds. For all $\varepsilon' \geqslant 0$, if there is an ℓ-tuple $\bar{w}'$ in $\mathfrak{A}$ and an FO-formula φ' with $k+\ell$ free variables of quantifier rank at most q such that $\mathrm{err}_T(h^{\mathfrak{A}}_{\varphi',\bar{w}'}) \leqslant \varepsilon'$, then there is also a tuple $\bar{w} \in \left(N^{\mathfrak{A}}_{4r+2}(X)\right)^\ell$ and an FO-formula φ with $k+\ell$ free variables of quantifier rank at most q such that $\mathrm{err}_T(h^{\mathfrak{A}}_{\varphi,\bar{w}}) \leqslant \varepsilon' + \varepsilon$.*

Furthermore, for every class $\mathcal{C}$ of structures that is closed under colour expansions and that has a fixed-parameter tractable FO model-checking problem, there is an fpt algorithm with parameter $k + \ell + q + 1/\varepsilon + |\sigma|$ that computes X on input $k, \ell, q, \varepsilon, \mathfrak{A}, T$ for $\mathfrak{A} \in \mathcal{C}$.

Proof. If the training examples are of length $k = 0$ or we have $\ell = 0$, that is, we do not use any parameters, then the statement is trivial. Thus, let $k, \ell \in \mathbb{N}_{\geqslant 1}$. We call an example $(\bar{v}, \gamma) \in T$ *critical* if there is another example $(\bar{v}', \gamma') \in T$ with a different label $\gamma' \neq \gamma$ that has the same local type $\mathrm{ltp}^{\mathfrak{A}}_{q,r}(\bar{v}') = \mathrm{ltp}^{\mathfrak{A}}_{q,r}(\bar{v})$. Let C be the subsequence of T that contains exactly the critical examples. Because of Fact 2.13, to distinguish two tuples $\bar{v}$, $\bar{v}'$ of the same local type by a formula of quantifier rank at most q, we need to find parameters $\bar{w}$ such that $\mathrm{ltp}^{\mathfrak{A}}_{q,r}(\bar{v}\bar{w}) \neq \mathrm{ltp}^{\mathfrak{A}}_{q,r}(\bar{v}'\bar{w})$. Therefore, by the Local Composition Lemma (Lemma 2.16), we know that we have to choose a parameter in the $(2r+1)$-neighbourhood of one of the two tuples. To identify the critical examples that are possibly affected when we choose a vertex w as a parameter, let $C(w)$ be the subsequence of C that contains all critical examples that have distance at most $2r+1$ from w in $\mathfrak{A}$.

To compute the set X, we iteratively find elements $x_1, \dots, x_p \in U(\mathfrak{A})$ (in a greedy fashion) with disjoint $(2r+1)$-neighbourhoods such that

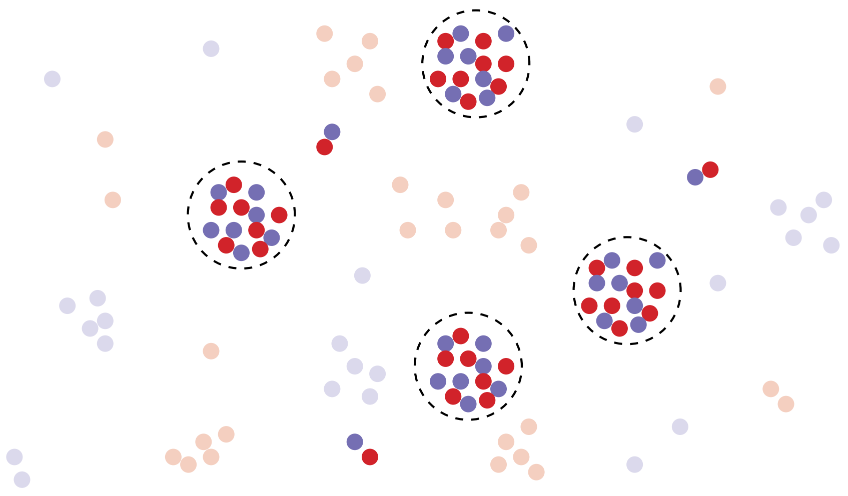


Figure 7.1: Construction of the set X in Lemma 7.6. Positive examples are shown in purple and negative examples are shown in red. Non-critical examples, i. e. those that can simply be classified by their local type, are grayed out. The neighbourhoods of the chosen vertices for the set X are shown as dashed circles. These contain almost all critical examples. Thus, the error is only increased slightly be restricting ourselves to parameters from these neighbourhoods.

the chosen elements affect as many critical examples as possible. See Figure 7.1 for an illustration. Formally, for every $i \geqslant 1$, we choose $x_i \in U(\mathfrak{A})$ such that $\operatorname{dist}(x_i, x_j) > 4r + 2$ for all already found x_j with $j < i$ and, subject to this condition, $|C(x_i)|$ is maximum. If no such x_i exists, set $p = i - 1$ and stop the construction.

Note that for every entry v of a k-tuple $\bar{v}$ in a critical example $(\bar{v}, \gamma)$, there is at most one of the x_i in the $(2r+1)$-neighbourhood of v. Since $\bar{v}$ contains at most k distinct elements, there are at most k of the x_i such that $(\bar{v}, \gamma) \in C(x_i)$. Thus, we have that $\sum_{i=1}^{p} |C(x_i)| \leqslant k\,|C|$. This implies that there are at most $k\ell/\varepsilon$ of the x_i with $|C(x_i)| \geqslant \frac{\varepsilon}{\ell}\,|C|$. Since the x_i are sorted by decreasing $|C(x_i)|$, we have $|C(x_i)| < \frac{\varepsilon}{\ell}\,|C|$ for all $i > k\ell/\varepsilon$. We choose $X = \{x_1, \dots, x_{\min\{p, k\ell/\varepsilon\}}\}$. Then, by construction, we have $|C(u)| < \frac{\varepsilon}{\ell}\,|C|$ for all $u \in U(\mathfrak{A}) \setminus N^{\mathfrak{A}}_{4r+2}(X)$.

To show that X satisfies the error bounds from the lemma, let $\bar{w}'$ be an ℓ-tuple in $\mathfrak{A}$ and let φ' be an FO-formula with $k + \ell$ free variables of quantifier rank at most q such that $\operatorname{err}_T(h^{\mathfrak{A}}_{\varphi', \bar{w}'}) \leqslant \varepsilon'$. Without loss of generality, let $\bar{w}' = \bar{w}^{\text{in}}\bar{w}^{\text{out}}$, where $\bar{w}^{\text{in}}$ only contains entries from $N^{\mathfrak{A}}_{4r+2}(X)$ and $\bar{w}^{\text{out}}$ only contains entries that are not contained in $N^{\mathfrak{A}}_{4r+2}(X)$. In the remainder of this proof, we show that the parameters from $\bar{w}^{\text{in}}$ suffice to distinguish enough examples from each other.

Claim 1. Let $\bar{v}, \bar{v}' \in \big(U(\mathfrak{A})\big)^k$ be such that $\operatorname{ltp}^{\mathfrak{A}}_{q,r}(\bar{v}\bar{w}^{\text{in}}) = \operatorname{ltp}^{\mathfrak{A}}_{q,r}(\bar{v}'\bar{w}^{\text{in}})$. If $\operatorname{ltp}^{\mathfrak{A}}_{q,r}(\bar{v}\bar{w}') \neq \operatorname{ltp}^{\mathfrak{A}}_{q,r}(\bar{v}'\bar{w}')$, then $\bar{v} \in C(w)$ or $\bar{v}' \in C(w)$ for some entry w of $\bar{w}^{\text{out}}$.

Proof. Let $\bar{v}, \bar{v}' \in \big(U(\mathfrak{A})\big)^k$ with $\mathrm{ltp}^{\mathfrak{A}}_{q,r}(\bar{v}\bar{w}^{\mathrm{in}}) = \mathrm{ltp}^{\mathfrak{A}}_{q,r}(\bar{v}'\bar{w}^{\mathrm{in}})$. We prove the claim by contraposition. Assume $\bar{v}, \bar{v}' \notin C(w)$ for all entries w of $\bar{w}^{\mathrm{out}}$. Then, by the definition of $C(w)$, $\mathrm{dist}(\bar{v}, \bar{w}^{\mathrm{out}}) > 2r+1$ and $\mathrm{dist}(\bar{v}', \bar{w}^{\mathrm{out}}) > 2r+1$. Thus, by Lemma 2.16, it follows that $\mathrm{ltp}^{\mathfrak{A}}_{q,r}(\bar{v}\bar{w}') = \mathrm{ltp}^{\mathfrak{A}}_{q,r}(\bar{v}\bar{w}^{\mathrm{in}}\bar{w}^{\mathrm{out}}) = \mathrm{ltp}^{\mathfrak{A}}_{q,r}(\bar{v}'\bar{w}^{\mathrm{in}}\bar{w}^{\mathrm{out}}) = \mathrm{ltp}^{\mathfrak{A}}_{q,r}(\bar{v}'\bar{w}')$. ⌟

By Corollary 2.14, there is a set Φ' of $(k+\ell)$-variable q-types such that, for every k-tuple $\bar{v}$ from $\mathfrak{A}$, we have that $\mathfrak{A} \models \varphi'[\bar{v}\bar{w}']$ if and only if $\mathrm{ltp}^{\mathfrak{A}}_{q,r}(\bar{v}\bar{w}') \in \Phi'$. Next, we define a set Φ^{in} that is a projection of the types in Φ' to $(k+|\bar{w}^{\mathrm{in}}|)$-variable q-types. For every example $(\bar{v},\lambda) \in T$, let Φ^{in} contain $\mathrm{ltp}^{\mathfrak{A}}_{q,r}(\bar{v}\bar{w}^{\mathrm{in}})$ if and only if for at least half of the examples $(\bar{v}',\lambda') \in T$ with the same local type $\mathrm{ltp}^{\mathfrak{A}}_{q,r}(\bar{v}'\bar{w}^{\mathrm{in}}) = \mathrm{ltp}^{\mathfrak{A}}_{q,r}(\bar{v}\bar{w}^{\mathrm{in}})$, we have that $\mathrm{ltp}^{\mathfrak{A}}_{q,r}(\bar{v}'\bar{w}') \in \Phi'$. Let $\bar{y}^{\mathrm{in}} := (y_1, \ldots, y_{|\bar{w}^{\mathrm{in}}|})$ and

$$\varphi^{\mathrm{in}}(\bar{x},\bar{y}^{\mathrm{in}}) := \bigvee_{\mathrm{ltp}^{\mathfrak{A}}_{q,r}(\bar{v}\bar{w}^{\mathrm{in}}) \in \Phi^{\mathrm{in}}} \quad \bigwedge_{\psi(\bar{x},\bar{y}^{\mathrm{in}}) \in \mathrm{ltp}^{\mathfrak{A}}_{q,r}(\bar{v}\bar{w}^{\mathrm{in}})} \psi(\bar{x},\bar{y}^{\mathrm{in}}).$$

Then, we let $\bar{w}$ be an ℓ-tuple with $w_i = w^{\mathrm{in}}_i$ for all $i \leqslant |\bar{w}^{\mathrm{in}}|$, and we set $w_i = a$ for all other entries for some arbitrary element a in $N^{\mathfrak{A}}_{4r(q)+2}(X)$. Furthermore, let $\varphi(\bar{x},\bar{y}) := \varphi^{\mathrm{in}}(\bar{x},\bar{y}^{\mathrm{in}})$.

Claim 2. It holds that $\mathrm{err}_T(h^{\mathfrak{A}}_{\varphi,\bar{w}}) \leqslant \varepsilon' + \varepsilon$.

Proof. We show that there are at most $\varepsilon \cdot m$ examples $(\bar{v},\lambda)$ in T with $h^{\mathfrak{A}}_{\varphi',\bar{w}'}(\bar{v}) = \lambda$ and $h^{\mathfrak{A}}_{\varphi,\bar{w}}(\bar{v}) \neq \lambda$. With this, the claim immediately follows.

Let $h := h^{\mathfrak{A}}_{\varphi,\bar{w}}$ and $h' := h^{\mathfrak{A}}_{\varphi',\bar{w}'}$. Due to the construction of Φ^{in} and φ, for every example $(\bar{v},\lambda)$ in T with $h(\bar{v}) \neq h'(\bar{v})$, there is a distinct corresponding example $(\bar{v}',\lambda')$ in T with the same local type $\mathrm{ltp}^{\mathfrak{A}}_{q,r}(\bar{v}'\bar{w}^{\mathrm{in}}) = \mathrm{ltp}^{\mathfrak{A}}_{q,r}(\bar{v}\bar{w}^{\mathrm{in}})$ such that $h(\bar{v}') = h'(\bar{v}')$. Now let $(\bar{v},\lambda) \in T$ be such that h' correctly classifies $\bar{v}$, i. e. $h'(\bar{v}) = \lambda$, and h misclassifies $\bar{v}$, i. e. $h(\bar{v}) \neq \lambda$. Let $(\bar{v}',\lambda')$ be the distinct corresponding example in T with $\mathrm{ltp}^{\mathfrak{A}}_{q,r}(\bar{v}'\bar{w}^{\mathrm{in}}) = \mathrm{ltp}^{\mathfrak{A}}_{q,r}(\bar{v}\bar{w}^{\mathrm{in}})$ and $h(\bar{v}') = h'(\bar{v}')$. Because of the equality of the local types and the construction of h, we have $h(\bar{v}) = h(\bar{v}')$. Thus, we can deduce that $h'(\bar{v}) \neq h'(\bar{v}')$ and $\mathrm{ltp}^{\mathfrak{A}}_{q,r}(\bar{v}'\bar{w}') \neq \mathrm{ltp}^{\mathfrak{A}}_{q,r}(\bar{v}\bar{w}')$. It follows from Claim 1 that $\bar{v} \in C(w)$ or $\bar{v}' \in C(w)$ for some entry w of $\bar{w}^{\mathrm{out}}$. Since

$$\Big|\bigcup_{w \in \bar{w}^{\mathrm{out}}} C(w)\Big| \leqslant \sum_{w \in \bar{w}^{\mathrm{out}}} |C(w)| < \ell \cdot \frac{\varepsilon}{\ell} |C| \leqslant \varepsilon \cdot m,$$

there are at most $\varepsilon \cdot m$ examples $(\bar{v},\lambda)$ in T with $h'(\bar{v}) = \lambda$ and $h(\bar{v}) \neq \lambda$. ⌟

It remains to show that the set X can be computed by an fpt algorithm. Let $\mathcal{C}$ be a class of structures that is closed under colour expansions and that has a fixed-parameter tractable FO model-checking problem. Furthermore, let $\mathfrak{A}$ be a σ-structure from $\mathcal{C}$. Up to equivalence, there are only finitely many $\mathrm{FO}[\sigma, q]$-formulas with k free variables. Analogously to the proof of Proposition 7.3, by encoding the inputs of the formulas using colours, we can use the model-checking result on $\mathcal{C}$ to check for every formula and for every tuple from the training examples whether the tuple satisfies the formula. Thus, for every tuple from the training examples, we can compute the local type by an fpt algorithm. Based on the local types, we can compute the sequence of all critical examples, the sequences $C(u)$ for all $u \in U(\mathfrak{A})$, and finally also the set X, by an fpt algorithm with parameter $k + \ell + q + 1/\varepsilon + |\sigma|$. $\square$

In the following, let $\mathcal{C}$ be an effectively nowhere dense class of structures and let $\lambda\colon \mathbb{N}_{\geqslant 1} \to \mathbb{N}_{\geqslant 1}$ be a computable function such that Splitter wins the λ-splitter game on $\mathcal{C}$. In our proof, we consider the splitter game with a larger radius

$$R(k, \ell^*, q^*) := 4^{\ell^*-1} \cdot (k+2)\big(2r(q^*)+1\big).$$

The specific choice of the radius is justified in the proof of Lemma 7.7. Based on this radius, for our functions L and Q that bound the number of parameters and the quantifier rank we may use, we choose

$$L(k, \ell^*, q^*) := \Big(2\lambda\big(R(k, \ell^*, q^*)\big) - 1\Big) \cdot \ell^*$$

and

$$Q(k, \ell^*, q^*) := q^* + \lambda\big(R(k, \ell^*, q^*)\big) \cdot \log R(k, \ell^*, q^*).$$

The choice of L comes from the fact that we go through the splitter game and choose $2\ell^*$ parameters in every round of the game, except for the last round, where ℓ^* parameters suffice. With the larger quantifier rank $Q(k, \ell^*, q^*)$, we want to be able to work with neighbourhoods of radius $R(k, \ell^*, q^*)$. The specific choice will also become clear in the proof of Lemma 7.7.

Let $\mathfrak{A}, T, k, \ell^*, q^*, \varepsilon$ be the input of p-FO-Learn-ERM(L, Q), where $\mathfrak{A}$ is a σ-structure from $\mathcal{C}$, T is a training sequence of length m, we have $k, \ell^*, q^* \in \mathbb{N}$, and $\varepsilon > 0$. With a slight abuse of notation, let $r := r(q^*)$, $R := R(k, \ell^*, q^*)$, and $\lambda := \lambda(R)$. Furthermore, let $\ell := L(k, \ell^*, q^*) = (2\lambda - 1) \cdot \ell^*$ and $q := Q(k, \ell^*, q^*) = q^* + \lambda \log R$.

Let $\varepsilon^* \geqslant 0$ be such that there is an FO-formula $\varphi^*(\bar{x}, \bar{y})$ of quantifier rank at most q^* with $k + \ell^*$ free variables and a tuple $\bar{w}^* \in \big(U(\mathfrak{A})\big)^{\ell^*}$ such that $\mathrm{err}_T(h^{\mathfrak{A}}_{\varphi^*, \bar{w}^*}) \leqslant \varepsilon^*$. Our goal is to find an FO-formula $\varphi(\bar{x}, \bar{y})$ of quantifier rank at most q with $k + \ell$ free variables and a tuple $\bar{w} \in \big(U(\mathfrak{A})\big)^{\ell}$ such that $\mathrm{err}_T(h^{\mathfrak{A}}_{\varphi, \bar{w}}) \leqslant \varepsilon^* + \varepsilon$. Equivalently, this means that the hypothesis $h^{\mathfrak{A}}_{\varphi, \bar{w}}$ misclassifies at most $m \cdot (\varepsilon^* + \varepsilon)$ examples from T.

As indicated above, we first look for the right parameters. Hence, our first goal is to find an ℓ-tuple $\bar{w}$ such that there is a formula φ of quantifier rank at most q with $\mathrm{err}_T(h^{\mathfrak{A}}_{\varphi, \bar{w}}) \leqslant \varepsilon^* + \varepsilon$. Once we have found such a tuple, we can simply go through all formulas of quantifier rank at most q and use the fixed-parameter tractability of the model-checking problem on nowhere dense classes to evaluate the resulting hypothesis on the training sequence. We can then return the hypothesis that minimises the training error. Thus, in the following, we describe a procedure to find such a tuple.

Our algorithm runs λ many steps that correspond to the moves in the splitter game. It computes structures $\mathfrak{A}_0, \dots, \mathfrak{A}_\lambda$ and training sequences $T_0, \dots, T_\lambda$. We start with $\mathfrak{A}_0 := \mathfrak{A}$ and $T_0 := T$. In each step, while reducing the search space for parameters using Lemma 7.6, the algorithm gives up on at most $\frac{\varepsilon}{\lambda} \cdot m$ many examples, in addition to those that the hypothesis $h^{\mathfrak{A}}_{\varphi^*, \bar{w}^*}$ is unable to classify correctly. In the last step, our algorithm will find a hypothesis that is at least as good as $h^{\mathfrak{A}}_{\varphi^*, \bar{w}^*}$ on the remaining examples that we have not given up on. Thus, all in all, the hypothesis found by our algorithm will make at most $(\varepsilon^* + \varepsilon) \cdot m$ errors on the training sequence.

Theorem 7.5 is a direct consequence of the following result.

Lemma 7.7. *There are signatures $\sigma_0, \dots, \sigma_\lambda$, structures $\mathfrak{A}_0, \dots, \mathfrak{A}_\lambda$ and training sequences $T_0, \dots, T_\lambda$ with $\sigma_0 = \sigma$, $\mathfrak{A}_0 = \mathfrak{A}$, and $T_0 = T$ such that the following holds.*

(1) For all $i \in [0, \lambda]$, the structure $\mathfrak{A}_i$ is a σ_i-structure. Moreover, σ_i only depends on σ, i, k, ℓ^*, and q^*. In particular, σ_i does not depend on $\mathfrak{A}$.

(2) For all $i \in [0, \lambda]$, Splitter has a winning strategy for the $(\lambda - i, R)$-splitter game on (the Gaifman graph of) $\mathfrak{A}_i$.

(3) For all $i \in [0, \lambda - 1]$, there is an ℓ^*-tuple $\bar{u}_i$ in $\mathfrak{A}_i$ and an $\mathsf{FO}[\sigma_i, q^*]$-formula ψ_i with $k + \ell^*$ free variables such that $\mathrm{err}_{T_i}\left(h^{\mathfrak{A}_i}_{\psi_i, \bar{u}_i}\right) \leqslant \varepsilon^* + \frac{i}{\lambda}\varepsilon$.

(4) In $\mathfrak{A}_\lambda$, there is an $\mathsf{FO}[\sigma_\lambda, q^*]$-formula ψ_λ with k free variables, i.e. without any free parameter variables, such that $\mathrm{err}_{T_\lambda}\left(h^{\mathfrak{A}_\lambda}_{\psi_\lambda}\right) \leqslant \varepsilon^* + \varepsilon$.

(5) For all $i \in [0, \lambda - 1]$, there is a $\big((2(\lambda - i) - 1)\ell^*\big)$-tuple $\bar{w}_i$ in $\mathfrak{A}_i$ and an $\mathsf{FO}[\sigma_i, q^* + (\lambda - i - 1) \cdot R]$-formula φ_i with $k + \big(2(\lambda - i) - 1\big) \cdot \ell^*$ free variables such that $\mathrm{err}_{T_i}\left(h^{\mathfrak{A}_i}_{\varphi_i, \bar{w}_i}\right) \leqslant \varepsilon^* + \varepsilon$.

Furthermore, there is an fpt algorithm with parameter $k + \ell^* + q^* + 1/\varepsilon + |\sigma|$ that computes $\mathfrak{A}_1, \dots, \mathfrak{A}_\lambda$, $T_1, \dots, T_\lambda$, and suitable $\bar{w}_{\lambda-1}, \dots, \bar{w}_0$ and $\varphi_{\lambda-1}, \dots, \varphi_0$ on input $k, \ell^*, q^*, \varepsilon, \mathfrak{A}, T$ for $\mathfrak{A} \in \mathcal{C}$.

Before we prove this result, we first use it to prove our main result.

Proof of Theorem 7.5. In order to solve p-FO-Learn-ERM(L, Q), we use Lemma 7.7 to compute the ℓ-tuple $\bar{w}_0$ and the $\mathsf{FO}[\sigma, q]$-formula φ_0 with $k + \ell$ free variables such that $\mathrm{err}_T\left(h^{\mathfrak{A}}_{\varphi_0, \bar{w}_0}\right) \leqslant \varepsilon^* + \varepsilon$. Then, we can simply return $\varphi := \varphi_0$ and $\bar{w} := \bar{w}_0$. By Item (5) of Lemma 7.7, this hypothesis fulfils the requirements of p-FO-LEARN-ERM(L, Q). $\square$

The remainder of this section is devoted to the proof of Lemma 7.7. In the proof of the lemma, we use the following result that is inspired by the so-called Vitali Covering Lemma [93].

Lemma 7.8. *Let $\mathfrak{A}$ be a structure, $X \subseteq U(\mathfrak{A})$, and $r \geqslant 1$. Then there is a set $Z \subseteq X$ and an $R = 4^p r$ for some $p \in [0, |X| - 1]$ such that $\mathcal{N}_r^{\mathfrak{A}}(X) \subseteq \mathcal{N}_R^{\mathfrak{A}}(Z)$ and $\operatorname{dist}^{\mathfrak{A}}(z, z') > 2R + 1$ for all distinct $z, z' \in Z$.*

Proof. Let $R_i := 4^i r$ for all $i \geqslant 0$. We inductively construct a sequence $Z_0 \supset Z_1 \supset \cdots \supset Z_p$ of subsets of X with $\mathcal{N}_r^{\mathfrak{A}}(X) \subseteq \mathcal{N}_{R_i}^{\mathfrak{A}}(Z_i)$ for all i such that Z_p is the desired set Z.

Let $Z_0 := X$. Now assume that Z_i is already defined. If $\operatorname{dist}^{\mathfrak{A}}(z, z') > 2R_i + 1$ for all distinct $z, z' \in Z_i$, we set $p := i$ and stop the construction. Otherwise, let $Z_{i+1} \supset Z_i$ be inclusion-wise maximal such that $\operatorname{dist}^{\mathfrak{A}}(z, z') > 2R_i + 1$ for all distinct $z, z' \in Z_{i+1}$. Then, for all $y \in Z_i$, there is a $z \in Z_{i+1}$ such that $\operatorname{dist}^{\mathfrak{A}}(y, z) \leqslant 2R_i + 1$ which implies that $\mathcal{N}_{R_i}^{\mathfrak{A}}(y) \subseteq \mathcal{N}_{3R_i+1}^{\mathfrak{A}}(z) \subseteq \mathcal{N}_{R_{i+1}}^{\mathfrak{A}}(z)$. Thus, $\mathcal{N}_r^{\mathfrak{A}}(X) \subseteq \mathcal{N}_{R_i}^{\mathfrak{A}}(Z_i) \subseteq \mathcal{N}_{R_{i+1}}^{\mathfrak{A}}(Z_{i+1})$.

Since Z_{i+1} is a proper subset of Z_i for all i, we have $p \leqslant |X| - 1$. $\square$

In addition to this result, we also use a modified version of the (ℓ, r)-splitter game. In round i of this game, Connector not only picks a vertex v_i, but they also choose a new radius $r_i \leqslant r$, and the game continues in $N_{r_i}^{G_{i-1}}(v_i)$. Clearly, reducing the radius does not help Connector. Thus, if Splitter has a winning strategy in the (ℓ, r)-splitter game on a graph G, then they also have a winning strategy in the modified (ℓ, r)-splitter game on G. We can now prove Lemma 7.7.

Proof of Lemma 7.7. At first, we describe how to construct the structures $\mathfrak{A}_0, \dots, \mathfrak{A}_\lambda$ and the training sequences $T_0, \dots, T_\lambda$.

Let $\mathfrak{A}_0 := \mathfrak{A}$ and $T_0 := T$. Then Items (1)–(3) of Lemma 7.7 hold for $i = 0$. We now describe the ith step of the algorithm for $i \in [\lambda]$. Assume that the σ_{i-1}-structure $\mathfrak{A}_{i-1}$ and the training sequence T_{i-1} have already been computed. Furthermore, assume that Item (3) of Lemma 7.7 holds for $i - 1$, i. e., assume that there is an ℓ^*-tuple $\bar{u}_{i-1}$ in $\mathfrak{A}_{i-1}$ and an FO$[\sigma_{i-1}, q^*]$-formula ψ_{i-1} with $k + \ell^*$ free variables such that $\operatorname{err}_{T_{i-1}}\left(h_{\psi_{i-1}, \bar{u}_{i-1}}^{\mathfrak{A}_{i-1}}\right) \leqslant \varepsilon^* + \frac{i-1}{\lambda}\varepsilon$.

We use Lemma 7.6 to compute a set X_i of size $|X_i| \leqslant k\ell^*\lambda/\varepsilon$ such that there is an ℓ^*-tuple $\bar{u}'_{i-1}$ in $\mathcal{N}_{4r+2}^{\mathfrak{A}_{i-1}}(X_i)$ and an FO$[\sigma_{i-1}, q^*]$-formula ψ'_{i-1} with $k + \ell^*$ free variables such that $\operatorname{err}_{T_{i-1}}\left(h_{\psi'_{i-1}, \bar{u}'_{i-1}}^{\mathfrak{A}_{i-1}}\right) \leqslant \varepsilon^* + \frac{i}{\lambda}\varepsilon$. Since the tuple $\bar{u}'_{i-1}$ has arity ℓ^*, there is a subset Y_i of X_i of size $\ell_i := |Y_i| \leqslant \ell^*$ such that $\bar{u}'_{i-1}$ is also contained in $\mathcal{N}_{4r+2}^{\mathfrak{A}_{i-1}}(Y_i)$. To find the right set $Y_i = \{y_{i,1}, \dots, y_{i,\ell_i}\}$, we try all possible choices. This adds a multiplicative cost of $|X_i|^{\ell^*} \leqslant \left(\frac{k\ell^*\lambda}{\varepsilon}\right)^{\ell^*}$ in every step of the algorithm, so all in all we check $\left(\frac{k\ell^*\lambda}{\varepsilon}\right)^{\lambda\ell^*}$ different combinations of vertices for the Y_i, which is allowed in an fpt algorithm. In the end, we choose the combination that minimises the error of the found hypothesis.

Next, we apply Lemma 7.8 and obtain a set $Z_i \subseteq Y_i$ and a radius $R_i = 4^j(k+2)(2r+1)$ for some $j \leqslant |Y_i| - 1 \leqslant \ell^* - 1$ such that $N^{\mathfrak{A}_{i-1}}_{(k+2)(2r+1)}(Y_i) \subseteq N^{\mathfrak{A}_{i-1}}_{R_i}(Z_i)$ and $\operatorname{dist}^{\mathfrak{A}_{i-1}}(z,z') > 2R_i + 1$ for all distinct $z, z' \in Z_i$. Note that $R_i \leqslant R$, where R is the radius from the (λ, R)-splitter game described above. Suppose that $Z_i = \{z_{i,1}, \dots, z_{i,\ell'_i}\}$ for some $\ell'_i \leqslant \ell_i \leqslant \ell^*$. For every $j \in [\ell'_i]$, let $w_{i,j}$ be Splitter's answer if Connector picks $z_{i,j}$ together with the radius R_i in the modified $(\lambda - i, R_i)$-splitter game on $\mathfrak{A}_{i-1}$. We set $\bar{w}_i := (w_{i,1}, \dots, w_{i,\ell^*})$ with $w_{i,j} = w_{i,\ell'_i}$ for all $j > \ell'_i$.

Now, we can describe the construction of $\mathfrak{A}_i$. We start with the induced R_i-neighbourhood structure $\mathfrak{A}'_i := \mathcal{N}^{\mathfrak{A}_{i-1}}_{R_i}(Z_i)$. Then, we obtain the next structure $\mathfrak{A}_i$ by the following four steps.

1. Expand the structure by fresh colours $D_{i,j,d}$ for $j \in [\ell_i]$ and $d \in \{0, \dots, (k+2)(2r+1)\}$. We let $D_{i,j,d}(\mathfrak{A}_i)$ be the set of all vertices v such that $\operatorname{dist}^{\mathfrak{A}_{i-1}}(v, y_j) = d$.

2. Expand the structure by fresh relation symbols $S_{i,I}$ for all relation symbols $S \in \sigma_{i-1}$ and sets of indices $I \subseteq [\operatorname{ar}(S)]$. For every tuple $\bar{a} = (a_1, \dots, a_{\operatorname{ar}(S)}) \in S(\mathfrak{A}_{i-1})$ where, for all indices $p \in I$, the entry a_p is equal to one of the vertices $w_{i,j}$, we let the relation $S_{i,I}$ contain the restriction $\bar{a}|_{[\operatorname{ar}(S)]\setminus I}$ of $\bar{a}$ that contains only indices not contained in I, so we leave out all entries a_p with $p \in I$. These relations are used to remember the connections of the vertices $w_{i,j}$ to the rest of the structure. This allows us to remove those connections in the next step.

3. For every relation symbol S, delete all tuples from the relation that contain at least one of the vertices $w_{i,j}$. Moreover, we add fresh colours $B_{i,j}$ for $j \in [\ell'_i]$ and set $B_{i,j}(\mathfrak{A}_i) := \{w_{i,j}\}$.

4. For each non-empty set $I \subseteq [k]$ and each $|I|$-variable q^*-type $\theta \in \operatorname{Tp}[\sigma_{i-1}, |I|, q^*]$, add an isolated vertex $t_{i,I,\theta}$ and a fresh colour $A_{i,I,\theta}$ with $A_{i,I,\theta}(\mathfrak{A}_i) = \{t_{i,I,\theta}\}$. These isolated vertices are used in the construction of the training sequence T_i to replace vertices that are not contained in $\mathfrak{A}_i$ any more (since they lie outside of the R_i-neighbourhood structure $\mathfrak{A}'_i$).

Before we continue with the construction of T_i, we can already prove Items (1) and (2) of Lemma 7.7. Item (1) of Lemma 7.7 follows directly from the construction of $\mathfrak{A}_i$. Next, we consider Item (2).

Claim 1. For all $i \in [0, \lambda]$, Splitter has a winning strategy for the $(\lambda - i, R)$-splitter game on (the Gaifman graph of) $\mathfrak{A}_i$.

Proof. As described above, the claim holds for $i = 0$ by assumption. Now let $i > 0$. Structurally, $\mathfrak{A}_i$ consists of neighbourhoods $N_{i,j} :=$

$N_{R_i}^{\mathfrak{A}_{i-1}}(z_{i,j}) \setminus \{w_{i,j}\}$ for $j \in [\ell'_i]$ and isolated vertices $w_{i,1}, \dots, w_{i,\ell'_i}$ and $t_{i,I,\theta}$ for I and θ as described above. By the construction of Z_i, the neighbourhoods $N_{i,j}$ and $N_{i,j'}$ are disconnected for all $j \neq j'$. Furthermore, the Gaifman graph of $\mathfrak{A}_i$, restricted to $N_{R_i}^{\mathfrak{A}_{i-1}}(Z_i)$, is a subgraph of the Gaifman graph of $\mathfrak{A}_{i-1}$.

Thus, Splitter's winning strategy on $\mathfrak{A}_{i-1}$ is still valid for $\mathfrak{A}_i$, but one of the steps of the (modified) splitter game, i. e., removing a vertex Splitter chose and continuing in the neighbourhood of a vertex Connector chose, has already been performed in each of the neighbourhoods $N_{i,j}$. Thus, if Splitter has a winning strategy for the $(\lambda-(i-1), R)$-splitter game on $\mathfrak{A}_{i-1}$, then Splitter has a winning strategy for the $(\lambda-i, R)$-splitter game in each of the neighbourhoods $N_{i,j}$ and hence also on $\mathfrak{A}_i$. Since Splitter has a winning strategy for the (λ, R)-splitter game on $\mathfrak{A}$, the claim follows by induction. ⌟

Now we describe the construction of the intermediate training sequence T_i. For every example $(\bar{v}, \gamma) \in T_{i-1}$ with $\bar{v} = (v_1, \dots, v_k) \in \left(U(\mathfrak{A}_{i-1})\right)^k$, we define a tuple $\bar{v}' = (v'_1, \dots, v'_k) \in \left(U(\mathfrak{A}_i)\right)^k$, which is a projection of $\bar{v}$ into $\mathfrak{A}_i$, and add $(\bar{v}', \gamma)$ to T_i. For that, we consider the graph $H_{\bar{v}}$ with vertex set $V(H_{\bar{v}}) = [k]$ and edges (a, b) for all $a, b \in [k]$ such that $1 \leqslant \mathrm{dist}^{\mathfrak{A}_{i-1}}(v_a, v_b) \leqslant 2r+1$. Let $I_1, \dots, I_p$ be the vertex sets of the connected components of $H_{\bar{v}}$. Using parameters from $N_{4r+2}^{\mathfrak{A}_{i-1}}(Y_i)$, we can only handle conflicting examples if they have at least one element in $N_{6r+3}^{\mathfrak{A}_{i-1}}(Y_i)$. Hence, for each component I_j, we proceed as follows. If there is some $j_0 \in I_j$ such that $v_{j_0} \in N_{6r+3}^{\mathfrak{A}_{i-1}}(Y_i)$, we let $v'_a :=$ v_a for all $a \in I_j$. Note that we have $\mathrm{dist}^{\mathfrak{A}_{i-1}}(v_a, v_{j_0}) \leqslant (k-1)(2r+1)$ for all $a \in I_j$ and hence, $\mathrm{dist}^{\mathfrak{A}_{i-1}}(v_a, Y_i) \leqslant 6r+3+(k-1)(2r+1) =$ $(k+2)(2r+1)$. Thus, $v'_a \in N_{(k+2)(2r+1)}^{\mathfrak{A}_{i-1}}(Y_i) \subseteq N_{R_i}^{\mathfrak{A}_{i-1}}(Z_i) \subseteq U(\mathfrak{A}_i)$. This is the point that determines the choice of the larger radius R of the splitter game we introduced earlier. Otherwise, if $v_a \notin N_{6r+3}^{\mathfrak{A}_{i-1}}(Y_i)$ for all $a \in I_j$, we consider the restriction $\bar{v}|_{I_j}$ of $\bar{v}$ to the indices in I_j, and we let $\theta := \mathrm{ltp}_{q^*,r^*}^{\mathfrak{A}_{i-1}}(\bar{v}|_{I_j})$. Then, we set $v'_a := t_{i,I_j,\theta}$ for all $a \in I_j$. Note that for two examples $(\bar{v}'_1, \lambda_1)$, $(\bar{v}'_2, \lambda_2)$ that appear in T_i, the tuples $\bar{v}'_1, \bar{v}'_2$ can only have the same type in $\mathfrak{A}_i$ if their counterparts $\bar{v}_1$ and $\bar{v}_2$ from T_{i-1} have the same type in $\mathfrak{A}_{i-1}$. Thus, our construction does not create any new conflicts in the examples. Moreover, note that $|T_i| = |T| = m$ for all $i \in [0, \lambda]$. We can now prove the remaining items from Lemma 7.7.

Claim 2. For all $i \in [0, \lambda-1]$, there is an ℓ^*-tuple $\bar{u}_i$ in $\mathfrak{A}_i$ and an $\mathrm{FO}[\sigma_i, q^*]$-formula ψ_i with $k+\ell^*$ free variables such that the training error of $h_{\psi_i, \bar{u}_i}^{\mathfrak{A}_i}$ on T_i is bounded by $\varepsilon^* + \frac{i}{\lambda}\varepsilon$.

Proof. We have already handled the case $i = 0$ above. Now let $i > 0$ and assume that the claim holds for $i-1$. As described above, using Lemma 7.6, there is an ℓ^*-tuple $\bar{u}'_{i-1}$ in $N_{4r+2}^{\mathfrak{A}_{i-1}}(X_i)$ and an $\mathrm{FO}[\sigma_{i-1}, q^*]$-formula ψ'_{i-1} with $k+\ell^*$ free variables such that the training error of $h^{\mathfrak{A}_{i-1}}_{\psi'_{i-1},\bar{u}'_{i-1}}$ on T_{i-1} is bounded by $\varepsilon^* + \frac{i}{\lambda}\varepsilon$. By the construction of $\mathfrak{A}_i$, the tuple $\bar{u}'_{i-1}$ is also contained in $\mathfrak{A}_i$. We set $\bar{u}_i := \bar{u}'_{i-1}$. As we have seen during the construction of T_i, for an example $(\bar{v}, \lambda)$ in T_{i-1}, the vertices of $\bar{v}$ outside $N^{\mathfrak{A}_{i-1}}_{(k+2)(2r+1)}(Y_i)$ can only influence the classification of $\bar{v}$ by their local type and do not interact with the parameters. Moreover, using the relations introduced in Steps 2 and 3 in the construction of $\mathfrak{A}_i$, we can interpret $N^{\mathfrak{A}_{i-1}}_{R_i}(Z_i)$ in $\mathfrak{A}_i$. All in all, using the distance information encoded in Step 1 together with the types encoded in Step 4 as well as the relations introduced in Steps 2 and 3 in the construction of $\mathfrak{A}_i$, there is an $\mathrm{FO}[\sigma_i, q^*]$-formula ψ_i with $k + \ell^*$ free variables such that $\mathrm{err}_{T_i}\left(h^{\mathfrak{A}_i}_{\psi_i,\bar{u}_i}\right) \leqslant \varepsilon^* + \frac{i}{\lambda}\varepsilon$. ⌟

Claim 3. In $\mathfrak{A}_\lambda$, there is an $\mathrm{FO}[\sigma_\lambda, q^*]$-formula ψ_λ with k free variables, i.e. without any free parameter variables, such that $\mathrm{err}_{T_\lambda}\left(h^{\mathfrak{A}_\lambda}_{\psi_\lambda}\right) \leqslant \varepsilon^* + \varepsilon$.

Proof. Since Splitter has a winning strategy for the (λ, R)-splitter game on $\mathfrak{A}$, we have $N^{\mathfrak{A}_{\lambda-1}}_{4r+2}(X_\lambda) \subseteq N^{\mathfrak{A}_{\lambda-1}}_{R_\lambda}(Z_\lambda) \subseteq \{w_{\lambda,1}, \dots, w_{\lambda,\ell^*}\}$ in the last step of our computation. Thus, there is an ℓ^*-tuple $\bar{u}'_{\lambda-1}$ containing only vertices from $\{w_{\lambda,1}, \dots, w_{\lambda,\ell^*}\}$ and an $\mathrm{FO}[\sigma_{\lambda-1}, q^*]$-formula $\psi'_{\lambda-1}$ with $k + \ell^*$ free variables such that $\mathrm{err}_{T_{\lambda-1}}\left(h^{\mathfrak{A}_{\lambda-1}}_{\psi'_{\lambda-1},\bar{u}'_{\lambda-1}}\right) \leqslant \varepsilon^* + \varepsilon$. The vertices $w_{\lambda,j}$ can be identified in $\mathfrak{A}_\lambda$ using the fresh colours $B_{\lambda,j}$ in σ_λ. Hence, there is also an $\mathrm{FO}[\sigma_\lambda, q^*]$-formula ψ_λ with k free variables such that $\mathrm{err}_{T_\lambda}\left(h^{\mathfrak{A}_\lambda}_{\psi_\lambda}\right) \leqslant \varepsilon^* + \varepsilon$. ⌟

Claim 4. For all $i \in [0, \lambda - 1]$, there is a $\Big(\big(2(\lambda - i) - 1\big)\ell^*\Big)$-tuple $\bar{w}_i$ in $\mathfrak{A}_i$ and an $\mathrm{FO}[\sigma_i, q^* + (\lambda - i - 1) \cdot R]$-formula φ_i with $k + \big(2(\lambda - i) - 1\big) \cdot \ell^*$ free variables such that $\mathrm{err}_{T_i}\left(h^{\mathfrak{A}_i}_{\varphi_i,\bar{w}_i}\right) \leqslant \varepsilon^* + \varepsilon$.

Proof. For $i = \lambda - 1$, we have seen in the proof of Claim 3 that there is an ℓ^*-tuple $\bar{u}'_{\lambda-1}$ containing only vertices from $\{w_{\lambda,1}, \dots, w_{\lambda,\ell^*}\}$ and an $\mathrm{FO}[\sigma_{\lambda-1}, q^*]$-formula $\psi'_{\lambda-1}$ with $k + \ell^*$ free variables such that $\mathrm{err}_{T_{\lambda-1}}\left(h^{\mathfrak{A}_{\lambda-1}}_{\psi'_{\lambda-1},\bar{u}'_{\lambda-1}}\right) \leqslant \varepsilon^* + \varepsilon$. Thus, we can choose $\bar{w}_{\lambda-1} = (w_{\lambda,1}, \dots, w_{\lambda,\ell^*})$. With slight modifications of $\psi'_{\lambda-1}$, we can obtain an $\mathrm{FO}[\sigma_{\lambda-1}, q^*]$-formula $\varphi_{\lambda-1}$ with $k + \ell^*$ free variables such that $h^{\mathfrak{A}_{\lambda-1}}_{\varphi_{\lambda-1},\bar{w}_{\lambda-1}}$ has the same property. These modifications might be needed, since the vertices in $\bar{w}_{\lambda-1}$ may appear in a different order than in $\bar{u}'_{\lambda-1}$.

Now assume that the claim holds for $i + 1$, so there is a $\Big(\big(2(\lambda - i - 1) - 1\big) \cdot \ell^*\Big)$-tuple $\bar{w}_{i+1}$ in $\mathfrak{A}_{i+1}$ and an $\mathrm{FO}[\sigma_i, q^* + (\lambda - i - 2) \cdot R]$-formula φ_{i+1} with $k + \big(2(\lambda - i - 1) - 1\big) \cdot \ell^*$ free variables such that $\mathrm{err}_{T_{i+1}}\left(h^{\mathfrak{A}_{i+1}}_{\varphi_{i+1},\bar{w}_{i+1}}\right) \leqslant \varepsilon^* + \varepsilon$. To obtain the $\Big(\big(2(\lambda - i) - 1\big) \cdot \ell^*\Big)$-tuple $\bar{w}_i$, we simply drop all vertices from $\bar{w}_{i+1}$ that are not contained in $\mathfrak{A}_i$ (or, more formally, replace them with arbitrary vertices from $\mathfrak{A}_i$ and do not use them in our formula) and append the vertices $w_{i,1}, \dots, w_{i,\ell^*}$ and $y_{i,1}, \dots, y_{i,\ell^*}$ with $y_j = y_{\ell_i}$ for $j > \ell_i$. Using these parameters, we can interpret $\mathfrak{A}_{i+1}$ in $\mathfrak{A}_i$. To deal with the distance information encoded in Step 1, we might need to increase the quantifier rank by $\log R$. Hence, there is an $\mathrm{FO}[\sigma_i, q^* + (\lambda - i - 1) \cdot R]$-formula φ_i with $k + \big(2(\lambda - i) - 1\big) \cdot \ell^*$ free variables such that $\mathrm{err}_{T_i}\left(h^{\mathfrak{A}_i}_{\varphi_i, \bar{w}_i}\right) \leqslant \varepsilon^* + \varepsilon$. ⌟

These claims prove Items (1)–(5) of Lemma 7.7. The sets X_i can be computed by an fpt algorithm. Furthermore, trying all possible choices for the sets Y_i adds a factor that only depends on the parameters. In the proof of Theorem 2.20, Grohe, Kreutzer, and Siebertz [52] show that Splitter's winning strategy can be computed by an fpt algorithm. With this, we obtain the vertices $w_{i,1}, \dots, w_{i,\ell^*}$. This is all we need to compute the structures and training sequences in our algorithm. To identify suitable parameters $\bar{w}_i$ and formulas φ_i, we can check all tuples found as candidates for parameters and all possible formulas with a suitable bound on the quantifier rank. The number of formulas to check only depends on k, ℓ, σ, and q. Furthermore, by Theorem 2.20, there is an fpt algorithm for model checking first-order formulas on nowhere dense classes.

Thus, there is an fpt algorithm with parameter $k + \ell^* + q^* + 1/\varepsilon + |\sigma|$ that computes $\mathfrak{A}_1, \dots, \mathfrak{A}_\lambda$, $T_1, \dots, T_\lambda$, and suitable $\bar{w}_{\lambda-1}, \dots, \bar{w}_0$ and $\varphi_{\lambda-1}, \dots, \varphi_0$ on input $k, \ell^*, q^*, \varepsilon, \mathfrak{A}, T$ for $\mathfrak{A} \in \mathcal{C}$. This completes the proof of Lemma 7.7. □

7.3 TRACTABILITY OF PAC LEARNING

In this section, we extend the tractability result from Theorem 7.5 to PAC learning. For that, we use the concept of the VC dimension of a hypothesis class that we already mentioned in Chapter 3. Let us briefly introduce this concept and discuss the related results that we use in this section.

Let X be an instance space and $\mathcal{H}$ be a hypothesis class of hypotheses $h\colon X \to \{0,1\}$ (cf. Chapter 3). For a subset $S \subseteq X$, we say that S is *shattered* by $\mathcal{H}$ if for every $S' \subseteq S$, there is a hypothesis $h \in \mathcal{H}$ such that for all $x \in S$ we have $h(x) = 1$ if and only if $x \in S'$. Intuitively, this means that $\mathcal{H}$, restricted to elements in S, contains every possible Boolean classification of the elements from S. The *Vapnik-Chervonenkis dimension* or *VC dimension* of the hypothesis class $\mathcal{H}$, denoted by $\mathrm{VC}(\mathcal{H})$, is the size of the largest set $S \subseteq X$ that is shattered by $\mathcal{H}$, or ∞ if arbitrarily large finite sets can be shattered by $\mathcal{H}$. For an $\mathrm{FO}[\sigma]$-formula φ with $k + \ell$ free variables and a σ-structure $\mathfrak{A}$, we let $\mathrm{VC}(\varphi, k, \ell, \mathfrak{A}) = \mathrm{VC}\left(\left\{h^{\mathfrak{A}}_{\varphi,\bar{w}} \mid \bar{w} \in \big(U(\mathfrak{A})\big)^{\ell}\right\}\right)$. We extend this to classes of structures $\mathcal{C}$ by $\mathrm{VC}(\varphi, k, \ell, \mathcal{C}) = \sup_{\mathfrak{A} \in \mathcal{C}} \mathrm{VC}(\varphi, k, \ell, \mathfrak{A})$ and to sets Φ of $\mathrm{FO}[\sigma]$-formulas by $\mathrm{VC}(\Phi, k, \ell, \mathcal{C}) = \sup_{\varphi \in \Phi} \mathrm{VC}(\varphi, k, \ell, \mathcal{C})$.

VC dimension

Grohe and Turán [57] initiated the analysis of the VC dimension $\mathrm{VC}(\varphi, k, \ell, \mathcal{C})$ for first-order and monadic second-order formulas on several classes of structures, including graphs of bounded genus and graphs of bounded degree. Based on this research, Adler and Adler [2] proved the following result for nowhere dense graph classes.

Theorem 7.9 ([2])**.** *Let $\mathcal{C}$ be a subgraph-closed class of graphs. Then $\mathrm{VC}(\varphi, k, \ell, \mathcal{C})$ is finite for all $k, \ell \in \mathbb{N}$ and for every FO-formula φ with $k + \ell$ free variables if and only if $\mathcal{C}$ is nowhere dense.*

To the best of our knowledge, it is still an open problem whether this theorem can be generalised from graphs to arbitrary relational structures with finite signatures.

The following result links the VC dimension of a hypothesis class with agnostic PAC learnability.

Theorem 7.10 (The Fundamental Theorem of Statistical Learning [85])**.** *A hypothesis class $\mathcal{H}$ is (agnostically) PAC-learnable if and only if it has finite VC dimension.*

Furthermore, if $\mathcal{H}$ has finite VC dimension, then $\mathcal{H}$ can be learned by any algorithm that follows the Empirical Risk Minimisation rule with sample complexity (i. e. the number of examples needed to fulfil the bounds required for agnostic PAC learnability) $m_{\mathcal{H}}(\varepsilon, \delta) \in \mathcal{O}\left(\frac{\mathrm{VC}(\mathcal{H}) + \log(1/\delta)}{\varepsilon^2}\right)$.

We can now combine Theorems 7.5, 7.9, and 7.10 to prove that the following PAC-learning problem is fixed-parameter tractable on nowhere dense graph classes.

p-FO-Learn-PAC(L, Q)

Input: σ-structure $\mathfrak{A}$, rational numbers $\varepsilon, \delta > 0$, probability distribution $\mathcal{D}$ on $(U(\mathfrak{A}))^k \times \{0,1\}$, $k, \ell^*, q^* \in \mathbb{N}$

Parameter: $k + \ell^* + q^* + \frac{1}{\varepsilon} + \frac{1}{\delta} + |\sigma|$

Problem: Return a formula $\varphi(\bar{x}, \bar{y}) \in \mathsf{FO}[\sigma, q]$ for some $q \leqslant Q(k, \ell^*, q^*)$ with $|\bar{x}| = k$ and $|\bar{y}| = \ell \leqslant L(k, \ell^*, q^*)$, and return a tuple $\bar{w} \in (U(\mathfrak{A}))^\ell$ such that, with probability of at least $1 - \delta$ over the choice of examples drawn i.i.d. from $\mathcal{D}$, it holds that

$$\mathrm{err}_{\mathcal{D}}\left(h^{\mathfrak{A}}_{\varphi, \bar{w}}\right) \leqslant \varepsilon^* + \varepsilon,$$

where

$$\varepsilon^* := \min\left\{\mathrm{err}_{\mathcal{D}}\left(h^{\mathfrak{A}}_{\varphi^*, \bar{w}^*}\right) \,\middle|\, h^{\mathfrak{A}}_{\varphi^*, \bar{w}^*} \in \mathcal{H}_{q^*, k, \ell^*}(\mathfrak{A})\right\}.$$

Theorem 7.11. *For every effectively nowhere dense graph class $\mathcal{C}$, there are functions $L, Q \colon \mathbb{N}^3 \to \mathbb{N}$ such that* p-FO-Learn-PAC(L, Q) *is fixed-parameter tractable on $\mathcal{C}$.*

Proof. Let $\mathcal{C}$ be an effectively nowhere dense graph class. Let L, Q be as in the proof of Theorem 7.5 and let Φ be the set of all $\mathrm{FO}[\sigma, q]$-formulas $\varphi(\bar{x}, \bar{y})$ with $|\bar{x}| = k$ and $|\bar{y}| = \ell = L(k, \ell^*, q^*)$. By Theorem 7.9, $\mathrm{VC}(\varphi, k, \ell, \mathcal{C})$ is finite for every $\varphi \in \Phi$. Since, up to equivalence, Φ only contains finitely many formulas, $\mathrm{VC}(\Phi, k, \ell, \mathcal{C})$ is finite as well. Hence, by Theorem 7.10, there is a constant $d \in \mathbb{N}$ such that for every $\mathfrak{A} \in \mathcal{C}$, the hypothesis class $\mathcal{H}_{\Phi,k,\ell}(\mathfrak{A})$ can be learned by any algorithm that follows the ERM rule with sample complexity $m_{\mathcal{H}_{\Phi,k,\ell}(\mathfrak{A})}(\varepsilon, \delta) \in \mathcal{O}\left(\frac{d+\log(1/\delta)}{\varepsilon^2}\right)$. Thus, we can use the algorithm from Theorem 7.5 to solve p-FO-Learn-PAC(L, Q). □

Because of the bounded VC dimension, the number of examples needed in Theorem 7.11 is independent of (the size of) the input graph.

We end this section with an extension of Theorem 7.11 to arbitrary relational structures. There, we cannot rely on Theorem 7.9 any more. Instead, to bound the number of needed examples, we proceed as in Chapter 4 by bounding the size of the hypothesis class. Although this number depends on the size of the input structure, it is still sufficient to yield a tractable learning result.

Theorem 7.12. *For every effectively nowhere dense class $\mathcal{C}$ of structures, there are functions $L, Q\colon \mathbb{N}^3 \to \mathbb{N}$ such that* p-FO-Learn-PAC(L, Q) *is fixed-parameter tractable on $\mathcal{C}$.*

Proof. Let L, Q be as in the proof of Theorem 7.5 and let Φ be the set of all $\mathrm{FO}[\sigma, q]$-formulas $\varphi(\bar{x}, \bar{y})$ with $|\bar{x}| = k$ and $|\bar{y}| = \ell = L(k, \ell^*, q^*)$. The number of formulas in Φ, up to equivalence, only depends on k, ℓ^*, q^*, and σ. Furthermore, for every input structure $\mathfrak{A}$, we can bound the number of possible parameter choices by $|\mathfrak{A}|^{\ell}$. Hence, the size of the hypothesis class $\mathcal{H}_{\Phi,k,\ell}(\mathfrak{A})$ can be bounded by $g(k + \ell^* + q^* \, |\sigma|) \cdot |\mathfrak{A}|^{\ell}$ for some function g. Analogously to the proof of Theorem 4.11, using Lemma 3.9, we can show that it suffices to query

$$m_{\mathcal{H}_{\Phi,k,\ell}(\mathfrak{A})}(\varepsilon, \delta) := \left\lceil \frac{2\log\left(2g(k + \ell^* + q^* + |\sigma|) \cdot |\mathfrak{A}|^{\ell} / \delta\right)}{\varepsilon^2} \right\rceil$$

many examples and then use an algorithm for p-FO-Learn-ERM(L, Q) to solve p-FO-Learn-PAC(L, Q).

Since $m_{\mathcal{H}_{\Phi,k,\ell}(\mathfrak{A})}(\varepsilon, \delta) \in \mathcal{O}\big(f(k + \ell^* + q^* + \frac{1}{\varepsilon} + \frac{1}{\delta} + |\sigma|) \cdot \log(|\mathfrak{A}|)\big)$ for some function f, and the running time of the algorithm from Theorem 7.5 is polynomial in the number of examples, this yields an fpt algorithm. □

8 CONCLUSION

In this thesis, we have studied the descriptive complexity of machine-learning problems, where the task is to learn a hypothesis from a class that can be defined in a certain logic.

We have proved sublinear-time non-learnability results for first-order logic and learnability results for two extensions of first-order logic with certain data-aggregation methods. We have also studied the parameterised complexity of learning first-order logic and seen both tractability and intractability results.

For the extension FOCN of first-order logic with counting quantifiers, we have proved that all three introduced types of problems—the consistent learning, the Empirical Risk Minimisation (ERM), and the PAC-learning problem—can be solved in sublinear time on classes of structures of small degree. For that, we utilised Hanf locality as well as a recently published isomorphism test.

To obtain learnability results for hypotheses that can combine relational and numerical information, we have introduced the logic FOWA, an extension of first-order logic with weight aggregation. We have shown that concepts definable in the fragment FOWA_1 can be learned in sublinear time over weighted structures of small degree. For the proof, we have provided locality results, namely Feferman-Vaught decompositions and a Gaifman normal form for the fragment FOW_1 as well as a localisation theorem for FOWA_1.

Towards a better understanding of the complexity of machine-learning problems on richer classes of structures, we have studied the parameterised complexity of learning first-order logic. On arbitrary relational structures and under common complexity-theoretic assumptions, the ERM problem is intractable. For nowhere dense classes of structures, however, we have proved that the ERM problem and the PAC-learning problem are in fact fixed-parameter tractable.

FUTURE WORK

In the proofs of the aforementioned fixed-parameter tractability results, we return hypotheses that may use more parameters and a larger quantifier rank than the optimal hypotheses that we compare them with. It remains open whether the results also hold for stricter requirements on the returned hypotheses.

Similarly to sublinear-time learnability, it would be desirable to generalise the fixed-parameter tractability results to stronger logics. Due to its fixed-parameter tractable model-checking problem, the counting logic FOC_1 from [56] would be a good candidate. For first-order logic with weight aggregation, a first step towards a tractable learning problem would be a model-checking result for suitable classes of structures.

In [53] and [47], the authors provide sublinear-time learning results for concepts definable in monadic second-order logic (MSO). It would be interesting to study also the parameterised complexity of this problem. Due to Courcelle's, Makowsky's, and Rotics's results [26, 27], it seems plausible that learning MSO-definable concepts is fixed-parameter tractable on classes of bounded treewidth, classes of bounded clique-width, and classes of bounded rank-width. This also raises the question whether there is a reduction from the MSO model-checking problem to MSO learning, similar to the reduction we have given for FO in Chapter 7.

Since, by the results presented in this thesis, FO model checking and FO learning are both intractable on arbitrary relational structures and fixed-parameter tractable on nowhere dense classes, the question arises whether the problems are actually equivalent in terms of their computational complexity on *all* classes of structures (that satisfy reasonable closure properties). We have already settled one of the two directions of this relationship, since the reduction from FO model checking to FO learning presented in Section 7.1 applies to all classes that are closed under union and colour expansions. For the other direction, we would need a reduction from FO learning to FO model checking. Our tractability result in Section 7.2 uses the model-checking problem, but also heavily depends on the characterisation of nowhere dense classes via the Splitter game. Thus, there is no straightforward generalisation to other classes of structures. To gain more insights on the relationship between model-checking and learning, one could study the parameterised complexity of the learning problem on dense graph classes with a tractable FO model-checking problem, such as classes of bounded clique-width or bounded rank-width [27], certain classes of interval graphs [43], or classes of structures obtained via FO interpretations or FO transductions [42]. While a unified approach using FO model checking to solve the learning problem would be desirable, it might also be the case that, similar to the relationship between model checking and counting [50], learning is computationally harder than model checking.

Finally, it would be interesting to study non-Boolean classification problems, where classifiers are described by terms instead of formulas. As for the generalisations of our fixed-parameter tractability results, the logic FOC_1 as well as suitable fragments of FOWA would be good starting points.

BIBLIOGRAPHY

[1] Azza Abouzied, Dana Angluin, Christos H. Papadimitriou, Joseph M. Hellerstein and Avi Silberschatz. 'Learning and verifying quantified boolean queries by example'. In: *Proceedings of the 32nd ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems, PODS 2013, New York, NY, USA - June 22 - 27, 2013*. Ed. by Richard Hull and Wenfei Fan. ACM, 2013, pp. 49–60. DOI: 10.1145/2463664.2465220.

[2] Hans Adler and Isolde Adler. 'Interpreting nowhere dense graph classes as a classical notion of model theory'. In: *Eur. J. Comb.* 36 (2014), pp. 322–330. DOI: 10.1016/j.ejc.2013.06.048.

[3] Isolde Adler and Polly Fahey. 'Faster Property Testers in a Variation of the Bounded Degree Model'. In: *40th IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science, FSTTCS 2020, December 14-18, 2020, BITS Pilani, K K Birla Goa Campus, Goa, India (Virtual Conference)*. Ed. by Nitin Saxena and Sunil Simon. Vol. 182. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020, 7:1–7:15. DOI: 10.4230/LIPIcs.FSTTCS.2020.7.

[4] Isolde Adler and Frederik Harwath. 'Property Testing for Bounded Degree Databases'. In: *35th Symposium on Theoretical Aspects of Computer Science, STACS 2018, February 28 to March 3, 2018, Caen, France*. Ed. by Rolf Niedermeier and Brigitte Vallée. Vol. 96. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2018, 6:1–6:14. DOI: 10.4230/LIPIcs.STACS.2018.6.

[5] Howard Aizenstein, Tibor Hegedüs, Lisa Hellerstein and Leonard Pitt. 'Complexity Theoretic Hardness Results for Query Learning'. In: *Comput. Complex.* 7.1 (1998), pp. 19–53. DOI: 10.1007/PL00001593.

[6] Bogdan Alexe, Balder ten Cate, Phokion G. Kolaitis and Wang Chiew Tan. 'Characterizing schema mappings via data examples'. In: *ACM Trans. Database Syst.* 36.4 (2011), 23:1–23:48. DOI: 10.1145/2043652.2043656.

[7] Dana Angluin. 'Queries and Concept Learning'. In: *Machine Learning* 2.4 (1987), pp. 319–342. DOI: 10.1007/BF00116828.

[8] Albert Atserias, Martin Grohe and Dániel Marx. 'Size Bounds and Query Plans for Relational Joins'. In: *SIAM J. Comput.* 42.4 (2013), pp. 1737–1767. DOI: 10.1137/110859440.

[9] Pablo Barceló, Alexander Baumgartner, Victor Dalmau and Benny Kimelfeld. 'Regularizing conjunctive features for classification'. In: *J. Comput. Syst. Sci.* 119 (2021), pp. 97–124. DOI: 10.1016/j.jcss.2021.01.003.

[10] Pablo Barceló and Miguel Romero. 'The Complexity of Reverse Engineering Problems for Conjunctive Queries'. In: *20th International Conference on Database Theory, ICDT 2017, March 21-24, 2017, Venice, Italy*. Ed. by Michael Benedikt and Giorgio Orsi. Vol. 68. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2017, 7:1–7:17. DOI: 10.4230/LIPIcs.ICDT.2017.7.

[11] Steffen van Bergerem. 'Learning Concepts Definable in First-Order Logic with Counting'. In: *34th Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2019, Vancouver, BC, Canada, June 24-27, 2019*. IEEE, 2019, pp. 1–13. DOI: 10.1109/LICS.2019.8785811.

[12] Steffen van Bergerem, Martin Grohe and Martin Ritzert. 'On the Parameterized Complexity of Learning First-Order Logic'. In: *PODS 2022: International Conference on Management of Data, Philadelphia, PA, USA, June 12-17, 2022*. ACM, 2022, pp. 337–346. DOI: 10.1145/3517804.3524151.

[13] Steffen van Bergerem and Nicole Schweikardt. 'Learning Concepts Described By Weight Aggregation Logic'. In: *29th EACSL Annual Conference on Computer Science Logic, CSL 2021, Ljubljana, Slovenia (Virtual Conference), January 25-28, 2021*. Vol. 183. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2021, 10:1–10:18. DOI: 10.4230/LIPIcs.CSL.2021.10.

[14] Anselm Blumer, Andrzej Ehrenfeucht, David Haussler and Manfred K. Warmuth. 'Learnability and the Vapnik-Chervonenkis dimension'. In: *J. ACM* 36.4 (1989), pp. 929–965. DOI: 10.1145/76359.76371.

[15] Angela Bonifati, Radu Ciucanu and Aurélien Lemay. 'Learning Path Queries on Graph Databases'. In: *Proceedings of the 18th International Conference on Extending Database Technology, EDBT 2015, Brussels, Belgium, March 23-27, 2015*. Ed. by Gustavo Alonso, Floris Geerts, Lucian Popa, Pablo Barceló, Jens Teubner, Martín Ugarte, Jan Van den Bussche and Jan Paredaens. OpenProceedings.org, 2015, pp. 109–120. DOI: 10.5441/002/edbt.2015.11.

[16] Angela Bonifati, Radu Ciucanu and Slawek Staworko. 'Learning Join Queries from User Examples'. In: *ACM Trans. Database Syst.* 40.4 (2016), 24:1–24:38. DOI: 10.1145/2818637.

[17] Angela Bonifati, Ugo Comignani, Emmanuel Coquery and Romuald Thion. 'Interactive Mapping Specification with Exemplar Tuples'. In: *ACM Trans. Database Syst.* 44.3 (2019), 10:1–10:44. DOI: 10.1145/3321485.

[18] Peter J. Cameron. *Combinatorics: Topics, Techniques, Algorithms*. Cambridge University Press, 1994.

[19] Nofar Carmeli, Shai Zeevi, Christoph Berkholz, Benny Kimelfeld and Nicole Schweikardt. 'Answering (Unions of) Conjunctive Queries using Random Access and Random-Order Enumeration'. In: *Proceedings of the 39th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems, PODS 2020, Portland, OR, USA, June 14-19, 2020*. Ed. by Dan Suciu, Yufei Tao and Zhewei Wei. ACM, 2020, pp. 393–409. DOI: 10.1145/3375395.3387662.

[20] Balder ten Cate and Victor Dalmau. 'Conjunctive Queries: Unique Characterizations and Exact Learnability'. In: *24th International Conference on Database Theory, ICDT 2021, March 23-26, 2021, Nicosia, Cyprus*. Ed. by Ke Yi and Zhewei Wei. Vol. 186. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2021, 9:1–9:24. DOI: 10.4230/LIPIcs.ICDT.2021.9.

[21] Balder ten Cate, Víctor Dalmau and Phokion G. Kolaitis. 'Learning schema mappings'. In: *ACM Trans. Database Syst.* 38.4 (2013), 28:1–28:31. DOI: 10.1145/2539032.2539035.

[22] Balder ten Cate, Phokion G. Kolaitis, Kun Qian and Wang-Chiew Tan. 'Active Learning of GAV Schema Mappings'. In: *Proceedings of the 37th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems, Houston, TX, USA, June 10-15, 2018*. Ed. by Jan Van den Bussche and Marcelo Arenas. ACM, 2018, pp. 355–368. DOI: 10.1145/3196959.3196974.

[23] Adrien Champion, Tomoya Chiba, Naoki Kobayashi and Ryosuke Sato. 'ICE-Based Refinement Type Discovery for Higher-Order Functional Programs'. In: *J. Autom. Reason.* 64.7 (2020), pp. 1393–1418. DOI: 10.1007/s10817-020-09571-y.

[24] Ashok K. Chandra and David Harel. 'Structure and Complexity of Relational Queries'. In: *XP1 Workshop on Relational Database Theory, 30 June - 2 July 1980, SUNY at Stony Brook, NY, USA*. Ed. by David Maier. 1980.

[25] William W. Cohen and C. David Page Jr. 'Polynomial Learnability and Inductive Logic Programming: Methods and Results'. In: *New Gener. Comput.* 13.3&4 (1995), pp. 369–409. DOI: 10.1007/BF03037231.

[26] Bruno Courcelle. 'The Monadic Second-Order Logic of Graphs. I. Recognizable Sets of Finite Graphs'. In: *Inf. Comput.* 85.1 (1990), pp. 12–75. DOI: 10.1016/0890-5401(90)90043-H.

[27] Bruno Courcelle, Johann A. Makowsky and Udi Rotics. 'Linear Time Solvable Optimization Problems on Graphs of Bounded Clique-Width'. In: *Theory Comput. Syst.* 33.2 (2000), pp. 125–150. DOI: 10.1007/s002249910009.

[28] Andrew Cropper, Sebastijan Dumancic, Richard Evans and Stephen H. Muggleton. ‘Inductive logic programming at 30’. In: *Mach. Learn.* 111.1 (2022), pp. 147–172. DOI: 10.1007/s10994-021-06089-1.

[29] Rodney G. Downey and Michael R. Fellows. ‘Fixed-Parameter Tractability and Completeness II: On Completeness for W[1]’. In: *Theor. Comput. Sci.* 141.1&2 (1995), pp. 109–131. DOI: 10.1016/0304-3975(94)00097-3.

[30] Rodney G. Downey, Michael R. Fellows and Udayan Taylor. ‘The Parameterized Complexity of Relational Database Queries and an Improved Characterization of W[1]’. In: *First Conference of the Centre for Discrete Mathematics and Theoretical Computer Science, DMTCS 1996, Auckland, New Zealand, December, 9-13, 1996*. Ed. by Douglas S. Bridges, Cristian S. Calude, Jeremy Gibbons, Steve Reeves and Ian H. Witten. Springer-Verlag, Singapore, 1996, pp. 194–213.

[31] Arnaud Durand, Nicole Schweikardt and Luc Segoufin. ‘Enumerating answers to first-order queries over databases of low degree’. In: *Proceedings of the 33rd ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems, PODS 2014, Snowbird, UT, USA, June 22-27, 2014*. Ed. by Richard Hull and Martin Grohe. ACM, 2014, pp. 121–131. DOI: 10.1145/2594538.2594539.

[32] Zdenek Dvorák, Daniel Král and Robin Thomas. ‘Deciding First-Order Properties for Sparse Graphs’. In: *51th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2010, October 23-26, 2010, Las Vegas, Nevada, USA*. IEEE Computer Society, 2010, pp. 133–142. DOI: 10.1109/FOCS.2010.20.

[33] Guy Even, Moti Medina and Dana Ron. ‘Deterministic Stateless Centralized Local Algorithms for Bounded Degree Graphs’. In: *Algorithms - ESA 2014 - 22th Annual European Symposium, Wroclaw, Poland, September 8-10, 2014. Proceedings*. Ed. by Andreas S. Schulz and Dorothea Wagner. Vol. 8737. Lecture Notes in Computer Science. Springer, 2014, pp. 394–405. DOI: 10.1007/978-3-662-44777-2_33.

[34] P. Ezudheen, Daniel Neider, Deepak D'Souza, Pranav Garg and P. Madhusudan. ‘Horn-ICE learning for synthesizing invariants and contracts’. In: *Proc. ACM Program. Lang.* 2.OOPSLA (2018), 131:1–131:25. DOI: 10.1145/3276501.

[35] Ronald Fagin. ‘Generalized first-order spectra and polynomial-time recognizable sets’. In: *Complexity of computation (Proc. SIAM-AMS Sympos., New York, 1973)*. SIAM-AMS Proc., Vol. VII. Amer. Math. Soc., Providence, R.I., 1974, pp. 43–73.

[36] Ronald Fagin, Larry J. Stockmeyer and Moshe Y. Vardi. 'On Monadic NP vs. Monadic co-NP'. In: *Inf. Comput.* 120.1 (1995), pp. 78–92. DOI: 10.1006/inco.1995.1100.

[37] Solomon Feferman and Robert L. Vaught. 'The first-order properties of products of algebraic systems'. In: *Fundamenta Mathematicae* 47 (1959), pp. 57–103.

[38] Jörg Flum, Markus Frick and Martin Grohe. 'Query evaluation via tree-decompositions'. In: *J. ACM* 49.6 (2002), pp. 716–752. DOI: 10.1145/602220.602222.

[39] Jörg Flum and Martin Grohe. *Parameterized Complexity Theory*. Texts in Theoretical Computer Science. An EATCS Series. Springer, 2006. ISBN: 978-3-540-29952-3. DOI: 10.1007/3-540-29953-X.

[40] Frank Fuhlbrück. 'Fixed-parameter tractability of the graph isomorphism and canonization problems'. Diploma thesis. Humboldt-Universität zu Berlin, 2013.

[41] Haim Gaifman. 'On Local and Non-Local Properties'. In: *Proceedings of the Herbrand Symposium*. Ed. by Jacques Stern. Vol. 107. Studies in Logic and the Foundations of Mathematics. North-Holland, 1982, pp. 105–135. DOI: 10.1016/S0049-237X(08)71879-2.

[42] Jakub Gajarský, Petr Hlinený, Jan Obdrzálek, Daniel Lokshtanov and M. S. Ramanujan. 'A New Perspective on FO Model Checking of Dense Graph Classes'. In: *ACM Trans. Comput. Log.* 21.4 (2020), 28:1–28:23. DOI: 10.1145/3383206.

[43] Robert Ganian, Petr Hlinený, Daniel Král, Jan Obdrzálek, Jarett Schwartz and Jakub Teska. 'FO Model Checking of Interval Graphs'. In: *Log. Methods Comput. Sci.* 11.4 (2015). DOI: 10.2168/LMCS-11(4:11)2015.

[44] Pranav Garg, Christof Löding, P. Madhusudan and Daniel Neider. 'ICE: A Robust Framework for Learning Invariants'. In: *Computer Aided Verification - 26th International Conference, CAV 2014, Held as Part of the Vienna Summer of Logic, VSL 2014, Vienna, Austria, July 18-22, 2014. Proceedings*. Ed. by Armin Biere and Roderick Bloem. Vol. 8559. Lecture Notes in Computer Science. Springer, 2014, pp. 69–87. DOI: 10.1007/978-3-319-08867-9_5.

[45] Oded Goldreich and Dana Ron. 'Property Testing in Bounded Degree Graphs'. In: *Algorithmica* 32.2 (2002), pp. 302–343. DOI: 10.1007/s00453-001-0078-7.

[46] Georg Gottlob and Pierre Senellart. 'Schema mapping discovery from data instances'. In: *J. ACM* 57.2 (2010), 6:1–6:37. DOI: 10.1145/1667053.1667055.

[47] Emilie Grienenberger and Martin Ritzert. 'Learning Definable Hypotheses on Trees'. In: *22nd International Conference on Database Theory, ICDT 2019, March 26-28, 2019, Lisbon, Portugal*. 2019, 24:1–24:18. DOI: 10.4230/LIPIcs.ICDT.2019.24.

[48] Martin Grohe. 'Generalized Model-Checking Problems for First-Order Logic'. In: *STACS 2001, 18th Annual Symposium on Theoretical Aspects of Computer Science, Dresden, Germany, February 15-17, 2001, Proceedings*. Ed. by Afonso Ferreira and Horst Reichel. Vol. 2010. Lecture Notes in Computer Science. Springer, 2001, pp. 12–26. DOI: 10.1007/3-540-44693-1_2.

[49] Martin Grohe. 'Logic, graphs, and algorithms'. In: *Logic and Automata: History and Perspectives [in Honor of Wolfgang Thomas]*. Ed. by Jörg Flum, Erich Grädel and Thomas Wilke. Vol. 2. Texts in Logic and Games. Amsterdam University Press, 2008, pp. 357–422.

[50] Martin Grohe. *Descriptive Complexity, Canonisation, and Definable Graph Structure Theory*. Vol. 47. Lecture Notes in Logic. Cambridge University Press, 2017. ISBN: 9781139028868. DOI: 10.1017/9781139028868.

[51] Martin Grohe. 'word2vec, node2vec, graph2vec, X2vec: Towards a Theory of Vector Embeddings of Structured Data'. In: *Proceedings of the 39th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems, PODS 2020, Portland, OR, USA, June 14-19, 2020*. Ed. by Dan Suciu, Yufei Tao and Zhewei Wei. ACM, 2020, pp. 1–16. DOI: 10.1145/3375395.3387641.

[52] Martin Grohe, Stephan Kreutzer and Sebastian Siebertz. 'Deciding First-Order Properties of Nowhere Dense Graphs'. In: *J. ACM* 64.3 (2017), 17:1–17:32. DOI: 10.1145/3051095.

[53] Martin Grohe, Christof Löding and Martin Ritzert. 'Learning MSO-definable hypotheses on strings'. In: *International Conference on Algorithmic Learning Theory, ALT 2017, 15-17 October 2017, Kyoto University, Kyoto, Japan*. 2017, pp. 434–451.

[54] Martin Grohe, Daniel Neuen and Pascal Schweitzer. 'A Faster Isomorphism Test for Graphs of Small Degree'. In: *59th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2018, Paris, France, October 7-9, 2018*. Ed. by Mikkel Thorup. IEEE Computer Society, 2018, pp. 89–100. DOI: 10.1109/FOCS.2018.00018.

[55] Martin Grohe and Martin Ritzert. 'Learning first-order definable concepts over structures of small degree'. In: *32nd Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2017, Reykjavik, Iceland, June 20-23, 2017*. 2017, pp. 1–12. DOI: 10.1109/LICS.2017.8005080.

[56] Martin Grohe and Nicole Schweikardt. 'First-Order Query Evaluation with Cardinality Conditions'. In: *Proceedings of the 37th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems, Houston, TX, USA, June 10-15, 2018*. 2018, pp. 253–266. DOI: 10.1145/3196959.3196970.

[57] Martin Grohe and György Turán. 'Learnability and Definability in Trees and Similar Structures'. In: *Theory Comput. Syst.* 37.1 (2004), pp. 193–220. DOI: 10.1007/s00224-003-1112-8.

[58] Aditya Grover and Jure Leskovec. 'node2vec: Scalable Feature Learning for Networks'. In: *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016*. Ed. by Balaji Krishnapuram, Mohak Shah, Alexander J. Smola, Charu C. Aggarwal, Dou Shen and Rajeev Rastogi. ACM, 2016, pp. 855–864. DOI: 10.1145/2939672.2939754.

[59] Yuri Gurevich. 'Toward logic tailored for computational complexity'. In: *Computation and Proof Theory*. Ed. by Egon Börger, Walter Oberschelp, Michael M. Richter, Brigitta Schinzel and Wolfgang Thomas. Springer Berlin Heidelberg, 1984, pp. 175–216.

[60] William Hanf. 'Model-theoretic methods in the study of elementary logic'. In: *The Theory of Models. Proceedings of the 1963 International Symposium at Berkeley*. Ed. by J.W. Addison, Leon Henkin and Alfred Tarski. Studies in logic and the foundations of mathematics. Amsterdam: North-Holland Publishing Company, 1965, pp. 132–145.

[61] David Haussler. 'Learning Conjunctive Concepts in Structural Domains'. In: *Mach. Learn.* 4 (1989), pp. 7–40. DOI: 10.1007/BF00114802.

[62] David Haussler. 'Decision Theoretic Generalizations of the PAC Model for Neural Net and Other Learning Applications'. In: *Inf. Comput.* 100.1 (1992), pp. 78–150. DOI: 10.1016/0890-5401(92)90010-D.

[63] Kouichi Hirata. 'On the Hardness of Learning Acyclic Conjunctive Queries'. In: *Algorithmic Learning Theory, 11th International Conference, ALT 2000, Sydney, Australia, December 11-13, 2000, Proceedings*. Ed. by Hiroki Arimura, Sanjay Jain and Arun Sharma. Vol. 1968. Lecture Notes in Computer Science. Springer, 2000, pp. 238–251. DOI: 10.1007/3-540-40992-0_18.

[64] Neil Immerman. 'Relational Queries Computable in Polynomial Time'. In: *Inf. Control.* 68.1-3 (1986), pp. 86–104. DOI: 10.1016/S0019-9958(86)80029-8.

[65] Neil Immerman. *Descriptive complexity*. Graduate texts in computer science. Springer, 1999. ISBN: 978-1-4612-6809-3. DOI: 10.1007/978-1-4612-0539-5.

[66] Jörg-Uwe Kietz and Saso Dzeroski. 'Inductive Logic Programming and Learnability'. In: *SIGART Bull.* 5.1 (1994), pp. 22–32. DOI: 10.1145/181668.181674.

[67] Benny Kimelfeld and Christopher Ré. 'A Relational Framework for Classifier Engineering'. In: *ACM Trans. Database Syst.* 43.3 (2018), 11:1–11:36. DOI: 10.1145/3268931.

[68] Stephan Kreutzer. 'Algorithmic meta-theorems'. In: *Finite and Algorithmic Model Theory*. Ed. by Javier Esparza, Christian Michaux and Charles Steinhorn. Vol. 379. London Mathematical Society Lecture Note Series. Cambridge University Press, 2011, pp. 177–270.

[69] Dietrich Kuske and Nicole Schweikardt. 'First-order logic with counting'. In: *32nd Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2017, Reykjavik, Iceland, June 20-23, 2017*. IEEE Computer Society, 2017, pp. 1–12. DOI: 10.1109/LICS.2017.8005133.

[70] Dietrich Kuske and Nicole Schweikardt. 'Gaifman Normal Forms for Counting Extensions of First-Order Logic'. In: *45th International Colloquium on Automata, Languages, and Programming, ICALP 2018, July 9-13, 2018, Prague, Czech Republic*. Ed. by Ioannis Chatzigiannakis, Christos Kaklamanis, Dániel Marx and Donald Sannella. Vol. 107. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2018, 133:1–133:14. DOI: 10.4230/LIPIcs.ICALP.2018.133.

[71] Reut Levi, Dana Ron and Ronitt Rubinfeld. 'Local Algorithms for Sparse Spanning Graphs'. In: *Algorithmica* 82.4 (2020), pp. 747–786. DOI: 10.1007/s00453-019-00612-6.

[72] Reut Levi, Ronitt Rubinfeld and Anak Yodpinyanee. 'Local Computation Algorithms for Graphs of Non-constant Degrees'. In: *Algorithmica* 77.4 (2017), pp. 971–994. DOI: 10.1007/s00453-016-0126-y.

[73] Christof Löding, P. Madhusudan and Daniel Neider. 'Abstract Learning Frameworks for Synthesis'. In: *Tools and Algorithms for the Construction and Analysis of Systems - 22nd International Conference, TACAS 2016, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2016, Eindhoven, The Netherlands, April 2-8, 2016, Proceedings*. Ed. by Marsha Chechik and Jean-François Raskin. Vol. 9636. Lecture Notes in Computer Science. Springer, 2016, pp. 167–185. DOI: 10.1007/978-3-662-49674-9_10.

[74] Johann A. Makowsky. 'Algorithmic uses of the Feferman-Vaught Theorem'. In: *Ann. Pure Appl. Log.* 126.1-3 (2004), pp. 159–213. DOI: 10.1016/j.apal.2003.11.002.

[75] Christopher Morris, Martin Ritzert, Matthias Fey, William L. Hamilton, Jan Eric Lenssen, Gaurav Rattan and Martin Grohe. 'Weisfeiler and Leman Go Neural: Higher-Order Graph Neural Networks'. In: *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*. AAAI Press, 2019, pp. 4602–4609. DOI: 10.1609/aaai.v33i01.33014602.

[76] Stephen Muggleton. 'Inductive Logic Programming'. In: *New Gener. Comput.* 8.4 (1991), pp. 295–318. DOI: 10.1007/BF03037089.

[77] Stephen Muggleton and Luc De Raedt. 'Inductive Logic Programming: Theory and Methods'. In: *J. Log. Program.* 19/20 (1994), pp. 629–679. DOI: 10.1016/0743-1066(94)90035-3.

[78] Jaroslav Nešetřil. 'Structural Properties of Sparse Graphs'. In: *Electron. Notes Discret. Math.* 31 (2008), pp. 247–251. DOI: 10.1016/j.endm.2008.06.050.

[79] Jaroslav Nešetřil and Patrice Ossona de Mendez. 'First order properties on nowhere dense structures'. In: *J. Symb. Log.* 75.3 (2010), pp. 868–887. DOI: 10.2178/jsl/1278682204.

[80] Jaroslav Nešetřil and Patrice Ossona de Mendez. 'On nowhere dense graphs'. In: *Eur. J. Comb.* 32.4 (2011), pp. 600–617. DOI: 10.1016/j.ejc.2011.01.006.

[81] Shimei Pan and Tao Ding. 'Social Media-based User Embedding: A Literature Review'. In: *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10-16, 2019*. Ed. by Sarit Kraus. ijcai.org, 2019, pp. 6318–6324. DOI: 10.24963/ijcai.2019/881.

[82] Ronitt Rubinfeld, Gil Tamir, Shai Vardi and Ning Xie. 'Fast Local Computation Algorithms'. In: *Innovations in Computer Science - ICS 2011, Tsinghua University, Beijing, China, January 7-9, 2011. Proceedings*. Ed. by Bernard Chazelle. Tsinghua University Press, 2011, pp. 223–238.

[83] Maximilian Schleich, Dan Olteanu, Mahmoud Abo Khamis, Hung Q. Ngo and XuanLong Nguyen. 'Learning Models over Relational Data: A Brief Tutorial'. In: *Scalable Uncertainty Management - 13th International Conference, SUM 2019, Compiègne, France, December 16-18, 2019, Proceedings*. Ed. by Nahla Ben Amor, Benjamin Quost and Martin Theobald. Vol. 11940. Lecture Notes in Computer Science. Springer, 2019, pp. 423–432. DOI: 10.1007/978-3-030-35514-2_32.

[84] Detlef Seese. 'Linear Time Computable Problems and First-Order Descriptions'. In: *Math. Struct. Comput. Sci.* 6.6 (1996), pp. 505–526.

[85] Shai Shalev-Shwartz and Shai Ben-David. *Understanding Machine Learning: From Theory to Algorithms*. New York, NY, USA: Cambridge University Press, 2014. ISBN: 9781107057135. DOI: 10.1017/CBO9781107298019.

[86] Robert H. Sloan, Balázs Szörényi and György Turán. 'Learning Boolean Functions with Queries'. In: *Boolean Models and Methods in Mathematics, Computer Science, and Engineering*. Ed. by Yves Crama and Peter L. Hammer. Cambridge University Press, 2010, pp. 221–256. DOI: 10.1017/cbo9780511780448.010.

[87] Slawek Staworko and Piotr Wieczorek. 'Learning twig and path queries'. In: *15th International Conference on Database Theory, ICDT 2012, Berlin, Germany, March 26-29, 2012*. Ed. by Alin Deutsch. ACM, 2012, pp. 140–154. DOI: 10.1145/2274576.2274592.

[88] Szymon Toruńczyk. 'Aggregate Queries on Sparse Databases'. In: *Proceedings of the 39th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems, PODS 2020, Portland, OR, USA, June 14-19, 2020*. Ed. by Dan Suciu, Yufei Tao and Zhewei Wei. ACM, 2020, pp. 427–443. DOI: 10.1145/3375395.3387660.

[89] Leslie G. Valiant. 'A Theory of the Learnable'. In: *Commun. ACM* 27.11 (1984), pp. 1134–1142. DOI: 10.1145/1968.1972.

[90] Vladimir Vapnik. 'Principles of Risk Minimization for Learning Theory'. In: *Advances in Neural Information Processing Systems 4, [NIPS Conference, Denver, Colorado, USA, December 2-5, 1991]*. 1991, pp. 831–838.

[91] Vladimir Vapnik and Alexey Ya. Chervonenkis. 'On the Uniform Convergence of Relative Frequencies of Events to Their Probabilities'. In: *Theory of Probability and its Applications* 16 (1971), pp. 264–280. DOI: 10.1137/1116025.

[92] Moshe Y. Vardi. 'The Complexity of Relational Query Languages (Extended Abstract)'. In: *Proceedings of the 14th Annual ACM Symposium on Theory of Computing, May 5-7, 1982, San Francisco, California, USA*. Ed. by Harry R. Lewis, Barbara B. Simons, Walter A. Burkhard and Lawrence H. Landweber. ACM, 1982, pp. 137–146. DOI: 10.1145/800070.802186.

[93] Giuseppe Vitali. 'Sui gruppi di punti e sulle funzioni di variabili reali'. Italian. In: *Torino Atti* 43 (1908), pp. 229–246.

[94] He Zhu, Stephen Magill and Suresh Jagannathan. 'A data-driven CHC solver'. In: *Proceedings of the 39th ACM SIGPLAN Conference on Programming Language Design and Implementation, PLDI 2018, Philadelphia, PA, USA, June 18-22, 2018*. Ed. by Jeffrey S. Foster

and Dan Grossman. ACM, 2018, pp. 707–721. DOI: 10.1145/3192366.3192416.