JONNI VIRTEMA

Approaches to Finite Variable Dependence

Expressiveness and Computational Complexity

■

ACADEMIC DISSERTATION
To be presented, with the permission of
the Board of the School of Information Sciences of the
University of Tampere,
for public discussion in the Auditorium Pinni B 1096,
Kanslerinrinne 1, Tampere, on June 5th, 2014, at 12 o'clock.

UNIVERSITY OF TAMPERE

JONNI VIRTEMA

Approaches to Finite Variable Dependence

Expressiveness and Computational Complexity

Acta Universitatis Tamperensis 1943
Tampere University Press
Tampere 2014

ACADEMIC DISSERTATION
University of Tampere
School of Information Sciences
Finland

The originality of this thesis has been checked using the Turnitin OriginalityCheck service in accordance with the quality management system of the University of Tampere.

Cover design by
Mikko Reinikka

Distributor:
kirjamyynti@juvenes.fi
http://granum.uta.fi

Acta Universitatis Tamperensis 1943
ISBN 978-951-44-9471-0 (print)
ISSN-L 1455-1616
ISSN 1455-1616

Acta Electronica Universitatis Tamperensis 1427
ISBN 978-951-44-9472-7 (pdf)
ISSN 1456-954X
http://tampub.uta.fi

Suomen Yliopistopaino Oy – Juvenes Print
Tampere 2014

Abstract

In this thesis we study computational aspects, expressive power and model-theoretic properties of fragments of various logics that are suitable for modeling diverse forms of variable dependence. In particular, we consider fragments of independence-friendly logic of Hintikka and Sandu, dependence logic of Väänänen, and first-order logic enriched with variants of partially-ordered connectives of Sandu and Väänänen closely related to the narrow Henkin quantifiers of Blass and Gurevich. Furthermore, we consider a novel variant of dependence logic called Boolean dependence logic.

The thesis consists of two principal chapters: Chapters 3 and 4. In Chapter 3 the focus is on the computational complexity and expressive power of finite variable fragments of independence-friendly logic and dependence logic. The main result of the chapter is the undecidability of the satisfiability problem for two-variable independence-friendly logic. In addition to studying the complexity of the satisfiability problem, we investigate the complexity of the related model checking problem. Moreover, we study certain model-theoretic properties of these logics: we consider the finite model property and the zero-one law. Chapter 4 is devoted on the study of the relationship between fragments of Boolean dependence logic and first-order logic enriched with partially-ordered connectives. We show that with respect to expressive power certain syntactic fragments of Boolean dependence logic correspond to natural logics enriched with partially-ordered connectives. Furthermore, we show that these fragments of Boolean dependence logic form a strict hierarchy with respect to expressive power. In addition, we establish that the expressive power of Boolean dependence logic and dependence logic coincide.

Acknowledgements

First of all, I wish to thank my supervisor Professor Lauri Hella for his support. Lauri has offered me a free and warm working environment.

I give my thanks to Antti Kuusisto with whom I have had many fruitful and sometimes heated conversations on multitude of matters. I wish to thank Juha Kontinen, who as well as being a valuable collaborator has always been a truly fun to travel with. I wish also to give my thanks to Kerkko Luosto for directing me to references for classical results I needed for this thesis. Moreover, I wish to thank all of my co-authors. In addition, I wish to give my thanks to Pentti Haukkanen, Henry Joutsijoki, Pertti Koivisto, Sirkka Laaksonen, Allen Mann, and Mika Mattila.

I give my special thanks to Professor Satoshi Tojo and Assistant Professor Katsuhiko Sano as a lion's share of my thesis was written when I was visiting Professor Sano at the Tojo Laboratory of the Japan Advanced Institute of Science and Technology. I highly appreciate the warm atmosphere that prevails at the laboratory of Professor Tojo. I wish also to give my thanks to the people of the former Department of Mathematics and Statistics of the University of Tampere.

Finally, I acknowledge the financial support of MALJA Graduate School in Mathematical Logic and Algebra, GSMAA Finnish National Doctoral Programme in Mathematics and its Applications, University of Tampere, Finnish Academy of Science and Letters: Vilho, Yrjö and Kalle Väisälä Foundation, and Academy of Finland.

Tampere, April 2014
Jonni Virtema

Table of Contents

Chapter 1

Introduction

Dependence is an important concept in various scientific disciplines. A multitude of formalisms have been designed to model dependences, for example, in database theory, social choice theory, and quantum mechanics. However, for a long time the research has been scattered and the same ideas have been discovered many times over in different fields of science. One important reason, albeit surely not the only one, for this scatteredness was the lack of unified logical background theory for the concept of dependence. Over the last decade the emergence of dependence logic and the extensive and rigorous research conducted on dependence logic and related formalisms have mended this shortcoming. A milestone was reached in 2013 when a Dagstuhl seminar, "Dependence Logic: Theory and Applications", was held in order to bring together people working on dependence logic and different application fields. The following quotation is from the introduction of the Dagstuhl seminar report [AKVV13].

> "The notions of logical dependence and independence are pervasive, and occur in many areas of science. The development of logical and semantical structures for these notions provides an opportunity for a systematic approach, which can expose surprising connections between different areas (e.g., quantum mechanics, social choice theory, and many more), and may lead to useful general results."

This thesis wishes to contribute to the basic research of logical background theory for the concept of dependence. Simultaneously, we also contribute to the research of fragments of second-order logic and extensions of finite variable first-order logics.

The most direct way to model dependences in first-order like logical systems is via dependences between variables in formulae. When using the game theoretic semantics[1] to evaluate the first-order formula

$$\forall x_0 \exists x_1 \forall x_2 \exists x_3 \varphi,$$

the value chosen by the Verifier for the variable x_1 can depend on the value chosen by the Falsifier for the variable x_0. Moreover, the value chosen by the Verifier for the variable x_3 can depend on the values chosen for each of the variables x_0, x_1 and x_2. In fact, the dependence relation for first-order formulae can be obtained canonically from the dominance relation of the syntactic tree of the formula. Consequently dependence relations for first-order formulae are quite simple. Furthermore, since each first-order formula has an equivalent formula in prenex normal form, each sentence of first-order logic has an equivalent formula for which the dependence relation is in fact a linear order. Therefore first-order logic is not suitable for modeling complex dependences.

In this thesis we consider extensions of first-order logic that are suitable for modeling richer forms of dependence. In particular, we consider independence-friendly logic of Hintikka and Sandu [Hin96, HS89], dependence logic of Väänänen [Vää07], and first-order logic enriched with variants of partially-ordered connectives of Sandu and Väänänen [SV92] closely related to the narrow Henkin quantifiers of Blass and Gurevich [BG86]. Furthermore, we introduce a new variant of dependence logic called Boolean dependence logic. Boolean dependence logic arises naturally from partially-ordered connectives in a similar manner as dependence logic can be seen arising from partially-ordered quantifiers of Henkin [Hen61]. Each of these logics have a unique way of generating a rich structure of variable dependence. Some of the methods are more straightforward and transparent than others. The most direct method is incorporated in dependence logic. In dependence logic the dependence relations between variables are written in terms of novel atomic formulae

$$=(x_1, \ldots, x_n, y)$$

called dependence atoms. The intended meaning of $=(x_1, \ldots, x_n, y)$ is that the value of the variable y depends solely on the values of the variables $x_1, \ldots, x_n$.

[1] Semantics for first-order logic can be defined by the so-called semantic game. The game is played between two players, i.e., the Verifier and the Falsifier. The goal of the Verifier is to show that a given formula is true in a given model and assignment, while the goal of the Falsifier is to show that the formula is false in the given model and assignment. For a detailed exposition to game-theoretic semantics see, e.g. [MSS11].

The method used in independence-friendly logic is a bit more subtle. Independence-friendly logic extends first-order logic in terms of so-called slashed quantifiers. For example, in

$$\forall x_0 \exists x_1 \forall x_2 \exists x_3 / \{x_0\} \varphi,$$

the intended meaning of the quantifier $\exists x_3/\{x_0\}$ is that x_3 is "independent" of x_0 in the sense that the choice for the value of x_3 should not depend on what the value of x_0 is. The fundamental idea behind independence-friendly logic and dependence logic is quite similar. However, the idea behind partially-ordered connectives differs slightly. Partially-ordered connective defined with Boolean variables is an expression of the form

$$\begin{pmatrix} \forall x_{11} & \dots & \forall x_{1n} & \exists \alpha_1 \\ \vdots & & \vdots & \vdots \\ \forall x_{m1} & \dots & \forall x_{mn} & \exists \alpha_m \end{pmatrix},$$

where $\alpha_1, \dots, \alpha_m$ are Boolean variables, i.e., variables that are evaluated on the set $\{\bot, \top\}$. The idea here is that the value of the Boolean variables α_i can only depend on the values of the first-order variables x_{ij}, $j \leq n$, and on the values of variables quantified before the connective. Hence, in partially-ordered connectives the attention is shifted from dependence between first-order variables to dependence between first-order variables and truth values. Boolean dependence logic incorporates this notion of dependence in a transparent and direct way. Boolean dependence logic extends first-order logic by special kind of restricted dependence atoms called Boolean dependence atoms. In Boolean dependence atoms

$$=(x_1, \dots, x_n, \alpha)$$

the antecedents $x_1, \dots, x_n$ are first-order variables, while the consequent α is a Boolean variable. The shift from dependence logic to Boolean dependence logic can also be seen as a shift from functional dependence to relational dependence.

It is a well known fact that the expressive powers of independence-friendly logic and dependence logic coincide with that of existential second-order logic [End70, Wal70, Hod97a, Vää07]. Since the corresponding translations are effective, and even polynomial, it follows that independence-friendly logic and dependence logic are not computationally feasible logics. Thus our interests lie on fragments of dependence logic, Boolean dependence logic, independence-friendly logic and first-order logic enriched with partially-ordered connectives. Our investigations focus on the expressive power and the computational complexity of these fragments. In particular, we are interested in the connections

between and the hierarchies within these different approaches to variable dependence.

Over the last decade the research related to independence-friendly logic and dependence logic has bloomed. A variety of closely related logics have been defined and various applications suggested, see e.g. journal articles [Abr07, AV09, Sev09, VH10, DK12, EK13, GV13, KV13, LV13] and conference reports [BK05, KKLV11, Bra13, EHLV13, EHM$^+$13, EKV13]. Furthermore, within the last five years six PhD-thesis have been published on closely related topics, see [Nur09, Kon10, Gal12, Loh12, Ebb14, Yan14]. See also the monographs [Vää07, MSS11]. Research related to partially-ordered connectives has been less active. For recent work, see e.g. [ST06, HST08, EHLV13].

Structure of this thesis

In Chapter 2 we recall some basic concepts and results relevant to this thesis. We start the chapter with a short survey on notation and conventions. We then give a short exposition on the development of logics of imperfect information, i.e., from partially-ordered quantifiers and connectives to independence-friendly logic and dependence logic. We conclude the chapter with an overview on expressivity and computational complexity.

Chapter 3 is the foremost chapter of this thesis. In it we present a comprehensive study on the finite variable fragments of independence-friendly logic and dependence logic. We compare the expressive powers of these logics to each other and to fragments of first-order logic and existential second-order logic. We then study the related satisfiability problem and the model checking problem. The main result of the chapter is the undecidability of the satisfiability problem for two-variable independence-friendly logic. We obtain the result already for the vocabulary consisting one binary relation symbol and nothing else. We also show that the corresponding problem for two-variable dependence logic is decidable. Since both dependence logic and independence-friendly logic are conservative extensions of first-order logic, we contribute to the study of the satisfiability problem for extensions of two-variable first-order logic. The chapter is based on the article [KKLV11]. However, the article has endured major revising and extending. For example, the above undecidability result for two-variable independence-friendly logic was generalized. In the article, we obtained the result for vocabularies consisting at least two binary relation symbols and infinitely many unary relation symbols. Furthermore, we give a solution for the problem

"Is it possible to define NP-complete problems in $\mathcal{D}^2$ or in $\mathcal{IF}^2$"

asked in [KKLV11]. A more detailed analysis concerning the relationship between Chapter 3 and article [KKLV11], among a more detailed description of the chapter, is described in the introduction of Chapter 3.

In Chapter 4 we study fragments of Boolean dependence logic and partially-ordered connectives. We show that, with respect to expressive power, certain syntactic fragments of Boolean dependence logic correspond to natural logics enriched with partially-ordered connectives. Furthermore, we show that these fragments of Boolean dependence logic form a strict hierarchy with respect to expressive power. In addition, we show that the expressive power of Boolean dependence logic and dependence logic coincide. Since the expressive power of dependence logic and existential second order logic is known to coincide, the results in this chapter can be seen as a contribution to the analysis of fragments of existential second-order logic. In particular, we are able to separate natural fragments of existential second-order logic. Chapter 4 is based on the conference article [EHLV13]. The article has endured major revising, the proofs and definitions have been made more clear and elaborate. More detailed account concerning the relationship of Chapter 4 and article [EHLV13], among a more detailed description of the chapter, is described in the introduction of Chapter 4.

Chapter 2

Preliminaries

In this chapter we recall some basic concepts and results relevant to this thesis. Although the thesis is quite self-contained, the reader is expected to be familiar with first-order logic and some basic concepts in complexity theory, i.e., to have an understanding regarding to the most common complexity classes and the concepts of decidability and undecidability. A short overview with a comprehensive list of further literature on first-order logic and computational complexity can be found, e.g in the book *Elements of Finite Model Theory* by Libkin [Lib04]. A good source on first-order logic is also the book *Mathematical logic* by Ebbinghaus, Flum, and Thomas [EFT94]. For complexity theory the reader can refer to the monograph *Computational complexity* by Papadimitriou [Pap94] and for computability to the book *Introduction to automata theory, languages and computation* by Hopcroft and Ullman [HU00]. While a short introduction on dependence logic and independence-friendly logic is included in this thesis, a reader unfamiliar to logics with team semantics may find some parts of the thesis arduous to read and to understand. For readers not accustomed to dependence logic or to independence-friendly logic, we recommend the monograph *Dependence logic - a new approach to independence friendly logic* by Väänänen [Vää07] and the monograph *Independence-friendly logic - a game-theoretic approach* by Mann, Sandu, and Sevenster [MSS11].

2.1 Notation

We start with a short survey on the notation and conventions used in this thesis. A vocabulary, denoted by symbols τ and σ, is a collection of constant symbols, and function and relation symbols with prescribed arities. A vocabulary is relational if it does not contain any function symbols, i.e., a vocabulary is relational if it contains only constant symbols and relation symbols with prescribed arities. In this thesis vocabularies are always considered relational. We denote constant symbols by $\{c_i\}_{i\in\mathbb{N}}$ and relation symbols by $\{R_i\}_{i\in\mathbb{N}}$. A τ-structure is a tuple

$$\mathfrak{A} = (A, \{c_i^{\mathfrak{A}}\}_{c_i\in\tau}, \{R_i^{\mathfrak{A}}\}_{R_i\in\tau}),$$

where A, called the domain of the structure, is a nonempty set and where

- the interpretation $c_i^{\mathfrak{A}}$ of each constant symbol $c_i \in \tau$ is an element from the set A and
- the interpretation $R_i^{\mathfrak{A}}$ of each relation symbol $R_i \in \tau$ with arity k is a k-ary relation on the set A, i.e., a subset of A^k.

We denote structures by uppercase letters of the typeface Fraktur, e.g $\mathfrak{A}$ and $\mathfrak{B}$, and the corresponding domains by the corresponding uppercase letters of the typeface Latin, e.g. A and B. Classes of structures are usually denoted by uppercase letters of a calligraphic font, e.g. $\mathcal{C}$ and $\mathcal{D}$. A structure $\mathfrak{A}$ is finite if A is a finite set. When τ is a vocabulary and $\mathcal{L}$ some logic, we denote by $\mathcal{L}(\tau)$ the set of all $\mathcal{L}$ formulae with vocabulary τ. When the vocabulary is clear from the context, we often write $\mathcal{L}$ instead of $\mathcal{L}(\tau)$. The lowercase letters x, y, z, and $\{x_i\}_{i\in\mathbb{N}}$ are reserved for first-order variables, while the uppercase letter X, Y, Z, and $\{X_i\}_{i\in\mathbb{N}}$ are used for second-order variables, i.e., relation variables. We sometimes use $\vec{x}$ and $\vec{X}$ to denote a tuple of variables of the corresponding type and of finite length.

2.2 First-order and second-order logic

In this section we recall the definitions of first-order logic and second-order logic. The syntax for first-order logic on a vocabulary τ, $\mathcal{FO}(\tau)$, is defined by the following grammar.

$$\varphi \quad ::= \quad x = y \mid R(x_1, \ldots, x_n) \mid \neg\varphi \mid (\varphi \wedge \varphi) \mid (\varphi \vee \varphi) \mid \exists x\varphi \mid \forall x\varphi,$$

where $R \in \tau$ is an n-ary relation symbol. The formulae $\varphi \to \psi$ and $\varphi \leftrightarrow \psi$ are shorthand notations for $(\neg\varphi \vee \psi)$ and $(\varphi \to \psi) \wedge (\psi \to \varphi)$, respectively. The syntax for second-order logic, $SO(\tau)$, extends the syntax for first-order logic by quantification of second-order variables, i.e., relation variables. In addition to the above rules for first-order logic, we have the following three rules, for each $n \in \mathbb{N}$:

$$\varphi \quad ::= \quad X(x_1, \ldots, x_n) \mid \exists X\varphi \mid \forall X\varphi,$$

where X is an n-ary relation variable. The syntax for monadic second-order logic, $\mathcal{MSO}$, extends the syntax for first-order logic by quantification of set variables, i.e., relation variables of arity 1. Existential second-order logic, $\mathcal{ESO}$, is the syntactic fragment of second-order logic in which the formulae are of the form

$$\exists X_1 \ldots \exists X_n \varphi,$$

where φ does not contain quantification of second-order variables. Existential monadic second-order logic, $\mathcal{EMSO}$, is the syntactic fragment of existential second-order logic in which the relation variables are all of arity 1. The set of *free variables* of an $\mathcal{SO}$ (or $\mathcal{FO}$) formula φ, $\mathrm{fr}(\varphi)$, is defined inductively as follows.

$$\begin{aligned}
\mathrm{fr}(x = y) &:= \{x, y\},\\
\mathrm{fr}\big(R(x_1, \ldots, x_n)\big) &:= \{x_1, \ldots, x_n\},\\
\mathrm{fr}\big(X(x_1, \ldots, x_n)\big) &:= \{X, x_1, \ldots, x_n\},\\
\mathrm{fr}(\neg\varphi) &:= \mathrm{fr}(\varphi),\\
\mathrm{fr}\big((\varphi \wedge \psi)\big) &:= \mathrm{fr}(\varphi) \cup \mathrm{fr}(\psi),\\
\mathrm{fr}\big((\varphi \vee \psi)\big) &:= \mathrm{fr}(\varphi) \cup \mathrm{fr}(\psi),\\
\mathrm{fr}(\exists x\varphi) &:= \mathrm{fr}(\varphi) \setminus \{x\},\\
\mathrm{fr}(\forall x\varphi) &:= \mathrm{fr}(\varphi) \setminus \{x\},\\
\mathrm{fr}(\exists X\varphi) &:= \mathrm{fr}(\varphi) \setminus \{X\},\\
\mathrm{fr}(\forall X\varphi) &:= \mathrm{fr}(\varphi) \setminus \{X\}.
\end{aligned}$$

Semantics for these logics is defined via models and assignments. An *assignment* over a model $\mathfrak{A}$ is a finite function that maps first-order variables to elements of A and a k-ary second-order variables to subsets of A^k. In the following, we denote by χ a variable that is either a first-order variable or a relation variable. If s is an assignment, x a first-order variable and $a \in A$ then

$s(a/x)$ denotes the assignment with domain $\mathrm{dom}(s) \cup \{x\}$ such that

$$s(a/x)(\chi) = \begin{cases} a & \text{if } \chi = x, \\ s(\chi) & \text{otherwise.} \end{cases}$$

Analogously, if s is an assignment, X an k-ary second-order variable and $B \subseteq A^k$ then $s(B/X)$ denotes the assignment with domain $\mathrm{dom}(s) \cup \{X\}$ such that

$$s(B/X)(\chi) = \begin{cases} B & \text{if } \chi = X, \\ s(\chi) & \text{otherwise.} \end{cases}$$

For the modified assignment $s(a_1/x_1)(a_2/x_2)\dots(a_n/x_n)$ we use the shorthand notations $s(a_1/x_1, \dots, a_n/x_n)$ and $s(\vec{a}/\vec{x})$, where $\vec{x} = (x_1, \dots, x_n)$ and $\vec{a} = (a_1, \dots, a_n)$. If s is an assignment and V a set of variables, we denote by $s \restriction V$ the assignment with domain $\mathrm{dom}(s) \cap V$ that agrees with s.

The semantics for first-order logic and second-order logic is defined by the following clauses. Note that when stating $\mathfrak{A}, s \models \varphi$, we always assume that $\mathrm{fr}(\varphi) \subseteq \mathrm{dom}(s)$.

$$\begin{aligned}
\mathfrak{A}, s \models x = y \quad &\text{iff} \quad s(x) = s(y). \\
\mathfrak{A}, s \models R(x_1, \dots, R_n) \quad &\text{iff} \quad (s(x_1), \dots, s(x_n)) \in R^{\mathfrak{A}}. \\
\mathfrak{A}, s \models \neg\varphi \quad &\text{iff} \quad \mathfrak{A}, s \not\models \varphi. \\
\mathfrak{A}, s \models (\varphi \wedge \psi) \quad &\text{iff} \quad \mathfrak{A}, s \models \varphi \text{ and } \mathfrak{A}, s \models \psi. \\
\mathfrak{A}, s \models (\varphi \vee \psi) \quad &\text{iff} \quad \mathfrak{A}, s \models \varphi \text{ or } \mathfrak{A}, s \models \psi. \\
\mathfrak{A}, s \models \exists x\varphi \quad &\text{iff} \quad \mathfrak{A}, s(a/x) \models \varphi \text{ for some } a \in A. \\
\mathfrak{A}, s \models \forall x\varphi \quad &\text{iff} \quad \mathfrak{A}, s(a/x) \models \varphi \text{ for all } a \in A.
\end{aligned}$$

For second-order logic we also have the following additional cases.

$$\begin{aligned}
\mathfrak{A}, s \models X(x_1, \dots, x_n) \quad &\text{iff} \quad (s(x_1), \dots, s(x_n)) \in s(X). \\
\mathfrak{A}, s \models \exists X\varphi \quad &\text{iff} \quad \mathfrak{A}, s(B/X) \models \varphi \text{ for some } B \subseteq A^k, \\
&\qquad \text{where } k \text{ is the arity of the relation variable } X. \\
\mathfrak{A}, s \models \forall X\varphi \quad &\text{iff} \quad \mathfrak{A}, s(B/X) \models \varphi \text{ for all } B \subseteq A^k, \\
&\qquad \text{where } k \text{ is the arity of the relation variable } X.
\end{aligned}$$

We sometimes use $\models_{\mathcal{FO}}$ to denote the satisfaction relation of first-order logic.

2.3 From perfect to imperfect information

Semantics for first-order and second-order logic can also be defined by the so-called semantic game. For a detailed exposition to game-theoretic semantics see, e.g. [MSS11]. The semantic game is played between two players, the Verifier and the Falsifier, on a model, an assignment and a formula. The goal of the Verifier is to show that a given formula is true in a given model and an assignment, while the goal of the Falsifier is to show that the formula is false in the given model and assignment. In a play of the game Verifier assigns values for the existentially quantified variables, while the Falsifier assigns values for the universally quantified variables. On conjunctions the Falsifier selects the conjunct on which the play continues, whereas on disjunctions the Verifier makes the choice. The play ends when an atomic formula is reached. The Verifier wins the play of the game if the model with the assignment built during the play satisfies the atomic formula that was reached. Otherwise the Falsifier wins the play. The formula is said to be true in a given model and an assignment if the Verifier can, regardless of the actions of the Falsifier, always win the play of the game.

In first-order logic (and second-order logic) the order in which quantifiers are written determines dependence relations between variables. For example, when using game-theoretic semantics to evaluate the first-order formula

$$\forall x_0 \exists x_1 \forall x_2 \exists x_3 \, \varphi,$$

the value the Verifier selects for x_1 can depend on the value chosen earlier by the Falsifier for x_0, and the value chosen for x_3 can depend on the values chosen for both universally quantified variables x_0 and x_2. On each round in the semantic game when the Verifier picks a value for a given variable he always knows the values of every variable that has been picked on earlier rounds in the game. In that sense first-order logic (and second-order logic) can be seen as a logic of perfect information.

In logics of perfect information the dependence relation between variables is a linear order. The first to consider more complex dependences between variables was Henkin [Hen61] with his partially-ordered quantifiers. A partially-ordered quantifier is an expression of the form

$$Q = \begin{pmatrix} \forall x_{11} & \dots & \forall x_{1n} & \exists y_1 \\ \vdots & & \vdots & \vdots \\ \forall x_{m1} & \dots & \forall x_{mn} & \exists y_m \end{pmatrix}.$$

When using game-theoretic semantics to evaluate the quantifier Q, the value chosen for the variable y_i can only depend on the values chosen for the variables x_{ij}, $j \leq n$, and on the values chosen for the variables evaluated before the quantifier Q. In particular, the value chosen for y_i is not allowed to depend on the values chosen for the variables x_{kj} nor the variable y_k if $k \neq i$. Enderton [End70] and Walkoe [Wal70] observed that exactly the properties definable in existential second-order logic ($\mathcal{ESO}$) can be expressed with $\mathcal{POQ}[\mathcal{FO}]$, first-order logic with a partially-ordered quantifier prefix. Building on the ideas of Henkin, Blass and Gurevich [BG86] introduced the narrow Henkin quantifier. The narrow Henkin quantifier is an expression of the form

$$N = \begin{pmatrix} \forall x_{11} & \dots & \forall x_{1n} & \exists \alpha_1 \\ \vdots & & \vdots & \vdots \\ \forall x_{m1} & \dots & \forall x_{mn} & \exists \alpha_m \end{pmatrix},$$

where the variables α_i are Boolean variables, i.e., variables that are evaluated on the set $\{\bot, \top\}$, and the variables x_{ij} are Boolean or first-order variables. The idea here is that the value of the Boolean variables α_i can only depend on the values of the first-order and Boolean variables x_{ij}, $j \leq n$, and on the values of the variables quantified before the narrow Henkin quantifier N. The idea of Blass and Gurevich was further developed by Sandu and Väänänen [SV92] by the introduction of partially-ordered connectives. According to the definition of Sandu and Väänänen, a partially-ordered connective is an expression of the form

$$D = \begin{pmatrix} \forall x_{11} & \dots & \forall x_{1n} & \bigvee_{b_1 \in \{0,1\}} \\ \vdots & & \vdots & \vdots \\ \forall x_{m1} & \dots & \forall x_{mn} & \bigvee_{b_m \in \{0,1\}} \end{pmatrix}$$

that binds a tuple $\gamma = (\varphi_{\vec{b}})_{\vec{b} \in \{0,1\}^m}$ of formulae. Here the idea is that the values of the universally quantified variables $x_{i1}, \dots, x_{in}$ determines the value of $b_i \in \{0, 1\}$. The value of the tuple $b_1, \dots, b_m$ then corresponds to a formula in the tuple $(\varphi_{\vec{b}})_{\vec{b} \in \{0,1\}^m}$.

The first to linearize the idea behind the syntax of partially-ordered quantifiers were Hintikka and Sandu [Hin96, HS89], who introduced independence-friendly logic ($\mathcal{IF}$). Independence-friendly logic extends first-order logic in terms of so-called slashed quantifiers. For example, in

$$\forall x_0 \exists x_1 \forall x_2 \exists x_3 / \{x_0\} \varphi, \tag{2.1}$$

the intended meaning of the quantifier $\exists x_3 / \{x_0\}$ is that x_3 is "independent" of x_0 in the sense that the choice for the value of x_3 should not depend on

what the value of x_0 is. Dependence logic, introduced by Väänänen [Vää07], was inspired by $\mathcal{IF}$-logic, but the approach of Väänänen provided a fresh perspective on quantifier dependence. In dependence logic the dependence relations between variables are written in terms of novel atomic dependence formulae

$$=(x_1, \ldots, x_n, y).$$

The intended meaning of the atomic formula $=(x_1, \ldots, x_n, y)$ is that the value of the variable y depends solely on the values of the variables $x_1, \ldots, x_n$. For example, the $\mathcal{IF}$-sentence (2.1) can be expressed in dependence logic by the sentence

$$\forall x_0 \exists x_1 \forall x_2 \exists x_3 (=(x_2, x_3) \wedge \varphi),$$

where the formula $=(x_2, x_3)$ indicates that x_3 depends on x_2 only.

Despite the first-order like syntax of dependence logic and independence-friendly logic, on the level of sentences, the expressive power of both of these logics coincide with that of existential second-order logic. The high expressive power and first-order like syntax however yield that the meaning of sentences of these logics is often difficult to parse.

2.4 Dependence and independence-friendly logic

In this section we define dependence logic and independence-friendly logic. We then present some fundamental properties of these logics. For dependence logic we follow the monograph [Vää07], while for independence-friendly logic we follow the exposition of [MSS11].

Definition 2.4.1. *Let τ be a relational vocabulary. The syntax for* independence-friendly *logic $\mathcal{IF}(\tau)$ is generated from τ according to the following grammar:*

$$\begin{aligned} \varphi \quad ::= \quad & x = y \mid \neg x = y \mid R(x_1, \ldots, x_n) \mid \neg R(x_1, \ldots, x_n) \mid \\ & (\varphi \wedge \varphi) \mid (\varphi \vee \varphi) \mid \forall x \varphi \mid \exists x / W \varphi, \end{aligned}$$

where W is a set of first-order variables. We identify existential first-order quantifiers with existential quantifiers with empty slash sets, and therefore if $W = \emptyset$ we simply write $\exists x \varphi(x)$ instead of $\exists x / W \varphi(x)$.

Definition 2.4.2. *Let τ be a relational vocabulary. The syntax for dependence logic $\mathcal{D}(\tau)$ is defined by the following grammar:*

$$\begin{aligned} \varphi \quad ::= \quad & x_1 = x_2 \mid \neg\, x_1 = x_2 \mid R(x_1, \ldots, x_n) \mid \neg R(x_1, \ldots, x_n) \mid \\ & =(x_1, \ldots, x_n) \mid (\varphi \vee \varphi) \mid (\varphi \wedge \varphi) \mid \forall x \varphi \mid \exists x \varphi. \end{aligned}$$

The set of *free variables* $\mathrm{fr}(\varphi)$ of an $\mathcal{IF}$- or $\mathcal{D}$-formula φ is defined inductively as follows. The cases for atomic formulae, Boolean connectives and first-order quantifiers are defined as in first-order logic. For dependence logic we have the additional clause

$$\mathrm{fr}\big(=(x_1,\ldots,x_n)\big) := \{x_1,\ldots,x_n\},$$

whereas for independence-friendly logic we have the additional clause

$$\mathrm{fr}(\exists x/W\psi) := \big(\mathrm{fr}(\psi) \setminus \{x\}\big) \cup W.$$

Note that if x is in W, then x is a free variable in $\exists x/W\psi$. As usual, we say that a formula φ is a sentence if $\mathrm{fr}(\varphi) = \emptyset$.

The semantics for independence-friendly logic and dependence logic is defined via models and teams, i.e., sets of assignments. Let A be a set and $\{x_1,\ldots,x_n\}$ a finite (possibly empty) set of variables. A *team* X of A with the domain

$$\mathrm{dom}(X) = \{x_1,\ldots,x_n\}$$

is any set of assignments from $\mathrm{dom}(X)$ into A. However, for the empty team $\emptyset$, we define that $\mathrm{dom}(\emptyset) = \emptyset$. If X is a team of A and $F\colon X \to A$ a function, we use

- $X(F/x)$ to denote the team $\{s(F(s)/x) \mid s \in X\}$, and
- $X(A/x)$ to denote the team $\{s(a/x) \mid s \in X \text{ and } a \in A\}$.

For a set $W \subseteq \mathrm{dom}(X)$, we call the function F *W-determined* if for every $s, s' \in X$ the implication

$$\forall x \in W : s(x) = s'(x) \quad \Rightarrow \quad F(s) = F(s')$$

holds. We call the function *W-independent* if for every $s, s' \in X$ the implication

$$\forall x \in \mathrm{dom}(X) \setminus W : s(x) = s'(x) \quad \Rightarrow \quad F(s) = F(s')$$

holds.

In the following definitions we always assume that the domain of X contains $\mathrm{fr}(\varphi)$.

Definition 2.4.3. *Let $\mathfrak{A}$ be a model and X a team of A. The satisfaction relation $\mathfrak{A} \models_X \varphi$ for independence-friendly logic is defined as follows:*

$$\begin{aligned}
\mathfrak{A} \models_X R(x_1,\dots,x_n) \quad &\textit{iff} \quad \forall s \in X : (s(x_1),\dots,s(x_n)) \in R^{\mathfrak{A}}.\\
\mathfrak{A} \models_X \neg R(x_1,\dots,x_n) \quad &\textit{iff} \quad \forall x \in X : (s(x_1),\dots,s(x_n)) \notin R^{\mathfrak{A}}.\\
\mathfrak{A} \models_X \psi \wedge \varphi \quad &\textit{iff} \quad \mathfrak{A} \models_X \psi \textit{ and } \mathfrak{A} \models_X \varphi.\\
\mathfrak{A} \models_X \psi \vee \varphi \quad &\textit{iff} \quad \mathfrak{A} \models_Y \psi \textit{ and } \mathfrak{A} \models_Z \varphi,\\
&\qquad \textit{for some } Y \textit{ and } Z \textit{ such that } Y \cup Z = X.\\
\mathfrak{A} \models_X \exists x/W \psi \quad &\textit{iff} \quad \mathfrak{A} \models_{X(F/x)} \psi, \textit{ for some } W\textit{-independent}\\
&\qquad \textit{function } F\colon X \to A.\\
\mathfrak{A} \models_X \forall x \psi \quad &\textit{iff} \quad \mathfrak{A} \models_{X(A/x)} \psi.
\end{aligned}$$

Definition 2.4.4. *Let $\mathfrak{A}$ be a model and X a team of A. The satisfaction relation $\mathfrak{A} \models_X \varphi$ for dependence logic is defined as follows:*

$$\begin{aligned}
\mathfrak{A} \models_X R(x_1,\dots,x_n) \quad &\textit{iff} \quad \forall s \in X : (s(x_1),\dots,s(x_n)) \in R^{\mathfrak{A}}.\\
\mathfrak{A} \models_X \neg R(x_1,\dots,x_n) \quad &\textit{iff} \quad \forall s \in X : (s(x_1),\dots,s(x_n)) \notin R^{\mathfrak{A}}.\\
\mathfrak{A} \models_X =(x_1,\dots,x_n) \quad &\textit{iff} \quad \forall s,t \in X : s(x_1) = t(x_1),\dots,s(x_{n-1}) = t(x_{n-1})\\
&\qquad \textit{implies that } s(x_n) = t(x_n).\\
\mathfrak{A} \models_X \psi \wedge \varphi \quad &\textit{iff} \quad \mathfrak{A} \models_X \psi \textit{ and } \mathfrak{A} \models_X \varphi.\\
\mathfrak{A} \models_X \psi \vee \varphi \quad &\textit{iff} \quad \mathfrak{A} \models_Y \psi \textit{ and } \mathfrak{A} \models_Z \varphi,\\
&\qquad \textit{for some } Y \textit{ and } Z \textit{ such that } Y \cup Z = X.\\
\mathfrak{A} \models_X \exists x \psi \quad &\textit{iff} \quad \mathfrak{A} \models_{X(F/x)} \psi \textit{ for some function } F\colon X \to A.\\
\mathfrak{A} \models_X \forall x \psi \quad &\textit{iff} \quad \mathfrak{A} \models_{X(A/x)} \psi.
\end{aligned}$$

We say that a sentence φ of independence-friendly logic or dependence logic is *true in a model* $\mathfrak{A}$, and write $\mathfrak{A} \models \varphi$, if $\mathfrak{A} \models_{\{\emptyset\}} \varphi$ holds.

The following proposition reveals that independence-friendly logic and dependence logic are conservative extensions of first-order logic, i.e., the semantics of independence-friendly logic and dependence logic coincide with that of first-order logic with regards to formulae that are syntactically first-order.

Proposition 2.4.5 ([Vää07, Hod97a])**.** *Let φ be a formula of dependence logic or independence-friendly logic without dependence atoms and without slashed quantifiers, i.e., φ is syntactically a first-order formula. Then for every model $\mathfrak{A}$, team X of $\mathfrak{A}$ and assignment s of $\mathfrak{A}$:*

1. $\mathfrak{A} \models_{\{s\}} \varphi \quad \Leftrightarrow \quad \mathfrak{A}, s \models_{\mathcal{FO}} \varphi$.

2. $\mathfrak{A} \models_X \varphi \quad \Leftrightarrow \quad \mathfrak{A}, t \models_{\mathcal{FO}} \varphi$, *for every* $t \in X$.

Let X be a team with domain $\{x_1, \ldots, x_k\}$ and $V \subseteq \{x_1, \ldots, x_k\}$. We denote by $X \upharpoonright V$ the team

$$\{s \upharpoonright V \mid s \in X\}$$

with domain V. The following proposition shows that truth of a dependence logic formula depends only on the interpretations of the variables occurring free in the formula.

Proposition 2.4.6 ([Vää07])**.** *Let φ be any formula of dependence logic and $V \supseteq \mathrm{fr}(\varphi)$. Now*

$$\mathfrak{A} \models_X \varphi \quad \Leftrightarrow \quad \mathfrak{A} \models_{X \upharpoonright V} \varphi.$$

The analogue of Proposition 2.4.6 does not hold for open formulae of independence-friendly logic. In other words, the truth of an $\mathcal{IF}$-formula may depend on the interpretations of variables that do not occur in the formula. For example, the truth of the formula

$$\varphi := \exists x / \{y\} x = y$$

in a team X with domain $\{x, y, z\}$ depends on the values of z in X, although z does not occur in φ. However, the analogue of Proposition 2.4.6 does hold for sentences of independence-friendly logic.

Proposition 2.4.7. *Let φ be a sentence of independence-friendly logic, $\mathfrak{A}$ a model and X a nonempty team of $\mathfrak{A}$. Then*

$$\mathfrak{A} \models_X \varphi \quad \Leftrightarrow \quad \mathfrak{A} \models_{\{\emptyset\}} \varphi.$$

Proof. For a proof see, e.g. [MSS11]. □

The following fact is a fundamental property of all formulae of dependence logic and independence-friendly logic.

Proposition 2.4.8 (Downward closure)**.** *Let φ be a formula of dependence logic or independence-friendly logic, $\mathfrak{A}$ a model, and $Y \subseteq X$ teams of $\mathfrak{A}$. Then*

$$\mathfrak{A} \models_X \varphi \quad \Rightarrow \quad \mathfrak{A} \models_Y \varphi.$$

Proof. For the proofs see, e.g. [Vää07, MSS11]. □

2.5 Expressive power and computational complexity

Two of the most fundamental aspects of any logic are its expressive power and computational complexity, i.e., what properties can be expressed with a given logic and how much resources, i.e., time and space, is needed to verify whether a described property holds or not. An ideal logic would of course be at the same time highly expressive and still computationally feasible. Unfortunately these concepts seem to be the opposite sides of the same coin. In most cases a more expressive logic also means a computationally more complex logic. And indeed, if the expressive power of a logic is great enough, many questions concerning the logic become even undecidable.

2.5.1 Expressive power and definability

The most common way to study the expressive power of a given logic is to compare it to the expressive power of other known logics. Due the overwhelming expressive power of second-order logic, most logics, including all of the logics considered in this thesis, can be seen as fragments of second-order logic. Hence second-order logic and its fragments are natural benchmark logics, a common ground, for studying the expressive power of almost any given logic.

Definition 2.5.1. *We say that a logic $\mathcal{L}'$ is* at least as expressive *as $\mathcal{L}$ and write $\mathcal{L} \leq \mathcal{L}'$ if for every sentence $\varphi \in \mathcal{L}$ there exists a sentence $\varphi' \in \mathcal{L}'$ such that for every structure $\mathfrak{A}$ it holds that*

$$\mathfrak{A} \models \varphi \quad \Leftrightarrow \quad \mathfrak{A} \models \varphi'.$$

We say $\mathcal{L}$ and $\mathcal{L}'$ are equivalent *and write $\mathcal{L} \equiv \mathcal{L}'$ if $\mathcal{L} \leq \mathcal{L}'$ and $\mathcal{L}' \leq \mathcal{L}$.*

Definition 2.5.2. *Let $\mathcal{L}$ be a logic such that the semantics for $\mathcal{L}$ is defined via models and assignments. Let $\varphi, \varphi' \in \mathcal{L}$ such that $\mathrm{fr}(\varphi) = \mathrm{fr}(\varphi')$. We say that φ and φ' are* equivalent *if for every model $\mathfrak{A}$ and every assignment s such that $\mathrm{fr}(\varphi) \subseteq \mathrm{dom}(s)$*

$$\mathfrak{A}, s \models \varphi \quad \Leftrightarrow \quad \mathfrak{A}, s \models \varphi'.$$

Let $\mathcal{C}$ be a class of structures. We say that φ and φ' are equivalent *in the class $\mathcal{C}$ if*

$$\mathfrak{A}, s \models \varphi \quad \Leftrightarrow \quad \mathfrak{A}, s \models \varphi'.$$

holds for every $\mathfrak{A} \in \mathcal{C}$ and assignment s such that $\mathrm{fr}(\varphi) \subseteq \mathrm{dom}(s)$.

Definition 2.5.3. *Let $\mathcal{L}$ be a logic such that the semantics for $\mathcal{L}$ is defined via models and teams. Let $\varphi, \varphi' \in \mathcal{L}$ such that* $\mathrm{fr}(\varphi) = \mathrm{fr}(\varphi')$. *We say that φ and φ' are* equivalent *if for every model $\mathfrak{A}$ and every team X of $\mathfrak{A}$ such that* $\mathrm{fr}(\varphi) \subseteq \mathrm{dom}(X)$

$$\mathfrak{A} \models_X \varphi \quad \Leftrightarrow \quad \mathfrak{A} \models_X \varphi'.$$

Let $\mathcal{C}$ be a class of structures. We say that φ and φ' are equivalent *in the class $\mathcal{C}$ if*

$$\mathfrak{A} \models_X \varphi \quad \Leftrightarrow \quad \mathfrak{A} \models_X \varphi'.$$

holds for every $\mathfrak{A} \in \mathcal{C}$ and team X of $\mathfrak{A}$ such that $\mathrm{fr}(\varphi) \subseteq \mathrm{dom}(X)$.

On the level of sentences the expressive power of dependence logic and independence-friendly coincide with that of existential second-order logic.

Theorem 2.5.4. $\mathcal{D} \equiv \mathcal{IF} \equiv \mathcal{ESO}$.

Proof. The fact $\mathcal{ESO} \leq \mathcal{D}$ (and $\mathcal{ESO} \leq \mathcal{IF}$) is based on the analogous result of [End70, Wal70] for partially-ordered quantifiers. For the converse inclusions, see [Vää07], [Hod97b] and [MSS11]. □

In addition to relative expressive power of logics we are of course interested in the concrete expressive power of a given logic, i.e., we are interested in the question "Which properties can be defined in a given logic?". Here we consider only Boolean queries, i.e., questions of the type "Does $\mathfrak{A} \in \mathcal{C}$ hold?", where $\mathcal{C}$ is some class of structures closed under isomorphisms. Examples of Boolean queries are: "Is the model finite?", "Is the graph connected?" and "Is the cardinality of the unary predicate P smaller than the cardinality of the unary predicate Q in the model?".

Definition 2.5.5. *Let τ be a vocabulary, $\mathcal{L}(\tau)$ a logic and $\mathcal{C}$ a class of τ-structures closed under isomorphisms. We say that the class $\mathcal{C}$ is* definable *in $\mathcal{L}(\tau)$ if there exists a sentence $\varphi \in \mathcal{L}(\tau)$ such that*

$$\mathfrak{A} \models \varphi \quad \Leftrightarrow \quad \mathfrak{A} \in \mathcal{C}$$

holds for every τ-structure $\mathfrak{A}$. We say that the class of structures is not definable *in $\mathcal{L}(\tau)$ if no such sentence exists.*

2.5.2 Decidability and complexity

Undoubtedly the proliferation of computers and the rise of computer science in general in the end of the 20th century has had a seminal effect for the importance of the study of decidability and computational complexity, also in the field of logic. In modern computer science logic is used, among other things, to model and to verify existing real life systems, in automatic reasoning and, for example, in handling queries in huge medical databases. Hence the need for computationally feasible logics is now greater than ever.

We consider the decidability and complexity of two essential problems concerning any logic. The satisfiability problem and the model checking problem. The former concentrates on the questions like: "Given a specification of a system is it possible to realize that specification?". In logical terms this translates to: "Given a sentence of some logic does there exist a model that satisfies it?". We consider two variants of the satisfiability problem, i.e., the general satisfiability problem and the finite satisfiability problem.

Definition 2.5.6. *Let* K *be a complexity class,* $\mathcal{L}$ *a logic and*

$$enc : \mathcal{L} \to \{0,1\}^*$$

some efficient encoding that maps $\mathcal{L}$*-formulae to binary strings. We say that* the satisfiability problem *of* $\mathcal{L}$ *is in* K *(is* K*-hard) if the language*

$$\{enc(\varphi) \mid \textit{there exists a model } \mathfrak{A} \textit{ such that } \mathfrak{A} \models \varphi\}$$

belongs in K *(is K-hard). Analogously, we say that* the finite satisfiability problem *of* $\mathcal{L}$ *is in* K *(is* K*-hard) if the language*

$$\{enc(\varphi) \mid \textit{there exists a finite model } \mathfrak{A} \textit{ such that } \mathfrak{A} \models \varphi\}$$

belongs in K *(is* K*-hard).*

Remember that a problem is K-*complete* if it is both in K and K-hard. For a logic $\mathcal{L}$, we denote by $\textsc{Sat}(\mathcal{L})$ and $\textsc{FinSat}(\mathcal{L})$ the satisfiability problem and the finite satisfiability problem of $\mathcal{L}$, respectively. We first state the folklore results concerning first-order and second-order logic, and then the corresponding results for dependence logic and independence-friendly logic.

Theorem 2.5.7.

1. *The satisfiability problem for $\mathcal{FO}$ and $\mathcal{ESO}$ is Π^0_1-complete.*
2. *The finite satisfiability problem for $\mathcal{FO}$ and $\mathcal{ESO}$ is Σ^0_1-complete.*

Proof. The results follow already from the work of Church and Turing [Chu36, Tur36]. □

The results for dependence logic and independence-friendly logic follow from Theorem 2.5.7 by the computable translations between $\mathcal{IF}$, $\mathcal{D}$ and $\mathcal{ESO}$, see [End70, Wal70, Vää07, MSS11].

Theorem 2.5.8.

1. *The satisfiability problem for $\mathcal{IF}$ and $\mathcal{D}$ is Π^0_1-complete.*
2. *The finite satisfiability problem for $\mathcal{IF}$ and $\mathcal{D}$ is Σ^0_1-complete.*

The model checking problem deals with problems of the sort: "Given a system and a specification, does the system satisfy the specification?". In logical terms this translates to: "Given a sentence of some logic and a model, does the model satisfy the sentence?". The model checking problem has two inputs, a finite model and a formula. Hence we can identify three different complexity measures for this problem. We can fix the model, the formula or neither. In the definition below $\mathfrak{A}$ is always a finite model.

Definition 2.5.9. *Let* K *be a complexity class, $\mathcal{L}$ a logic and and enc some efficient encoding that maps $\mathcal{L}$-formulae and finite models to binary strings.*

- *The data complexity of $\mathcal{L}$ is in* K *if for every sentence φ of $\mathcal{L}$ the language*

  $$\{enc(\mathfrak{A}) \mid \mathfrak{A} \models \varphi\}$$

  belongs to K*, and it is* K*-hard if there exists a sentence φ such that*

  $$\{enc(\mathfrak{A}) \mid \mathfrak{A} \models \varphi\}$$

  is a K*-hard problem.*

- *The expression complexity of $\mathcal{L}$ is in* K *if for every finite structure $\mathfrak{A}$ the language*

  $$\{enc(\varphi) \mid \mathfrak{A} \models \varphi\}$$

  belongs to K*, and it is* K*-hard if there exists a finite model $\mathfrak{A}$ such that*

  $$\{enc(\varphi) \mid \mathfrak{A} \models \varphi\}$$

  is a K*-hard problem.*

- *The combined complexity of $\mathcal{L}$ is in* K *(is K-hard) if the language*

$$\{(enc(\mathfrak{A}), enc(\varphi)) \mid \mathfrak{A} \models \varphi\}$$

belongs to K *(is K-hard).*

We conclude with some related complexity results. We first state the folklore results concerning first-order and existential second-order logic and then finish off with known results concerning dependence logic and independence-friendly logic.

Theorem 2.5.10.

1. *The data complexity for $\mathcal{FO}$ is in AC^0.*
2. *The expression complexity of $\mathcal{FO}$ is* PSPACE*-complete.*
3. *The combined complexity of $\mathcal{FO}$ is* PSPACE*-complete.*

Proof. For proofs see, e.g. [Lib04]. □

Theorem 2.5.11 ([Var82])**.**

1. *The data complexity of $\mathcal{ESO}$ is* NP*-complete.*
2. *The expression complexity of $\mathcal{ESO}$ is* NEXPTIME*-complete.*
3. *The combined complexity of $\mathcal{ESO}$ is* NEXPTIME*-complete.*

Theorem 2.5.12.

1. *The data complexity for $\mathcal{D}$ is* NP*-complete.*
2. *The expression complexity for $\mathcal{D}$ is* NEXPTIME*-complete.*
3. *The combined complexity of $\mathcal{D}$ is* NEXPTIME*-complete.*

Proof. The first claim follows directly from Theorems 2.5.4 and 2.5.11. The second and the third claim was shown by Grädel [Grä13]. □

By Theorems 2.5.4 and 2.5.11, it also follows that the data complexity for $\mathcal{IF}$ is NP-complete.

Chapter 3

Finite Variable Dependence Logic and Independence-Friendly Logic

In this chapter we present a comprehensive study on the finite variable fragments of independence-friendly logic and dependence logic. We compare the expressive powers of these logics to each other and to fragments of first-order logic and existential second-order logic. We then study the related satisfiability problem and the model checking problem. Since both independence-friendly logic and dependence logic are conservative extensions of first-order logic, i.e., they agree with first-order logic on sentences which are syntactically first-order, our results contribute to the understanding of extensions of finite variable first-order logics. Since the expressive powers of existential second-order logic, independence-friendly logic and dependence logic coincide, our results also contribute to the study of fragments of existential second-order logic. Furthermore, we contribute to the study of the logical background theory for the concept of dependence. The main result of this chapter is the discovery of an undecidability barrier between the satisfiability problems of two-variable dependence logic and two-variable independence-friendly logic. Hence we particularly contribute to the study of the satisfiability problem for extensions of two-variable first-order logic. We also answer "yes, to both accounts" to the following question asked in [KKLV11]: "Is it possible to define NP-complete problems in $\mathcal{D}^2$ or in $\mathcal{IF}^2$".

The satisfiability problem of first-order logic was shown to be undecidable by Church and Turing [Chu36, Tur36], and ever since logicians have been searching for decidable fragments of first-order logic. The fragments $\mathcal{FO}^k$, for $k \geq 3$, were easily seen to be undecidable but the case for $k = 2$ remained open. Scott [Sco62] showed that $\mathcal{FO}^2$ without equality is decidable. Mortimer [Mor75] extended the result to $\mathcal{FO}^2$ with equality and showed that every satis-

fiable $\mathcal{FO}^2$ formula has a model whose size is doubly exponential in the length of the formula. His result established that the satisfiability and the finite satisfiability problems of $\mathcal{FO}^2$ are contained in 2NEXPTIME. Finally, Grädel, Kolaitis, and Vardi [GKV97] improved the result of Mortimer by establishing that every satisfiable $\mathcal{FO}^2$-formula has a model of exponential size. Furthermore, they showed that the satisfiability problem for $\mathcal{FO}^2$ is NEXPTIME-complete.

The decidability of the satisfiability problem of various extensions of $\mathcal{FO}^2$ has been studied extensively (e.g. [GOR97b, GO99, EVW02, KO05]). One such interesting extension $\mathcal{FOC}^2$ is acquired by extending $\mathcal{FO}^2$ with counting quantifiers $\exists^{\geq i}$. The meaning of a formula of the form

$$\exists^{\geq i} x \varphi(x)$$

is that $\varphi(x)$ is satisfied by at least i distinct elements. The satisfiability problem for the logic $\mathcal{FOC}^2$ was shown to be decidable by Grädel, Otto, and Rosen [GOR97a], and shown to be in 2NEXPTIME by Pacholski, Szwast, and Tendera [PST97]. Finally, Pratt-Hartmann [PH05] established that the problem is NEXPTIME-complete. We will later use the result of Pratt-Hartmann to determine the complexity of the satisfiability problem of the two-variable fragment of dependence logic.

With two-variable dependence logic, we identify an expressive but still decidable extension of $\mathcal{FO}^2$ that neither has the finite model property nor the zero-one law. The former already with the empty vocabulary and latter in the presence of one unary predicate symbol. Illustrating how close to the undecidability barrier $\mathcal{D}^2$ really is, we succeed on showing that two-variable independence-friendly logic is already undecidable.

The structure of this chapter is as follows. In Section 3.1 we define the logics relevant to this chapter, i.e., k-variable fragments of first-order logic, first-order logic with counting, existential second-order logic, dependence logic, and independence-friendly logic. In Section 3.2 we describe a general method of interpreting structures inside other structures called uniform interpretation. Section 3.3 begins with a comparison of the expressive powers of finite variable dependence logics and independence-friendly logics to fragments of first-order logic. We then continue by comparing the expressive powers of finite variable dependence logics to finite variable independence-friendly logics. We conclude the section by comparing finite variable dependence logics and independence-friendly logics to fragments of existential second-order logic. In Section 3.4 we discuss on the criterion when finite variable dependence logics and independence-friendly logics have the finite model property and the zero-one law. In Section 3.5 we study the satisfiability and the finite satisfiability

problems for finite variable dependence logics. In Section 3.6 we introduce a general method of proving undecidability called the tiling method. Sections 3.7 and 3.8 are devoted on the study of the satisfiability and the finite satisfiability problems for finite variable independence-friendly logics. Section 3.7 is devoted for preparatory matters and lemmata needed in the main theorems of Section 3.8. Finally, in Section 3.9 we study the model checking problem, i.e., data complexity, expression complexity and combined complexity, of finite variable dependence logics and independence-friendly logics. Section 3.10 summarizes the results of the current chapter.

This chapter is based on the article [KKLV11]. However, the article has endured major revising and extending. In [KKLV11] only two-variable dependence logic and independence-logic is considered. In the article we show the following:

1. We show that the satisfiability problem and the finite satisfiability problem for $\mathcal{IF}^2$ is undecidable.

2. We establish that the satisfiability problem and the finite satisfiability problem for $\mathcal{D}^2$ is NEXPTIME-complete.

3. We show that $\mathcal{D}^2 < \mathcal{IF}^2 \leq \mathcal{D}^3$ with respect to expressive power.

Most of the results in this thesis concerning k-variable logics where $k \neq 2$ are new. Furthermore, most of the results concerning expressive power and all of the results concerning model checking are new. All proofs concerning the above results $(1) - (3)$ have been thoroughly rewritten. The results concerning the satisfiability problem and the finite satisfiability problem of $\mathcal{IF}^2$ have been generalized. In [KKLV11] we show that these problems are undecidable if the vocabulary includes at least two binary relation symbols and infinitely many unary relation symbols. In this chapter we show that it suffices to have only one binary relation symbol, in particular, no unary relation symbols are needed.

3.1 Finite variable logics

In this section we define the logics relevant to this chapter, i.e., k-variable fragments of first-order logic, first-order logic with counting, existential second-order logic, dependence logic, and independence-friendly logic. We start by defining an expansion of first-order logic called first-order logic with counting. A counting quantifier is an expression of the form

$$\exists^{\geq i} x,$$

where x is a first-order variable and $i \in \mathbb{N}$. First-order logic with counting, denoted by $\mathcal{FOC}$, is the extension of first-order logic with all counting quantifiers. In other words, the syntax of $\mathcal{FOC}$ extends the syntax of first-order logic by the clause

$$\varphi ::= \exists^{\geq i} x\, \varphi,$$

where x is a first-order variable and $i \in \mathbb{N}$. The semantics for $\mathcal{FOC}$ is defined in the same manner as for first-order logic with the following additional clause for counting quantifiers:

$$\mathfrak{A}, s \models \exists^{\geq i} x\, \varphi \quad \text{iff} \quad |\{a \in A \mid \mathfrak{A}, s(a/x) \models \varphi\}| \geq i.$$

Note that a counting quantifier

$$\exists^{\leq i} x$$

is regarded as a shorthand notation for the expression

$$\neg \exists^{\geq i+1} x.$$

The k-variable first-order logic, denoted by $\mathcal{FO}^k$, is the syntactic fragment of first-order logic in which only variables form the set $\{x_1, \ldots, x_k\}$ can appear in formulae. In the case $k = 2$ we denote these variables by x and y. The k-variable fragment $\mathcal{FOC}^k$ of $\mathcal{FOC}$ is defined analogously. The k-variable existential second-order logic, $\mathcal{ESO}^k$, is the syntactic fragment of $\mathcal{ESO}$ in which only first-order variables from the set $\{x_1, \ldots, x_k\}$ can appear in formulae. Note that there is no restrictions concerning the usage of second-order variables. Also the alternative notation $\Sigma^1_1(\mathcal{FO}^k)$ is used for $\mathcal{ESO}^k$. By $\Sigma^1_1(\mathcal{FOC}^k)$ we denote the extension of $\mathcal{ESO}^k$ with counting quantifiers for $x_1, \ldots, x_k$.

The syntax for k-variable dependence logic and independence-friendly logic is given below. For the semantics see Definitions 2.4.3 and 2.4.4 in Chapter 2.

Definition 3.1.1. *Let τ be a relational vocabulary. The set of formulae for* k-variable independence-friendly *logic $\mathcal{IF}^k(\tau)$ is generated from τ according to the following grammar:*

$$\begin{aligned}\varphi \quad ::= \quad & x_i = x_j \mid \neg x_i = x_j \mid R(x_{i_1}, \ldots, x_{i_n}) \mid \neg R(x_{i_1}, \ldots, x_{i_n}) \mid \\ & (\varphi \wedge \varphi) \mid (\varphi \vee \varphi) \mid \forall x_i \varphi \mid \exists x_i / W \varphi,\end{aligned}$$

where $R \in \tau$ is an n-ary relation symbol, $W \subseteq \{x_1, \ldots, x_k\}$ is a set of first-order variables and $x_i, x_j, x_{i_1}, \ldots, x_{i_n} \in \{x_1, \ldots, x_k\}$. We identify existential first-order quantifiers with existential quantifiers with empty slash sets, and therefore if $W = \emptyset$ we simply write $\exists x \varphi(x)$ instead of $\exists x / W \varphi(x)$.

Definition 3.1.2. *Let τ be a relational vocabulary. The set of formulae for* k-variable dependence logic *$\mathcal{D}^k(\tau)$ is defined by the following grammar:*

$$\begin{aligned}\varphi \quad ::= \quad & x_i = x_j \mid \neg\, x_i = x_j \mid R(x_{i_1}, \ldots, x_{i_n}) \mid \neg R(x_{i_1}, \ldots, x_{i_n}) \mid \\ & =\!(x_{i_1}, \ldots, x_{i_n}) \mid (\varphi \vee \varphi) \mid (\varphi \wedge \varphi) \mid \forall x \varphi \mid \exists x \varphi,\end{aligned}$$

where $R \in \tau$ is an n-ary relation symbol and $x_i, x_j, x_{i_1}, \ldots, x_{i_n} \in \{x_1, \ldots, x_k\}$.

3.2 Uniform interpretation

In this section we describe a general method of interpreting structures inside other structures with first-order formulae. We give a restricted definition of this method suitable for our needs. However, it is easy to see that this method can be straightforwardly generalized.

Definition 3.2.1. *Let σ and τ be relational vocabularies with relations of maximum arity 2. Let $\mathcal{A}$ be a nonempty class of σ-structures and $\mathcal{C}$ a nonempty class of τ-structures. Assume that there exists a surjective map*

$$F : \mathcal{C} \to \mathcal{A}$$

and an $\mathcal{FO}^2(\tau)$-formula $\varphi_{Dom}(x)$ in one free variable, x, such that for each structure $\mathfrak{B} \in \mathcal{C}$, there is a bijection f from the domain of $F(\mathfrak{B})$ to the set

$$\{\ b \in Dom(\mathfrak{B}) \mid \mathfrak{B} \models \varphi_{Dom}(b)\ \}.$$

Assume furthermore that for each binary relation symbol $R \in \sigma$ and for each unary relation symbol $P \in \sigma$, there exists an $\mathcal{FO}^2(\tau)$-formula $\varphi_R(x_1, x_2)$ and

$\varphi_P(x)$, *respectively, such that*

$$R^{F(\mathfrak{B})}(a_1, a_2) \;\Leftrightarrow\; \mathfrak{B} \models \varphi_R(f(a_1), f(a_2)) \text{ and}$$
$$P^{F(\mathfrak{B})}(a) \;\Leftrightarrow\; \mathfrak{B} \models \varphi_R(f(a))$$

for every tuple $(a_1, a_2) \in \Big(Dom\big(F(\mathfrak{B})\big)\Big)^2$ *and every element* $a \in Dom\big(F(\mathfrak{B})\big)$. *We then say that the class* $\mathcal{A}$ *is* uniformly $\mathcal{FO}^2$-interpretable in $\mathcal{C}$. *If* $\mathcal{A}$ *is a singleton class* $\{\mathfrak{A}\}$, *we say that* $\mathfrak{A}$ *is* uniformly $\mathcal{FO}^2$-interpretable in $\mathcal{C}$.

Definition 3.2.2. *Let* σ *and* τ *be relational vocabularies with relations of maximum arity* 2. *Let* $\mathcal{A}$ *be a nonempty class of* σ*-structures and* $\mathcal{C}$ *a nonempty class of* τ*-structures such that* $\mathcal{A}$ *is uniformly* $\mathcal{FO}^2$*-interpretable in* $\mathcal{C}$. *Define a map* I *from the set of* $\mathcal{FO}^2(\sigma)$*-formulae to the set of* $\mathcal{FO}^2(\tau)$*-formulae as follows.*

1. *If* $P \in \sigma$ *is a unary relation symbol then*

$$I\big(P(x)\big) := \varphi_P(x),$$

   where $\varphi_P(x)$ *is the* $\mathcal{FO}^2$*-formula for* P *witnessing the fact that* $\mathcal{A}$ *is uniformly* $\mathcal{FO}^2$*-interpretable in* $\mathcal{C}$.

2. *If* $R \in \sigma$ *is a binary relation symbol then*

$$I\big(R(x_1, x_2)\big) := \varphi_R(x_1, x_2),$$

   where $\varphi_R(x_1, x_2)$ *is the* $\mathcal{FO}^2$ *formula for* R *witnessing the fact that* $\mathcal{A}$ *is uniformly* $\mathcal{FO}^2$*-interpretable in* $\mathcal{C}$.

3. $I(x = y) := x = y$.

4. $I(\neg\varphi) := \neg I(\varphi)$.

5. $I\big((\varphi \wedge \psi)\big) := \big(I(\varphi) \wedge I(\psi)\big)$.

6. $I\big(\exists x\, \psi(x)\big) := \exists x\Big(\varphi_{Dom}(x) \wedge I\big(\psi(x)\big)\Big)$.

We call the map I *a uniform* $\mathcal{FO}^2$*-interpretation of* $\mathcal{A}$ *in* $\mathcal{C}$. *When* $\mathcal{A}$ *and* $\mathcal{C}$ *are known from the context, we may call* I *simply an* interpretation.

Lemma 3.2.3. *Let σ and τ be relational vocabularies of arity at most 2. Let $\mathcal{A}$ be a nonempty class of σ-structures and $\mathcal{C}$ a nonempty class of τ-structures. Assume that $\mathcal{A}$ is uniformly $\mathcal{FO}^2$-interpretable in $\mathcal{C}$ and let I denote a related interpretation. Let φ be an $\mathcal{FO}^2(\sigma)$-sentence. The following conditions are equivalent.*

1. *There exists a structure $\mathfrak{A} \in \mathcal{A}$ such that $\mathfrak{A} \models \varphi$.*
2. *There exists a structure $\mathfrak{B} \in \mathcal{C}$ such that $\mathfrak{B} \models I(\varphi)$.*

Proof. Straightforward. □

3.3 Expressivity

It is well known that the expressive power of both $\mathcal{D}$ and $\mathcal{IF}$ coincide with the expressive power of existential second-order logic (e.g. [Vää07, MSS11]). However, not much is known about the expressive powers of finite variable fragments of dependence logic and independence-friendly logic. In this section we study the expressive powers of these logics. We start by comparing these logics to fragments of first-order logic. We show that while one-variable fragments of $\mathcal{D}$ and $\mathcal{IF}$ are contained in first-order logic, two-variable fragments are not. We then establish a hierarchy with regard to the expressive powers of the finite variable logics $\mathcal{D}^k$ and $\mathcal{IF}^k$. We show that for $k \geq 2$

$$\mathcal{D}^k \leq \mathcal{IF}^k < \mathcal{D}^{k+1}.$$

Finally, we study the relationships between fragments of existential second-order logic and finite variable fragments of dependence logic and independence-friendly logic.

3.3.1 Fragments of first-order logic

In this section we compare the expressive powers of finite variable dependence logics and independence-friendly logics to fragments of first-order logic. We show that while the expressive power of one-variable $\mathcal{IF}$-logic coincides with that of one-variable first-order logic, the picture for one-variable dependence logic looks quite different. We show that while $\mathcal{D}^1$ is contained in $\mathcal{FO}$, for each $k \in \mathbb{N}$, there are properties definable in $\mathcal{D}^1$ that cannot be defined in $\mathcal{FO}^k$. In addition, we show that the two-variable fragments of $\mathcal{IF}$ and $\mathcal{D}$ are not contained even in monadic second-order logic. We also describe some interesting properties already definable in $\mathcal{D}^2$.

The following self-evident proposition follows directly from Proposition 2.4.5.

Proposition 3.3.1. *For every $k \in \mathbb{N}$, $\mathcal{FO}^k \leq \mathcal{D}^k$ and $\mathcal{FO}^k \leq \mathcal{IF}^k$.*

Proposition 3.3.2. $\mathcal{IF}^1 \equiv \mathcal{FO}^1$.

Proof. The direction $\mathcal{FO}^1 \leq \mathcal{IF}^1$ follows from Proposition 3.3.1. For the other direction, we will first show that the equivalence

$$\mathfrak{A} \models_X \exists x/\{x\}\, \varphi(x) \quad \Leftrightarrow \quad \mathfrak{A} \models_X \exists x\, \varphi(x)$$

holds for every model $\mathfrak{A}$, every $\mathcal{FO}^1$-formula $\varphi(x)$ and every nonempty team X of $\mathfrak{A}$ such that $\mathrm{dom}(X) \subseteq \{x\}$. Let $\varphi(x)$ and X be as above. Now, the following equivalences hold.

$$\begin{aligned}
&\mathfrak{A} \models_X \exists x/\{x\}\, \varphi(x)\\
&\quad \Leftrightarrow \mathfrak{A} \models_{X(F/x)} \varphi(x), \text{ for some } \{x\}\text{-independent function } F : X \to A\\
&\quad \Leftrightarrow \mathfrak{A} \models_{\{s\}} \varphi(x), \text{ for some assignment } s : \{x\} \to A\\
&\quad \Leftrightarrow \mathfrak{A}, s \models_{\mathcal{FO}} \varphi(x), \text{ for some assignment } s : \{x\} \to A\\
&\quad \Leftrightarrow \mathfrak{A}, s \models_{\mathcal{FO}} \exists x\, \varphi(x), \text{ for the empty assignment } s : \emptyset \to A\\
&\quad \Leftrightarrow \mathfrak{A}, s \models_{\mathcal{FO}} \exists x\, \varphi(x), \text{ for all } s \in X\\
&\quad \Leftrightarrow \mathfrak{A} \models_X \exists x\, \varphi(x).
\end{aligned}$$

The first equivalence is due to the semantics for slashed quantifiers. The second equivalence is due to the observation that, since X is nonempty, $\mathrm{dom}(X) \subseteq \{x\}$ and F is $\{x\}$-independent, then in fact $X(F/x) = \{s\}$ for some assignment $s : \{x\} \to A$. The third and the last equivalence follow from Proposition 2.4.5, since both $\varphi(x)$ and $\exists x\, \varphi(x)$ are syntactically $\mathcal{FO}$-formulae. The fourth and the fifth equivalence follow from properties of first-order logic and from the facts that $\exists x\, \varphi(x)$ is a sentence and X nonempty.

It now follows, by an easy inductive argument, that the following translation $\varphi \mapsto \varphi^*$ translates every $\mathcal{IF}^1$ sentence to an equivalent $\mathcal{FO}^1$ sentence. Hence we obtain that $\mathcal{IF}^1 \leq \mathcal{FO}^1$. The translation is defined as follows. For first-order literals the translation is the identity. The remaining cases are defined as follows:

$$\begin{aligned}
(\varphi \wedge \psi) &\mapsto (\varphi^* \wedge \psi^*),\\
(\varphi \vee \psi) &\mapsto (\varphi^* \vee \psi^*),\\
\exists x/W\varphi &\mapsto \exists x\varphi^*,\\
\forall x\varphi &\mapsto \forall x\varphi^*.
\end{aligned}$$

□

While, on the level of sentences, the expressive power of $\mathcal{IF}^1$ and $\mathcal{FO}^1$ coincide, the following proposition demonstrates that the situation regarding $\mathcal{D}^1$ is completely different.

Proposition 3.3.3. *For each $k \in \mathbb{N}$, there is a sentence φ_k in $\mathcal{D}^1$ such that*

$$\mathfrak{A} \models \varphi_k \quad \textit{iff} \quad |A| \leq k.$$

Proof. For each $k \in \mathbb{N}$, let φ_k denote the formula

$$\forall x \, (\bigvee_{1 \leq i \leq k} =(x)).$$

The following equivalences hold.

$\mathfrak{A} \models \varphi_k$

$\Leftrightarrow$ $\mathfrak{A} \models_{\{\emptyset\}(A/x)} \bigvee_{1 \leq i \leq k} =(x)$

$\Leftrightarrow$ there exist teams $X_1, \ldots, X_k$ such that $X_1 \cup \cdots \cup X_k = \{\emptyset\}(A/x)$ and such that, for every $i \leq k$, $\mathfrak{A} \models_{X_i} =(x)$

$\Leftrightarrow$ there exist teams $X_1, \ldots, X_k$ such that $X_1 \cup \cdots \cup X_k = \{\emptyset\}(A/x)$ and such that, for every $i \leq k$, $|X_i| \leq 1$

$\Leftrightarrow$ $|A| \leq k$.

Hence the claim follows. □

As an immediate corollary of Proposition 3.3.3, we obtain the following result.

Corollary 3.3.4. *For each $k \in \mathbb{N}$ it holds that $\mathcal{D}^1 \not\leq \mathcal{FO}^k$.*

Proof. The expressive power of $\mathcal{FO}^k$ can be characterized by the k-pebble game. See, e.g. [Lib04]. By using the k-pebble game, it is easy to show that on the empty vocabulary $\mathcal{FO}^k$ cannot differentiate a model of size k from a model of size $k+1$. Hence, the claim follows from Proposition 3.3.3. □

We will use the following folklore result to show that $\mathcal{D}^1 \leq \mathcal{FO}$.

Proposition 3.3.5. *Let τ be a finite unary vocabulary. Then the equivalence $\mathcal{MSO}(\tau) \equiv \mathcal{FO}(\tau)$ holds.*

Proof. The following claim is proved in the proof of [Lib04, Proposition 7.12].

If $\mathfrak{A}$ and $\mathfrak{B}$ are models of the empty vocabulary and $|A|, |B| \geq 2^k$, then for every $\mathcal{MSO}$-sentence φ of quantifier depth at most k

$$\mathfrak{A} \models \varphi \quad \textit{iff} \quad \mathfrak{B} \models \varphi.$$

The proof of [Lib04, Proposition 7.12] can be straightforwardly generalized for a proof of the following claim.

Let τ be a finite unary vocabulary. For every subset σ of τ, we define that

$$\varphi_\sigma(x) := \bigwedge_{P \in \sigma} P(x) \wedge \bigwedge_{P \in \tau \setminus \sigma} \neg P(x).$$

If $\mathfrak{A}$ and $\mathfrak{B}$ are τ-models, and $k \in \mathbb{N}$ such that for every $\sigma \subseteq \tau$ and $n < 2^k$

$$|\{a \in A \mid \mathfrak{A} \models \varphi_\sigma(a)\}| = n \quad \text{iff} \quad |\{b \in B \mid \mathfrak{B} \models \varphi_\sigma(b)\}| = n,$$

then for every $\mathcal{MSO}$-sentence φ of quantifier depth at most k

$$\mathfrak{A} \models \varphi \quad \text{iff} \quad \mathfrak{B} \models \varphi.$$

From this claim $\mathcal{MSO}(\tau) \equiv \mathcal{FO}(\tau)$ clearly follows. □

In the proof of the following theorem, we use the result of Theorem 3.3.15 from Section 3.3.3 that states that

$$\mathcal{D}^1 \leq \Sigma^1_1(\mathcal{FOC}^1).$$

By the proof of Lemma 3.3.14 from Section 3.3.3, the above result can be strengthened to the statement that

$$\mathcal{D}^1 \leq \mathcal{EMSO}(\mathcal{FOC}^1).$$

Proposition 3.3.6. *$\mathcal{D}^1 < \mathcal{FO}$.*

Proof. By proof of Lemma 3.3.14 and by Theorem 3.3.15 from Section 3.3.3, we have that

$$\mathcal{D}^1 \leq \mathcal{EMSO}(\mathcal{FOC}^1).$$

Hence, it suffices to show that

$$\mathcal{EMSO}(\mathcal{FOC}^1) < \mathcal{FO}.$$

Without loss of generality, we can restrict our attention to structures of unary vocabulary. By Proposition 3.3.5, we have that $\mathcal{EMSO} \leq \mathcal{FO}$ on unary vocabularies. Furthermore, it is well known that $\mathcal{FO} \equiv \mathcal{FOC}$. Hence

$$\mathcal{EMSO}(\mathcal{FOC}^1) \leq \mathcal{FO}$$

follows. Now, since clearly nonemptiness of a binary relation is expressible in $\mathcal{FO}$, but is not expressible in $\mathcal{EMSO}(\mathcal{FOC}^1)$, it follows that

$$\mathcal{EMSO}(\mathcal{FOC}^1) < \mathcal{FO}.$$

Therefore $\mathcal{D}^1 < \mathcal{FO}$. □

We have shown that the expressive power of one-variable dependence logic and independence-friendly logic is strictly below the expressive power of first-order logic. This however does not hold for the two-variable fragments. Next we will describe non-first-order properties that can be expressed in $\mathcal{D}^2$. In Section 3.3.2 we will show that $\mathcal{D}^2 \leq \mathcal{IF}^2$. Thus, the properties described here in $\mathcal{D}^2$ can also be expressed in $\mathcal{IF}^2$. Case 2 of the proposition below is an improved version of a result of [KKLV11]. In the earlier result one constant symbol was needed in the vocabulary.

Proposition 3.3.7. *The following properties can be expressed in $\mathcal{D}^2$.*

1. *For unary relation symbols P and Q, we can express that $|P| \leq |Q|$ and hence that $|P| = |Q|$.*
2. *We can express that the model is infinite.*

Proof. For claim 1, we will show that a model $\mathfrak{A}$ satisfies the $\mathcal{D}^2$ formula

$$\varphi := \forall x \exists y \Big(=(y, x) \wedge (\neg P(x) \vee Q(y)) \Big)$$

if and only if $|P^{\mathfrak{A}}| \leq |Q^{\mathfrak{A}}|$. Notice first that, by the semantics of universal and existential quantifiers,

$$\mathfrak{A} \models \varphi$$

if and only if there exists a function $F : \{\emptyset\}(A/x) \to A$ such that

$$\mathfrak{A} \models_{\{\emptyset\}(A/x, F/y)} \Big(=(y, x) \wedge (\neg P(x) \vee Q(y)) \Big).$$

This holds if and only if there exists an injective function $F : \{\emptyset\}(A/x) \to A$ such that

$$\mathfrak{A} \models_{\{\emptyset\}(A/x, F/y)} (\neg P(x) \vee Q(y)).$$

Moreover, this holds if and only if there exists an injective function $F\colon A \to A$ such that $F[P^{\mathfrak{A}}] \subseteq Q^{\mathfrak{A}}$. This, of course, is equivalent with the claim that $|P^{\mathfrak{A}}| \leq |Q^{\mathfrak{A}}|$.

For part 2, we use a similar idea as above. We will show that a model $\mathfrak{A}$ satisfies

$$\psi := \forall x \exists y \Big(=(y,x) \wedge \exists x (=(x) \wedge \neg x = y) \Big)$$

if and only is A is infinite. Again, notice that

$$\mathfrak{A} \models \psi$$

if and only if there exists a function $F : \{\emptyset\}(A/x) \to A$ such that

$$\mathfrak{A} \models_{\{\emptyset\}(A/x, F/y)} \Big(=(y,x) \wedge \exists x (=(x) \wedge \neg x = y) \Big).$$

This holds if and only if there exists an injective function $F : \{\emptyset\}(A/x) \to A$ such that

$$\mathfrak{A} \models_{\{\emptyset\}(A/x, F/y)} \exists x (=(x) \wedge \neg x = y).$$

Furthermore, this is equivalent to the claim that there exists an injective function $F\colon A \to A$ and $a \in A$ such that $a \notin F[A]$. Clearly this is equivalent with the claim that A is infinite. □

As an immediate consequence of Proposition 3.3.7 we obtain the following corollary. The result for $\mathcal{IF}^2$ follows from the fact that $\mathcal{D}^2 \leq \mathcal{IF}^2$ proven later on in Theorem 3.3.11 of Section 3.3.2

Corollary 3.3.8. *$\mathcal{D}^2 \not\leq \mathcal{MSO}$ and $\mathcal{IF}^2 \not\leq \mathcal{MSO}$ (and hence $\mathcal{D}^2 \not\leq \mathcal{FO}$ and $\mathcal{IF}^2 \not\leq \mathcal{FO}$).*

Proof. In the proof of [Lib04, Proposition 7.12.] it is shown that if $\mathfrak{A}$ and $\mathfrak{B}$ are models of the empty vocabulary and $|A|, |B| \geq 2^k$ then for every $\mathcal{MSO}$-sentence φ of quantifier depth at most k

$$\mathfrak{A} \models \varphi \quad \text{iff} \quad \mathfrak{B} \models \varphi.$$

Hence, infinity is not an $\mathcal{MSO}$ definable property. Therefore, the claim for $\mathcal{D}^2$ follows from Proposition 3.3.7. Moreover, since we will establish in Theorem 3.3.11 of Section 3.3.2 that $\mathcal{D}^2 \leq \mathcal{IF}^2$, the claim holds also for $\mathcal{IF}^2$. □

3.3.2 Hierarchy of expressive power

In this section we study the relationship between finite variable dependence logics and independence-friendly logics. We show that for every $k \geq 2$

$$\mathcal{D}^k \leq \mathcal{IF}^k < \mathcal{D}^{k+1}.$$

For $k = 2$, we show that also the first inclusion is strict.

Lemma 3.3.9. *For every $k \geq 2$ and every formula $\varphi \in \mathcal{D}^k$ there is a formula $\varphi^+ \in \mathcal{IF}^k$ such that for all structures $\mathfrak{A}$ and teams X of $\mathfrak{A}$, where $\mathrm{dom}(X) = \{x_1, \ldots, x_k\}$, it holds that*

$$\mathfrak{A} \models_X \varphi \quad \Leftrightarrow \quad \mathfrak{A} \models_X \varphi^+.$$

Proof. We will define a translation $tr_k : \mathcal{D}^k \mapsto \mathcal{IF}^k$ separately for each $k \geq 2$. By VAR_k we denote the set $\{x_1, \ldots, x_k\}$. The translation tr_k is defined as follows. For first-order literals the translation is the identity. The remaining cases are defined as follows:

$$\begin{aligned}
(\varphi \wedge \psi) &\mapsto \big(tr_k(\varphi) \wedge tr_k(\psi)\big),\\
(\varphi \vee \psi) &\mapsto \big(tr_k(\varphi) \vee tr_k(\psi)\big),\\
\exists x\varphi &\mapsto \exists x\, tr_k(\varphi),\\
\forall x\varphi &\mapsto \forall x\, tr_k(\varphi).
\end{aligned}$$

The dependence atom $=(y_1, \ldots, y_n, y)$ is translated to a formula

$$\exists x_j / (\mathrm{VAR}_k \setminus \{y_1, \ldots, y_n\})\, x_j = y,$$

where $j \geq 1$ is the smallest natural number such that x_j and y are distinct. The claim can now be proved by using induction on φ. The only nontrivial case is the case for dependence atoms.

Let $\varphi := =(y_1, \ldots, y_n, y)$ and let $j \geq 1$ be the smallest natural number such that $x_j \neq y$. Let us assume first that

$$\mathfrak{A} \models_X \varphi.$$

Hence there exists a function $F\colon A^n \to A$ such that

$$\text{for all } s \in X : \ s(y) = F\Big(\big(s(y_1), \ldots, s(y_n)\big)\Big). \tag{3.1}$$

Now define the function $F'\colon X \to A$ by setting that

$$F'(s) := F\Big(\big(s(y_1), \ldots, s(y_n)\big)\Big). \tag{3.2}$$

Notice that, for every $s, s' \in X$, it holds that if

$$s(y_1) = s'(y_1), \ldots, s(y_n) = s'(y_n),$$

then

$$F\Big((s(y_1), \ldots, s(y_n))\Big) = F\Big((s'(y_1), \ldots, s'(y_n))\Big)$$

and hence

$$F'(s) = F'(s').$$

Therefore F' is a $\mathrm{VAR}_k \setminus \{y_1, \ldots, y_n\}$-independent function. It remains to show that

$$\mathfrak{A} \models_{X(F'/x_j)} x_j = y. \tag{3.3}$$

Let $s \in X(F'/x_j)$. Then

$$s = s'\big(F'(s')/x_j\big), \text{ for some } s' \in X. \tag{3.4}$$

Now

$$s(x_j) \overset{(3.4)}{=} F'(s') \overset{(3.2)}{=} F\Big((s'(y_1), \ldots, s'(y_n))\Big) \overset{(3.1)}{=} s'(y) \overset{(3.4)}{=} s(y).$$

Therefore, (3.3) holds, and hence

$$\mathfrak{A} \models_X \exists x_j / (\mathrm{VAR}_k \setminus \{y_1, \ldots, y_n\})\, x_j = y.$$

Suppose then that $\mathfrak{A} \not\models_X \varphi$. Then there must exist $s, s' \in X$ such that

$$s(y_1) = s'(y_1), \ldots, s(y_n) = s'(y_n) \text{ but } s(y) \neq s'(y).$$

We claim now that

$$\mathfrak{A} \not\models_X \exists x_j / (\mathrm{VAR}_k \setminus \{y_1, \ldots, y_n\})\, x_j = y. \tag{3.5}$$

Let $F\colon X \to A$ be an arbitrary $\mathrm{VAR}_k \setminus \{y_1, \ldots, y_n\}$-independent function. Now, from $\mathrm{VAR}_k \setminus \{y_1, \ldots, y_n\}$-independence of F it follows that $F(s) = F(s')$. Since additionally $s(y) \neq s'(y)$, we have that

$$F(s) \neq s(y) \text{ or } F(s') \neq s'(y).$$

Since j was picked such that x_j and y are different variables, we conclude that

$$s(F(s)/x_j)(x_j) \neq s(F(s)/x_j)(y) \text{ or } s'(F(s')/x_j)(x_j) \neq s'(F(s')/x_j)(y).$$

Now since clearly $s\big(F(s)/x_j\big)$, $s'\big(F(s')/x_j\big) \in X(F/x_j)$, this implies that

$$\mathfrak{A} \not\models_{X(F/x_j)} x_j = y.$$

Since F was arbitrary, we may conclude that (3.5) holds. □

Next we show a translation from $\mathcal{IF}^k$ to $\mathcal{D}^{k+1}$.

Lemma 3.3.10. *For every $k \geq 1$ and every formula $\varphi \in \mathcal{IF}^k$ there exists a formula $\varphi^* \in \mathcal{D}^{k+1}$ such that for every structure $\mathfrak{A}$ and team X of $\mathfrak{A}$, where $\mathrm{dom}(X) = \{x_1, \ldots, x_k\}$, it holds that*

$$\mathfrak{A} \models_X \varphi \quad \Leftrightarrow \quad \mathfrak{A} \models_X \varphi^*.$$

Proof. Fix $k \geq 1$. The claim follows by the following translation $\varphi \mapsto \varphi^*$. For atomic and negated atomic formulae the translation is the identity, and for propositional connectives and first-order quantifiers the translation is defined in the obvious inductive way, i.e.,

$$\begin{aligned} (\varphi \wedge \psi) &\mapsto (\varphi^* \wedge \psi^*), \\ (\varphi \vee \psi) &\mapsto (\varphi^* \vee \psi^*), \\ \exists x\varphi &\mapsto \exists x\varphi^*, \\ \forall x\varphi &\mapsto \forall x\varphi^*. \end{aligned}$$

The only nontrivial case is the case for the slashed quantifiers. Remember that the variables used in the syntax of $\mathcal{IF}^k$ and $\mathcal{D}^{k+1}$ are $x_1, \ldots, x_k$ and $x_1, \ldots, x_k, x_{k+1}$, respectively. If W is a set of variables then by $W[z/x]$ we denote the set of variables in W with x replaced by z. Note that if neither x nor z is in W, then z is not in $W[z/x]$. In the following, we also use the shorthand notation $\overline{W}$ for the relative complement $\{x_1, \ldots, x_k\} \setminus W$ of W. By $\overline{W}[z/x]$, we mean the set of variables that is obtained from W by first taking the relative complement of W and then replacing the variable x by z.

The case for slashed quantifiers in the translation is the following:

$$\exists x/W\psi \quad \mapsto \quad \exists x_{k+1}\Big(x_{k+1} = x \wedge \exists x\big(=\!(\overline{W}[x_{k+1}/x], x) \wedge \psi^*\big)\Big).$$

The claim is proved by using induction on φ. The cases for atomic formulae, Boolean connectives and first-order quantifiers are trivial. We will prove here the case where φ is of the form $\exists x/W\psi$, where $x \in \{x_1, \ldots, x_k\}$ and $W \subseteq \{x_1, \ldots, x_k\}$.

Assume first that $\mathfrak{A} \models_X \varphi$. Hence there exists a W-independent function $F\colon X \to A$ such that

$$\mathfrak{A} \models_{X(F/x)} \psi. \tag{3.6}$$

By the definition of W-independence, for every $s, s' \in X$ it holds that

$$\text{if } s(z) = s'(z) \text{ holds for every } z \in \overline{W}$$

then $F(s) = F(s')$. Our goal is to show that

$$\mathfrak{A} \models_X \exists x_{k+1}\Big(x_{k+1} = x \wedge \exists x(=(\overline{W}[x_{k+1}/x], x) \wedge \psi^*)\Big). \tag{3.7}$$

Now, (3.7) holds if for $G\colon X \to A$, defined by $G(s) = s(x)$ for every $s \in X$, it holds that

$$\mathfrak{A} \models_{X(G/x_{k+1})} \exists x(=(\overline{W}[x_{k+1}/x], x) \wedge \psi^*). \tag{3.8}$$

Define the function $F'\colon X(G/x_{k+1}) \to A$ by setting that

$$F'(s) = F(s \upharpoonright \{x_1, \ldots, x_n\}).$$

We claim that

$$\mathfrak{A} \models_{X(G/x_{k+1})(F'/x)} (=(\overline{W}[x_{k+1}/x], x) \wedge \psi^*), \tag{3.9}$$

implying (3.8) and hence (3.7).

We will first show that

$$\mathfrak{A} \models_{X(G/x_{k+1})(F'/x)} =(\overline{W}[x_{k+1}/x], x). \tag{3.10}$$

At this point it is helpful to note that every $s \in X(G/x_{k+1})(F'/x)$ arises from an $s' \in X$ by first copying the value of x to x_{k+1} and then replacing the value of x by $F(s')$, i.e., that

$$s(x_{k+1}) = s'(G(s')/x_{k+1})(x_{k+1}) = G(s') = s'(x) \tag{3.11}$$

and $s(x) = F(s')$. Now, to show (3.10), let $s_1, s_2 \in X(G/x_{k+1})(F'/x)$ be such that

$$\text{for all } y \in \overline{W}[x_{k+1}/x] \text{ it holds that } s_1(y) = s_2(y). \tag{3.12}$$

We will show that

$$s_1(x) = s_2(x), \tag{3.13}$$

from which (3.10) follows. Let $s_1', s_2' \in X$ be as above, i.e., s_1 (resp. s_2) arises from s_1' (resp. s_2'). Remember that $\overline{W} \subseteq \{x_1, \ldots, x_k\}$. Clearly for every $y \in \overline{W} \cap \overline{W}[x_{k+1}/x]$ $(= \overline{W} \setminus \{x, x_{k+1}\})$

$$s_1(y) = s_1'(y) \text{ and } s_2(y) = s_2'(y).$$

Now from this, together with (3.12), it follows that

$$s_1'(y) = s_2'(y) \text{ for all } y \in \overline{W} \cap \overline{W}[x_{k+1}/x].$$

Note that, if $\overline{W} \neq \overline{W}[x_{k+1}/x]$ then $x \in \overline{W}$, $x_{k+1} \in \overline{W}[x_{k+1}/x]$ and furthermore

$$\overline{W} = (\overline{W} \cap \overline{W}[x_{k+1}/x]) \cup \{x\}.$$

Thus

$$s_1'(x) \overset{(3.11)}{=} s_1(x_{k+1}) \overset{(3.12)}{=} s_2(x_{k+1}) \overset{(3.11)}{=} s_2'(x)$$

follows from the fact that $x_{k+1} \in \overline{W}[x_{k+1}/x]$. Therefore we conclude that

$$s_1'(y) = s_2'(y) \text{ for all } y \in \overline{W}. \tag{3.14}$$

Now since F is W-independent, we have that $F(s_1') = F(s_2')$. Furthermore, since $s_1(x) = F(s_1')$ and $s_2(x) = F(s_2')$, we conclude that (3.13) holds. Therefore (3.10) holds.

Let us then show that

$$\mathfrak{A} \models_{X(G/x_{k+1})(F'/x)} \psi^*. \tag{3.15}$$

Note first that, by the definition of the mapping $\varphi \mapsto \varphi^*$, the variable x_{k+1} cannot appear free in ψ^*. By Proposition 2.4.6, the satisfaction of any $\mathcal{D}$-formula ϑ only depends on those variables in a team that appear free in ϑ. Therefore, (3.15) holds if and only if

$$\mathfrak{A} \models_{X(G/x_{k+1})(F'/x)\restriction\{x_1,\dots,x_k\}} \psi^*. \tag{3.16}$$

We have chosen G and F' in such a way that

$$X(G/x_{k+1})(F'/x) \restriction \{x_1, \dots, x_k\} = X(F/x),$$

hence (3.16) now follows from (3.6) and the induction hypothesis. From (3.10) and (3.16) we conclude that (3.9) holds, from which (3.7) finally follows.

Assume then that $\mathfrak{A} \models_X \varphi^*$. Hence for $G\colon X \to A$, defined by $G(s) = s(x)$ for all $s \in X$, it holds that

$$\mathfrak{A} \models_{X(G/x_{k+1})} \exists x\big(=(\overline{W}[x_{k+1}/x], x) \wedge \psi^*\big). \tag{3.17}$$

Therefore, there exists a function $F\colon X(G/x_{k+1}) \to A$ such that

$$\mathfrak{A} \models_{X(G/x_{k+1})(F/x)} \big(=(\overline{W}[x_{k+1}/x], x) \wedge \psi^*\big). \tag{3.18}$$

Define the function $F'\colon X \to A$ such that

$$F'(s) = F\Big(s\big(G(s)/x_{k+1}\big)\Big). \tag{3.19}$$

We will show that F' is W-independent and that

$$\mathfrak{A} \models_{X(F'/x)} \psi. \tag{3.20}$$

From this

$$\mathfrak{A} \models_X \exists x/W\psi$$

follows. We will first show that F' is W-independent. Let $s_1, s_2 \in X$ be such that $s_1(y) = s_2(y)$ for all $y \in \overline{W}$. We need to show that

$$F'(s_1) = F'(s_2).$$

By (3.19), this is equivalent to showing that

$$F\Big(s_1\big(G(s_1)/x_{k+1}\big)\Big) = F\Big(s_2\big(G(s_2)/x_{k+1}\big)\Big). \tag{3.21}$$

Now let $s_1', s_2' \in X(G/x_{k+1})(F/x)$ be the assignments that arise from s_1 and s_2 by (re)evaluating x_{k+1} and x according to G and F, respectively. Note that s_1' and s_2' are obtained from s_1 and s_2 by first copying the value for x to x_{k+1} and then reevaluating x according to F, respectively. Hence

$$s_1'(y) = s_2'(y) \text{ for all } y \in \overline{W}[x_{k+1}/x].$$

Therefore, from (3.18), it follows that $s_1'(x) = s_2'(x)$. Now, since in fact

$$s_1'(x) = F\Big(s_1\big(G(s_1)/x_{k+1}\big)\Big) \text{ and } s_2'(x) = F\Big(s_2\big(G(s_2)/x_{k+1}\big)\Big)$$

we conclude that (3.21) holds and hence F' is W-independent.

To prove (3.20), notice first the following two facts: Firstly, by the definition of the mapping $\varphi \mapsto \varphi^*$, the variable x_{k+1} cannot appear free in ψ^*. Hence, by Proposition 2.4.6,

$$\mathfrak{A} \models_{X(G/x_{k+1})(F/x)} \psi^* \quad \text{iff} \quad \mathfrak{A} \models_{X(G/x_{k+1})(F/x)\restriction\{x_1,\dots,x_k\}} \psi^*.$$

Secondly, notice that

$$X(G/x_{k+1})(F/x) \restriction \{x_1, \dots, x_k\} = X(F'/x).$$

Hence

$$\mathfrak{A} \models_{X(G/x_{k+1})(F/x)} \psi^* \quad \text{iff} \quad \mathfrak{A} \models_{X(F'/x)} \psi^*. \tag{3.22}$$

Now from (3.18), (3.22) and the induction hypothesis we conclude that (3.20) holds. □

For sentences Lemmas 3.3.9 and 3.3.10 imply the following.

Theorem 3.3.11. *For every $k \geq 2$, $\mathcal{D}^k \leq \mathcal{IF}^k < \mathcal{D}^{k+1}$.*

Proof. Fix $k \geq 2$. When φ is a $\mathcal{D}^k$-formula, we denote by φ^+ the related $\mathcal{IF}^k$-formula of Lemma 3.3.9. Similarly, when φ is an $\mathcal{IF}^k$-formula, we denote by φ^* the related $\mathcal{D}^{k+1}$-formula of Lemma 3.3.10. Note first that, by Propositions 2.4.6 and 2.4.7, for every $\mathcal{IF}$- and $\mathcal{D}$-sentence φ, every model $\mathfrak{A}$ and every nonempty team X of $\mathfrak{A}$

$$\mathfrak{A} \models_X \varphi \quad \text{iff} \quad \mathfrak{A} \models_{\{\emptyset\}} \varphi. \tag{3.23}$$

It is important to note that even if $\varphi \in \mathcal{D}^k$ is a sentence, it may happen that the translation φ^+ has free variables. This is due to the fact that the variables in W are regarded as free in subformulae of φ^+ of the form $\exists x/W\psi$. Similarly if $\varphi \in \mathcal{IF}^k$ is a sentence, it may happen that the $\mathcal{D}^{k+1}$-formula φ^* has free variables. However, we will show that this is not a problem.

We will first show that $\mathcal{D}^k \leq \mathcal{IF}^k$. Let $\varphi \in \mathcal{D}^k$ be a sentence and Y be the set of all assignments of $\mathfrak{A}$ with the domain $\{x_1, \ldots, x_k\}$. Now

$$\begin{aligned} \mathfrak{A} \models_{\{\emptyset\}} \varphi \quad &\overset{(3.23)}{\Leftrightarrow} \quad \mathfrak{A} \models_Y \varphi \\ &\overset{\text{Lemma 3.3.9}}{\Leftrightarrow} \quad \mathfrak{A} \models_Y \varphi^+ \\ &\Leftrightarrow \quad \mathfrak{A} \models_{\{\emptyset\}} \forall x_1 \ldots \forall x_k \, \varphi^+, \end{aligned}$$

where the last equivalence holds by the semantics of universal quantifiers.

The proof for $\mathcal{IF}^k \leq \mathcal{D}^{k+1}$ is completely analogous. Let $\varphi \in \mathcal{IF}^k$ and Y be the set of all assignments of $\mathfrak{A}$ with the domain $\{x_1, \ldots, x_k\}$. Now

$$\begin{aligned} \mathfrak{A} \models_{\{\emptyset\}} \varphi \quad &\overset{(3.23)}{\Leftrightarrow} \quad \mathfrak{A} \models_Y \varphi \\ &\overset{\text{Lemma 3.3.10}}{\Leftrightarrow} \quad \mathfrak{A} \models_Y \varphi^* \\ &\Leftrightarrow \quad \mathfrak{A} \models_{\{\emptyset\}} \forall x_1 \ldots \forall x_k \varphi^*, \end{aligned}$$

where the last equivalence holds by the semantics of universal quantifiers. Note that, by the definition of the translation $\varphi \mapsto \varphi^*$ of Lemma 3.3.10, the variable x_{k+1} cannot appear free in the formula φ^*. Hence $\forall x_1 \ldots \forall x_k \varphi^*$ is surely a $\mathcal{D}^{k+1}$-sentence. Now $\mathcal{IF}^k < \mathcal{D}^{k+1}$ since, e.g. non-emptiness of a $(k+1)$-ary relation is not expressible in $\mathcal{IF}^k$ and it is clearly expressible in $\mathcal{D}^{k+1}$. □

The following proposition follows by deeper results of Sections 3.5 and 3.8 concerning a decidability barrier between two-variable dependence logic and

independence-friendly logic. We will later show that while the satisfiability problem and the finite satisfiability problem for $\mathcal{D}^2$ are decidable, for $\mathcal{IF}^2$ both of these problems are undecidable.

Proposition 3.3.12. *$\mathcal{D}^2 < \mathcal{IF}^2$. This holds already in the finite.*

Proof. By Theorem 3.3.11, we know that $\mathcal{D}^2 \leq \mathcal{IF}^2$. Hence, to proof the claim it is enough to show that the following two conditions hold.

1. There exists an $\mathcal{IF}^2$-sentence such that there does not exist any equivalent $\mathcal{D}^2$-sentence.

2. There exists an $\mathcal{IF}^2$-sentence φ such that there does not exist any $\mathcal{D}^2$-sentence ψ such that φ is equivalent to ψ in the class of all finite models.

The property of being grid-like (see Definition 3.7.6) can be expressed in $\mathcal{IF}^2$ but not in $\mathcal{D}^2$. It is easy to see, from the proof of Theorem 3.8.2, that if there would be a $\mathcal{D}^2$-sentence equivalent to the $\mathcal{IF}^2$-sentence $\varphi_{gridlike}$ then the satisfiability problem for $\mathcal{D}^2$ would be undecidable. Since, by Theorem 3.5.4, the satisfiability problem for $\mathcal{D}^2$ is decidable there cannot be any $\mathcal{D}^2$-sentence equivalent to $\varphi_{gridlike}$.

Analogously, in the finite there exists no $\mathcal{D}^2$-sentence equivalent to the $\mathcal{IF}^2$-sentence $\varphi_{\text{fingridlike}}$ (see Section 3.7.2). It is easy to see, from the proof of Theorem 3.8.4, that if there would be a $\mathcal{D}^2$-sentence equivalent to $\varphi_{fingridlike}$, in the finite, then the finite satisfiability problem for $\mathcal{D}^2$ would be undecidable. Since, by Theorem 3.5.4, the finite satisfiability problem for $\mathcal{D}^2$ is decidable there cannot be any $\mathcal{D}^2$-sentence equivalent to the $\mathcal{IF}^2$-sentence $\varphi_{fingridlike}$, in the finite. □

Interestingly, while for every $k \geq 2$, it holds that $\mathcal{D}^k \leq \mathcal{IF}^k$, for $k = 1$, the reverse holds.

Proposition 3.3.13. *$\mathcal{IF}^1 < \mathcal{D}^1$.*

Proof. By Proposition 3.3.1 and Corollary 3.3.4, we have that $\mathcal{FO}^1 < \mathcal{D}^1$. Now since by Proposition 3.3.2 we have that $\mathcal{IF}^1 \equiv \mathcal{FO}^1$, we conclude that $\mathcal{IF}^1 < \mathcal{D}^1$. □

3.3.3 Fragments of existential second-order logic

In this section we study the relationship of finite variable dependence logics and independence-friendly logics with fragments of existential second-order logic.

Väänänen established in [Vää07] that every sentence of dependence logic can be translated into an equivalent sentence of existential second-order logic. Using the translation of Väänänen, it can be shown that each sentence of $\mathcal{D}^k$ can be translated into an equivalent sentence of $\Sigma^1_1(\mathcal{FO}^{2k})$. However, this translation is not designed to minimize the number of distinct first-order variables used. We show that by optimizing the use of variables, a translation can be made into $\Sigma^1_1(\mathcal{FO}^{k+1})$. In addition, we will show that by using first-order counting quantifiers even this small increase of needed first-order variables is eliminated, i.e., we show that for each $\mathcal{D}^k$-sentence there exists an equivalent sentence in $\Sigma^1_1(\mathcal{FOC}^k)$. This translation is optimal since clearly $\Sigma^1_1(\mathcal{FOC}^{k-1})$ does not suffice, e.g nonemptiness of a k-ary relation can be expressed in $\mathcal{D}^k$ but not in $\Sigma^1_1(\mathcal{FOC}^{k-1})$. Theorem 3.3.11 implies the related results also for finite variable independence-friendly logics.

We will first show how to translate $\mathcal{D}^k$-sentences into equivalent sentences of $\Sigma^1_1(\mathcal{FOC}^k)$. We will then modify this translation to show how to translate $\mathcal{D}^k$-sentences into equivalent sentences of $\Sigma^1_1(\mathcal{FO}^{k+1})$. Notice that the counting quantifier $\exists^{\leq 1}$ (or in fact the expression $\neg\exists^{\geq 2}$) is the only counting quantifier used in the translation below.

Lemma 3.3.14. *Assume that $k \geq 1$. Let τ be a relational vocabulary and let $R \notin \tau$ be a k-ary relation symbol. For every formula $\varphi \in \mathcal{D}^k(\tau)$ there exists a sentence*

$$\varphi^* \in \Sigma^1_1(\mathcal{FOC}^k)(\tau \cup \{R\})$$

such that for every model $\mathfrak{A}$ and team X of $\mathfrak{A}$ with $\mathrm{dom}(X) = \{x_1, \ldots, x_k\}$

$$\mathfrak{A} \models_X \varphi \quad \textit{iff} \quad (\mathfrak{A}, \mathrm{rel}(X)) \models \varphi^*, \tag{3.24}$$

where $(\mathfrak{A}, \mathrm{rel}(X))$ is the expansion $\mathfrak{A}'$ of $\mathfrak{A}$ into vocabulary $\tau \cup \{R\}$ defined by

$$R^{\mathfrak{A}'} := \mathrm{rel}(X)^1.$$

Proof. The proof is a modification of the proof of [Vää07, Theorem 6.2]. Fix $k \geq 1$. We will define a translation

$$tr_k : \mathcal{D}^k(\tau) \mapsto \Sigma^1_1(\mathcal{FOC}^k)(\tau \cup \{R\})$$

inductively. Below we always assume that the quantified relations S and T are fresh, i.e., they are assumed not to appear in $tr_k(\psi)$ or $tr_k(\vartheta)$. Notice that for every $\mathcal{D}^k$-formula φ

$$tr_k(\varphi) = \exists S_1 \ldots \exists S_n \varphi',$$

[1] By $\mathrm{rel}(X)$, we mean the canonical representation of the team X as a relation, i.e., $(a_1, \ldots, a_n) \in \mathrm{rel}(X)$ if and only if $s \in X$, where $s(x_i) = a_i$, for every $1 \leq i \leq n$.

for some k-ary relation variables $S_1 \dots S_n$, $n \in \mathbb{N}$, and for some syntactically $\mathcal{FOC}^k$-formula φ'. The translation tr_k is defined as follows.

1. If φ is a literal then $tr_k(\varphi)$ is defined as
$$\forall x_1 \dots \forall x_k \big(R(x_1, \dots, x_k) \to \varphi(x_1, \dots, x_k)\big).$$
2. Assume that φ is a dependence atom $=(\vec{x}, x_i)$, where $\vec{x} \in \{x_1, \dots, x_k\}^t$ and $t \in \mathbb{N}$.
   (a) If x_i occurs in $\vec{x}$ then $tr_k(\varphi)$ is defined as
   $$\exists x_1 (x_1 = x_1).$$
   (b) If x_i does not occur in $\vec{x}$ then $tr_k(\varphi)$ is defined as
   $$\forall \vec{x}\, \exists^{\leq 1} x_i\, \exists \vec{z}\, R(x_1, \dots, x_k),$$
   where $\vec{z}$ is exactly the sequence of variables in $\{x_1, \dots, x_k\} \setminus \{x_i\}$ (in alphabetical order) that do not occur in $\vec{x}$.
3. Assume that $tr_k(\psi) = \exists S_1 \dots \exists S_n \psi'$ and $tr_k(\vartheta) = \exists T_1 \dots \exists T_m \vartheta'$, where ψ' and ϑ' are syntactically $\mathcal{FOC}^k$-formulae. Furthermore, assume that $S_1, \dots S_n, T_1, \dots, T_m$ are all distinct.
   (a) If φ is of the form $(\psi \vee \vartheta)$, then $tr_k(\varphi)$ is defined as
   $$\exists S \exists T \exists S_1 \dots \exists S_n \exists T_1 \dots \exists T_m \Big(\forall x_1 \dots \forall x_k \Big(R(x_1, \dots, x_k) \to \big(S(x_1, \dots, x_k) \vee T(x_1, \dots, x_k)\big)\Big) \wedge \psi'(S/R) \wedge \vartheta'(T/R)\Big).$$
   (b) If φ is of the form $(\psi \wedge \vartheta)$, then $tr_k(\varphi)$ is defined as
   $$\exists S_1 \dots \exists S_n \exists T_1 \dots \exists T_m (\psi' \wedge \vartheta').$$
4. If φ is of the form $\exists x_i \psi$ and $tr_k(\psi) = \exists S_1 \dots \exists S_n \psi'$, where ψ' is syntactically an $\mathcal{FOC}^k$-formula, then $tr_k(\varphi)$ is defined as
$$\exists S \exists S_1 \dots \exists S_n \Big(\forall x_1 \dots \forall x_k \big(R(x_1, \dots, x_k) \to \exists x_i S(x_1, \dots, x_k)\big) \wedge \psi'(S/R)\Big).$$
5. If φ is of the form $\forall x_i \psi$ and $tr_k(\psi) = \exists S_1 \dots \exists S_n \psi'$, where ψ' is syntactically an $\mathcal{FOC}^k$-formula, then $tr_k(\varphi)$ is defined as
$$\exists S \exists S_1 \dots \exists S_n \Big(\forall x_1 \dots \forall x_k \big(R(x_1, \dots, x_k) \to \forall x_i S(x_1, \dots, x_k)\big) \wedge \psi'(S/R)\Big).$$

A straightforward induction on φ shows that for every model $\mathfrak{A}$ and every team X of $\mathfrak{A}$ such that $\mathrm{dom}(X) = \{x_1, \ldots, x_k\}$, we have that

$$\mathfrak{A} \models_X \varphi \quad \text{iff} \quad (\mathfrak{A}, \mathrm{rel}(X)) \models tr_k(\varphi).$$

We will proof here the cases 2b and 4. Assume that φ is $=(\vec{x}, x_i)$, $\vec{x} \in \{x_1, \ldots, x_k\}^t$ and $t \in \mathbb{N}$. Furthermore, assume that x_i does not occur in $\vec{x}$. Let $\vec{z}$ denote the sequence of variables in $\{x_1, \ldots, x_k\} \setminus \{x_i\}$ that do not occur in the tuple $\vec{z}$. Let $\mathfrak{A}$ be a model and X a team of $\mathfrak{A}$ with domain $\{x_1, \ldots, x_m\}$. The claim follows from the following equivalences.

$$\begin{aligned}
\mathfrak{A} \models_X =(\vec{x}, x_i) \quad &\Leftrightarrow \quad \text{For every } s_1, s_2 \in X\colon\ s_1(\vec{x}) = s_2(\vec{x}) \Rightarrow s_1(x_i) = s_2(x_i).\\
&\Leftrightarrow \quad \text{For every assignment } s \text{ defined by } s(\vec{x}) = \vec{a} \text{ there exists}\\
&\qquad\quad \text{at most one } b \in A \text{ such that there exists an expansion}\\
&\qquad\quad \text{of the assignment } s(b/x_i) \text{ in } X.\\
&\Leftrightarrow \quad (\mathfrak{A}, \mathrm{rel}(X)) \models \forall \vec{x}\, \exists^{\leq 1} x_i\, \exists \vec{z}\, R(x_1, \ldots, x_k).
\end{aligned}$$

For the proof of case 4, assume that φ is $\exists x_i \psi$ and $tr_k(\psi) = \exists S_1 \ldots \exists S_n \psi'$, where ψ' is syntactically an $\mathcal{FOC}^k$-formula. Let $\mathfrak{A}$ be a model and X a team of $\mathfrak{A}$ with domain $\{x_1, \ldots, x_k\}$. The claim follows by the following equivalences.

$$\begin{aligned}
\mathfrak{A} \models_X \exists x_i \psi \quad &\Leftrightarrow \quad \mathfrak{A} \models_{X(F/x_i)} \psi, \text{ for some function } F : X \to A.\\
&\Leftrightarrow \quad (\mathfrak{A}, \mathrm{rel}(X(F/x_i))) \models tr_k(\psi), \text{ for some function } F : X \to A.\\
&\Leftrightarrow \quad (\mathfrak{A}, \mathrm{rel}(X)) \models \exists S \Big(\forall x_1 \ldots \forall x_k (R(x_1, \ldots, x_k)\\
&\qquad\qquad\qquad \to \exists x_i S(x_1, \ldots, x_k)) \wedge tr_k(\psi)(S/R) \Big).\\
&\Leftrightarrow \quad (\mathfrak{A}, \mathrm{rel}(X)) \models \exists S \Big(\forall x_1 \ldots \forall x_k (R(x_1, \ldots, x_k)\\
&\qquad\qquad\qquad \to \exists x_i S(x_1, \ldots, x_k)) \wedge \exists S_1 \ldots \exists S_n \psi'(S/R) \Big).\\
&\Leftrightarrow \quad (\mathfrak{A}, \mathrm{rel}(X)) \models \exists S \exists S_1 \ldots \exists S_n \Big(\forall x_1 \ldots \forall x_k (R(x_1, \ldots, x_k)\\
&\qquad\qquad\qquad \to \exists x_i S(x_1, \ldots, x_k)) \wedge \psi'(S/R) \Big).
\end{aligned}$$

The first equivalence is due to the semantics of existential quantifiers and the second follows from the induction hypothesis. The third equivalence follows from the definition of $X(F/x_i)$. The fourth and the last equivalence is trivial. □

Theorem 3.3.15. *For every $k \geq 1$ it holds that $\mathcal{D}^k \leq \Sigma^1_1(\mathcal{FOC}^k)$.*

Proof. Let τ be a relational vocabulary and $k \geq 1$. Let φ be a $\mathcal{D}^k(\tau)$-sentence and

$$\varphi^* = \exists R_1, \dots R_n \psi$$

the related $\Sigma^1_1(\mathcal{FOC}^k)(\tau \cup \{R\})$-sentence given by Lemma 3.3.14. The following conditions are equivalent.

1. $\mathfrak{A} \models \varphi$.
2. $\mathfrak{A} \models_X \varphi$ for some team X such that $\mathrm{dom}(X) = \{x_1, \dots, x_k\}$.
3. $(\mathfrak{A}, \mathrm{rel}(X)) \models \varphi^*$ for some team X such that $\mathrm{dom}(X) = \{x_1, \dots, x_k\}$.
4. $(\mathfrak{A}, R) \models \exists R_1 \dots \exists R_n (\exists x_1 \dots \exists x_k R(x_1, \dots, x_k) \wedge \psi)$ for some nonempty $R \subseteq A^k$.
5. $\mathfrak{A} \models \exists R \exists R_1 \dots \exists R_n (\exists x_1 \dots \exists x_k R(x_1, \dots, x_k) \wedge \psi)$.

The equivalence of 1 and 2 follows from Proposition 2.4.6 and the fact that $\mathrm{fr}(\varphi) = \emptyset$. By Lemma 3.3.14, conditions 2 and 3 are equivalent. The equivalence of 3 and 4 follows from the fact that the relations $\mathrm{rel}(X)$ with $\mathrm{dom}(X) = \{x_1, \dots, x_k\}$ are exactly nonempty subsets of A^k. The conditions 4 are 5 clearly equivalent. □

Hence, by Theorem 3.3.11, we obtain the following corollary. Note that the strictness of the inclusion follows due to the fact that $\mathcal{IF}^{k+1}$ can, and $\mathcal{IF}^k$ cannot, express nonemptiness of a $(k+1)$-ary relation.

Corollary 3.3.16. *For every $k \geq 1$ it holds that $\mathcal{IF}^k < \Sigma^1_1(\mathcal{FOC}^{k+1})$.*

The following theorem follows from the proofs of Lemma 3.3.14 and Theorem 3.3.15 with a small modification. It also follows from the proof of [Vää07, Theorem 6.2] after a small adjustment.

Theorem 3.3.17. *For every $k \geq 1$ it holds that $\mathcal{D}^k < \Sigma^1_1(\mathcal{FO}^{k+1})$.*

Proof. For each $k \geq 1$, the translation

$$tr_k : \mathcal{D}^k(\tau) \mapsto \Sigma^1_1(\mathcal{FOC}^k)(\tau \cup \{R\})$$

defined in the proof of Lemma 3.3.14 can be easily modified to a translation

$$tr'_k : \mathcal{D}^k(\tau) \mapsto \Sigma^1_1(\mathcal{FO}^{k+1})(\tau \cup \{R\})$$

by replacing case 2b of the translation tr_k by the following case 2b′.

2. Assume that φ is a dependence atom $=(\vec{x}, x_i)$, where $\vec{x} \in \{x_1, \ldots, x_k\}^t$ and $t \in \mathbb{N}$.

   (b′) If x_i does not occur in $\vec{x}$ then $tr_k(\varphi)$ is defined as

$$\forall \vec{x} \forall x_i \forall x_{k+1} \Big((\exists \vec{z}\, R(x_1, \ldots, x_k) \\ \wedge \exists \vec{z}\, R(x_1, \ldots, x_{i-1}, x_{k+1}, x_{i+1}, \ldots, x_k)) \rightarrow x_i = x_{k+1} \Big),$$

   where $\vec{z}$ is exactly the sequence of variables in $\{x_1, \ldots, x_k\} \setminus \{x_i\}$ (in alphabetical order) that do not occur in $\vec{x}$.

Clearly the proofs of Lemma 3.3.14 and Theorem 3.3.15 can be straightforwardly modified to handle the translation tr'_k. □

Hence, by Theorem 3.3.11, we obtain the following corollary.

Corollary 3.3.18. *For every $k \geq 1$ it holds that $\mathcal{IF}^k < \Sigma^1_1(\mathcal{FO}^{k+2})$.*

3.4 Finite model property and zero-one law

In this section we will show that the separation between one-variable and higher-variable logics, already manifested in Section 3.3.1, also manifests itself in regards to the zero-one law and the finite model property. We show that the logics $\mathcal{IF}^k$ and $\mathcal{D}^k$ have the zero-one law and the finite model property if and only if $k = 1$.

We start by giving the definitions of the zero-one law and the finite model property. We then recall the related folklore results concerning first-order logic.

Definition 3.4.1. *Let τ be a finite relational vocabulary and $\mathcal{L}(\tau)$ a logic. We denote by $\mathrm{Str}_n(\tau)$ the class of all τ structures with domain $\{1, \ldots, n\}$. We say that $\mathcal{L}(\tau)$ has the zero-one law if for every $\mathcal{L}(\tau)$-formula φ the asymptotic probability that φ is true in a model is 0 or 1, i.e., if*

$$\lim_{n \to \infty} \frac{|\{\mathfrak{A} \in \mathrm{Str}_n(\tau) \mid \mathfrak{A} \models \varphi\}|}{|\{\mathfrak{A} \in \mathrm{Str}_n(\tau)\}|} \in \{0, 1\}.$$

Definition 3.4.2. *Let $\mathcal{L}$ be a logic. We say that $\mathcal{L}$ has the finite model property if every $\mathcal{L}$-sentence φ that has a model has a finite model.*

We now recall the related folklore results concerning first-order logic.

Theorem 3.4.3 ([GKLT69, Fag76])**.** *First-order logic has the zero-one law.*

Proof. For a proof see, e.g. [EF99]. □

Theorem 3.4.4 ([Mor75])**.** *$\mathcal{FO}^2$ has the finite model property.*

Proof. For a proof see, e.g. [EF99]. □

The following proposition is a self-evident corollary of Theorem 3.4.4.

Proposition 3.4.5. *$\Sigma^1_1(\mathcal{FO}^2)$ has the finite model property.*

Proof. Let τ be a relational vocabulary and let φ be a $\Sigma^1_1(\mathcal{FO}^2)(\tau)$-sentence. We will show that if φ has a model then is has a finite model. Assume that there exists a model $\mathfrak{A}$ of vocabulary τ such that

$$\mathfrak{A} \models \varphi.$$

Since φ is a $\Sigma^1_1(\mathcal{FO}^2)(\tau)$-sentence

$$\varphi = \exists X_1 \dots \exists X_k \, \psi,$$

for some relation variables $X_1, \dots, X_k$ and some syntactically $\mathcal{FO}^2$-formula ψ. Hence

$$\mathfrak{A} \models \exists X_1 \dots \exists X_k \psi.$$

Let $R_1, \dots, R_k \notin \tau$ be fresh relation symbols with arities corresponding to $X_1, \dots, X_k$, respectively. Then clearly

$$\mathfrak{A}^* \models \psi(R_1/X_1, \dots, R_k/X_k),$$

for some expansion $\mathfrak{A}^*$ of $\mathfrak{A}$ to the vocabulary $\tau \cup \{R_1, \dots, R_k\}$. Since clearly $\psi(R_1/X_1, \dots, R_k/X_k)$ is an $\mathcal{FO}^2(\tau \cup \{R_1, \dots, R_k\})$-sentence, by Theorem 3.4.4, there exists a finite $\tau \cup \{R_1, \dots, R_k\}$-model $\mathfrak{B}$ such that

$$\mathfrak{B} \models \psi(R_1/X_1, \dots, R_k/X_k).$$

Hence, for the τ reduct $\mathfrak{B}^*$ of $\mathfrak{B}$, it holds that $\mathfrak{B}^* \models \varphi$. Since $\mathfrak{B}^*$ is a finite τ-model, we are done. □

We are now ready to show that the k-variable fragments of $\mathcal{IF}$ and $\mathcal{D}$ have the zero-one law and the finite model property only when $k = 1$.

Proposition 3.4.6. *Let $k \geq 1$. The logics $\mathcal{IF}^k$ and $\mathcal{D}^k$ have the zero-one law if and only if $k = 1$.*

Proof. By Theorem 3.4.3, first-order logic has the zero-one law. Furthermore, by Propositions 3.3.2 and 3.3.6, we have that $\mathcal{D}^1 \leq \mathcal{FO}$ and $\mathcal{IF}^1 \leq \mathcal{FO}$. Hence it follows that both $\mathcal{D}^1$ and $\mathcal{IF}^1$ also have the zero-one law. Claim 1 of Proposition 3.3.7 with Theorem 3.3.11 implies that neither $\mathcal{D}^2$ nor $\mathcal{IF}^2$ has the zero-one law. This is due to the fact that the $\mathcal{D}^2$ and $\mathcal{IF}^2$ expressible property $|P^{\mathfrak{A}}| \leq \frac{1}{2}|A|$ (i.e., $|P^{\mathfrak{A}}| \leq |(\neg P)^{\mathfrak{A}}|$) has the limit probability $\frac{1}{2}$. Now since neither $\mathcal{IF}^2$ nor $\mathcal{D}^2$ has the zero-one law it follows that, for all $k \geq 2$, neither $\mathcal{IF}^k$ nor $\mathcal{D}^k$ has the zero-one law. □

Proposition 3.4.7. *Let $k \geq 1$. The logics $\mathcal{IF}^k$ and $\mathcal{D}^k$ have the finite model property if and only if $k = 1$.*

Proof. By Theorem 3.3.17, each sentence of $\mathcal{D}^1$ has an equivalent sentence in $\Sigma^1_1(\mathcal{FO}^2)$. By Proposition 3.4.5, $\Sigma^1_1(\mathcal{FO}^2)$ has the finite model property, and hence $\mathcal{D}^1$ has the finite model property. Similarly since by Proposition 3.3.2 $\mathcal{IF}^1 \equiv \mathcal{FO}^1$, also $\mathcal{IF}^1$ has the finite model property. As observed in Proposition 3.3.7, there is a $\mathcal{D}^2(\emptyset)$-sentence expressing that the domain of the model is infinite. Hence, for every $k \geq 2$, it holds that $\mathcal{D}^k$ does not have the finite model property. By Theorem 3.3.11, this also holds for $\mathcal{IF}^k$, for every $k \geq 2$. □

3.5 Satisfiability of finite variable dependence logics

In this section we study the complexity of the satisfiability problem and the finite satisfiability problem for finite variable fragments of dependence logic. We will show that for $\mathcal{D}^1$ both of these problems are NP-complete, while for $\mathcal{D}^2$ these problems are NEXPTIME-complete. For $k \geq 3$, we will show that the satisfiability problem for $\mathcal{D}^k$ is Π^0_1-complete, whereas the finite satisfiability problem for $\mathcal{D}^k$ is Σ^0_1-complete. Our proofs rely on the translations from $\mathcal{D}^k$ into $\Sigma^1_1(\mathcal{FOC}^k)$ described in Section 3.3.3 and the corresponding complexity results for the k-variable first-order logic and the k-variable first-order logic with counting.

We start by stating some well known complexity results from the literature.

Theorem 3.5.1 ([Coo71, PH08])**.** SAT($\mathcal{FO}^1$), FINSAT($\mathcal{FO}^1$), SAT($\mathcal{FOC}^1$) *and* FINSAT($\mathcal{FOC}^1$) *are all* NP*-complete.*

Theorem 3.5.2 ([GKV97, PH05])**.** SAT($\mathcal{FO}^2$), FINSAT($\mathcal{FO}^2$), SAT($\mathcal{FOC}^2$) *and* FINSAT($\mathcal{FOC}^2$) *are all* NEXPTIME*-complete.*

Theorem 3.5.3. *The following hold for every $k \geq 3$.*

1. $\textsc{Sat}(\mathcal{FO}^k)$ *and* $\textsc{Sat}(\mathcal{FOC}^k)$ *are both* Π^0_1*-complete.*

2. $\textsc{FinSat}(\mathcal{FO}^k)$ *and* $\textsc{FinSat}(\mathcal{FOC}^k)$ *are both* Σ^0_1*-complete.*

Proof. For a proof see, e.g [BGG97]. □

We will then use the connections between finite variable dependence logics and fragments of first-order and second-order logic, from Section 3.3, together with the results of Theorems 3.5.1, 3.5.2 and 3.5.3 to prove the corresponding complexity results for finite variable dependence logics.

Theorem 3.5.4.

1. $\textsc{Sat}(\mathcal{D}^1)$ *and* $\textsc{FinSat}(\mathcal{D}^1)$ *are* NP*-complete.*

2. $\textsc{Sat}(\mathcal{D}^2)$ *and* $\textsc{FinSat}(\mathcal{D}^2)$ *are* NEXPTIME*-complete.*

3. $\textsc{Sat}(\mathcal{D}^k)$ *is* Π^0_1*-complete and* $\textsc{FinSat}(\mathcal{D}^k)$ *is* Σ^0_1*-complete, for* $k \geq 3$.

Proof. By Proposition 3.3.1, $\mathcal{FO}^k \leq \mathcal{D}^k$, for every $k \geq 1$. Hence, the corresponding hardness result for each claim follows from the corresponding hardness results for $\textsc{Sat}(\mathcal{FO}^k)$ and $\textsc{FinSat}(\mathcal{FO}^k)$, i.e., Theorems 3.5.1, 3.5.2 and 3.5.3, and the fact that the translation from $\mathcal{FO}^k$ to $\mathcal{D}^k$ is polynomial (in fact it is the identity function).

The corresponding inclusions are due to the translations from $\mathcal{D}^k$ into $\Sigma^1_1(\mathcal{FOC}^k)$ described in Section 3.3.3. Let $\varphi \in \mathcal{D}^k(\tau)$ be a sentence. By Theorem 3.3.15, there exists an equivalent $\Sigma^1_1(\mathcal{FOC}^k)(\tau)$-sentence φ'. Now since φ' is a $\Sigma^1_1(\mathcal{FOC}^k)(\tau)$-sentence, it is of the form

$$\exists R_1 \dots \exists R_k \psi,$$

where ψ is some $\mathcal{FOC}^k(\tau \cup \{R_1, \dots, R_k\})$-sentence. Note that we identify the second-order variables $R_1, \dots, R_k$ with second-order relation symbols with corresponding arities. The following are clearly equivalent.

1. The $\mathcal{D}^k(\tau)$-sentence φ is (finitely) satisfiable.

2. The $\Sigma^1_1(\mathcal{FOC}^k)(\tau)$-sentence φ' is (finitely) satisfiable.

3. The $\mathcal{FOC}^k(\tau \cup \{R_1, \dots, R_k\})$-sentence ψ is (finitely) satisfiable.

Now since the transformation from φ to ψ is clearly computable in polynomial time, see the translation used in Lemma 3.3.14, we conclude by Theorem 3.5.1 that

$$\textsc{Sat}(\mathcal{D}^1), \textsc{FinSat}(\mathcal{D}^1) \in \text{NP},$$

by Theorem 3.5.2 that

$$\textsc{Sat}(\mathcal{D}^2), \textsc{FinSat}(\mathcal{D}^2) \in \text{NEXPTIME},$$

and by Theorem 3.5.3 that for every $k \geq 3$

$$\textsc{Sat}(\mathcal{D}^k) \in \Pi^0_1 \text{ and } \textsc{FinSat}(\mathcal{D}^k) \in \Sigma^0_1.$$

□

3.6 Undecidability via tiling

In this section we introduce structures and methods that we will later employ to prove undecidability of the satisfiability and the finite satisfiability problem for two-variable independence-friendly logic. We start by defining some important structures used in the proofs later on. We then introduce a powerful and flexible method of proving undecidability called the tiling method. The advantage of the tiling method is its adaptiveness to accommodate multitude of logics with ease. Finally, we describe two undecidable problems called the tiling problem and the finite tiling problem.

We start by defining structures and classes of structures utilized in the tiling method. *The grid* is the structure $\mathfrak{G} = (\mathbb{N}^2, V, H)$, where

$$\begin{aligned} V &= \{((i,j),(i,j+1)) \in \mathbb{N}^2 \times \mathbb{N}^2 \mid i,j \in \mathbb{N}\} \text{ and} \\ H &= \{((i,j),(i+1,j)) \in \mathbb{N}^2 \times \mathbb{N}^2 \mid i,j \in \mathbb{N}\}. \end{aligned}$$

Recall that, for every $k \in \mathbb{N}$, $k = \{0, 1, \ldots, k-1\}$. For each $n, m \in \mathbb{N}$, we call the structure $\mathfrak{m} \times \mathfrak{n} = (m \times n, V^{\mathfrak{m}\times\mathfrak{n}}, H^{\mathfrak{m}\times\mathfrak{n}})$, where

$$\begin{aligned} V^{\mathfrak{m}\times\mathfrak{n}} &= \{((i,j),(i,j+1)) \in (m \times n)^2 \mid i < m,\, j < (n-1)\} \text{ and} \\ H^{\mathfrak{m}\times\mathfrak{n}} &= \{((i,j),(i+1,j)) \in (m \times n)^2 \mid i < (m-1),\, j < n\}, \end{aligned}$$

the $m \times n$ grid. A structure is called *a finite grid* if it is an $m \times n$ grid for some $m, n \in \mathbb{N}$.

Let T be a set of unary predicate symbols. We call an expansion $\mathfrak{G}'$ of $\mathfrak{G}$ to the vocabulary $\{V, H\} \cup T$ a T-labelled grid, if each point u of $\mathfrak{G}'$ belongs

to the extension of exactly one predicate symbol $P \in T$. Analogously, we say that a structure

$$\mathfrak{A} = (A, V, H, \{P\}_{P \in T})$$

is a T-labelled finite grid if the $\{V, H\}$ reduct of $\mathfrak{A}$ is a finite grid and if each point in A belongs to the extension of exactly one predicate symbol $P \in T$. By $\mathcal{G}_T$ and $\mathcal{F}_T$, we denote the class of all T-labelled grids and the class of all T-labelled finite grids, respectively.

Definition 3.6.1. *Let R and S be binary relation symbols and let T be a set of unary predicate symbols.*

- *We say that a structure $\mathfrak{A} = (A, R, S)$ is a* supergrid *if there is a homomorphism from the grid $\mathfrak{G}$ to $\mathfrak{A}$.*
- *We say that a structure $\mathfrak{B} = (B, R, S)$ is a* superfingrid *if there is a component C of $\mathfrak{B}$ such that $\mathfrak{B} \restriction C$ is isomorphic to some finite grid. By a component, we mean some minimal nonempty subset C of B such that if a pair (a, b) is in either of the relations $R^{\mathfrak{B}}$ or $S^{\mathfrak{B}}$ then a is in C if and only if b is in C.*
- *We call the expansion $\mathfrak{A}'$ of $\mathfrak{A}$ to the vocabulary $\{R, S\} \cup T$ a* T-labelled supergrid *if $\mathfrak{A}$ is a supergrid and each point u of $\mathfrak{A}'$ belongs to the extension of exactly one predicate symbol $P \in T$.*
- *We call the expansion $\mathfrak{B}'$ of $\mathfrak{B}$ to the vocabulary $\{R, S\} \cup T$ a* T-labelled superfingrid *if $\mathfrak{B}$ is a superfingrid and each point u of $\mathfrak{B}'$ belongs to the extension of exactly one predicate symbol $P \in T$.*

We will now describe the tiling method. A function $t : 4 \longrightarrow \mathbb{N}$ is called a *tile type*. Define the set

$$\text{TILES} \;:=\; \{\; P_t \mid t \text{ is a tile type} \;\}$$

of unary relation symbols. The unary relation symbols in the set TILES are called *tiles*. The numbers $t(i)$ of a tile P_t are the *colours* of P_t. The number $t(0)$ is the *top colour*, $t(1)$ the *right colour*, $t(2)$ the *bottom colour*, and $t(3)$ the *left colour* of P_t.

Let T be a finite nonempty set of tiles and V and H binary relation symbols. We say that a structure $\mathfrak{A} = (A, V, H)$ is *T-tilable*, if there exists an expansion of $\mathfrak{A}$ to the vocabulary $\{H, V\} \cup \{\; P_t \mid P_t \in T \;\}$ such that the following conditions hold for each $u, v \in A$.

1. The point u belongs to the extension of exactly one symbol P_t in T.
2. If uHv, $P_t(u)$ and $P_s(v)$ hold then the right colour of the tile P_t is the same as the left colour of the tile P_s.
3. If uVv, $P_t(u)$ and $P_s(v)$ hold then the top colour of the tile P_t is the same as the bottom colour of the tile P_s.

Let T be a finite nonempty set of tiles and $P_b, P_f \in T$. We say that a finite structure $\mathfrak{A}$ of vocabulary $\{V, H\}$ is (T, b, f)-*tilable*, if there exists an expansion of $\mathfrak{A}$ to the vocabulary $\{H, V\} \cup \{\ P_t \mid P_t \in T\ \}$ such that the following conditions hold for each $u, v \in A$.

1. The point u belongs to the extension of exactly one symbol P_t in T.
2. The point u belongs to the extension of P_b if there does not exist a point w such that wHu or wVu holds.
3. The point u belongs to the extension of P_f if there does not exist a point w such that uHw or uVw holds.
4. If uHv, $P_t(u)$ and $P_s(v)$ hold then the right colour of the tile P_t is the same as the left colour of the tile P_s.
5. If uVv, $P_t(u)$ and $P_s(v)$ hold then the top colour of the tile P_t is the same as the bottom colour of the tile P_s.

We are now ready to define two undecidable problems called the tiling problem and the finite tiling problem. Let $\mathcal{F}$ denote the set of finite, nonempty subsets of TILES, and let

$$\mathcal{S} \ := \ \{\ (T, b, f) \mid T \in \mathcal{F},\ P_b, P_f \in T\ \}.$$

We then define that

$$\begin{aligned} \mathcal{T} \ &:= \{\ T \in \mathcal{F} \ \mid \ \mathfrak{G} \text{ is } T\text{-tilable}\ \}, \\ \mathcal{FT} \ &:= \{\ (T, b, f) \in \mathcal{S} \ \mid \ \text{there exists a finite grid } \mathfrak{m} \times \mathfrak{n} \text{ such that} \\ &\qquad\qquad\qquad\qquad \mathfrak{m} \times \mathfrak{n} \text{ is } (T, b, f)\text{-tilable}\}. \end{aligned}$$

The *tiling problem* is the membership problem of the set $\mathcal{T}$ with the input set $\mathcal{F}$. The *finite tiling problem* is the membership problem of the set $\mathcal{FT}$ with the input set $\mathcal{S}$.

Theorem 3.6.2 ([Ber66])**.** *The tiling problem is Π^0_1-complete.*

Theorem 3.6.3 ([LPRT95])**.** *The finite tiling problem is Σ^0_1-complete.*

Lemma 3.6.4. *There is a computable function associating each input T to the tiling problem with an $\mathcal{FO}^2$-sentence φ_T of vocabulary $\tau := \{H, V\} \cup T$ such that for every structure $\mathfrak{A}$ of vocabulary $\{H, V\}$ it holds that the structure $\mathfrak{A}$ is T-tilable iff there exists an expansion $\mathfrak{A}^*$ of $\mathfrak{A}$ to the vocabulary τ such that $\mathfrak{A}^* \models \varphi_T$.*

Proof. Straightforward. □

Lemma 3.6.5. *There is a computable function associating each input (T, b, f) to the finite tiling problem with an $\mathcal{FO}^2$-sentence $\varphi_{(T,b,f)}$ of vocabulary $\tau := \{H, V\} \cup T$ such that for every finite structure $\mathfrak{A}$ of vocabulary $\{V, H\}$ it holds that, the structure $\mathfrak{A}$ is (T, b, f)-tilable iff there exists an expansion $\mathfrak{A}^*$ of $\mathfrak{A}$ to the vocabulary τ such that $\mathfrak{A}^* \models \varphi_{(T,b,f)}$.*

Proof. Straightforward. □

It is easy to see that the grid $\mathfrak{G}$ is T-tilable iff there exists a supergrid that is T-tilable. Likewise, there exists a finite grid that is (T, b, f)-tilable iff there exists a superfingrid that is (T, b, f)-tilable.

3.7 Encoding supergrids and superfingrids

In this section we show how to encode tiled supergrids and superfingrids in structures with just one binary relation. We show that for every finite set of tiles T there exists an $\mathcal{IF}^2$-definable class of structures $\mathcal{C}_T$ of vocabulary $\{R\}$ such that a sufficiently rich class of T-tiled supergrids is uniformly $\mathcal{FO}^2$-interpretable in $\mathcal{C}_T$. Likewise, we show that for every finite set of tiles T there exists an $\mathcal{IF}^2$-definable class of structures $\mathcal{E}_T$ of vocabulary $\{R\}$ such that a sufficiently rich class of T-tiled superfingrids is uniformly $\mathcal{FO}^2$-interpretable in $\mathcal{E}_T$. The structure of this section is the following. We start by defining some auxiliary formulae used in the encodings later on. In Section 3.7.1 we show how to encode supergrids in structures with just one binary relation. In Section 3.7.2 we modify this encoding to handle superfingrids. In Section 3.7.3 we show how the encodings of Sections 3.7.1 and 3.7.2 can be expanded to manage tiled supergrids and superfingrids. Finally in Section 3.7.4 we show that our encodings work as planned.

We start by defining some auxiliary formulae used later on in this section. First note that a binary relation R can be partitioned in three subrelations R_s, R_d and R_l such that

$$\begin{aligned} R_s &= \{(a,b) \in R \mid (b,a) \in R \text{ and } a \neq b\}, \\ R_d &= \{(a,b) \in R \mid (b,a) \notin R\}, \\ R_l &= \{(a,b) \in R \mid a = b\}. \end{aligned}$$

Note also that each of these relations can be defined from R by an $\mathcal{FO}^2$-formula. We define the following abbreviations:

$$\begin{aligned} \varphi_{directed}(x,y) &:= R(x,y) \wedge \neg R(y,x), \\ \varphi_{symmetric}(x,y) &:= R(x,y) \wedge R(y,x) \wedge x \neq y. \end{aligned}$$

The formula $\varphi_{directed}$ defines the directed part R_d of R, while $\varphi_{symmetric}$ defines the symmetric part R_s of R. There is no need to define an abbreviation for a formula defining R_l, since R_l is definable from R by a simple atomic formula $R(x,x)$.

We will next define the formulae we use to encode the domain, and the horizontal and the vertical relations of supergrids and superfingrids. Define that

$$\begin{aligned} \varphi_{gridpoint}(x) &:= \exists y\big(R(y,y) \wedge \varphi_{symmetric}(x,y)\big) \\ &\quad \wedge \exists y\big(\neg R(y,y) \wedge \varphi_{symmetric}(x,y)\big), \\ \varphi_{horizontal}(x,y) &:= \varphi_{directed}(x,y) \wedge \big(R(x,x) \leftrightarrow R(y,y)\big), \\ \varphi_{vertical}(x,y) &:= \varphi_{directed}(x,y) \wedge \big(R(x,x) \leftrightarrow \neg R(y,y)\big). \end{aligned}$$

The idea here is that the directed part of the relation R is used to encode both the horizontal relation and the vertical relation of supergrids and superfingrids, reflexive loops are used to differentiate the relations. Points of the domain are encoded as three point gadgets, see Figure 3.1. Figure 3.2 is an illustration of the grid $\mathfrak{G} = (\mathbb{N}^2, V, H)$, while Figure 3.3 illustrates an encoding of the grid by one binary relation.

Definition 3.7.1. *Let R be a binary relation symbol, $\mathfrak{A}$ a structure of vocabulary $\{R\}$ and $u, v \in A$. We say that u is a gridpoint of $\mathfrak{A}$ if*

$$\varphi_{gridpoint}(u) \text{ holds in } \mathfrak{A}.$$

We say that v is a vertical successor *of u and that u is a* vertical predecessor *of v in $\mathfrak{A}$ if*

$$\varphi_{vertical}(u,v) \text{ holds in } \mathfrak{A}.$$

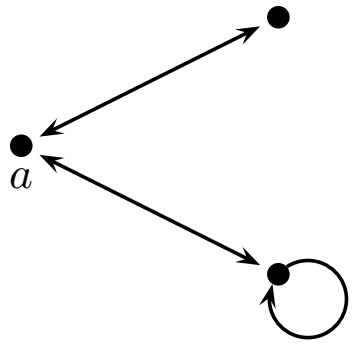

Figure 3.1: Illustration of a gadget encoding a gridpoint a. Here a is a point satisfying $\varphi_{gridpoint}(a)$.

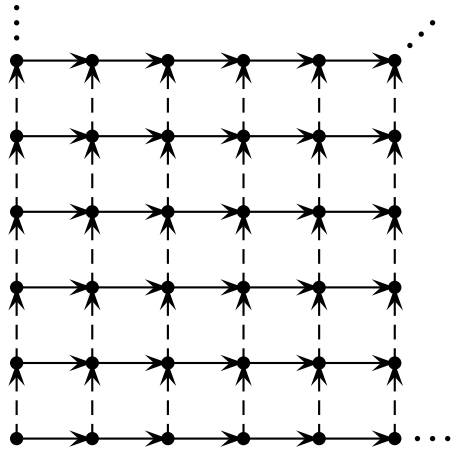

Figure 3.2: Illustration of the grid. The dashed lines correspond to the vertical relation V while the continuous lines correspond to the horizontal relation H.

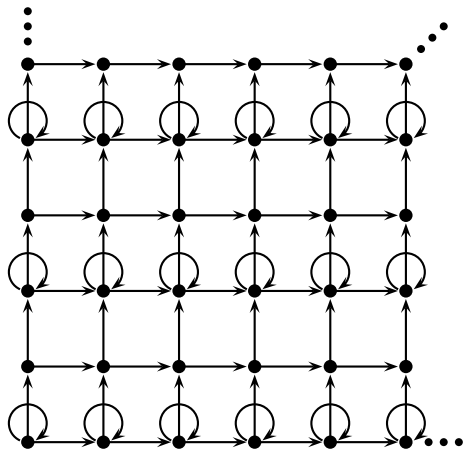

Figure 3.3: Illustration of an encoding of the grid. The lines connecting points with loops to points with loops, and points without loops to points without loops correspond to the horizontal relation H. The lines connecting points with loops to points without loops, and vice versa, correspond to the vertical relation V.

Analogously, we say that v is a horizontal successor *of u and that u is a* horizontal predecessor *of v in $\mathfrak{A}$ if*

$$\varphi_{horizontal}(u, v) \text{ holds in } \mathfrak{A}.$$

We call the relations

$$\{(u, v) \in A \times A \mid \ v \text{ is a vertical successor of } u \text{ in } \mathfrak{A}\}$$
$$\{(u, v) \in A \times A \mid \ v \text{ is a horizontal successor of } u \text{ in } \mathfrak{A}\}$$

the vertical and horizontal successor relation *of $\mathfrak{A}$, respectively. Note that we are not asserting (yet) that the relations actually are successor relations.*

3.7.1 Gridlike structures

We will now introduce the structures we call gridlike structures. Gridlike structures will be used to encode supergrids. We first give an informal description, a formal definition is given after the informal one.

Description 3.7.2. *Informally, we call a structure $\mathfrak{A} = (A, R)$ gridlike if it satisfies the following intuitive statements.*

1. *There exists exactly one gridpoint $u \in A$ that has neither vertical nor horizontal predecessors. We call this point the origo.*
2. *If a point $u \in A$ has a vertical successor v and a horizontal successor w then v and w are distinct.*
3. *The horizontal and vertical relations of $\mathfrak{A}$ are partial injections.*
4. *Each gridpoint has a horizontal and a vertical successor.*
5. *Each horizontal successor and vertical successor of a gridpoint is a gridpoint.*
6. *If a point $u \in A$ does not have a vertical predecessor then its horizontal successor does not have a vertical predecessor.*
7. *If a point $u \in A$ does not have a horizontal predecessor then its vertical successor does not have a horizontal predecessor.*
8. *Every gridpoints horizontal successors vertical successor is also its vertical successors horizontal successor.*

We will show how to formalize these conditions one by one. First note that, by the definitions of $\varphi_{horizontal}$ and $\varphi_{vertical}$, condition 2 always holds. Define then that

$$\varphi_{functional}(X) := \forall x \forall y (X(x,y) \to \exists x / \{y\}\, x = y) \text{ and}$$
$$\varphi_{injective}(X) := \forall x \forall y (X(y,x) \to \exists x / \{y\}\, x = y).$$

It is straightforward to check that the formulae $\varphi_{functional}(X)$ and $\varphi_{injective}(X)$ assert that the interpretation of the binary relation X is functional and injective, respectively. Hence the formula

$$\varphi_{injection}(X) := \varphi_{functional}(X) \wedge \varphi_{injective}(X)$$

asserts that the interpretation of the relation X is a partial injection. Define then that

$$\varphi_{HVinjection} := \varphi_{injection}(\varphi_{horizontal}/X) \wedge \varphi_{injection}(\varphi_{vertical}/X),$$

where $\varphi_{injection}(\varphi_{horizontal}/X)$ and $\varphi_{injection}(\varphi_{vertical}/X)$ are the formulae obtained from $\varphi_{injection}(X)$ by substituting each occurrence of $X(x,y)$ by the formula $\varphi_{horizontal}(x,y)$ and $\varphi_{vertical}(x,y)$, respectively. Clearly

$$\mathfrak{A} \models \varphi_{HVinjection}$$

if and only if condition 3 of Description 3.7.2 holds in $\mathfrak{A}$. Define then that

$$\varphi_{origo}(x) := \varphi_{gridpoint}(x) \wedge \forall y(\neg\varphi_{horizontal}(y,x) \wedge \neg\varphi_{vertical}(y,x)),$$
$$\varphi_{1-origo} := \exists x\Big(\varphi_{origo}(x) \wedge \forall y(x \neq y \rightarrow \neg\varphi_{origo}(y))\Big).$$

It is easy to see that

$$\mathfrak{A} \models \varphi_{1-origo}$$

if and only if condition 1 of Description 3.7.2 holds in $\mathfrak{A}$. Now let

$$\varphi_{gridinfinity} := \forall x\Big(\varphi_{gridpoint}(x) \rightarrow \Big(\exists y(\varphi_{horizontal}(x,y) \wedge \varphi_{gridpoint}(y)) \wedge \exists y(\varphi_{vertical}(x,y) \wedge \varphi_{gridpoint}(y))\Big)\Big).$$

Notice that if the horizontal and vertical successor relations are partial injections, i.e., if condition 3 of Description 3.7.2 holds in $\mathfrak{A}$, then

$$\mathfrak{A} \models \varphi_{gridinfinity}$$

if and only if conditions 4 and 5 of Description 3.7.2 hold in $\mathfrak{A}$. Conditions 6 and 7 of Description 3.7.2 are also straightforward to formalize. Let

$$\varphi_{borders} := \varphi_{W-border} \wedge \varphi_{S-border},$$

where $\varphi_{W-border}$ is a shorthand notation for the formula

$$\forall x\Big(\exists y(\varphi_{horizontal}(y,x) \wedge \forall x\neg\varphi_{vertical}(x,y)) \rightarrow \forall y\neg\varphi_{vertical}(y,x)\Big)$$

and $\varphi_{S-border}$ is a shorthand notation for the formula

$$\forall x\Big(\exists y(\varphi_{vertical}(y,x) \wedge \forall x\neg\varphi_{horizontal}(x,y)) \rightarrow \forall y\neg\varphi_{horizontal}(y,x)\Big).$$

It is easy to check that

$$\mathfrak{A} \models \varphi_{borders}$$

if and only if conditions 6 and 7 of Description 3.7.2 hold in $\mathfrak{A}$.

We have now formalized conditions 1-7 of Description 3.7.2. It is straightforward to check that each of the formulae above can be also written in $\mathcal{D}^2$. However we will show that condition 8 of Description 3.7.2 can be formalized in $\mathcal{IF}^2$ but not in $\mathcal{D}^2$. We will show that this small difference in the expressive power of these logics yields a massive difference in the complexity of these logics. We will see that while the satisfiability problem for $\mathcal{D}^2$ is decidable, the

problem for $\mathcal{IF}^2$ is not. The bottom reason for this is that $\mathcal{D}^2$ cannot express condition 8 of Description 3.7.2.

Define that

$$\varphi_{almostgridlike} := \varphi_{1-origo} \wedge \varphi_{HVinjection} \wedge \varphi_{gridinfinity} \wedge \varphi_{borders}.$$

Definition 3.7.3. *We call a structure* $\mathfrak{A} = (A, R)$ almost gridlike *if*

$$\mathfrak{A} \models \varphi_{almostgridlike}.$$

Lemma 3.7.4. *A structure* $\mathfrak{A} = (A, R)$ *is an almost gridlike structure if and only if conditions 1 - 7 of Description 3.7.2 hold in* $\mathfrak{A}$.

Proof. Straightforward by the arguments given above. □

We will next show how condition 8 can be formalized in $\mathcal{IF}^2$. Define that

$$\begin{aligned}
\varphi_{evenjoin}(x) :=& R(x,x) \wedge \forall y\Big(\big(\varphi_{horizontal}(x,y) \vee \varphi_{vertical}(x,y)\big) \\
&\to \exists x/\{y\}\,\big(\neg R(x,x) \wedge \big(\varphi_{horizontal}(y,x) \vee \varphi_{vertical}(y,x)\big)\big)\Big), \\
\varphi_{oddjoin}(x) :=& \neg R(x,x) \wedge \forall y\Big(\big(\varphi_{horizontal}(x,y) \vee \varphi_{vertical}(x,y)\big) \\
&\to \exists x/\{y\}\,\big(R(x,x) \wedge \big(\varphi_{horizontal}(y,x) \vee \varphi_{vertical}(y,x)\big)\big)\Big), \\
\varphi_{join} :=& \forall x\Big(\varphi_{gridpoint}(x) \to \big(\varphi_{evenjoin}(x) \vee \varphi_{oddjoin}(x)\big)\Big).
\end{aligned}$$

Lemma 3.7.5. *Let* $\mathfrak{A} = (A, R)$ *be an almost gridlike structure. Then*

$$\mathfrak{A} \models \varphi_{join}$$

if and only if condition 8 of Description 3.7.2 holds in $\mathfrak{A}$.

Proof. By Lemma 3.7.4, conditions 1-7 of Description 3.7.2 hold in every gridlike structure. Since $\varphi_{gridpoint}(x)$ is a first-order formula, by Proposition 2.4.5, φ_{join} is equivalent to the formula

$$\begin{aligned}
&\forall x\Big(\varphi_{gridpoint}(x) \\
&\quad\to \Big(\big(\varphi_{gridpoint}(x) \wedge \varphi_{evenjoin}(x)\big) \vee \big(\varphi_{gridpoint}(x) \wedge \varphi_{oddjoin}(x)\big)\Big)\Big).
\end{aligned} \tag{3.25}$$

Notice then that when restricted to teams with domain $\{x\}$, the $\mathcal{IF}^2$-formula

$$\varphi_{gridpoint}(x) \wedge \varphi_{evenjoin}(x)$$

is equivalent to the first-order formula

$$\varphi_{gridpoint}(x) \wedge R(x,x) \wedge \exists x' \forall y \Big(\big(\varphi_{horizontal}(x,y) \vee \varphi_{vertical}(x,y)\big) \rightarrow \big(\neg R(x',x') \wedge (\varphi_{horizontal}(y,x') \vee \varphi_{vertical}(y,x'))\big)\Big). \tag{3.26}$$

Now, by conditions 2, 3 and 4 of Description 3.7.2, the formula

$$\begin{aligned} &\varphi_{gridpoint}(x) \wedge R(x,x) \wedge \exists x' \exists y_1 \exists y_2 \\ &\Big(y_1 \neq y_2 \wedge \varphi_{horizontal}(x,y_1) \wedge \varphi_{vertical}(x,y_2) \wedge \neg R(x',x') \\ &\quad \wedge (\varphi_{horizontal}(y_1,x') \vee \varphi_{vertical}(y_1,x')) \\ &\quad \wedge (\varphi_{horizontal}(y_2,x') \vee \varphi_{vertical}(y_2,x'))\Big) \end{aligned} \tag{3.27}$$

is equivalent to the formula (3.26) in the class of almost gridlike structures. Due to condition 3, neither

$$\varphi_{horizontal}(y_1,x') \wedge \varphi_{horizontal}(y_2,x') \text{ nor } \varphi_{vertical}(y_1,x') \wedge \varphi_{vertical}(y_2,x')$$

can be true in an almost gridlike structure if $y_1 \neq y_2$. Hence, the formula (3.27) is equivalent to the formula

$$\begin{aligned} &\varphi_{gridpoint}(x) \wedge R(x,x) \wedge \exists x' \exists y_1 \exists y_2 \Big(y_1 \neq y_2 \wedge \neg R(x',x') \\ &\wedge \Big(\big(\varphi_{horizontal}(x,y_1) \wedge \varphi_{vertical}(x,y_2) \wedge \varphi_{horizontal}(y_1,x') \wedge \varphi_{vertical}(y_2,x')\big) \\ &\vee \big(\varphi_{horizontal}(x,y_1) \wedge \varphi_{vertical}(x,y_2) \wedge \varphi_{vertical}(y_1,x') \wedge \varphi_{horizontal}(y_2,x')\big)\Big)\Big) \end{aligned} \tag{3.28}$$

in the class of almost gridlike structures. Now recall the definition of $\varphi_{horizontal}$. If $R(x,x)$ and $\neg R(x',x')$ hold then

$$\varphi_{horizontal}(x,y) \wedge \varphi_{horizontal}(y,x')$$

cannot hold, for any y. Therefore, the formula (3.28) is equivalent to the formula

$$\begin{aligned} &\varphi_{gridpoint}(x) \wedge R(x,x) \wedge \exists x' \exists y_1 \exists y_2 \big(y_1 \neq y_2 \wedge \neg R(x',x') \\ &\wedge \varphi_{horizontal}(x,y_1) \wedge \varphi_{vertical}(x,y_2) \wedge \varphi_{vertical}(y_1,x') \wedge \varphi_{horizontal}(y_2,x')\big). \end{aligned} \tag{3.29}$$

Due to condition 2 of Description 3.7.2 and the definitions of $\varphi_{horizontal}$ and $\varphi_{vertical}$, we conclude that the simple formula

$$\begin{aligned} \varphi_{evenjoin^*}(x) := \varphi_{gridpoint}(x) \wedge R(x,x) \wedge \exists x' \exists y_1 \exists y_2 \big(&\varphi_{horizontal}(x,y_1) \\ &\wedge \varphi_{vertical}(x,y_2) \wedge \varphi_{vertical}(y_1,x') \wedge \varphi_{horizontal}(y_2,x')\big) \end{aligned}$$

is equivalent to (3.29) in the class of almost gridlike structures. Therefore, the formula

$$\varphi_{gridpoint}(x) \wedge \varphi_{evenjoin}(x)$$

is equivalent to $\varphi_{evenjoin^*}$ in the class of almost gridlike structures and when restricted to teams with domain $\{x\}$. By an analogous argument, the formula

$$\varphi_{gridpoint}(x) \wedge \varphi_{oddjoin}(x)$$

is equivalent to the formula

$$\begin{aligned}\varphi_{oddjoin^*}(x) :=\ & \varphi_{gridpoint}(x) \wedge \neg R(x,x) \wedge \exists x' \exists y_1 \exists y_2 (\varphi_{horizontal}(x,y_1) \\ & \wedge \varphi_{vertical}(x,y_2) \wedge \varphi_{vertical}(y_1,x') \wedge \varphi_{horizontal}(y_2,x'))\end{aligned}$$

in the class of almost gridlike structures and when restricted to teams with domain $\{x\}$. Hence, by the above equivalences, (3.25) is equivalent to

$$\forall x \Big(\varphi_{gridpoint}(x) \to (\varphi_{evenjoin^*}(x) \vee \varphi_{oddjoin^*}(x)) \Big),$$

in the class of almost gridlike structures. Since the above formula is first-order it follows, by Proposition 2.4.5, that it is equivalent to

$$\begin{aligned}\forall x \Big(& \varphi_{gridpoint}(x) \to \exists x' \exists y_1 \exists y_2 (\varphi_{horizontal}(x,y_1) \wedge \varphi_{vertical}(x,y_2) \\ & \wedge \varphi_{vertical}(y_1,x') \wedge \varphi_{horizontal}(y_2,x')) \Big),\end{aligned} \tag{3.30}$$

Thus (3.30) is equivalent to φ_{join} in the class of almost gridlike structures. From this the claim clearly follows. □

We are now ready to give a formal definition of a gridlike structure. Let

$$\varphi_{gridlike} := \varphi_{almostgridlike} \wedge \varphi_{join}.$$

Definition 3.7.6. *Let $\mathfrak{A}$ be a structure of vocabulary $\{R\}$. We say that $\mathfrak{A}$ is a gridlike structure if*

$$\mathfrak{A} \models \varphi_{gridlike}.$$

Lemma 3.7.7. *A structure $\mathfrak{A} = (A, R)$ is gridlike if and only if the conditions of Description 3.7.2 hold in $\mathfrak{A}$.*

Proof. Follows directly from Lemmas 3.7.4 and 3.7.5. □

3.7.2 Fingridlike structures

In this section we define structures we will later use to encode superfingrids. We name these structures as fingridlike structures. This sections is a modification of Section 3.7.1.

We will first give an informal description of fingridlike structures. A formal definition is given after the informal one.

Description 3.7.8. *Informally, we call a finite structure $\mathfrak{A} = (A, R)$ fingridlike if it satisfies the following intuitive statements.*

1. *There exists exactly one gridpoint $u \in A$ that has neither vertical nor horizontal predecessors. We call this point the origo of $\mathfrak{A}$.*

2. *If a point $u \in A$ has a vertical successor v and a horizontal successor w then u and w are distinct.*

3. *The horizontal and vertical relations of $\mathfrak{A}$ are partial injections.*

4. *Every horizontal successor and vertical successor of a gridpoint is a gridpoint.*

5. *If a point does not have a vertical predecessor then its horizontal successor does not have a vertical predecessor.*

6. *If a point does not have a horizontal predecessor then its vertical successor does not have a horizontal predecessor.*

7. *If a point does not have a vertical successor then its horizontal successor does not have a vertical successor.*

8. *If a point does not have a horizontal successor then its vertical successor does not have a horizontal successor.*

9. *If a gridpoint u has a vertical successor v and a horizontal successor v' then there exist a point w such that w is a horizontal successor of v and a vertical successor of v'.*

We will make use of abbreviations defined in Section 3.7.1, i.e., $\varphi_{gridpoint}$, $\varphi_{vertical}$, $\varphi_{1-origo}$, $\varphi_{HVinjection}$, $\varphi_{W-border}$, and $\varphi_{S-border}$. We define that

$$\varphi_{finborders} := \varphi_{W-border} \wedge \varphi_{N-border} \wedge \varphi_{E-border} \wedge \varphi_{S-border},$$

where $\varphi_{E-border}$ is a shorthand notation for the formula

$$\forall x\Big(\exists y(\varphi_{horizontal}(y,x) \wedge \forall x\neg\varphi_{vertical}(y,x)) \rightarrow \forall y\neg\varphi_{vertical}(x,y)\Big)$$

and $\varphi_{N-border}$ is a shorthand notation for the formula

$$\forall x\Big(\exists y(\varphi_{vertical}(y,x) \wedge \forall x\neg\varphi_{horizontal}(y,x)) \rightarrow \forall y\neg\varphi_{horizontal}(x,y)\Big).$$

Furthermore, let

$$\varphi_{gridsuccessor} := \forall x\forall y\Big(\varphi_{gridpoint}(x) \wedge (\varphi_{horizontal}(x,y) \vee \varphi_{vertical}(x,y))\Big) \rightarrow \varphi_{gridpoint}(y)$$

and define the abbreviation

$$\varphi_{almostfingridlike} := \varphi_{1-origo} \wedge \varphi_{HVinjection} \wedge \varphi_{finborders} \wedge \varphi_{gridsuccessor}.$$

Definition 3.7.9. *We say that a finite structure $\mathfrak{A} = (A, R)$ is* almost fingrid-like *if*

$$\mathfrak{A} \models \varphi_{almostfingridlike}.$$

Lemma 3.7.10. *A finite structure $\mathfrak{A} = (A, R)$ is almost fingridlike if and only if it satisfies conditions 1-8 of Description 3.7.8.*

Proof. First note that the following equivalences carry over from Section 3.7.1.

Condition 1 of Description 3.7.8 holds	iff	$\mathfrak{A} \models \varphi_{1-origo}$.
Condition 3 of Description 3.7.8 holds	iff	$\mathfrak{A} \models \varphi_{HVinjection}$.

Note also that condition 2 of Description 3.7.8 holds always. It is straightforward to check that condition 4 of Description 3.7.8 holds if and only if

$$\mathfrak{A} \models \varphi_{gridsuccessor}.$$

Likewise, it is easy to check that

$$\mathfrak{A} \models \varphi_{finborders}$$

if and only if conditions 5 - 8 of Description 3.7.8 hold in $\mathfrak{A}$. Hence the claim holds. □

Next we define a variant of φ_{join} suitable for the finite case. Recall that

$$\begin{aligned}\varphi_{evenjoin}(x) :=& R(x,x) \wedge \forall y\Big((\varphi_{horizontal}(x,y) \vee \varphi_{vertical}(x,y)) \\ &\to \exists x/\{y\}\,(\neg R(x,x) \wedge (\varphi_{horizontal}(y,x) \vee \varphi_{vertical}(y,x)))\Big), \\ \varphi_{oddjoin}(x) :=& \neg R(x,x) \wedge \forall y\Big((\varphi_{horizontal}(x,y) \vee \varphi_{vertical}(x,y)) \\ &\to \exists x/\{y\}\,(R(x,x) \wedge (\varphi_{horizontal}(y,x) \vee \varphi_{vertical}(y,x)))\Big).\end{aligned}$$

Define then that

$$\begin{aligned}\varphi_{finjoin} := \forall x\Big((\varphi_{gridpoint}(x) \wedge \exists y\,\varphi_{horizontal}(x,y) \wedge \exists y\,\varphi_{vertical}(x,y)) \\ \to (\varphi_{evenjoin}(x) \vee \varphi_{oddjoin}(x))\Big).\end{aligned}$$

Lemma 3.7.11. *Let $\mathfrak{A} = (A,R)$ be an almost fingridlike structure. Then*

$$\mathfrak{A} \models \varphi_{finjoin}$$

if and only if condition 9 of Description 3.7.8 holds in $\mathfrak{A}$.

Proof. By Lemma 3.7.10, conditions 1-8 of Description 3.7.8 hold in every fingridlike structure. Let

$$\varphi_{successors}(x) := \exists y\,\varphi_{horizontal}(x,y) \wedge \exists y\,\varphi_{vertical}(x,y).$$

Since $\varphi_{successors}(x)$ is a first-order formula it follows, by Proposition 2.4.5, that $\varphi_{finjoin}$ is equivalent to the formula

$$\begin{aligned}\forall x\Big((\varphi_{gridpoint}(x) \wedge \varphi_{successors}(x)) \\ \to \Big((\varphi_{successors}(x) \wedge \varphi_{evenjoin}(x)) \vee (\varphi_{successors}(x) \wedge \varphi_{oddjoin}(x))\Big)\Big).\end{aligned} \tag{3.31}$$

Notice then that when restricted to teams with domain $\{x\}$, the formula

$$\varphi_{successors}(x) \wedge \varphi_{evenjoin}(x)$$

is equivalent to the first-order formula

$$\begin{aligned}\varphi_{successors}(x) \wedge R(x,x) \wedge \exists x'\forall y\Big((\varphi_{horizontal}(x,y) \vee \varphi_{vertical}(x,y)) \\ \to (\neg R(x',x') \wedge (\varphi_{horizontal}(y,x') \vee \varphi_{vertical}(y,x')))\Big).\end{aligned} \tag{3.32}$$

Furthermore, by conditions 2 and 3 of Description 3.7.8, the formula

$$\begin{aligned}&\varphi_{successors}(x) \wedge R(x,x) \wedge \exists x' \exists y_1 \exists y_2 \\ &\Big(y_1 \neq y_2 \wedge \varphi_{horizontal}(x,y_1) \wedge \varphi_{vertical}(x,y_2) \wedge \neg R(x',x') \\ &\quad \wedge \big(\varphi_{horizontal}(y_1,x') \vee \varphi_{vertical}(y_1,x')\big) \\ &\quad \wedge \big(\varphi_{horizontal}(y_2,x') \vee \varphi_{vertical}(y_2,x')\big)\Big)\end{aligned} \tag{3.33}$$

is equivalent to the formula (3.32) in the class of almost fingridlike structures. Note that if $y_1 \neq y_2$ then, by condition 3 of Description 3.7.8, neither

$$\varphi_{horizontal}(y_1,x') \wedge \varphi_{horizontal}(y_2,x') \text{ nor } \varphi_{vertical}(y_1,x') \wedge \varphi_{vertical}(y_2,x')$$

can be true in an almost fingridlike structure. Hence the formula (3.33) is equivalent to the formula

$$\begin{aligned}&\varphi_{successors}(x) \wedge R(x,x) \wedge \exists x' \exists y_1 \exists y_2 \Big(y_1 \neq y_2 \wedge \neg R(x',x') \\ &\big(\varphi_{horizontal}(x,y_1) \wedge \varphi_{vertical}(x,y_2) \wedge \varphi_{horizontal}(y_1,x') \vee \varphi_{vertical}(y_2,x')\big) \\ &\vee \big(\varphi_{horizontal}(x,y_1) \wedge \varphi_{vertical}(x,y_2) \wedge \varphi_{vertical}(y_1,x') \wedge \varphi_{horizontal}(y_2,x')\big)\Big)\end{aligned} \tag{3.34}$$

in the class of almost fingridlike structures. Recall the definition of $\varphi_{horizontal}$. Now, if $R(x,x)$ and $\neg R(x',x')$ hold then

$$\varphi_{horizontal}(x,y) \wedge \varphi_{horizontal}(y,x')$$

cannot hold, for any y. Hence the formula (3.34) is equivalent to the formula

$$\begin{aligned}&\varphi_{successors}(x) \wedge R(x,x) \wedge \exists x' \exists y_1 \exists y_2 (y_1 \neq y_2 \wedge \neg R(x',x') \\ &\wedge \varphi_{horizontal}(x,y_1) \wedge \varphi_{vertical}(x,y_2) \wedge \varphi_{vertical}(y_1,x') \wedge \varphi_{horizontal}(y_2,x')).\end{aligned} \tag{3.35}$$

Due to condition 2 of Description 3.7.8 and the definitions of $\varphi_{horizontal}$ and $\varphi_{vertical}$, we conclude that the simple formula

$$\begin{aligned}\varphi_{evenfinjoin}(x) :=\ &\varphi_{successors}(x) \wedge R(x,x) \wedge \exists x' \exists y_1 \exists y_2 (\varphi_{horizontal}(x,y_1) \\ &\wedge \varphi_{vertical}(x,y_2) \wedge \varphi_{vertical}(y_1,x') \wedge \varphi_{horizontal}(y_2,x'))\end{aligned}$$

is equivalent to (3.35) in the class of almost fingridlike structures. Therefore, the $\mathcal{IF}^2$-formula

$$\varphi_{successors}(x) \wedge \varphi_{evenjoin}(x)$$

is equivalent to the formula $\varphi_{evenfinjoin}$ in the class of almost fingridlike structures when restricted to teams with domain $\{x\}$. By an analogous argument, the formula

$$\varphi_{successors}(x) \wedge \varphi_{oddjoin}(x)$$

is equivalent to the formula

$$\begin{aligned}\varphi_{oddfinjoin}(x) := \varphi_{successors}(x) \wedge \neg R(x,x) \wedge \exists x' \exists y_1 \exists y_2 \big(&\varphi_{horizontal}(x,y_1) \\ &\wedge \varphi_{vertical}(x,y_2) \wedge \varphi_{vertical}(y_1,x') \wedge \varphi_{horizontal}(y_2,x')\big)\end{aligned}$$

in the class of almost fingridlike structures when restricted to teams with domain $\{x\}$. By the above equivalences, we conclude that (3.31) is equivalent to

$$\forall x \Big(\big(\varphi_{gridpoint}(x) \wedge \varphi_{successors}(x)\big) \to \big(\varphi_{evenfinjoin}(x) \vee \varphi_{oddfinjoin}(x)\big)\Big)$$

in the class of almost fingridlike structures. Since the above formula is first-order it follows, by Proposition 2.4.5, that it is equivalent to

$$\begin{aligned}\forall x \Big(\big(\varphi_{gridpoint}(x) \wedge \varphi_{successors}(x)\big) \to \exists x' \exists y_1 \exists y_2 \big(&\varphi_{horizontal}(x,y_1) \\ &\wedge \varphi_{vertical}(x,y_2) \wedge \varphi_{vertical}(y_1,x') \wedge \varphi_{horizontal}(y_2,x')\big)\Big).\end{aligned}$$

Hence, the above formula is equivalent to $\varphi_{finjoin}$ in the class of almost fingridlike structures. From this the claim clearly follows. □

We are now ready to give a formal definition of fingridlike structures. Let

$$\varphi_{fingridlike} := \varphi_{almostfingridlike} \wedge \varphi_{finjoin}.$$

Definition 3.7.12. *We say that a finite structure $\mathfrak{A} = (A, R)$ is* fingridlike *if*

$$\mathfrak{A} \models \varphi_{fingridlike}.$$

Lemma 3.7.13. *A finite structure $\mathfrak{A} = (A, R)$ is fingridlike if and only if the conditions of Description 3.7.8 hold in $\mathfrak{A}$.*

Proof. Follows directly from Lemmas 3.7.10 and 3.7.11. □

3.7.3 Labelled structures

We will next show how to expand the encodings defined in Sections 3.7.1 and 3.7.2 to handle T-labelled supergrids and superfingrids. The idea is that proposition symbols are encoded by gadgets of the following type. For $n \geq 1$, let $\mathfrak{G}_n = (G_n, R_n)$ be the structure such that $G_n = \{k \in \mathbb{N} \mid k \leq n\}$ and

$$R_n = \{(0,1),(1,0)\} \cup \{(i, i+1) \subseteq \mathbb{N}^2 \mid 1 \leq i < n\}.$$

Definition 3.7.14. *We call a structure $\mathfrak{A}$ an* n-gadget *if it is isomorphic to $\mathfrak{G}_n$ and we call a structure a* gadget *if it is an n-gadget for some $n \in \mathbb{N}$. Let $\mathfrak{A}$ be a* gadget *and f a related isomorphism. We call the point $a \in A$ such that $f(a) = 0$ a* root of the gadget. *Note that for $n \geq 2$ the root of an n-gadget $\mathfrak{A}$ is unique.*

Remember that a tile type is just a function $t : 4 \to \mathbb{N}$. Hence we can define a canonical ordering $\prec$ on the set of all tiles. Let $\delta : \text{TILES} \to \mathbb{Z}_+$ be a function that arises from $\prec$. For each tile P_t we use $\delta(P_t)$-gadgets to encode the extension of the proposition symbol P_t. The idea is that a point a is in the extension of the proposition symbol P_t if and only if the encoding of a is a root of some $\delta(P_t)$-gadget in the encoding. See Figure 3.4 for an illustration. We first define the following auxiliary formulae recursively.

$$\begin{aligned}
\varphi_{0-path}(x) &:= \forall y\, \neg\varphi_{horizontal}(x,y),\\
\varphi_{0-path}(y) &:= \forall x\, \neg\varphi_{horizontal}(y,x),\\
\varphi_{n-path}(x) &:= \exists y\big(\varphi_{horizontal}(x,y) \wedge \varphi_{(n-1)-path}(y)\big),\\
\varphi_{n-path}(y) &:= \exists x\big(\varphi_{horizontal}(y,x) \wedge \varphi_{(n-1)-path}(x)\big).
\end{aligned}$$

We then define that

$$\varphi_{P_t-gadget}(x) := \exists y\big(\neg R(y,y) \wedge \varphi_{symmetric}(x,y) \wedge \varphi_{\left(\delta(P_t)-1\right)-path}(y)\big).$$

The intended meaning of the formula $\varphi_{P_t-gadget}(x)$ is that the interpretation of x is a root of some $\delta(P_t)$-gadget. For each finite set of tiles T we define that

$$\begin{aligned}
\varphi_{T-labelled} := \forall x \Big(\varphi_{gridpoint}(x) \to \Big(&\bigwedge_{P_t,P_s \in T, t \neq s} \neg\big(\varphi_{P_t-gadget}(x) \wedge \varphi_{P_s-gadget}(x)\big)\\
&\wedge \bigvee_{P_t \in T} \varphi_{P_t-gadget}(x)\Big)\Big)
\end{aligned}$$

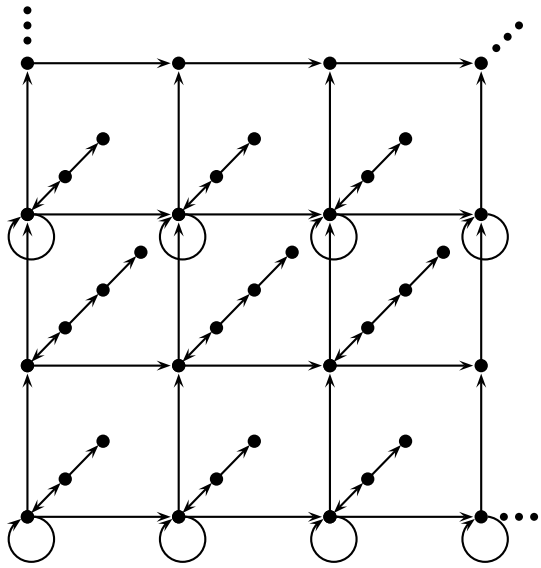

Figure 3.4: Illustration of an encoding of a labelled grid.

Definition 3.7.15. *Let T be a finite set of tiles. We call a structure $\mathfrak{A} = (A, R)$ a* T-labelled gridlike structure *if*

$$\mathfrak{A} \models \varphi_{gridlike} \wedge \varphi_{T-labelled}.$$

Analogously, we say that a finite structure $\mathfrak{B} = (B, R')$ is a T-labelled fingridlike *structure if*

$$\mathfrak{B} \models \varphi_{fingridlike} \wedge \varphi_{T-labelled}.$$

3.7.4 Interpreting labelled supergrids and superfingrids

We will now show that for every finite set of tiles T a sufficiently rich class of T-labelled supergrids is uniformly $\mathcal{FO}^2$-interpretable in the class of T-labelled gridlike structures. Analogously, we show that for every finite set of tiles T a sufficiently rich class of T-labelled superfingrids is uniformly $\mathcal{FO}^2$-interpretable in the class of T-labelled fingridlike structures.

Lemma 3.7.16. *There is a computable function I such that for each nonempty set T of tile symbols there is a class $\mathcal{S}_T$ of T-labelled supergrids such that $\mathcal{G}_T \subseteq \mathcal{S}_T$ and the function I is a uniform $\mathcal{FO}^2$-interpretation of $\mathcal{S}_T$ in the class of T-labelled gridlike structures.*

Proof. Let T be a finite nonempty set of tiles. Define the formulae

$$\begin{aligned}\varphi_{Dom}(x) &:= \varphi_{gridpoint}(x),\\ \varphi_H(x,y) &:= \varphi_{horizontal}(x,y),\\ \varphi_V(x,y) &:= \varphi_{vertical}(x,y), \text{ and}\\ \varphi_{P_t}(x) &:= \varphi_{P_t-gadget}(x) \text{ for each } P_t \in T.\end{aligned}$$

The formulae φ_{Dom}, φ_H φ_V and φ_{P_t} define the uniform $\mathcal{FO}^2$-interpretation I. Let $\mathfrak{A} = (A, R)$ be a T-labelled gridlike structure. Call

$$S_{\mathfrak{A}} := \{\ u \in A \mid \mathfrak{A} \models \varphi_{Dom}(u)\ \},$$

and define the structure

$$\mathfrak{S}_{\mathfrak{A}} = (S_{\mathfrak{A}}, H^{\mathfrak{S}_{\mathfrak{A}}}, V^{\mathfrak{S}_{\mathfrak{A}}}, (P_t^{\mathfrak{S}_{\mathfrak{A}}})_{P_t \in T}),$$

where

$$\begin{aligned}H^{\mathfrak{S}_{\mathfrak{A}}} &:= \{\ (u,v) \in S_{\mathfrak{A}} \times S_{\mathfrak{A}} \mid \mathfrak{A} \models \varphi_H(u,v)\ \},\\ V^{\mathfrak{S}_{\mathfrak{A}}} &:= \{\ (u,v) \in S_{\mathfrak{A}} \times S_{\mathfrak{A}} \mid \mathfrak{A} \models \varphi_V(u,v)\ \},\\ P^{\mathfrak{S}_{\mathfrak{A}}} &:= \{\ u \in S_{\mathfrak{A}} \mid \mathfrak{A} \models \varphi_{P_t}(u)\ \} \text{ for each } P_t \in T.\end{aligned}$$

From Lemma 3.7.7 and Description 3.7.2 it is easy to see that there is a homomorphism from some T-labelled grid into $\mathfrak{S}_{\mathfrak{A}}$. Hence $\mathfrak{S}_{\mathfrak{A}}$ is a T-labelled supergrid. Let $f : S_{\mathfrak{A}} \to A$ be the identity function. Now clearly f is a bijection from $S_{\mathfrak{A}}$ to

$$\{u \in A \mid \mathfrak{A} \models \varphi_{Dom}(u)\}$$

such that the following conditions hold for all $u, v \in S_{\mathfrak{A}}$ and $P_t \in T$:

1. $(u,v) \in H^{\mathfrak{S}_{\mathfrak{A}}} \ \Leftrightarrow\ \varphi_H\big(f(u), f(v)\big)$,
2. $(u,v) \in V^{\mathfrak{S}_{\mathfrak{A}}} \ \Leftrightarrow\ \varphi_V\big(f(u), f(v)\big)$,
3. $u \in P_t^{\mathfrak{S}_{\mathfrak{A}}} \ \Leftrightarrow\ \varphi_{P_t}\big(f(u)\big)$.

Let

$$\mathcal{S}_T := \{\mathfrak{S}_{\mathfrak{A}} \in Str(\{H, V\} \cup T) \mid \mathfrak{A} \text{ is a } T\text{-labelled gridlike structure }\}.$$

The function I is a uniform $\mathcal{FO}^2$-interpretation of $\mathcal{S}_T$ in the class of all T-labelled gridlike structures. For each T-labelled grid $\mathfrak{G}'$ it is easy to construct a T-labelled gridlike structure $\mathfrak{A}$ such that $\mathfrak{S}_{\mathfrak{A}} = \mathfrak{G}'$. Hence $\mathcal{G}_T \subseteq \mathcal{S}_T$. □

Lemma 3.7.17. *There is a computable function J such that for each nonempty set T of tile symbols there is a class $\mathcal{SF}_T$ of finite T-labelled superfingrids such that $\mathcal{F}_T \subseteq \mathcal{SF}_T$ and the function J is a uniform $\mathcal{FO}^2$-interpretation of $\mathcal{SF}_T$ in the class of all finite T-labelled fingridlike structures.*

Proof. Let T be a finite nonempty set of tiles. Define the formulae

$$\begin{aligned}
\varphi_{Dom}(x) &:= \varphi_{gridpoint}(x),\\
\varphi_{H}(x,y) &:= \varphi_{horizontal}(x,y),\\
\varphi_{V}(x,y) &:= \varphi_{vertical}(x,y), \text{ and}\\
\varphi_{P_t}(x) &:= \varphi_{P_t-gadget}(x) \text{ for each } P_t \in T.
\end{aligned}$$

The formulae φ_{Dom}, φ_H φ_V and φ_{P_t} define the uniform $\mathcal{FO}^2$-interpretation J. Let $\mathfrak{A} = (A, R)$ be a finite T-labelled fingridlike structure. Call

$$S_{\mathfrak{A}} := \{\ u \in A \mid \mathfrak{A} \models \varphi_{Dom}(u)\ \},$$

and define the structure

$$\mathfrak{S}_{\mathfrak{A}} = (S_{\mathfrak{A}}, H^{\mathfrak{S}_{\mathfrak{A}}}, V^{\mathfrak{S}_{\mathfrak{A}}}, (P_t^{\mathfrak{S}_{\mathfrak{A}}})_{P_t \in T}),$$

where

$$\begin{aligned}
H^{\mathfrak{S}_{\mathfrak{A}}} &:= \{\ (u,v) \in S_{\mathfrak{A}} \times S_{\mathfrak{A}} \mid \mathfrak{A} \models \varphi_H(u,v)\ \},\\
V^{\mathfrak{S}_{\mathfrak{A}}} &:= \{\ (u,v) \in S_{\mathfrak{A}} \times S_{\mathfrak{A}} \mid \mathfrak{A} \models \varphi_V(u,v)\ \},\\
P_t^{\mathfrak{S}_{\mathfrak{A}}} &:= \{\ u \in S_{\mathfrak{A}} \mid \mathfrak{A} \models \varphi_{P_t}(u)\ \} \text{ for each } P_t \in T.
\end{aligned}$$

From Lemma 3.7.13 and Description 3.7.8 it is easy to see that there is a connected component in $\mathfrak{S}_{\mathfrak{A}}$ that is isomorphic to some T-labelled finite grid. Hence $\mathfrak{S}_{\mathfrak{A}}$ is a finite T-labelled superfingrid. Let $f : S_{\mathfrak{A}} \to A$ be the identity function. Now clearly f is a bijection from $S_{\mathfrak{A}}$ to $\{u \in A \mid \mathfrak{A} \models \varphi_{Dom}(u)\}$ such that the following conditions hold for all $u, v \in S_{\mathfrak{A}}$ and $P_t \in T$:

1. $(u,v) \in H^{\mathfrak{S}_{\mathfrak{A}}} \ \Leftrightarrow\ \varphi_H\big(f(u), f(v)\big)$,
2. $(u,v) \in V^{\mathfrak{S}_{\mathfrak{A}}} \ \Leftrightarrow\ \varphi_V\big(f(u), f(v)\big)$,
3. $u \in P_t^{\mathfrak{S}_{\mathfrak{A}}} \ \Leftrightarrow\ \varphi_{P_t}\big(f(u)\big)$.

Let

$$\mathcal{SF}_T := \{\mathfrak{S}_{\mathfrak{A}} \in Str(\{H,V\} \cup T) \mid \mathfrak{A} \text{ is a finite } T\text{-labelled fingridlike structure}\ \}.$$

The function J is a uniform $\mathcal{FO}^2$-interpretation of $\mathcal{FS}_T$ in the class of all finite T-labelled fingridlike structures. For each T-labelled finite grid $\mathfrak{G}'$ it is easy to construct a finite T-labelled fingridlike structure $\mathfrak{A}$ such that $\mathfrak{S}_{\mathfrak{A}} = \mathfrak{G}'$. Hence $\mathcal{F}_T \subseteq \mathcal{SF}_T$. □

3.8 Satisfiability of finite variable $\mathcal{IF}$-logics

In this section we study the complexity of the satisfiability problem and the finite satisfiability problem for finite variable independence-friendly logics. We show that while both problems are NP-complete for the one-variable case, the problems become undecidable already for two-variable $\mathcal{IF}$-logic. More precisely, we show that, for $k \geq 2$, the satisfiability problem and the finite satisfiability problem for $\mathcal{IF}^k$ are Π^0_1-complete and Σ^0_1-complete, respectively. We show that this holds already in the case where the vocabulary consists of just one binary relation symbol and nothing else. This improves our corresponding result from [KKLV11] where two binary relations and infinitely many unary relations are needed.

Theorem 3.8.1. $\textsc{Sat}(\mathcal{IF}^1)$ *and* $\textsc{FinSat}(\mathcal{IF}^1)$ *are both* NP*-complete.*

Proof. By Proposition 3.3.2, we know that $\mathcal{IF}^1 \equiv \mathcal{FO}^1$. Since the translations are polynomial, the result follows from the corresponding result for $\mathcal{FO}^1$, i.e., Theorem 3.5.1. □

The rest of this section is devoted on showing that the satisfiability problem and the finite satisfiability problem for $\mathcal{IF}^2$ are Π^0_1-complete and Σ^0_1-complete, respectively. Remember that $\textsc{Sat}(\mathcal{IF})$ and $\textsc{FinSat}(\mathcal{IF})$ are Π^0_1-complete and Σ^0_1-complete, respectively (Theorem 2.5.8). Hence if $\textsc{Sat}(\mathcal{IF}^2)$ and $\textsc{FinSat}(\mathcal{IF}^2)$ are Π^0_1-complete and Σ^0_1-complete, respectively, it follows that $\textsc{Sat}(\mathcal{IF}^k)$ and $\textsc{FinSat}(\mathcal{IF}^k)$ are Π^0_1-complete and Σ^0_1-complete, respectively, for every $k \geq 2$.

Theorem 3.8.2. *Let R be a binary relation symbol. The satisfiability problem for $\mathcal{IF}^2(\{R\})$ is Π^0_1-complete.*

Proof. The upper bound follows directly from Theorem 2.5.8. For the lower bound, let $\sigma = \{H, V\}$ be the vocabulary of supergrids and let $\tau = \{R\}$ be the vocabulary of gridlike structures. There is a computable function that associates each input T to the tiling problem with an $\mathcal{IF}^2(\tau)$-sentence

$$\varphi_{gridlike} \wedge \varphi_{T-labelled}$$

that defines the class of T-labelled gridlike structures with respect to the class of all τ-structures, for the construction see Sections 3.7.1 and 3.7.3. By Lemma 3.6.4, there is a computable function that associates each input T to the tiling problem with an $\mathcal{FO}^2(\sigma \cup T)$-sentence φ_T such that a structure $\mathfrak{A}$ of vocabulary σ is T-tilable if and only if there is an expansion $\mathfrak{A}^*$ of the structure $\mathfrak{A}$ to the vocabulary $\sigma \cup T$ such that $\mathfrak{A}^* \models \varphi_T$. Note that, by Proposition 2.4.5, for any syntactically first-order sentence it holds that

$$\mathfrak{A} \models_{\mathcal{FO}} \varphi \quad \text{iff} \quad \mathfrak{A} \models \varphi.$$

By Lemma 3.7.16, there exists a computable function I such that for each input T to the tiling problem there exists a class of T-labelled supergrids $\mathcal{S}_T$ such that $\mathcal{G}_T \subseteq \mathcal{S}_T$ and such that the function I is a uniform $\mathcal{FO}^2$-interpretation of $\mathcal{S}_T$ in the class of all T-labelled gridlike structures.

Let T be an input to the tiling problem. Define the $\mathcal{IF}^2(\tau)$-sentence

$$\psi_T := \varphi_{gridlike} \wedge \varphi_{T-labelled} \wedge I(\varphi_T).$$

We will prove that, for each input T to the tiling problem, the following conditions are equivalent.

1. There exists a τ structure $\mathfrak{B}$ such that $\mathfrak{B} \models \psi_T$.
2. The grid $\mathfrak{G}$ is T-tilable.

Thereby we establish that there exists a computable reduction from the tiling problem to $\textsc{Sat}(\mathcal{IF}^2(\{R\}))$. Since the tiling problem is Π^0_1-complete, this shows that $\textsc{Sat}(\mathcal{IF}^2(\{R\}))$ is Π^0_1-hard.

Let T be an input to the tiling problem. Assume first that the grid $\mathfrak{G}$ is T-tilable. Therefore there exists an expansion $\mathfrak{G}^*$ of the grid $\mathfrak{G}$ to the vocabulary $\{H, V\} \cup T$ such that $\mathfrak{G}^* \models_{\mathcal{FO}} \varphi_T$. Now since $\mathfrak{G}^* \in \mathcal{G}_T \subseteq \mathcal{S}_T$, by Lemma 3.2.3, there exists a T-labelled gridlike structure $\mathfrak{A}$ such that $\mathfrak{A} \models_{\mathcal{FO}} I(\varphi_T)$. Since $\mathfrak{A}$ is a T-labelled gridlike structure, we have that

$$\mathfrak{A} \models \varphi_{gridlike} \wedge \varphi_{T-labelled}.$$

Therefore $\mathfrak{A} \models \psi_T$.

For the converse, assume that there exists a τ-structure $\mathfrak{B}$ such that $\mathfrak{B} \models \psi_T$. Therefore

$$\mathfrak{B} \models \varphi_{gridlike} \wedge \varphi_{T-labelled} \quad \text{and} \quad \mathfrak{B} \models_{\mathcal{FO}} I(\varphi_T).$$

Since $\mathfrak{B} \models \varphi_{gridlike} \wedge \varphi_{T-labelled}$, the structure $\mathfrak{B}$ is a T-labelled gridlike structure. Therefore, and since $\mathfrak{B} \models_{\mathcal{FO}} I(\varphi_T)$, we conclude by Lemma 3.2.3 that

$$\mathfrak{A} \models_{\mathcal{FO}} \varphi_T,$$

for some T-labelled supergrid $\mathfrak{A} \in \mathcal{S}_T$. Thus there exists a supergrid that is T-tilable. Hence the grid $\mathfrak{G}$ is T-tilable.

□

Corollary 3.8.3. $\textsc{Sat}(\mathcal{IF}^k)$ *is* Π^0_1*-complete, for every* $k \geq 2$.

Theorem 3.8.4. *Let* R *be a binary relation symbol. The finite satisfiability problem for* $\mathcal{IF}^2(\{R\})$ *is* Σ^0_1*-complete.*

Proof. The upper bound follows directly from Theorem 2.5.8. For the lower bound, let $\sigma = \{H, V\}$ be the vocabulary of superfingrids and let $\tau = \{R\}$ be the vocabulary of fingridlike structures. There is a computable function that associates each input (T, b, f) to the finite tiling problem with an $\mathcal{IF}^2(\tau)$-sentence

$$\varphi_{fingridlike} \wedge \varphi_{T-labelled}$$

that defines the class of finite T-labelled fingridlike structures with respect to the class of all finite τ-structures. The construction of the sentence is described in Sections 3.7.2 and 3.7.3. By Lemma 3.6.4, there is a computable function that associates each input (T, b, f) to the finite tiling problem with an $\mathcal{FO}^2(\sigma \cup T)$-sentence $\varphi_{(T,b,f)}$ such that a finite grid $\mathfrak{F}$ is (T, b, f)-tilable if and only if there is an expansion $\mathfrak{F}^*$ of the structure $\mathfrak{F}$ to the vocabulary $\sigma \cup T$ such that $\mathfrak{F}^* \models \varphi_{(T,b,f)}$. Remember that, by Proposition 2.4.5, for any syntactically first-order sentence it holds that

$$\mathfrak{F} \models_{\mathcal{FO}} \varphi \quad \text{iff} \quad \mathfrak{F} \models \varphi.$$

By Lemma 3.7.17, there exists a computable function J such that for each input (T, b, f) to the finite tiling problem there exists a class of finite T-labelled superfingrids $\mathcal{SF}_T$ such that $\mathcal{F}_T \subseteq \mathcal{SF}_T$ and the function J is a uniform $\mathcal{FO}^2$-interpretation of $\mathcal{SF}_T$ in the class of all finite T-labelled fingridlike structures.

Let (T, b, f) be an input to the finite tiling problem. Define the $\mathcal{IF}^2(\tau)$-sentence

$$\psi_{(T,b,f)} := \varphi_{fingridlike} \wedge \varphi_{T-labelled} \wedge J(\varphi_{(T,b,f)}).$$

We will prove that, for each input (T, b, f) to the finite tiling problem, the following conditions are equivalent.

1. There exists a finite τ structure $\mathfrak{B}$ such that $\mathfrak{B} \models \psi_{(T,b,f)}$.

2. There exists a finite grid $\mathfrak{F}$ such that $\mathfrak{F}$ is (T, b, f)-tilable.

Thereby we establish that there exists a computable reduction from the finite tiling problem to FINSAT($\mathcal{IF}^2(\{R\})$). Since the finite tiling problem is Σ^0_1-complete, this shows that the finite satisfiability problem for $\mathcal{IF}^2(\{R\})$ is Σ^0_1-hard.

Let (T, b, f) be an input to the finite tiling problem. Assume first that there exists a finite grid $\mathfrak{F}$ such that it is (T, b, f)-tilable. Therefore there exists an expansion $\mathfrak{F}^*$ of $\mathfrak{F}$ to the vocabulary $\{H, V\} \cup T$ such that

$$\mathfrak{F}^* \models_{\mathcal{FO}} \varphi_{(T,b,f)}.$$

Now since $\mathfrak{F}^* \in \mathcal{F}_T \subseteq \mathcal{SF}_T$ and since J is a uniform $\mathcal{FO}^2$-interpretation of $\mathcal{SF}_T$ in the class of all finite T-labelled fingridlike structures, we conclude by Lemma 3.2.3 that there exists a finite T-labelled fingridlike structure $\mathfrak{A}$ such that

$$\mathfrak{A} \models_{\mathcal{FO}} J(\varphi_{(T,b,f)}).$$

Since $\mathfrak{A}$ is a finite T-labelled fingridlike structure, we have that

$$\mathfrak{A} \models \varphi_{\mathit{fingridlike}} \wedge \varphi_{T-\mathit{labelled}}.$$

Therefore $\mathfrak{A} \models \psi_{(T,b,f)}$.

For the converse, assume that there exists a finite τ-structure $\mathfrak{B}$ such that

$$\mathfrak{B} \models \psi_{(T,b,f)}.$$

Therefore

$$\mathfrak{B} \models \varphi_{\mathit{fingridlike}} \wedge \varphi_{T-\mathit{labelled}} \quad \text{and} \quad \mathfrak{B} \models_{\mathcal{FO}} J(\varphi_{(T,b,f)}).$$

Since $\mathfrak{B}$ is finite and $\mathfrak{B} \models \varphi_{\mathit{fingridlike}} \wedge \varphi_{T-\mathit{labelled}}$, it is a finite T-labelled fingridlike structure. Now since J is a uniform $\mathcal{FO}^2$-interpretation of $\mathcal{SF}_T$ in the class of all finite T-labelled fingridlike structures, and since $\mathfrak{B} \models_{\mathcal{FO}} J(\varphi_{(T,b,f)})$, we conclude, by Lemma 3.2.3, that $\mathfrak{A} \models_{\mathcal{FO}} \varphi_{(T,b,f)}$, for some finite T-labelled superfingrid $\mathfrak{A} \in \mathcal{SF}_T$. Thus, there exists a superfingrid that is (T, b, f)-tilable. Hence, there exists a finite grid that is (T, b, f)-tilable.

□

Corollary 3.8.5. FINSAT($\mathcal{IF}^k$) *is* Σ^0_1*-complete for every* $k \geq 2$.

3.9 Complexity of model checking

In this section we study variants of the model checking problem for finite variable dependence logics and independence-friendly logics. We show that, for $k \geq 2$, the data complexity and the combined complexity for $\mathcal{D}^k$ and $\mathcal{IF}^k$ are all NP-complete. Hence we answer "yes and yes" to a question asked in [KKLV11], i.e., is it possible to define NP-complete problems in $\mathcal{D}^2$ and $\mathcal{IF}^2$. We give a reduction from the dominating set problem, an NP-complete problem from graph theory, to the model checking problem with a fixed $\mathcal{D}^2$-sentence. For related work see the Ph.D. thesis of Jarmo Kontinen [Kon10] and the article [Grä13] by Erich Grädel.

We first recall the definitions for graphs and dominating sets. A finite graph is a structure

$$G = (V, E),$$

where V is a finite set and E is a subset of the set

$$\{\{u, v\} \mid u, v \in V\}.$$

The elements of V are called *vertices* and the elements in E are called *edges*. A set $D \subseteq V$ is called a *dominating set* for the graph G if for every vertex $v \in V$ either $v \in D$ holds or there exists some vertex $u \in D$ such that $\{u, v\} \in E$ holds.

We are now ready to describe the NP-complete problem to be used in the proof of Proposition 3.9.4 to show that the data complexity for $\mathcal{D}^2$ is NP-hard.

Definition 3.9.1. *The dominating set problem is the following decision problem: Given a graph G and a number $n \in \mathbb{N}$, does there exist a dominating set for G of size at most n.*

Theorem 3.9.2. *The dominating set problem is* NP*-complete.*

Proof. For a proof, see [GJ90]. □

Every graph $G = (V, E)$ can be naturally identified with a first-order structure $\mathfrak{A}_G = (A_G, R_G)$ of vocabulary $\{R\}$, where

$$A_G := V \quad \text{and} \quad R_G := \{(u, v) \in A_G^2 \mid \{u, v\} \in E\}.$$

We call $\mathfrak{A}_G$ the *first-order structure corresponding to* G. Let P be a unary relation symbol and $\mathfrak{A}_G^*$ some expansion of $\mathfrak{A}_G$ to the vocabulary $\{R, P\}$. We call the structure $\mathfrak{A}_G^*$ a *first-order structure corresponding to* (G, n) if the cardinality of the extension of the proposition symbol P in $\mathfrak{A}_G^*$ is $\min\{n, |V|\}$.

Lemma 3.9.3. *There exists a polynomial time algorithm that constructs from any given finite graph $G = (V, E)$ and a natural number n a first-order structure $\mathfrak{A}_{(G,n)}$ corresponding to (G, n).*

Proof. Straightforward. □

We are now ready to show that the data complexity for $\mathcal{D}^2$ is NP-hard.

Proposition 3.9.4. *The data complexity for $\mathcal{D}^2$ is* NP*-hard.*

Proof. We will prove the NP-hardness by a polynomial time reduction from the dominating set problem to the model checking problem with a fixed $\mathcal{D}^2$-sentence. By Lemma 3.9.3, there is a polynomial time algorithm that constructs from each input (G, n) to the dominating set problem a first-order structure $\mathfrak{A}_{(G,n)}$ corresponding to (G, n). Let φ_{dsp} denote the following $\mathcal{D}^2$ sentence

$$\forall x \exists y \Big(\big(E(x, y) \vee x = y\big) \wedge \exists x \big(=\!(x, y) \wedge P(x)\big) \Big).$$

It is straightforward to check that the following conditions are equivalent for any finite graph $G = (V, E)$ and $n \in \mathbb{N}$.

1. G has a dominating set of size at most n.
2. $\mathfrak{A} \models \varphi_{dsg}$ for some first-order structure $\mathfrak{A}$ corresponding to (G, n).
3. $\mathfrak{A} \models \varphi_{dsg}$ for all first-order structures $\mathfrak{A}$ corresponding to (G, n).

Hence the following conditions are equivalent for every input (G, n) to the dominating set problem.

1. G has a dominating set of size at most n.
2. $\mathfrak{A}_{(G,n)} \models \varphi_{dsg}$.

Now, since the dominating set problem is an NP-complete problem and $\mathfrak{A}_{(G,n)}$ is constructed from G by a polynomial time algorithm, it follows that the data complexity of $\mathcal{D}^2$ is NP-hard. □

Since finite variable fragments of independence-friendly logic and dependence logic translate into finite variable fragments of existential second-order logic, the following theorem provides us upper bounds for the complexities of the model checking problems for finite variable independence-friendly logics and dependence logics.

Theorem 3.9.5 ([Var82])**.** *The expression complexity and the combined complexity for $\mathcal{ESO}^k$ is* NP*-complete, for every $k \geq 1$.*

For finite variable dependence logics we have the following theorem.

Theorem 3.9.6. *The following hold for every $k \geq 2$.*

1. *The data complexity for $\mathcal{D}^k$ is* NP*-complete.*
2. *The expression complexity for $\mathcal{D}^k$ is in* NP*.*
3. *The combined complexity for $\mathcal{D}^k$ is* NP*-complete.*

Proof. Fix $k \geq 2$. We will first show the lower bounds for the data complexity and the combined complexity for $\mathcal{D}^k$. By Proposition 3.9.4, the data complexity for $\mathcal{D}^2$ is NP-hard. Therefore the data complexity of $\mathcal{D}^k$ is NP-hard. Now since the combined complexity is bound below by the data complexity, it follows that the combined complexity for $\mathcal{D}^k$ is NP-hard. For the upper bounds, note first that, by Theorem 3.9.5, the combined complexity for $\mathcal{ESO}^n$ is in NP, for every $n \in \mathbb{N}$. Therefore, and since the translation from $\mathcal{D}^k$ to $\mathcal{ESO}^{k+1}$ defined in Theorem 3.3.17 is clearly polynomial, we conclude that the combined complexity for $\mathcal{D}^k$ is in NP. Since the data complexity and the expression complexity is bound above by the combined complexity, it follows that both the data complexity and the expression complexity for $\mathcal{D}^k$ is in NP. □

As a corollary, we obtain the corresponding results also for finite variable independence-friendly logics.

Corollary 3.9.7. *The following hold for every $k \geq 2$.*

1. *The data complexity for $\mathcal{IF}^k$ is* NP*-complete.*
2. *The expression complexity for $\mathcal{IF}^k$ is in* NP*.*
3. *The combined complexity for $\mathcal{IF}^k$ is* NP*-complete.*

Proof. Straightforward by the polynomial time translations from $\mathcal{D}^k$ into $\mathcal{IF}^k$ and from $\mathcal{IF}^k$ into $\mathcal{D}^{k+1}$, Lemmas 3.3.9 and 3.3.10, and the corresponding complexity results for $\mathcal{D}^k$, Theorem 3.9.6. □

Furthermore, since $\mathcal{IF}^1$ is essentially the same as $\mathcal{FO}^1$, the model checking for $\mathcal{IF}^1$ and $\mathcal{FO}^1$ coincide. Moreover, by Proposition 3.3.6, each $\mathcal{D}^1$ sentence is equivalent to some first-order sentence. Therefore and since, by Theorem 2.5.10, the data complexity for $\mathcal{FO}$ is in AC^0, it follows that the data complexity for $\mathcal{D}^1$ is also in AC^0.

3.10 Conclusion

In this chapter we studied the expressive power and the computational complexity of finite variable fragments of dependence logic and independence-friendly logic. We compared the expressive powers of these logics to fragments of first-order logic and existential second-order logic. We presented a comprehensive study on the computational complexities of different variants of the satisfiability problem and the model checking problem for these logics. In addition, we addressed some model theoretic properties of these logics.

We established that k-variable fragments of independence-friendly logic and dependence logic have the zero-one law and the finite model property only when $k = 1$. We showed that while the expressive powers of one-variable independence-friendly logic and one-variable first-order logic coincide, the expressive power of two-variable independence-friendly logic is not bounded even by that of monadic second-order logic. We obtain that, for each $k \geq 1$,

$$\mathcal{IF}^k < \Sigma^1_1(\mathcal{FOC}^{k+1}) \quad \text{and} \quad \mathcal{IF}^k < \Sigma^1_1(\mathcal{FO}^{k+2}).$$

We showed that while the expressive power of one-variable dependence logic is bounded by that of first-order logic, for each $k \geq 1$,

$$\mathcal{D}^1 \not\leq \mathcal{FO}^k.$$

For the two-variable fragment we established that $\mathcal{D}^2 \not\leq \mathcal{MSO}$. In addition, we showed that, for each $k \geq 1$,

$$\mathcal{D}^k \leq \Sigma^1_1(\mathcal{FOC}^k) \quad \text{and} \quad \mathcal{D}^k < \Sigma^1_1(\mathcal{FO}^{k+1}).$$

Moreover, we discovered a hierarchy between the expressive powers of finite variable dependence logics and independence-friendly logics. We showed that, for every $k \geq 2$,

$$\mathcal{D}^k \leq \mathcal{IF}^k < \mathcal{D}^{k+1}.$$

For $k = 2$, we showed that also the first inclusion is strict. The result was a by-product of a deeper result of detecting a decidability barrier between $\mathcal{IF}^2$ and $\mathcal{D}^2$. Moreover as a peculiar quirk, we noticed that for the one-variable fragments the order is flipped, i.e., that $\mathcal{IF}^1 < \mathcal{D}^1$.

In addition to questions concerning expressive power, we studied the computational complexity of the satisfiability problem and the model checking problem. We established the following trichotomy concerning the satisfiability and the finite satisfiability problem for finite variable dependence logics.

- The satisfiability and finite satisfiability problem for $\mathcal{D}^1$ is NP-complete.
- The satisfiability and finite satisfiability problem for $\mathcal{D}^2$ is NEXPTIME-complete.
- The satisfiability and finite satisfiability problem for $\mathcal{D}^k$ is undecidable for every $k \geq 3$. The satisfiability problem is Π^0_1-complete, while the finite satisfiability problem is Σ^0_1-complete.

For finite variable fragments of independence-friendly logic, we have the following dichotomy.

- The satisfiability and finite satisfiability problem for $\mathcal{IF}^1$ is NP-complete.
- The satisfiability and finite satisfiability problem for $\mathcal{IF}^k$ is undecidable for every $k \geq 2$. The satisfiability problem is Π^0_1-complete, while the finite satisfiability problem is Σ^0_1-complete.

Concerning the model checking problem, we established the following results, for every $k \geq 2$.

- The data complexity for $\mathcal{D}^k$ and $\mathcal{IF}^k$ is NP-complete.
- The expression complexity for $\mathcal{D}^k$ and $\mathcal{IF}^k$ is in NP.
- The combined complexity for $\mathcal{D}^k$ and $\mathcal{IF}^k$ is NP-complete.

Many interesting related problems remain open. We conclude with three of them.

1. What is the computational complexity of the validity problem for finite variable dependence logics and independence-friendly logics.
2. Does $\mathcal{D}^k < \mathcal{IF}^k$ hold for $k > 2$ and does $\mathcal{IF}^k < \mathcal{D}^{k+1}$ hold for vocabularies of fixed arity.
3. Is the expression complexity for $\mathcal{IF}^k$ and $\mathcal{D}^k$ NP-hard.

Chapter 4

Boolean Dependence Logic and Partially-Ordered Connectives

In this chapter we introduce a new variant of dependence logic called Boolean dependence logic ($\mathcal{BD}$). Boolean dependence logic extends first-order logic with special restricted versions of dependence atoms which we call Boolean dependence atoms. While all variables occurring in dependence atoms

$$=(x_1, \ldots, x_n, y)$$

of dependence logic are first-order variables, in Boolean dependence atoms

$$=(x_1, \ldots, x_n, \alpha)$$

of Boolean dependence logic only the antecedents $x_1, \ldots, x_n$ are first-order variables, whereas the consequent α is a Boolean variable. A Boolean variable is special kind of variable with values that range over the set $\{\top, \bot\}$, i.e., Boolean variables as assigned a value *true* or *false*.

Boolean dependence atoms provide a direct way to express partially-ordered connectives in a similar manner as dependence atoms express partially-ordered quantifiers. To make this connection more clear, we define a syntactic variant of the partially-ordered connectives of Sandu and Väänänen [SV92], closely related to the narrow Henkin quantifier of Blass and Gurevich [BG86]. We show that our definition of partially-ordered connectives is, in a strong sense, equivalent to that of Sandu and Väänänen. We then establish a novel connection between partially-ordered connectives and Boolean dependence logic. For example, the partially-ordered connective

$$\begin{pmatrix} \forall x & \exists \alpha \\ \forall y & \exists \beta \end{pmatrix} \varphi,$$

defined with Boolean variables, can be expressed in Boolean dependence logic by the formula

$$\forall x \exists \alpha \forall y \exists \beta (=(x, \alpha) \wedge =(y, \beta) \wedge \varphi).$$

Intuitively, the dependence atoms of dependence logic express quantification over functions. The meaning of the dependence atom

$$=(x_1, \ldots, x_n, y)$$

is that there exists a k-ary function that maps the values of the variables $x_1, \ldots, x_n$ to the value of the variable y. Analogously, Boolean dependence atoms can be interpreted as expressing quantification of relations or, more precisely, characteristic functions of relations. In this sense, the meaning of the Boolean dependence atom

$$=(x_1, \ldots, x_n, \alpha)$$

is that there exists a characteristic function of an n-ary relation that maps the values of the variables $x_1, \ldots, x_n$ to the value $\bot$ or $\top$. Since the expressive powers of dependence logic and existential second-order logic coincide, and since in existential second-order logic it is clear that functions and relations are interdefinable, the question arises whether there is any significant difference between dependence logic and Boolean dependence logic. In fact, in terms of expressive power there is no difference, we show that the expressive power of dependence logic and Boolean dependence logic coincide. On the other hand, natural fragments of Boolean dependence logic directly correspond to logics enriched with partially-ordered connectives. We show that in terms of expressive power, certain fragments of Boolean dependence logic coincide with natural logics enriched with partially-ordered connectives. In addition, we show that these fragments of Boolean dependence logic form a strict hierarchy with respect to expressive power.

The results of this chapter can be seen as a contribution to the analysis of fragments of existential second-order logic. In particular, we are able to separate natural fragments of existential second-order logic. We also give new insight concerning interdefinability of functions and relations in the framework of dependence logic, and henceforth contribute to the basic research of the dependence phenomenon.

The structure of this chapter is as follows. In Sections 4.1 and 4.2 we give a formal definition of Boolean dependence logic and state some of its elementary properties. In Section 4.3 we first briefly discuss the origin of partially-ordered

connectives. We then give two alternative definitions for partially-ordered connectives, one familiar from the literature, and a syntactic variant that makes the comparison to Boolean dependence logic more straightforward. Finally, we show that, in a rather strong sense, these two definitions are equivalent. In Section 4.4 we define three natural fragments of Boolean dependence logic. We name them as bounded Boolean dependence logic ($\mathcal{BBD}$), restricted Boolean dependence logic ($\mathcal{RBD}$) and universal Boolean dependence logic ($\forall$-$\mathcal{BD}$). In Section 4.5 we show a normal form for bounded Boolean dependence logic, and in Section 4.6 we use this normal form to show that the expressive power of $\mathcal{BBD}$, $\mathcal{RBD}$ and $\forall$-$\mathcal{BD}$ coincide with the expressive power of natural logics enriched with partially-ordered connectives, namely $\mathcal{FO}(\mathcal{POC}^+)$, $\mathcal{POC}[\mathcal{FO}]$ and $\mathcal{POC}[\mathcal{QF}]$, respectively. In Section 4.7 we show that $\mathcal{BD}$, $\mathcal{BBD}$, $\mathcal{RBD}$ and $\forall$-$\mathcal{BD}$ form a strict hierarchy with respect to the expressive power.

This chapter is a revised and extended version of the conference article [EHLV13]. The style of the article [EHLV13] is extremely compact and hence in some parts arduous to read and to understand. Due to page limitations, multitude of proofs were forced to be shortened or even omitted. The objective of this chapter is to present a more precise and less cumbersome presentation of the topic. The main improvements are listed below.

1. The proofs concerning the equivalence of partially-ordered connectives by Sandu and Väänänen and partially-ordered connectives with Boolean variables, i.e., Lemmas 4.3.3 and 4.3.4, are new.

2. The definitions concerning occurrences of subformulae, i.e., Definitions 4.4.1 and 4.4.2, have been made more precise. This improvement makes the definition for $\mathcal{BBD}$ more understandable, and also improves the readability of many proofs henceforth.

3. The proof of Proposition 4.5.11 has endured a major revision, e.g the concept of an evaluation (Definition 4.5.8) was introduced in order to simplify the proof.

Furthermore a number of lesser inaccuracies were corrected and weaknesses improved. The research leading to the article [EHLV13] was a joint effort by the author of this thesis, Johannes Ebbing, Lauri Hella and Peter Lohmann. The core ideas were formed in joint sessions by the authors. Most of the technical work done in Sections 4.1 - 4.6 was done by the author of this thesis. The definition of partially-ordered connectives with Boolean variables was formulated by professor Lauri Hella. The original idea behind the proof of Theorem 4.6.2

was by the author of this thesis, whereas the first formulation of the proof was by Hella. The first draft of the proof of Theorem 4.6.5 was done by Johannes Ebbing based on the ideas of Lohmann. Most of the work concerning Section 4.7 was done by professor Hella.

4.1 Boolean dependence logic

Boolean dependence logic ($\mathcal{BD}$) is a variant of dependence logic in which the consequents of dependence atoms are Boolean variables instead of first-order variables. We denote Boolean variables by the Greek letters α and β, whereas we use x, y, z to denote first-order variables as usual. We use χ to denote a variable that is either a first-order variable or a Boolean variable. Tuples of variables are denoted by $\vec{\alpha}, \vec{\beta}$, $\vec{x}, \vec{y}$ and $\vec{\chi}$, respectively.

Definition 4.1.1. *Let τ be a relational vocabulary. The syntax of Boolean dependence logic $\mathcal{BD}(\tau)$ is defined from τ by the following grammar:*

$$\begin{aligned}\varphi \quad ::= \quad & x_1 = x_2 \mid \neg\, x_1 = x_2 \mid \alpha \mid \neg\alpha \mid =\!(x_1, \ldots, x_n, \alpha) \mid \\ & R(x_1, \ldots, x_n) \mid \neg R(x_1, \ldots, x_n) \mid \\ & (\varphi \vee \varphi) \mid (\varphi \wedge \varphi) \mid \forall x\varphi \mid \exists x\varphi \mid \exists\alpha\varphi.\end{aligned}$$

Analogously to dependence logic, the semantics for Boolean dependence logic is defined in terms of teams, i.e., sets of assignments. The only difference here is that in Boolean dependence logic we are also assigning values for Boolean variables. Hence, in this chapter, *assignments* over $\mathfrak{A}$ are finite functions that map first-order variables to elements of A and Boolean variables to elements of $\{\bot, \top\}$. We further assume that $A \cap \{\bot, \top\}$ is always empty. Notice that Boolean variables are never assigned a value from the domain A and first-order variables are never assigned a value $\bot$ or $\top$. If s is an assignment, x a first-order variable and $a \in A$, we denote by $s(a/x)$ the assignment with domain $\mathrm{dom}(s) \cup \{x\}$ such that

$$s(a/x)(\chi) = \begin{cases} s(\chi) & \text{if } \chi \neq x \\ a & \text{if } \chi = x. \end{cases}$$

Analogously, if s is an assignment, α a Boolean variable and $a \in \{\bot, \top\}$, we denote by $s(a/\alpha)$ the assignment with domain $\mathrm{dom}(s) \cup \{\alpha\}$ such that

$$s(a/\alpha)(\chi) = \begin{cases} s(\chi) & \text{if } \chi \neq \alpha \\ a & \text{if } \chi = \alpha. \end{cases}$$

Let A be a set and $\{x_1, \ldots, x_n, \alpha_1, \ldots, \alpha_m\}$ a finite (possibly empty) set of variables. A *team* X of A with domain

$$\mathrm{dom}(X) = \{x_1, \ldots, x_n, \alpha_1, \ldots, \alpha_m\}$$

is any set of assignments from $\mathrm{dom}(X)$ into $A \cup \{\bot, \top\}$. However, we fix that, for the empty team $\emptyset$, $\mathrm{dom}(\emptyset) = \emptyset$. If X is a team of A, and $F\colon X \to A$ and $G\colon X \to \{\bot, \top\}$ are functions, we use

- $X(F/x)$ to denote the team $\{s(F(s)/x) \mid s \in X\}$,
- $X(G/\alpha)$ to denote the team $\{s(G(s)/\alpha) \mid s \in X\}$, and
- $X(A/x)$ to denote the team $\{s(a/x) \mid s \in X \text{ and } a \in A\}$.

Let X be a team of $\mathfrak{A}$, $W \subseteq \mathrm{dom}(X)$ and $F : X \to A$ a function. We say that the function F is *W-determined* if for every assignment $s, s' \in X$ the implication

$$\forall\, \chi \in W : s(\chi) = s'(\chi) \quad \Rightarrow \quad F(s) = F(s')$$

holds.

Definition 4.1.2. *Let $\mathfrak{A}$ be a model and X a team of $\mathfrak{A}$. The satisfaction relation $\mathfrak{A} \models_X \varphi$ for Boolean dependence logic is defined as follows.*

$$\begin{array}{lcl}
\mathfrak{A} \models_X R(x_1, \ldots, x_n) & \Leftrightarrow & \forall s \in X : \big(s(x_1), \ldots, s(x_n)\big) \in R^{\mathfrak{A}}. \\
\mathfrak{A} \models_X \neg R(x_1, \ldots, x_n) & \Leftrightarrow & \forall s \in X : \big(s(x_1), \ldots, s(x_n)\big) \notin R^{\mathfrak{A}}. \\
\mathfrak{A} \models_X =(x_1, \ldots, x_n, \alpha) & \Leftrightarrow & \forall s, t \in X : s(x_1) = t(x_1), \ldots, s(x_n) = t(x_n) \\
 & & \textit{implies that } s(\alpha) = t(\alpha). \\
\mathfrak{A} \models_X \alpha & \Leftrightarrow & \forall s \in X : s(\alpha) = \top. \\
\mathfrak{A} \models_X \neg\alpha & \Leftrightarrow & \forall s \in X : s(\alpha) = \bot. \\
\mathfrak{A} \models_X (\varphi \wedge \psi) & \Leftrightarrow & \mathfrak{A} \models_X \varphi \textit{ and } \mathfrak{A} \models_X \psi. \\
\mathfrak{A} \models_X (\varphi \vee \psi) & \Leftrightarrow & \mathfrak{A} \models_Y \varphi \textit{ and } \mathfrak{A} \models_Z \psi, \\
 & & \textit{for some } Y \textit{ and } Z \textit{ such that } Y \cup Z = X. \\
\mathfrak{A} \models_X \exists x\psi & \Leftrightarrow & \mathfrak{A} \models_{X(F/x)} \psi \textit{ for some } F\colon X \to A. \\
\mathfrak{A} \models_X \exists\alpha\psi & \Leftrightarrow & \mathfrak{A} \models_{X(F/\alpha)} \psi \textit{ for some } F\colon X \to \{\bot, \top\}. \\
\mathfrak{A} \models_X \forall x\psi & \Leftrightarrow & \mathfrak{A} \models_{X(A/x)} \psi.
\end{array}$$

Remark 4.1.3. *Note that we do not allow universal quantification of Boolean variables in the syntax of Boolean dependence logic. We have chosen this convention in order to make the comparison to logics with partially-ordered connectives more straightforward. Furthermore, allowing universal quantification*

of Boolean variables would not add anything essential to Boolean dependence logic, since universal quantification of Boolean variables can be simulated by universal first-order and existential Boolean quantifiers. It is easy to check that, with respect to models with cardinality at least 2, the formulae $\forall\alpha\varphi$ and

$$\forall x\forall y\exists\alpha\Big(((x = y \wedge \alpha) \vee (\neg x = y \wedge \neg\alpha)) \wedge \varphi\Big),$$

where x and y are fresh first-order variables that do not occur in φ, are equivalent.

The set $\mathrm{fr}(\varphi)$ of *free variables* of a $\mathcal{BD}$-formula φ is defined recursively in the obvious manner, i.e., in addition to the rules familiar from first-order logic, we have that

$$\begin{aligned}
&\mathrm{fr}(=(x_1,\dots,x_k,\alpha)) = \{x_1,\dots,x_k,\alpha\},\\
&\mathrm{fr}(\neg\alpha) = \mathrm{fr}(\alpha) = \{\alpha\},\\
&\mathrm{fr}(\exists\alpha\,\varphi) = \mathrm{fr}(\varphi) \setminus \{\alpha\}.
\end{aligned}$$

If $\mathrm{fr}(\varphi) = \emptyset$, we call φ a sentence. We say that the sentence φ is true in the model $\mathfrak{A}$ and write

$$\mathfrak{A} \models \varphi,$$

if $\mathfrak{A} \models_{\{\emptyset\}} \varphi$ holds. We define the following abbreviation.

Definition 4.1.4. *Let $V = \{x_{i_1},\dots,x_{i_n}\}$ where $i_j \leq i_{j+1}$, for all $j < n$. By $=(V,\alpha)$ and $=(V,y)$ we denote $=(x_{i_1},\dots,x_{i_n},\alpha)$ and $=(x_{i_1},\dots,x_{i_n},y)$, respectively.*

Note that while the precise ordering of the n first first-order variables in the dependence atom corresponding to $=(V,\alpha)$ or $=(V,y)$ is irrelevant, to be precise, we have to fix some ordering. Hence we choose the most canonical one.

4.2 Elementary properties of Boolean dependence logic

In this section we state some elementary results on Boolean dependence logic familiar from dependence logic. In Boolean dependence logic, as well as in dependence logic, the truth of a formula depends only on the interpretations of the variables occurring free in the formula.

Proposition 4.2.1. *Let φ be a $\mathcal{BD}$-formula of vocabulary τ, $\mathfrak{A}$ a τ-model and X a team of $\mathfrak{A}$. If $V \supseteq \mathrm{fr}(\varphi)$, then*

$$\mathfrak{A} \models_X \varphi \quad \textit{iff} \quad \mathfrak{A} \models_{X\restriction V} \varphi.$$

Proof. The proof is essentially the same as the proof of the corresponding proposition for dependence logic, [Vää07, Lemma 3.27], with just one additional analogous case for the existential Boolean quantifiers. □

Boolean dependence logic is a conservative extension of first-order logic.

Proposition 4.2.2. *Let φ be a formula of $\mathcal{BD}$ without Boolean variables, i.e., φ is syntactically a first-order formula. Then for all models $\mathfrak{A}$, teams X of $\mathfrak{A}$ and assignments s of $\mathfrak{A}$:*

1. $\mathfrak{A} \models_{\{s\}} \varphi \quad \textit{iff} \quad \mathfrak{A}, s \models_{\mathcal{FO}} \varphi$.
2. $\mathfrak{A} \models_X \varphi \quad \textit{iff} \quad \mathfrak{A}, t \models_{\mathcal{FO}} \varphi$ *for every* $t \in X$.

Proof. Follows directly from the corresponding proposition for dependence logic, i.e., Proposition 2.4.5. □

The next two lemmas state that in Boolean dependence logic one can freely substitute subformulae by equivalent formulae and free variables by fresh variables with the same extension.

Lemma 4.2.3. *Suppose that φ, ψ and ϑ are $\mathcal{BD}$-formulae such that $\mathrm{fr}(\varphi) = \mathrm{fr}(\psi)$ and $\varphi \equiv \psi$. Then*

$$\vartheta \equiv \vartheta(\varphi/\psi),$$

where $\vartheta(\varphi/\psi)$ is the formula obtained from ϑ by substituting each occurrence of ψ by φ.

Proof. The proof is essentially the same as the proof of the corresponding lemma for dependence logic, [Vää07, Lemma 3.25]. □

Lemma 4.2.4. *Let φ be a $\mathcal{BD}$-formula and $x_1, \ldots, x_n, \alpha_1, \ldots, \alpha_m$ the free variables of φ. Let $y_1, \ldots, y_n, \beta_1, \ldots, \beta_m$ be distinct variables that do not occur in φ. If s is an assignment with domain $\{x_1, \ldots, x_n, \alpha_1, \ldots, \alpha_m\}$, let s' denote the assignment with domain $\{y_1, \ldots, y_n, \beta_1, \ldots, \beta_m\}$ defined as*

$$s'(\chi) = \begin{cases} s(x_i) & \textit{if } \chi = y_i, \\ s(\alpha_1) & \textit{if } \chi = \beta_i. \end{cases}$$

If X is a team with domain $\{x_1, \ldots, x_n, \alpha_1, \ldots, \alpha_m\}$, define that $X' := \{s' \mid s \in X\}$. Now, for every model $\mathfrak{A}$ and team X of $\mathfrak{A}$ with domain $\mathrm{fr}(\varphi)$ it holds that

$$\mathfrak{A} \models_X \varphi \quad \textit{iff} \quad \mathfrak{A} \models_{X'} \varphi(y_1/x_1, \ldots, y_n/x_n, \beta_1/\alpha_1, \ldots, \beta_m/\alpha_m),$$

where $\varphi(y_1/x_1, \ldots, y_n/x_n, \beta_1/\alpha_1, \ldots, \beta_m/\alpha_m)$ is obtained from φ by substituting each free occurrence of x_i by y_i and α_j by β_j, $i \leq n$, $j \leq m$.

Proof. Straightforward by Proposition 4.2.1 and the semantics of Boolean dependence logic. □

4.3 Partially-ordered connectives

We will first give a short exposition to the origin of partially-ordered connectives. We will then recall the definition of partially-ordered connectives familiar from the literature. After this, we will introduce a notational variant of partially-ordered connectives based on Boolean variables. Finally, we will show that these two variants, in a rather strong sense, coincide.

4.3.1 Partially-ordered connectives by Sandu and Väänänen

Partially-ordered connectives were introduced in [SV92] by Sandu and Väänänen. The starting point for their definition was the Henkin quantifier

$$\begin{pmatrix} \forall x & \exists y \\ \forall y & \exists v \end{pmatrix},$$

and the idea to replace the existential quantifiers in the Henkin quantifier by disjunctions. Hence they arrived to the following expression

$$\begin{pmatrix} \forall x & \bigvee_{i\in\{0,1\}} \\ \forall y & \bigvee_{j\in\{0,1\}} \end{pmatrix} (\varphi_{ij}(x,y))_{i,j\in\{0,1\}},$$

which they call the partially-ordered connective $D_{1,1}$. The connective $D_{1,1}$ bounds a tuple of formulae of length 4. The subscript of $D_{1,1}$ reveals the number of rows and the number of universal quantifiers in a given row in the connective. For example, the partially-ordered connective $D_{4,3,3}$ is an expression of the form

$$\begin{pmatrix} \forall x_1 \forall x_2 \forall x_3 \forall x_4 & \bigvee_{i\in\{0,1\}} \\ \forall y_1 \forall y_2 \forall y_3 & \bigvee_{j\in\{0,1\}} \\ \forall z_1 \forall z_2 \forall z_3 & \bigvee_{k\in\{0,1\}} \end{pmatrix},$$

that binds a tuple of formulae $(\varphi_{ijk}(x_1, x_2, x_3, x_4, y_1, y_2, y_3, z_1, z_2, z_3))_{i,j,k\in\{0,1\}}$. Hence, more generally, a partially-ordered connective, according to Sandu and Väänänen, is an expression of the form

$$D = \begin{pmatrix} \forall x_{11} & \dots & \forall x_{1n_1} & \bigvee_{b_1\in\{0,1\}} \\ \vdots & & \vdots & \vdots \\ \forall x_{m1} & \dots & \forall x_{mn_m} & \bigvee_{b_m\in\{0,1\}} \end{pmatrix}$$

that binds a tuple $\gamma = (\varphi_{\vec{b}})_{\vec{b}\in\{0,1\}^m}$ of formulae[1]. Note that it is assumed that the variables x_{ij} are distinct. Denoting the tuples $(x_{i1}, \dots, x_{in_i})$ by $\vec{x}_i$, $1 \le i \le m$, the semantics of the partially-ordered connective D can be written as follows:

$$\begin{aligned} \mathfrak{A}, s \models D\,\gamma \quad \Leftrightarrow \quad & \text{there exist functions } g_i : A^{n_i} \to \{0,1\}, 1 \le i \le m, \\ & \text{s.t. for all } \vec{a}_i \in A^{n_i},\ 1 \le i \le m,\ \mathfrak{A}, s' \models \varphi_{\vec{b}}, \text{ where} \\ & s' = s(\vec{a}_1/\vec{x}_1, \dots, \vec{a}_m/\vec{x}_m) \text{ and } b_i = g_i(\vec{a}_i),\ 1 \le i \le m. \end{aligned}$$

We denote the set of all such partially-ordered connectives D by D. By $\mathcal{FO}(\mathsf{D})$ we denote the extension of first-order logic by all partially-ordered connectives $D \in \mathsf{D}$, i.e., the syntax of $\mathcal{FO}(\mathsf{D})(\tau)$ is defined from τ by the following grammar:

$$\begin{aligned} \varphi \quad ::= \quad & x_1 = x_2 \mid \neg x_1 = x_2 \mid R(x_1, \dots, x_n) \mid \neg R(x_1, \dots, x_n) \mid \\ & (\varphi \vee \varphi) \mid (\varphi \wedge \varphi) \mid D\vec{\varphi} \mid \neg D\vec{\varphi} \mid \forall x\varphi \mid \exists x\varphi \end{aligned}$$

where $D \in \mathsf{D}$ and $\vec{\varphi}$ is a vector of formulae of the appropriate length. By $\mathcal{FO}(\mathsf{D}^+)$, we denote the fragment of $\mathcal{FO}(\mathsf{D})$ that allows only positive occurrences of partially-ordered connectives, i.e., the syntax of $\mathcal{FO}(\mathsf{D}^+)$ is defined as the syntax for $\mathcal{FO}(\mathsf{D})$ but without the rule $\neg D\vec{\varphi}$. Furthermore, by $\mathsf{D}[\mathcal{FO}]$ and $\mathsf{D}[\mathcal{QF}]$ we denote the logics consisting of formulae of the form

$$D\,(\varphi_{\vec{b}})_{\vec{b}\in\{0,1\}^m},$$

where $D \in \mathsf{D}$ and $\varphi_{\vec{b}}$, for every $\vec{b} \in \{0,1\}^m$, is a formula of first-order logic or a quantifier free formula of first-order logic, respectively. The semantics for these logics is defined in terms of models and assignments in the usual way, i.e., in the same manner as for first-order logic, with the additional clause for the partially-ordered connectives D.

[1]Sandu and Väänänen consider also even more general versions of partially-ordered connectives in which the indices b_i range over a set of size $k \in \mathbb{N}$. However, these can be easily simulated by the partially-ordered connectives described here.

4.3.2 Partially-ordered connectives with Boolean variables

We will deviate from the definitions of [SV92] in two ways: First, we will replace the disjunctions $\bigvee_{b_i \in \{0,1\}}$ by existentially quantified Boolean variables $\exists \alpha_i$; this makes it easier to relate logics with partially-ordered connectives to fragments of Boolean dependence logic. Second, in order to simplify the proofs in Section 4.6, we will relax the restriction that the variables x_{ij} should be distinct. This approach to partially-ordered connectives is closely related to the narrow Henkin quantifier introduced by Blass and Gurevich [BG86].

Definition 4.3.1. *Let $\vec{x}_i = (x_{i1}, \dots, x_{in_i})$, $1 \leq i \leq m$, be tuples of first-order variables, and let α_i, $1 \leq i \leq m$, be distinct Boolean variables. Then*

$$C = \begin{pmatrix} \forall \vec{x}_1 & \exists \alpha_1 \\ \vdots & \vdots \\ \forall \vec{x}_m & \exists \alpha_m \end{pmatrix}$$

is a partially-ordered connective. *The* pattern *of C is $\pi = (n_1, \dots, n_m, E)$, where E describes the identities between the variables in the tuples $\vec{x}_1, \dots, \vec{x}_m$, i.e.,*

$$E = \{(i, j, k, l) \mid 1 \leq i, k \leq m, 1 \leq j \leq n_i, 1 \leq l \leq n_k, x_{ij} = x_{kl}\}.$$

If C is a partially-ordered connective with pattern π, we denote the connective C by $N_\pi \, \vec{x}_1 \alpha_1 \dots \vec{x}_m \alpha_m$.

Definition 4.3.2. *Let τ be a relational vocabulary. Syntax of $\mathcal{FO}(\mathcal{POC})(\tau)$ is defined from τ by the following grammar:*

$$\begin{aligned} \varphi \quad ::= \quad & \alpha \mid \neg\alpha \mid x_1 = x_2 \mid \neg x_1 = x_2 \mid R(x_1, \dots, x_n) \mid \\ & \neg R(x_1, \dots, x_n) \mid (\varphi \vee \varphi) \mid (\varphi \wedge \varphi) \mid \forall x \varphi \mid \exists x \varphi \mid \\ & N_\pi \, \vec{x}_1 \alpha_1 \dots \vec{x}_m \alpha_m \, \varphi \mid \neg N_\pi \, \vec{x}_1 \alpha_1 \dots \vec{x}_m \alpha_m \, \varphi. \end{aligned}$$

For $N_\pi \, \vec{x}_1 \alpha_1 \dots \vec{x}_m \alpha_m \, \varphi$ to be a syntactically correct formula, we require that the identities between the variables in $\vec{x}_1, \dots, \vec{x}_m$ are exactly those described in π. Additionally the Boolean variables $\alpha_1, \dots, \alpha_m$ are all required to be distinct.

$\mathcal{FO}(\mathcal{POC}^+)$ is the syntactic fragment of $\mathcal{FO}(\mathcal{POC})$ that allows only positive occurrences of partially-ordered connectives. In other words, syntax for $\mathcal{FO}(\mathcal{POC}^+)$ is defined by the grammar

$$\begin{aligned} \varphi \quad ::= \quad & \alpha \mid \neg\alpha \mid x_1 = x_2 \mid \neg x_1 = x_2 \mid R(x_1, \dots, x_n) \mid \neg R(x_1, \dots, x_n) \mid \\ & (\varphi \vee \varphi) \mid (\varphi \wedge \varphi) \mid \forall x \varphi \mid \exists x \varphi \mid N_\pi \, \vec{x}_1 \alpha_1 \dots \vec{x}_m \alpha_m \, \varphi. \end{aligned}$$

The fragment $\mathcal{POC}[\mathcal{FO}]$ of $\mathcal{FO}(\mathcal{POC})$ consists of exactly the formulae of the form

$$N_\pi \vec{x}_1\alpha_1 \dots \vec{x}_m\alpha_m \varphi,$$

where $N_\pi \vec{x}_1\alpha_1 \dots \vec{x}_m\alpha_m$ is a partially-ordered connective and $\varphi \in \mathcal{FO}$. Analogously, the logic $\mathcal{POC}[\mathcal{QF}]$ consists of exactly the formulae of the form

$$N_\pi \vec{x}_1\alpha_1 \dots \vec{x}_m\alpha_m \varphi,$$

where $N_\pi \vec{x}_1\alpha_1 \dots \vec{x}_m\alpha_m$ is a partially-ordered connective and φ is a quantifier free formula of $\mathcal{FO}$.

The semantics for these logics is defined in terms of models and assignments in the usual way, i.e., in the same manner as for first-order logic. The clause for formulae starting with a partially-ordered connective $C = N_\pi \vec{x}_1\alpha_1 \dots \vec{x}_m\alpha_m$ with pattern $\pi = (n_1, \dots, n_m, E)$ is the following:

$$\begin{aligned} \mathfrak{A}, s \models C\,\varphi \quad \Leftrightarrow \quad & \text{there exist functions } f_i : A^{n_i} \to \{\bot, \top\}, 1 \le i \le m, \\ & \text{such that for all tuples } \vec{a}_i \in A^{n_i}, 1 \le i \le m : \\ & \text{if } \vec{a}_1 \dots \vec{a}_m \text{ is of pattern } \pi, \text{ then } \mathfrak{A}, s' \models \varphi, \text{where} \\ & s' = s(\vec{a}_1/\vec{x}_1, \dots, \vec{a}_m/\vec{x}_m, f_1(\vec{a}_1)/\alpha_1, \dots, f_m(\vec{a}_m)/\alpha_m). \end{aligned}$$

Here "$\vec{a}_1 \dots \vec{a}_m$ is of pattern π" means that $a_{ij} = a_{kl}$ whenever $(i, j, k, l) \in E$. Note that s' is well-defined for all tuples $\vec{a}_1 \dots \vec{a}_m$ that are of pattern π.

It is straightforward to prove that the expressive powers of $\mathcal{FO}(\mathcal{POC})$, $\mathcal{FO}(\mathcal{POC}^+)$, $\mathcal{POC}[\mathcal{FO}]$ and $\mathcal{POC}[\mathcal{QF}]$ coincide with that of $\mathcal{FO}(\mathsf{D})$, $\mathcal{FO}(\mathsf{D}^+)$, $\mathsf{D}[\mathcal{FO}]$ and $\mathsf{D}[\mathcal{QF}]$, respectively. However the related proofs are quite technical.

Lemma 4.3.3. *For every formula $\varphi \in \mathcal{FO}(\mathsf{D})$ there exists a formula $\varphi^* \in \mathcal{FO}(\mathcal{POC})$ such that for every model $\mathfrak{A}$ and assignment s of $\mathfrak{A}$ it holds that*

$$\mathfrak{A}, s \models \varphi \quad \Leftrightarrow \quad \mathfrak{A}, s \models \varphi^*.$$

Proof. The claim follows by the following translation $\varphi \mapsto \varphi^*$. For atomic and negated atomic formulas the translation is the identity. For propositional connectives and first-order quantifiers the translation is defined in the obvious inductive way, i.e.,

$$\begin{aligned} \neg\varphi &\mapsto \neg\varphi^*, \\ (\varphi \wedge \psi) &\mapsto (\varphi^* \wedge \psi^*), \\ (\varphi \vee \psi) &\mapsto (\varphi^* \vee \psi^*), \\ \exists x\,\varphi &\mapsto \exists x\,\varphi^*, \\ \forall x\,\varphi &\mapsto \forall x\,\varphi^*. \end{aligned}$$

The only nontrivial case is the case for the partially-ordered connectives $D \in \mathsf{D}$. Let $D \in \mathsf{D}$ be the partially-ordered connective

$$\begin{pmatrix} \forall \vec{x}_1 & \bigvee_{b_1 \in \{0,1\}} \\ \vdots & \vdots \\ \forall \vec{x}_m & \bigvee_{b_m \in \{0,1\}} \end{pmatrix}$$

and let $\gamma = (\varphi_{\vec{b}})_{\vec{b} \in \{0,1\}^m}$ be a tuple of $\mathcal{FO}(\mathsf{D})$-formulae. Let π be the pattern that arises from the tuples $\vec{x}_1, \ldots, \vec{x}_m$ of variables. We define that

$$(D\,\gamma)^* := N_\pi\, \vec{x}_1 \alpha_1 \ldots \vec{x}_m \alpha_m \bigwedge_{K \subseteq \{1,\ldots,m\}} \Big((\bigwedge_{i \in K} \alpha_i \wedge \bigwedge_{\substack{1 \leq i \leq m \\ i \notin K}} \neg \alpha_i) \to (\varphi_{\vec{b}_K})^* \Big),$$

where $\vec{b}_K = (b_1, \ldots, b_m) \in \{0,1\}^m$ is the tuple such that $b_i = 1 \;\Leftrightarrow\; i \in K$.

The claim now follows by a simple induction on the structure of the formulae. The only nontrivial case is the case for partially-ordered connectives. Let $D \in \mathsf{D}$ be the partially-ordered connective

$$\begin{pmatrix} \forall \vec{x}_1 & \bigvee_{b_1 \in \{0,1\}} \\ \vdots & \vdots \\ \forall \vec{x}_m & \bigvee_{b_m \in \{0,1\}} \end{pmatrix}$$

and let $\gamma = (\varphi_{\vec{b}})_{\vec{b} \in \{0,1\}^m}$ be a tuple of $\mathcal{FO}(\mathsf{D})$-formulae. Let $\pi = (n_1, \ldots, n_m, E)$ be the pattern that arises from the tuples $\vec{x}_1, \ldots, \vec{x}_m$ of variables. Note that since the variables in $\vec{x}_1, \ldots, \vec{x}_m$ are all distinct the pattern π does not give rise to any nontrivial identities between variables, i.e., other identities than those of the type $x_{ij} = x_{ij}$. Assume that the claim holds for each formula in the tuple γ. By the semantics of the partially-ordered connective D,

$$\mathfrak{A}, s \models D\,\gamma$$

if and only if there exists functions

$$g_i : A^{n_i} \to \{0,1\},$$

$1 \leq i \leq m$, such that for all $\vec{a}_i \in A^{n_i}$, $1 \leq i \leq m$,

$$\mathfrak{A}, s(\vec{a}_1/\vec{x}_1, \ldots, \vec{a}_m/\vec{x}_m) \models \varphi_{\vec{b}},$$

where $\vec{b} = (g_1(\vec{a}_1), \ldots, g_m(\vec{a}_m))$. By the induction hypothesis, this holds if and only if there exists functions

$$g_i : A^{n_i} \to \{0,1\},$$

$1 \leq i \leq m$, such that for all $\vec{a}_i \in A^{n_i}$, $1 \leq i \leq m$,

$$\mathfrak{A}, s(\vec{a}_1/\vec{x}_1, \ldots, \vec{a}_m/\vec{x}_m) \models \varphi^*_{\vec{b}},$$

where $\vec{b} = \big(g_1(\vec{a}_1), \ldots, g_m(\vec{a}_m)\big)$. At this point it is useful to note that functions from A^{n_i} to $\{0,1\}$ and functions from A^{n_i} to $\{\bot, \top\}$ are essentially the same functions. Also note that a tuple $\vec{a}_1 \ldots \vec{a}_m$ is always of pattern π. Hence the above holds if and only if there exists functions

$$f_i : A^{n_i} \to \{\bot, \top\},$$

$1 \leq i \leq m$, such that for all $\vec{a}_i \in A^{n_i}$, $1 \leq i \leq m$, if $\vec{a}_1 \ldots \vec{a}_m$ is of pattern π, then

$$\mathfrak{A}, s' \models \bigwedge_{K \subseteq \{1,\ldots,m\}} \Big(\big(\bigwedge_{i \in K} \alpha_i \wedge \bigwedge_{\substack{1 \leq i \leq m \\ i \notin K}} \neg \alpha_i \big) \to (\varphi_{\vec{b}_K})^* \Big),$$

where

$$s' = s(\vec{a}_1/\vec{x}_1, \ldots, \vec{a}_m/\vec{x}_m, f_1(\vec{a}_1)/\alpha_1, \ldots, f_m(\vec{a}_m)/\alpha_m)$$

and $\vec{b}_K = (b_1, \ldots, b_m) \in \{0,1\}^m$ is the tuple such that $b_i = 1 \;\Leftrightarrow\; i \in K$. Furthermore, by the semantics of the partially-ordered connective N_π, the above holds if and only if

$$\mathfrak{A}, s \models N_\pi \, \vec{x}_1 \alpha_1 \ldots \vec{x}_m \alpha_m \bigwedge_{K \subseteq \{1,\ldots,m\}} \Big(\big(\bigwedge_{i \in K} \alpha_i \wedge \bigwedge_{\substack{1 \leq i \leq m \\ i \notin K}} \neg \alpha_i \big) \to (\varphi_{\vec{b}_K})^* \Big).$$

□

Lemma 4.3.4. *For every formula $\varphi \in \mathcal{FO}(\mathcal{POC})$ without free Boolean variables there exists a formula $\varphi^+ \in \mathcal{FO}(\mathsf{D})$ such that for every model $\mathfrak{A}$ and assignment s of $\mathfrak{A}$ it holds that*

$$\mathfrak{A}, s \models \varphi \quad \Leftrightarrow \quad \mathfrak{A}, s \models \varphi^+.$$

Proof. The claim follows by the following translation $\varphi \mapsto \varphi^+$. Note that the way we handle partially-ordered connectives ensures that we do not need a clause for Boolean variables nor for negated Boolean variables. For atomic and negated atomic formulae the translation is the identity. For propositional connectives and first-order quantifiers the translation is defined in the obvious

inductive way, i.e.,

$$\begin{aligned} \neg\varphi &\mapsto \neg\varphi^+, \\ (\varphi \wedge \psi) &\mapsto (\varphi^+ \wedge \psi^+), \\ (\varphi \vee \psi) &\mapsto (\varphi^+ \vee \psi^+), \\ \exists x\, \varphi &\mapsto \exists x\, \varphi^+, \\ \forall x\, \varphi &\mapsto \forall x\, \varphi^+. \end{aligned}$$

The only nontrivial case is the case for the partially-ordered connectives N_π. Let $\pi = (n_1, \ldots, n_m, E)$ be a pattern and let

$$N_\pi\, \vec{x}_1 \alpha_1 \ldots \vec{x}_m \alpha_m \psi$$

be an $\mathcal{FO}(\mathcal{POC})$-formula without free Boolean variables. Furthermore, let $\vec{y}_1, \ldots, \vec{y}_m$ be tuples of distinct fresh variables such that $\vec{y}_i = \{y_{i1}, \ldots, y_{in_i}\}$, for every $i \leq m$. We define that

$$(N_\pi\, \vec{x}_1 \alpha_1 \ldots \vec{x}_m \alpha_m\, \psi)^+ := \begin{pmatrix} \forall \vec{y}_1 & \bigvee_{b_1 \in \{0,1\}} \\ \vdots & \vdots \\ \forall \vec{y}_m & \bigvee_{b_m \in \{0,1\}} \end{pmatrix} (\psi^+_{\vec{b}})_{\vec{b} \in \{0,1\}^m},$$

where, for each $\vec{b} = (b_1, \ldots, b_m) \in \{0,1\}^m$, $\psi_{\vec{b}}$ is the formula

$$\Big(\bigwedge_{(i,j,k,l) \in E} y_{ij} = y_{kl} \Big) \to \psi'(\vartheta_{b_1}/\alpha_1, \ldots, \vartheta_{b_m}/\alpha_m),$$

where ψ' is the formula obtained from ψ by replacing each free occurrence of the variable x_{ij} by some variable y_{kl} such that $(i,j,k,l) \in E$, ϑ_0 is the formula $\neg(y_{11} = y_{11})$ and ϑ_1 is the formula $y_{11} = y_{11}$.

The claim now follows by a simple induction on the nesting depth of partially-ordered connectives in formulae, i.e., the highest number of nested partially-ordered connectives in formulae. The case without partially-ordered connectives is trivial. The cases for first-order operations are easy. The only nontrivial case is the case for partially-ordered connectives. Let

$$N_\pi\, \vec{x}_1 \alpha_1 \ldots \vec{x}_m \alpha_m$$

be a partially-ordered connective with pattern $\pi = (n_1, \ldots, n_m, E)$ and let ψ be a $\mathcal{FO}(\mathcal{POC})$-formula such that

$$\varphi := N_\pi\, \vec{x}_1 \alpha_1 \ldots \vec{x}_m \alpha_m\, \psi$$

does not have free Boolean variables. Furthermore, assume that the claim holds for every $\mathcal{FO}(\mathcal{POC})$-formula that does not have free boolean variables and has a lower nesting depth of partially-ordered connectives than φ. Let $\vec{y}_1, \ldots, \vec{y}_m$ be tuples of distinct fresh variables not occurring in φ such that $\vec{y}_i = \{y_{i1}, \ldots, y_{in_i}\}$, for every $i \leq m$. Now, by the semantics of the partially-ordered connective N_π,

$$\mathfrak{A}, s \models N_\pi \, \vec{x}_1 \alpha_1 \ldots \vec{x}_m \alpha_m \, \psi$$

if and only if there exist functions

$$f_i : A^{n_i} \to \{\bot, \top\},$$

$1 \leq i \leq m$, such that for all tuples $\vec{a}_i \in A^{n_i}$, $1 \leq i \leq m$, if $\vec{a}_1 \ldots \vec{a}_m$ is of pattern π then

$$\mathfrak{A}, s' \models \psi,$$

where $s' = s(\vec{a}_1/\vec{x}_1, \ldots, \vec{a}_m/\vec{x}_m, f_1(\vec{a}_1)/\alpha_1, \ldots, f_m(\vec{a}_m)/\alpha_m)$. Clearly this holds if and only if there exist functions

$$f_i : A^{n_i} \to \{\bot, \top\},$$

$1 \leq i \leq m$, such that for all tuples $\vec{a}_i \in A^{n_i}$, $1 \leq i \leq m$,

$$\mathfrak{A}, t' \models (\bigwedge_{(i,j,k,l) \in E} y_{ij} = y_{kl}) \to \psi',$$

where $t' = s(\vec{a}_1/\vec{y}_1, \ldots, \vec{a}_m/\vec{y}_m, f_1(\vec{a}_1)/\alpha_1, \ldots, f_m(\vec{a}_m)/\alpha_m)$ and ψ' is obtained from ψ by replacing each variable x_{ij} occurring free in ψ by some variable y_{kl} such that $(i, j, k, l) \in E$. At this point it is helpful to note that functions from A^{n_i} to $\{\bot, \top\}$ and functions from A^{n_i} to $\{0, 1\}$ are essentially the same functions. Hence the above holds if and only if there exist functions

$$g_i : A^{n_i} \to \{0, 1\},$$

$1 \leq i \leq m$, such that for all tuples $\vec{a}_i \in A^{n_i}$, $1 \leq i \leq m$,

$$\mathfrak{A}, t \models (\bigwedge_{(i,j,k,l) \in E} y_{ij} = y_{kl}) \to \psi'(\vartheta_{b_1}/\alpha_1, \ldots, \vartheta_{b_m}/\alpha_m),$$

where $t = s(\vec{a}_1/\vec{y}_1, \ldots, \vec{a}_m/\vec{y}_m)$, $b_i = g_i(\vec{a}_i)$ for each $i \leq m$, ϑ_0 denotes the formula $\neg(y_{11} = y_{11})$ and ϑ_1 denotes the formula $y_{11} = y_{11}$. Remember that for $\vec{b} = (b_1, \ldots, b_m) \in \{0, 1\}^m$ we defined that

$$\psi_{\vec{b}} = (\bigwedge_{(i,j,k,l) \in E} y_{ij} = y_{kl}) \to \psi'(\vartheta_{b_1}/\alpha_1, \ldots, \vartheta_{b_m}/\alpha_m).$$

Hence, by the inductive hypothesis, the above holds if and only if there exist functions

$$g_i : A^{n_i} \to \{0,1\},$$

$1 \leq i \leq m$, such that for all tuples $\vec{a}_i \in A^{n_i}$, $1 \leq i \leq m$,

$$\mathfrak{A}, t \models \psi^+_{\vec{b}}$$

where $t = s(\vec{a}_1/\vec{y}_1, \ldots, \vec{a}_m/\vec{y}_m)$ and $\vec{b} = (g_1(\vec{a}_1), \ldots, g_m(\vec{a}_m))$. Finally, by the semantics of the partially-ordered connectives D, this holds if and only if

$$\mathfrak{A}, s \models \begin{pmatrix} \forall \vec{y}_1 & \bigvee_{b_1 \in \{0,1\}} \\ \vdots & \vdots \\ \forall \vec{y}_m & \bigvee_{b_m \in \{0,1\}} \end{pmatrix} (\psi^+_{\vec{b}})_{\vec{b} \in \{0,1\}^m}.$$

□

Proposition 4.3.5. $\mathcal{FO}(\mathcal{POC}) \equiv \mathcal{FO}(\mathsf{D})$.

Proof. Follows directly from Lemmas 4.3.3 and 4.3.4. □

The translations defined in Lemmas 4.3.3 and 4.3.4 directly yield the following equivalences between fragments of $\mathcal{FO}(\mathcal{POC})$ and $\mathcal{FO}(\mathsf{D})$ as well.

Proposition 4.3.6. *The following equivalences hold:* $\mathcal{FO}(\mathsf{D}^+) \equiv \mathcal{FO}(\mathcal{POC}^+)$, $\mathcal{POC}[\mathcal{FO}] \equiv \mathsf{D}[\mathcal{FO}]$ *and* $\mathcal{POC}[\mathcal{QF}] \equiv \mathsf{D}[\mathcal{QF}]$.

Hella, Sevenster and Tulenheimo [HST08] established that $\mathcal{FO}(\mathsf{D})$ has the zero-one law. Thus, by Proposition 4.3.5, the same is true for $\mathcal{FO}(\mathcal{POC})$ and all its fragments as well.

Theorem 4.3.7. $\mathcal{FO}(\mathcal{POC})$ *has the zero-one law. Therefore,* $\mathcal{FO}(\mathcal{POC}^+)$, $\mathcal{POC}[\mathcal{FO}]$ *and* $\mathcal{POC}[\mathcal{QF}]$ *also have the zero-one law.*

4.4 Fragments of Boolean dependence logic

In this section we define fragments of Boolean dependence logic that correspond to the fragments $\mathcal{FO}(\mathcal{POC}^+)$, $\mathcal{POC}[\mathcal{FO}]$ and $\mathcal{POC}[\mathcal{QF}]$ of $\mathcal{FO}(\mathcal{POC})$, with respect to expressive power. We restrict our attention to sentences, i.e., to formulae without free variables.

We will first demonstrate an exemplary translation from $\mathcal{FO}(\mathcal{POC}^+)$ to Boolean dependence logic in order to visualize the fragments of $\mathcal{BD}$ that correspond to $\mathcal{FO}(\mathcal{POC}^+)$, $\mathcal{POC}[\mathcal{FO}]$ and $\mathcal{POC}[\mathcal{QF}]$, respectively. Let φ be the $\mathcal{FO}(\mathcal{POC}^+)$-sentence

$$\forall x_0 \forall x_1 \exists x_2 \begin{pmatrix} \forall \vec{y} & \exists \alpha \\ \forall \vec{z} & \exists \beta \end{pmatrix} \psi,$$

where ψ is syntactically first-order. Clearly φ is equivalent to the sentence

$$\vartheta := \forall x_0 \forall x_1 \exists x_2 \forall \vec{y} \forall \vec{z} \exists \alpha \exists \beta (=(x_0, x_1, x_2, \vec{y}, \alpha) \wedge =(x_0, x_1, x_2, \vec{z}, \beta) \wedge \psi)$$

of Boolean dependence logic. After examining the syntactic form of ϑ we notice some regularity in the Boolean dependence atoms of ϑ. The existential first-order quantifier $\exists x_2$ seems to partition the set of first-order variables quantified in ϑ into two parts. Every variable before the quantifier $\exists x_2$ including the variable x_2 itself are in the antecedent of every Boolean dependence atom in ϑ. The variables after $\exists x_2$ are not in the antecedent of every Boolean dependence atom of ϑ. This observation leads us to the following somewhat technical definitions.

Definition 4.4.1. *Let φ be a formula of $\mathcal{BD}$ or $\mathcal{FO}(\mathcal{POC})$, ψ a subformula of φ and $n \in \mathbb{N}$. Note that we may consider formulae as just strings of symbols of some prescribed vocabulary. Hence, we may order the occurrences of subformulae of a given formula. By $[\psi, \varphi]_n$ we denote the nth occurrence of the formula ψ in φ. If there is just one occurrence of ψ in φ, we may drop the subscript and just write $[\psi, \varphi]$.*

Definition 4.4.2. *Let φ be a formula of $\mathcal{BD}$ or $\mathcal{FO}(\mathcal{POC})$, and $[\psi, \varphi]_n$ an occurrence of the subformula ψ of φ. We define $V([\psi, \varphi]_n)$ to be the set of all first-order variables x such that x is free in φ or the occurrence $[\psi, \varphi]_n$ of ψ is in a scope of $\forall x$ or $\exists x$ in φ.*

Hence, for sentences $V([\psi, \varphi]_n)$ is the set of quantified variables that dominate $[\psi, \varphi]_n$ in the syntactic tree of φ. We are now ready to define the fragments of $\mathcal{BD}$ that correspond to $\mathcal{FO}(\mathcal{POC}^+)$, $\mathcal{POC}[\mathcal{FO}]$ and $\mathcal{POC}[\mathcal{QF}]$, respectively.

Definition 4.4.3. *We define the following syntactic fragments of Boolean dependence logic.*

1. Bounded Boolean dependence logic, *$\mathcal{BBD}$, is the syntactic restriction of $\mathcal{BD}$ to formulae φ such that the following condition holds.*

If $[\exists x\psi, \varphi]_n$ *is an occurrence of a subformula* $\exists x\psi$ *of* φ *and a dependence atom* $=(x_1, \ldots, x_n, \alpha)$ *is a subformula of* ψ *then*

$$V([\psi, \varphi]_t) \subseteq \{x_1, \ldots, x_n\},$$

where $[\psi, \varphi]_t$ *is the occurrence of* ψ *that occurs in* $[\exists x\psi, \varphi]_n$.

2. Restricted Boolean dependence logic, *$\mathcal{RBD}$, is the restriction of $\mathcal{BD}$ to formulae where no dependence atoms occur inside the scope of an existential first-order quantifier.*

3. Universal Boolean dependence logic, *$\forall$-$\mathcal{BD}$, is the restriction of $\mathcal{BD}$ to formulae without existential quantification of first-order variables.*

In Section 4.6 we establish that the above definitions indeed fit for our purposes. We show that

$$\forall\text{-}\mathcal{BD} \equiv \mathcal{POC}[\mathcal{QF}],\ \mathcal{RBD} \equiv \mathcal{POC}[\mathcal{FO}] \text{ and } \mathcal{BBD} \equiv \mathcal{FO}(\mathcal{POC}^+).$$

It is easy to see, that every $\forall$-$\mathcal{BD}$ formula is an $\mathcal{RBD}$ formula, every $\mathcal{RBD}$ formula is a $\mathcal{BBD}$ formula and every $\mathcal{BBD}$ formula is a $\mathcal{BD}$ formula. Hence, once we realize that every $\mathcal{BD}$-sentence can be simulated by a sentence of dependence logic, we obtain the following hierarchy.

Proposition 4.4.4. $\forall\text{-}\mathcal{BD} \leq \mathcal{RBD} \leq \mathcal{BBD} \leq \mathcal{BD} \leq \mathcal{D}$.

Proof. The first three inclusions follow by the observation made above. For the last inclusion we give a translation $\varphi \mapsto \varphi^+$ that maps $\mathcal{BD}$-sentences to equivalent $\mathcal{D}$ sentences. We will establish that for every $\mathcal{BD}$-sentence φ and every structure $\mathfrak{A}$ it holds that

$$\mathfrak{A} \models \varphi \quad \text{iff} \quad \mathfrak{A} \models \varphi^+. \tag{4.1}$$

For each Boolean variable α and for the symbols $\bot$ and $\top$, we introduce distinct fresh first-order variables x_α, $x_\bot$ and $x_\top$. Without loss of generality, we may assume that these first-order variables do not appear in the formulae of Boolean dependence logic. We may, without loss of generality, restrict our attention to models with at least two elements. For a sentence $\varphi \in \mathcal{BD}$ we define that

$$\varphi^+ := \exists x_\bot \exists x_\top (x_\bot \neq x_\top \wedge \varphi^*),$$

where φ^* is the sentence obtained from φ by the following recursive translation. For first-order literals, the translation is the identity. The remaining clauses are as follows:

$$\begin{aligned}
=(x_1,\dots,x_k,\alpha)^* &:= =(x_1,\dots,x_k,x_\alpha),\\
\alpha^* &:= x_\alpha = x_\top,\\
(\neg\alpha)^* &:= x_\alpha = x_\bot,\\
(\varphi\wedge\psi)^* &:= (\varphi^*\wedge\psi^*),\\
(\varphi\vee\psi)^* &:= (\varphi^*\vee\psi^*),\\
(\exists\alpha\varphi)^* &:= \exists x_\alpha\varphi^*,\\
(\exists x\varphi)^* &:= \exists x\varphi^*,\\
(\forall x\varphi)^* &:= \forall x\varphi^*.
\end{aligned}$$

Clearly, if φ is $\mathcal{BD}$-sentence then φ^+ is a $\mathcal{D}$-sentence. Furthermore, it is easy to see that (4.1) holds for every $\mathcal{BD}$-sentence φ and every model $\mathfrak{A}$ of cardinality at least two. □

4.5 Dependence normal form

In this section we define a normal form for bounded Boolean dependence logic and show that for each $\mathcal{BBD}$-formula there exists an equivalent $\mathcal{BBD}$-formula in this normal form. We use this normal form in Section 4.6 to establish a translation from $\mathcal{BBD}$ to $\mathcal{FO}(\mathcal{POC}^+)$. As a byproduct we also obtain translations from $\mathcal{RBD}$ into $\mathcal{POC}[\mathcal{FO}]$ and from $\forall$-$\mathcal{BD}$ into $\mathcal{POC}[\mathcal{QF}]$.

We start by introducing a normal form that does not allow any reuse of variables.

Definition 4.5.1. *A formula $\varphi\in\mathcal{BD}$ is in* variable normal form *if no variable in* $\mathrm{fr}(\varphi)$ *is quantified in φ, and if each variable quantified in φ is quantified exactly once.*

Note that, if a $\mathcal{BD}$-formula φ is in variable normal form and ψ is a subformula of φ that has at least one quantifier in it then φ has exactly one occurrence of the subformula ψ.

Lemma 4.5.2. *For every $\mathcal{BD}$ ($\mathcal{BBD}$, $\mathcal{RBD}$, $\forall$-$\mathcal{BD}$, respectively) formula there exists an equivalent $\mathcal{BD}$ ($\mathcal{BBD}$, $\mathcal{RBD}$, $\forall$-$\mathcal{BD}$, respectively) formula in variable normal form.*

Proof. Follows from Proposition 4.2.1 and Lemmas 4.2.3 and 4.2.4. □

The following normal form can be seen as a kind of a local prenex normal form for $\mathcal{BBD}$. The idea is that each universal first-order and existential Boolean quantifier is pulled toward a preceding existential first-order quantifier and then using Boolean dependence atoms each universal first-order quantifier is pulled past the preceding Boolean quantifiers.

Definition 4.5.3. *A sentence $\varphi \in \mathcal{BD}$ is in* Q-normal form *if φ is in variable normal form and there exists a formula $\vartheta \in \mathcal{BD}$ such that the following holds.*

1. *$\varphi = \forall\vec{x}\exists\vec{\alpha}\vartheta$, for some (possibly empty) block of universal quantifiers $\forall\vec{x}$ followed by a (possibly empty) block of existential Boolean quantifiers $\exists\vec{\alpha}$.*
2. *Each quantifier in ϑ occurs in some block of quantifiers $\exists\vec{x}\forall\vec{y}\exists\vec{\alpha}$, where at least $\vec{x}$ is nonempty.*

The following lemmas are used to prove the Q-normal form for $\mathcal{BBD}$, i.e., Proposition 4.5.6.

Lemma 4.5.4. *Let φ and ϑ be formulae of Boolean dependence logic such that $x, \alpha \notin \mathrm{fr}(\vartheta)$. The following equivalences hold.*

1. $(\forall x\varphi \vee \vartheta) \equiv \forall x(\varphi \vee \vartheta)$.
2. $(\forall x\varphi \wedge \vartheta) \equiv \forall x(\varphi \wedge \vartheta)$.
3. $(\exists\alpha\varphi \vee \vartheta) \equiv \exists\alpha(\varphi \vee \vartheta)$.
4. $(\exists\alpha\varphi \wedge \vartheta) \equiv \exists\alpha(\varphi \wedge \vartheta)$.

Proof. Each claim follows straightforwardly from Proposition 4.2.1. We prove here claim 1. Claims 2–4 are completely analogous.

Let $\mathfrak{A}$ be a model and X a team of A. The claim follows from the following chain of equivalences.

$\mathfrak{A} \models_X (\forall x\varphi \vee \vartheta)$
$\Leftrightarrow \mathfrak{A} \models_Y \forall x\varphi$ and $\mathfrak{A} \models_Z \vartheta$, for some Y and Z such that $Y \cup Z = X$
$\Leftrightarrow \mathfrak{A} \models_{Y(A/x)} \varphi$ and $\mathfrak{A} \models_{Z(A/x)} \vartheta$, for some Y and Z such that $Y \cup Z = X$
$\Leftrightarrow \mathfrak{A} \models_{Y'} \varphi$ and $\mathfrak{A} \models_{Z'} \vartheta$, for some Y' and Z' such that $Y' \cup Z' = X(A/x)$
$\Leftrightarrow \mathfrak{A} \models_{X(A/x)} (\varphi \vee \vartheta)$
$\Leftrightarrow \mathfrak{A} \models_X \forall x(\varphi \vee \vartheta)$.

The first and the fourth equivalence is due to the semantics of disjunctions. The second equivalence follows from the semantics of universal quantifiers, Proposition 4.2.1 and the fact that $x \notin \mathrm{fr}(\vartheta)$. The third equivalence follows from the observation that $Y(A/x) \cup Z(A/x) = X(A/x)$, from Proposition 4.2.1 and the fact that $x \notin \mathrm{fr}(\vartheta)$. Finally, the last equivalence is due to the semantics of universal quantifiers. □

Lemma 4.5.5. *Let φ be a $\mathcal{BD}$-sentence and $\psi = \exists\alpha\forall\vec{x}\,\exists\vec{\beta}\,\vartheta$ a subformula of φ. Let $[\psi,\varphi]_n$ denote an occurrence of ψ in φ and let φ^* denote the formula obtained form φ by substituting the occurrence $[\psi,\varphi]_n$ of ψ by*

$$\forall\vec{x}\,\exists\alpha\exists\vec{\beta}\,(=\!(V([\psi,\varphi]_n),\alpha) \wedge \vartheta).$$

Then $\varphi \equiv \varphi^*$.

Proof. Straightforward. □

We are now ready to prove that for every $\mathcal{BBD}$-sentence there exists an equivalent $\mathcal{BBD}$-sentence in Q-normal form.

Proposition 4.5.6. *For every $\mathcal{BBD}$-sentence there exists an equivalent $\mathcal{BBD}$-sentence in Q-normal form.*

Proof. Let $\varphi \in \mathcal{BBD}$ be a sentence. By Lemma 4.5.2 we can assume that φ is in variable normal form. We will give an algorithm that transforms φ into an equivalent $\mathcal{BBD}$-sentence in Q-normal form.

We will first transform φ to an equivalent $\mathcal{BBD}$-sentence φ^* such that

$$\varphi^* = \overrightarrow{Q\xi}\psi, \tag{4.2}$$

where $\overrightarrow{Q\xi}$ is a (possibly empty) vector of universal first-order and existential Boolean quantifiers. Furthermore, in ψ every universal first-order or existential Boolean quantifier $Q\chi$ occurs in a subformula ϑ of ψ such that

$$\vartheta = Q'\eta Q\chi\gamma,$$

where $Q'\eta$ is a quantifier and γ is a $\mathcal{BBD}$ formula. In order to obtain φ^* from φ, we use the equivalences from Lemma 4.5.4 repetitively substituting subformulae with equivalent subformulae. More precisely, there exists a natural number $n \in \mathbb{N}$ and a tuple $(\varphi_i)_{i\leq n}$ of $\mathcal{BBD}$-sentences such that $\varphi_0 = \varphi$ and $\varphi_n = \varphi^*$. Furthermore,

1. for each $i < n$ there exist subformulae ϑ, ψ_1 and ψ_2 of φ_i such that

$$\vartheta = (Q\chi\psi_1 \otimes \psi_2) \text{ (or } \vartheta = (\psi_1 \otimes Q\chi\psi_2) \text{)},$$

   where $Q\chi \in \{\forall x, \exists\alpha\}$ and $\otimes \in \{\vee, \wedge\}$, and φ_{i+1} is obtained from φ_i by substituting ϑ by $Q\chi(\psi_1 \otimes \psi_2)$.

It is easy to see, that for each $\mathcal{BFD}$-sentence the substitution procedure described in 1 terminates, i.e., there exists some $n \in \mathbb{N}$ such that there are no subformulae ϑ, ψ_1 and ψ_2 of φ_n such that the substitution described in 1 can be carried out. Clearly the sentence φ_n is then in the form described in (4.2). By induction it is easy to show, that since φ_0 is in variable normal form it follows that φ_i is in variable normal form, for all $i \geq 0$. Hence, the assumptions on free variables needed for Lemma 4.5.4 hold for each φ_i. By Lemmas 4.5.4 and 4.2.3, we conclude that, for each $i < n$, the sentences φ_i and φ_{i+1} are equivalent. Hence the sentences φ_0 and φ_n are equivalent.

We still need to transform the sentence φ^* into an equivalent sentence φ' in Q-normal form. In order to obtain φ' from φ^* we use the equivalence from Lemma 4.5.5 repetitively. More precisely, there exists a natural number $m \in \mathbb{N}$ and a tuple $(\varphi_i^*)_{i \leq m}$ of $\mathcal{BFD}$-sentences such that $\varphi_0^* = \varphi^*$ and $\varphi_m^* = \varphi'$. Furthermore,

2. for each $i < m$ there exists subformulae ϑ and ψ of φ_i^*, and a quantifier $\exists\alpha$ such that

$$\vartheta = \exists\alpha\forall\vec{x}\exists\vec{\beta}\psi,$$

   where $\forall\vec{x}$ is a nonempty vector of universal first-order quantifiers and ψ does not start with a Boolean existential or universal first-order quantifier, and φ_{i+1}^* is obtained from φ_i^* by substituting ϑ by

$$\forall\vec{x}\,\exists\alpha\,\exists\vec{\beta}\,(=\!(V([\vartheta, \varphi_i^*]), \alpha) \wedge \psi).$$

By Lemmas 4.5.5 and 4.2.3, we conclude that, for each i, the sentences φ_i^* and φ_{i+1}^* are equivalent. It is easy to see that, for each $\mathcal{BFD}$-sentence the substitution procedure described above terminates, i.e., there exists some $m \in \mathbb{N}$ such that there are no subformulae of φ_m^* that can be substituted as described in 2. Now clearly φ_m^* is in Q-normal form. □

We are finally ready to define dependence normal form. The idea here is that a $\mathcal{BFD}$-sentence in Q-normal form is in dependence normal form if there is one-to-one correspondence between Boolean existential quantifiers and Boolean

dependence atoms such that each quantifier $\exists\alpha$ is immediately followed by the corresponding dependence atom $=(\vec{x},\alpha)$, and conversely each dependence atom is directly preceded by the corresponding Boolean quantifier.

Definition 4.5.7. *A sentence $\varphi \in \mathcal{BD}$ is in* dependence normal form *if*

1. *φ is in Q-normal form,*
2. *for every Boolean variable α it holds that if $[=(\vec{x},\alpha),\varphi]_t$ and $[=(\vec{y},\alpha),\varphi]_l$ are occurrences in φ then $[=(\vec{x},\alpha),\varphi]_t = [=(\vec{y},\alpha),\varphi]_l$,*
3. *for every maximal nonempty block of Boolean existential quantifiers $\exists\vec{\alpha}$ in φ there exists a subformula*

$$\exists\vec{\alpha}\Big(\big(\bigwedge_{1\leq i\leq n} =(\vec{x}_i,\alpha_i)\big)\wedge\psi\Big)$$

of φ such that the Boolean variables α_i, $1 \leq i \leq n$, are exactly the variables quantified in $\exists\vec{\alpha}$.

To simplify the proof of Proposition 4.5.11, we introduce the concepts of evaluation and satisfying evaluation.

Definition 4.5.8. *Let $\mathfrak{A}$ be a model, X a team of $\mathfrak{A}$ and φ a $\mathcal{BD}$-formula. Define that*

$$SubOc(\varphi) := \{[\psi,\varphi]_t \mid [\psi,\varphi]_t \text{ is an occurrence of } \psi \text{ in } \varphi\}$$
$$Teams(\mathfrak{A}) := \{Y \mid Y \text{ is a team of } \mathfrak{A}\}.$$

We say that a function $e : SubOc(\varphi) \to Teams(\mathfrak{A})$ is an evaluation *of φ on the model $\mathfrak{A}$ and team X if the following recursive conditions hold.*

1. *$e([\varphi,\varphi]) = X$.*
2. *If $e([\psi\vee\vartheta,\varphi]_t) = Y$, and $[\psi,\varphi]_n$ and $[\vartheta,\varphi]_m$ are the occurrences of ψ and ϑ in $[\psi\vee\vartheta,\varphi]_t$, then there exists Y_0 and Y_1 such that $Y_0 \cup Y_1 = Y$ and $e([\psi,\varphi]_n) = Y_0$ and $e([\vartheta,\varphi]_m) = Y_1$.*
3. *If $e([\psi\wedge\vartheta,\varphi]_t) = Y$, and $[\psi,\varphi]_n$ and $[\vartheta,\varphi]_m$ are the occurrences of ψ and ϑ in $[\psi\wedge\vartheta,\varphi]_t$, then $e([\psi,\varphi]_n) = Y$ and $e([\vartheta,\varphi]_m) = Y$.*
4. *If $e([\exists x\psi,\varphi]_t) = Y$ and $[\psi,\varphi]_n$ is the occurrence of ψ in $[\exists x\psi,\varphi]_t$ then $e([\psi,\varphi]_n) = Y(F/x)$ for some function $F : Y \to A$.*

5. *If* $e([\exists\alpha\psi,\varphi]_t) = Y$ *and* $[\psi,\varphi]_n$ *is the occurrence of* ψ *in* $[\exists\alpha\psi,\varphi]_t$ *then* $e([\psi,\varphi]_n) = Y(F/\alpha)$ *for some function* $F : Y \to \{\bot,\top\}$.

6. *If* $e([\forall x\psi,\varphi]_t) = Y$ *and* $[\psi,\varphi]_n$ *is the occurrence of* ψ *in* $[\forall x\psi,\varphi]_t$ *then* $e([\psi,\varphi]_n) = Y(A/x)$.

We say that the evaluation e *is an* successful evaluation *if for each occurrence* $[\psi,\varphi]_t$ *such that* ψ *is a Boolean dependence atom or a literal*

$$\mathfrak{A} \models_{e([\psi,\vartheta]_t)} \psi.$$

The following results follow directly from the semantics of Boolean dependence logic and the definition of successful evaluation.

Proposition 4.5.9. *Let* $\mathfrak{A}$ *be a model,* X *a team of* $\mathfrak{A}$*,* φ *a* $\mathcal{BD}$*-formula and* e *a successful evaluation of* φ *on the model* $\mathfrak{A}$ *and team* X*. For every occurrence* $[\psi,\varphi]_t \in SubOc(\varphi)$

$$\mathfrak{A} \models_{e([\psi,\varphi]_t)} \psi.$$

Theorem 4.5.10. *Let* $\mathfrak{A}$ *be a model,* X *a team of* $\mathfrak{A}$ *and* φ *a* $\mathcal{BD}$*-formula. The following are equivalent:*

1. $\mathfrak{A} \models_X \varphi$.

2. *There exists a successful evaluation of* φ *on the model* $\mathfrak{A}$ *and team* X*.*

Hence the concepts of satisfaction and successful evaluation coincide. We are now ready to prove that every $\mathcal{BBD}$-sentence there exists an equivalent $\mathcal{BBD}$-sentence in dependence normal form. The proof is quite long and technical.

Proposition 4.5.11. *For every* $\mathcal{BBD}$*-sentence there exists an equivalent* $\mathcal{BBD}$*-sentence in dependence normal form.*

Proof. Let $\varphi \in \mathcal{BBD}$ be a sentence. By Proposition 4.5.6, we may assume that φ is in Q-normal form. We will give an algorithm that transforms φ to an equivalent $\mathcal{BBD}$-sentence φ^+ in dependence normal form. We show that there exists a natural number $n \in \mathbb{N}$ and a tuple $(\varphi_i)_{i\leq n}$ of equivalent $\mathcal{BBD}$-sentences in Q-normal form such that $\varphi_0 = \varphi$ and $\varphi_n = \varphi^+$. The sentence φ_{i+1} is obtained from φ_i by the procedure described below.

Assume that φ_i is not in dependence normal form. Assume first that this is due to the fact that there exists an occurrence

$$[=(\vec{x},\alpha),\varphi_i]_t$$

of some dependence atom $=(\vec{x},\alpha)$ in φ_i that violates the conditions of Definition 4.5.7, i.e., there exists some other occurrence of a dependence atom with a consequent α or $[=(\vec{x},\alpha),\varphi_i]_t$ is not a conjunct in a conjunction of dependence atoms immediately following a block of existentially quantified Boolean variables in which $\exists\alpha$ occurs. Hence, there exists a formula ψ and maximal quantifier blocks $\exists\vec{y}$, $\forall\vec{z}$ and $\exists\vec{\beta}$ such that $\exists\vec{\beta}$ is nonempty and

$$\vartheta := \exists\vec{y}\forall\vec{z}\exists\vec{\beta}\,\psi$$

is a subformula of φ_i, where the occurrence $[=(\vec{x},\alpha),\varphi_i]_t$ of $=(\vec{x},\alpha)$ is a subformula of ψ and is not bound by any quantifier in ψ. Let U denote the set of variables that are in both $\vec{x}$ and $\vec{z}$. By $\vec{u}$ we denote the canonical ordering of the variables in U. Since φ_i is a $\mathcal{BBD}$-sentence, the variables in $\vec{x}$ are exactly those that are in $V([\forall\vec{z}\exists\vec{\beta}\,\psi,\varphi_i])\cup U$. Hence the formulae

$$=(V([\forall\vec{z}\exists\vec{\beta}\,\psi,\varphi_i])\cup U,\alpha) \quad \text{and} \quad =(\vec{x},\alpha)$$

are equivalent. Therefore, due to Lemma 4.2.3, we may assume that $=(\vec{x},\alpha)$ is $=(V([\forall\vec{z}\exists\vec{\beta}\,\psi,\varphi_i])\cup U,\alpha)$.

Let $\vec{w}$ be a tuple of fresh distinct first-order variables of the same length as $\vec{u}$, W the set of variables in $\vec{w}$ and β' a fresh Boolean variable. Define then that

$$\vartheta' := \exists\vec{y}\forall\vec{z}\forall\vec{w}\exists\vec{\beta}\exists\beta'\,(=(V([\forall\vec{z}\exists\vec{\beta}\,\psi,\varphi_i])\cup W,\beta')\wedge\psi'),$$

where ψ' is obtained from ψ by substituting the occurrence $[=(\vec{x},\alpha),\varphi_i]_t$ of $=(\vec{x},\alpha)$ by $\vec{u}=\vec{w}\to\alpha=\beta'$. We will show that the formulae ϑ and ϑ' are equivalent. First observe that, since the variables $\vec{w},\beta'$ do not occur in ψ, it is easy to conclude, by Proposition 4.2.1, that ϑ is equivalent to the formula

$$\gamma := \exists\vec{y}\forall\vec{z}\forall\vec{w}\exists\vec{\beta}\exists\beta'\,(=(V([\forall\vec{z}\exists\vec{\beta}\,\psi,\varphi_i])\cup W,\beta')\wedge\psi).$$

We still need to show that γ is equivalent to ϑ'. Note that ϑ' can be obtained from γ by substituting one occurrence of $=(\vec{x},\alpha)$ by $\vec{u}=\vec{w}\to\alpha=\beta'$. Notice also that

$$\mathfrak{A}\models_Z =(V([\forall\vec{z}\exists\vec{\beta}\,\psi,\varphi_i])\cup W,\beta')\wedge\psi, \tag{4.3}$$

if and only if

$$\mathfrak{A}\models_{Z'} =(V([\forall\vec{z}\exists\vec{\beta}\,\psi,\varphi_i])\cup W,\beta')\wedge\psi, \tag{4.4}$$

where $\mathfrak{A}$ is a model, and Z and Z' are teams of $\mathfrak{A}$ such that

$$\begin{aligned} &Z \upharpoonright \mathrm{fr}(\psi) = Z' \upharpoonright \mathrm{fr}(\psi), \\ &\mathfrak{A} \models_Z =(V([\forall \vec{z} \exists \vec{\beta}\, \psi, \varphi_i]) \cup W, \beta'), \text{ and} \\ &\mathfrak{A} \models_{Z'} =(V([\forall \vec{z} \exists \vec{\beta}\, \psi, \varphi_i]) \cup W, \beta'). \end{aligned}$$

Therefore and since $|\vec{w}| = |\vec{u}|$, we can encode a partial function related to the occurrence $[=(\vec{x}, \alpha), \varphi_i]_t$ of the dependence atom $=(\vec{x}, \alpha)$ using $\vec{w}$ and β'.

Assume first that $\mathfrak{A} \models_X \gamma$. Hence

$$\mathfrak{A} \models_Y =(V([\forall \vec{z} \exists \vec{\beta}\, \psi, \varphi_i]) \cup W, \beta') \wedge \psi,$$

for some team Y that can be obtained from X by evaluating the quantifier prefix of γ. Therefore, by Theorem 4.5.10, there exists some successful evaluation e of

$$=(V([\forall \vec{z} \exists \vec{\beta}\, \psi, \varphi_i]) \cup W, \beta') \wedge \psi$$

on the model $\mathfrak{A}$ and team Y. Hence, by Proposition 4.5.9,

$$\mathfrak{A} \models_{e\left([=(\vec{x},\alpha),\varphi_i]_t\right)} =(\vec{x}, \alpha),$$

i.e.,

$$\mathfrak{A} \models_{e\left([=(\vec{x},\alpha),\varphi_i]_t\right)} =(V([\forall \vec{z} \exists \vec{\beta}\, \psi, \varphi_i]) \cup U, \alpha).$$

Therefore, there exists a partial function

$$f_e : A^{|V([\forall \vec{z} \exists \vec{\beta}\, \psi, \varphi_i]) \cup U|} \to \{\bot, \top\}$$

that maps the values of the variables of $V([\forall \vec{z} \exists \vec{\beta}\, \psi, \varphi_i]) \cup U$ to the value of α in the team $e([=(\vec{x}, \alpha), \varphi_i]_t)$. Remember that $|U| = |W|$. Let

$$g_e : A^{|V([\forall \vec{z} \exists \vec{\beta}\, \psi, \varphi_i]) \cup W|} \to \{\bot, \top\}$$

be a function such that $f_e \subseteq g_e$, and let Y' be the variant of Y for which the values for β' have been picked by using the function g_e, i.e., such that $Y \upharpoonright \mathrm{fr}(\psi) = Y' \upharpoonright \mathrm{fr}(\psi)$ and such that, for every $s \in Y'$,

$$s(\beta') = g_e(s(V([\forall \vec{z} \exists \vec{\beta}\, \psi, \varphi_i]) \cup W|)).$$

Hence the equivalence of (4.3) and (4.4) can be applied here. Therefore and since $\mathfrak{A} \models_Y =(V([\forall \vec{z} \exists \vec{\beta}\, \psi, \varphi_i]) \cup W, \beta')$, we conclude that

$$\mathfrak{A} \models_{Y'} =(V([\forall \vec{z} \exists \vec{\beta}\, \psi, \varphi_i]) \cup W, \beta') \wedge \psi.$$

Now since $Y \upharpoonright \mathrm{fr}(\psi) = Y' \upharpoonright \mathrm{fr}(\psi)$, it follows from Lemma 4.2.1 and Theorem 4.5.10 that there exists some successful evaluation e' of

$$=(V([\forall\vec{z}\exists\vec{\beta}\,\psi,\varphi_i]) \cup W, \beta') \wedge \psi$$

on the model $\mathfrak{A}$ and team Y' such that

$$e'([=(\vec{x},\alpha),\varphi]_t) \upharpoonright \mathrm{fr}(\psi) \;=\; e([=(\vec{x},\alpha),\varphi]_t) \upharpoonright \mathrm{fr}(\psi).$$

By Proposition 4.5.9,

$$\mathfrak{A} \models_{e'([=(\vec{x},\alpha),\varphi]_t)} =(\vec{x},\alpha),$$

i.e.,

$$\mathfrak{A} \models_{e'([=(\vec{x},\alpha),\varphi]_t)} =(V([\forall\vec{z}\exists\vec{\beta}\,\psi,\varphi_i]) \cup U, \alpha).$$

Therefore and since the values for β' in Y' were picked by applying the expansion g_e of f_e to the values of $V([\forall\vec{z}\exists\vec{\beta}\,\psi,\varphi_i])$ and $\vec{w}$, it follows that the values of β' in $e'([=(\vec{x},\alpha),\varphi]_t)$ can be obtained by applying the expansion g_e of f_e to the values of $V([\forall\vec{z}\exists\vec{\beta}\,\psi,\varphi_i])$ and $\vec{w}$. Thus

$$\mathfrak{A} \models_{e'([=(\vec{x},\alpha),\varphi]_t)} \vec{w} = \vec{u} \to \alpha = \beta'.$$

Hence, we conclude that there exists a successful evaluation of

$$=(V([\forall\vec{z}\exists\vec{\beta}\,\psi,\varphi_i]) \cup W, \beta') \wedge \psi'$$

on the model $\mathfrak{A}$ and team Y'. Therefore, by Theorem 4.5.10,

$$\mathfrak{A} \models_{Y'} =(V([\forall\vec{z}\exists\vec{\beta}\,\psi,\varphi_i]) \cup W, \beta') \wedge \psi'.$$

Clearly Y' can be obtained form X by evaluating the quantifier prefix of ϑ'. Hence

$$\mathfrak{A} \models_X \vartheta'.$$

Assume then that $\mathfrak{A} \models_X \vartheta'$. Hence

$$\mathfrak{A} \models_Y =(V([\forall\vec{z}\exists\vec{\beta}\,\psi,\varphi_i]) \cup W, \beta') \wedge \psi',$$

for some team Y that can be obtained from X by evaluating the quantifier prefix of ϑ'. Thus, there exists some successful evaluation h of ψ' on the model $\mathfrak{A}$ and team Y. Hence

$$\mathfrak{A} \models_{h([\vec{w}=\vec{u}\to\alpha=\beta',\vartheta'])} \vec{w} = \vec{u} \to \alpha = \beta'.$$

Since the variables in $\vec{w}$ and β' do not occur in other subformulae of ψ' other than $\vec{w} = \vec{u} \to \alpha = \beta'$, we may assume, by Proposition 4.2.1, that for each assignment $s \in h([\vec{w} = \vec{u} \to \alpha = \beta', \vartheta'])$ and $\vec{a} \in A^{|\vec{w}|}$ the modified assignment $s' \in Y$ of s that maps $\vec{w}$ to $\vec{a}$ and β' to $\bot$ or $\top$ is also in $h([\vec{w} = \vec{u} \to \alpha = \beta', \vartheta'])$. Now since

$$\mathfrak{A} \models_Y =(V([\forall \vec{z} \exists \vec{\beta}\, \psi, \varphi_i]) \cup W, \beta'),$$
$$\mathfrak{A} \models_{h([\vec{w}=\vec{u}\to\alpha=\beta',\vartheta'])} \vec{w} = \vec{u} \to \alpha = \beta',$$

and $h([\vec{w} = \vec{u} \to \alpha = \beta', \vartheta']) \subseteq Y$, we conclude that

$$\mathfrak{A} \models_{h([\vec{w}=\vec{u}\to\alpha=\beta',\vartheta'])} =(V([\forall \vec{z} \exists \vec{\beta}\, \psi, \varphi_i]) \cup U, \alpha).$$

Remember that ψ can be obtained from ψ' by substituting $\vec{w} = \vec{u} \to \alpha = \beta'$ with $=(V([\forall \vec{z} \exists \vec{\beta}\, \psi, \varphi_i]) \cup U, \alpha)$. Thus, h can be modified into a successful evaluation of ψ on the model $\mathfrak{A}$ and team Y. Therefore, by Theorem 4.5.10,

$$\mathfrak{A} \models_Y \psi.$$

Since $\mathfrak{A} \models_Y =(V([\forall \vec{z} \exists \vec{\beta}\, \psi, \varphi_i]) \cup W, \beta')$ and Y can clearly be obtained from X by evaluating the quantifier prefix of γ, we conclude that $\mathfrak{A} \models_X \gamma$. Thus we have shown that γ and ϑ' are equivalent. Since γ and ϑ are equivalent, we can finally conclude that ϑ and ϑ' are equivalent.

Let φ_{i+1} be the sentence obtained from φ_i by substituting ϑ with ϑ'. Since ϑ and ϑ' are equivalent, it follows from Lemma 4.2.3 that φ_i and φ_{i+1} are equivalent. Notice that, if φ_i is in Q-normal form, then φ_{i+1} is also in Q-normal form. Furthermore, in φ_{i+1} there is strictly less[2] occurrences of dependence atoms that violate the condition of Definition 4.5.7 than in φ_i. Hence for large enough k, the formula φ_k does not have any dependence atoms that violate the conditions of Definition 4.5.7. Hence, if φ_k is not in dependence normal form there exists a subformula

$$\exists \beta\, \exists \vec{\alpha}\, \psi$$

of φ_k such that $\vec{\alpha}$ is maximal and such that $=(\vec{x}, \beta)$ is not a subformula of ψ for any $\vec{x}$. Let φ_{k+1} denote the formula obtained from φ_k by substituting

$$\exists \beta\, \exists \vec{\alpha}\, \psi \quad \text{by} \quad \exists \beta\, \exists \vec{\alpha}\, (=(V([\exists \beta\, \exists \vec{\alpha}\, \psi, \varphi_k]), \beta) \wedge \psi).$$

Clearly φ_k and φ_{k+1} are equivalent. It is easy to see that the procedure described here terminates and finally produces an equivalent sentence in dependence normal form. □

[2]To be precise, to achieve this we may need to reorder the conjunction in which $[=(\vec{x}, \alpha), \varphi_i]_t$ was a conjunct of.

4.6 Fragments of $\mathcal{FO}(\mathcal{POC})$ and $\mathcal{BD}$ coincide

In this section we use the normal form for bounded Boolean dependence logic from Section 4.5 to establish that

$$\mathcal{BBD} \equiv \mathcal{FO}(\mathcal{POC}^+),\ \mathcal{RBD} \equiv \mathcal{POC}[\mathcal{FO}] \text{ and } \forall\text{-}\mathcal{BD} \equiv \mathcal{POC}[\mathcal{QF}].$$

In addition, we show that $\mathcal{BD} \equiv \mathcal{D}$.

Definition 4.6.1. *Let $\varphi \in \mathcal{BBD}$ be a sentence in dependence normal form. We say that a subformula ψ of φ is* dependence maximal *(with respect to φ) if either ψ does not contain any dependence atoms, or*

$$\psi = \forall \vec{x}\, \exists \vec{\alpha}\, \vartheta,$$

where $\vec{\alpha}$ is nonempty and neither $\forall y \psi$ nor $\exists \beta \exists \vec{\alpha}\, \vartheta$ is a subformula of φ, for any y or β.

Theorem 4.6.2. $\mathcal{BBD} \equiv \mathcal{FO}(\mathcal{POC}^+)$.

Proof. We will first prove that $\mathcal{BBD} \leq \mathcal{FO}(\mathcal{POC}^+)$. Let φ be a $\mathcal{BBD}$-sentence. By Proposition 4.5.11 we may assume that φ is in dependence normal form. We will translate φ into an equivalent $\mathcal{FO}(\mathcal{POC}^+)$ sentence φ^* by substituting each maximal block $\forall \vec{x}\, \exists \vec{\alpha}$ of quantifiers along with the corresponding dependence atoms in φ by a partially-ordered connective.

More precisely, we define a translation $\psi \mapsto \psi^*$ for all subformulae ψ of φ that are dependence maximal or can be obtained from dependence maximal subformulae of φ by first-order operations, i.e., by taking conjunctions, disjunctions and first-order quantifications. Note that, every $\mathcal{BBD}$-sentence that is in dependence normal form can be build from its dependence maximal subformulae by using only first-order operations. The translation is defined recursively as follows:

(i) If ψ is a formula without dependence atoms then $\psi^* := \psi$.

(ii) If

$$\psi = \forall \vec{x}\, \exists \vec{\alpha} \Big(\big(\bigwedge_{1 \leq i \leq m} {=}(V([\psi, \varphi]) \cup \{\vec{x}_i\}, \alpha_i) \big) \wedge \vartheta \Big)$$

is dependence maximal and $\vec{\alpha} = (\alpha_1, \ldots, \alpha_m)$, we define that

$$\psi^* := N_\pi\, \vec{x}_0 \alpha_0 \vec{x}_1 \alpha_1 \ldots \vec{x}_m \alpha_m\, \vartheta^*, \tag{4.5}$$

where $\vec{x}_0$ are exactly those variables in $\vec{x}$ that are not in any $\vec{x}_i$, $1 \leq i \leq m$, and α_0 is a fresh Boolean variable not occurring in ϑ^* nor φ. The patter π of the connective is obtained canonically form the identities between the variables in the tuples $\vec{x}_i$, $i \leq m$.

(iii) If $\psi = (\vartheta \wedge \eta)$, we define that $\psi^* := (\vartheta^* \wedge \eta^*)$.

(iv) If $\psi = (\vartheta \vee \eta)$, we define that $\psi^* := (\vartheta^* \vee \eta^*)$.

(v) If $\psi = \exists x\, \eta$, we define that $\psi^* := \exists x\, \eta^*$.

(vi) If $\psi = \forall x\, \eta$ and ψ is not dependence maximal, we define that $\psi^* := \forall x\, \eta^*$.

Note that since φ is in dependence normal form, φ^* is defined, and clearly $\varphi^* \in \mathcal{FO}(\mathcal{POC}^+)$. Thus, it suffices to show that for every formula ψ that can be obtained from dependence maximal subformulae of φ by using conjunctions, disjunctions and first-order quantifications, for every model $\mathfrak{A}$ and for every team X of $\mathfrak{A}$ such that $\mathrm{fr}(\psi) \subseteq \mathrm{dom}(X)$

$$\mathfrak{A} \models_X \psi \quad \Leftrightarrow \quad \mathfrak{A}, s \models \psi^* \text{ for all } s \in X.$$

The proof is done by induction on the definition of the translation.
(i) If ψ is without dependence atoms, the claim holds by Proposition 4.2.2.
(ii) Assume that

$$\psi = \forall \vec{x}\, \exists \vec{\alpha} \Big(\big(\bigwedge_{1 \leq i \leq m} =(V([\psi, \varphi]) \cup \{\vec{x}_i\}, \alpha_i) \big) \wedge \vartheta \Big)$$

is dependence maximal and that $\vec{\alpha} = (\alpha_1, \ldots, \alpha_m)$. Then

$$\psi^* = N_\pi\, \vec{x}_0 \alpha_0 \vec{x}_1 \alpha_1 \ldots \vec{x}_m \alpha_m\, \vartheta^*,$$

where π is the pattern determined by the tuple $\vec{x}_0, \vec{x}_1, \ldots, \vec{x}_m$. We will show that

$$\mathfrak{A} \models_X \psi \quad \Leftrightarrow \quad \mathfrak{A}, s \models N_\pi\, \vec{x}_0 \alpha_0 \vec{x}_1 \alpha_1 \ldots \vec{x}_m \alpha_m\, \vartheta^* \text{ for all } s \in X,$$

for every model $\mathfrak{A}$ and every team X of $\mathfrak{A}$ such that $\mathrm{fr}(\psi) \subseteq \mathrm{dom}(X)$.

Let $\mathfrak{A}$ be a model and X a team of $\mathfrak{A}$ such that $\mathrm{fr}(\psi) \subseteq \mathrm{dom}(X)$. Furthermore, define that $n := |\vec{x}|$ and that $n_i := |\vec{x}_i|$, for each $i \leq m$. Now, by the semantics of the quantifiers,

$$\mathfrak{A} \models_X \psi$$

if and only if for each i, $1 \leq i \leq m$, there exists a function

$$F_i : X(A^n/\vec{x}, F_1/\alpha_1, \ldots, F_{i-1}/\alpha_{i-1}) \to \{\bot, \top\}$$

such that

$$\mathfrak{A} \models_Y \Big(\bigwedge_{1 \leq i \leq m} {=}(V([\psi, \varphi]) \cup \{\vec{x}_i\}, \alpha_i) \Big) \wedge \vartheta, \tag{4.6}$$

where $Y = X(A^n/\vec{x}, \vec{F}/\vec{\alpha})$. For each i, $1 \leq i \leq m$, we define

$$G_i : X(A^n/\vec{x}) \to \{\bot, \top\}$$

to be the unique function obtained from F_i such that, for every $s \in \mathrm{dom}(F_i)$,

$$G_i(s \restriction \mathrm{dom}(G_i)) = F_i(s).$$

Assume first that (4.6) holds. For each $s \in X$ and $1 \leq i \leq m$, let

$$f_i^s : A^{n_i} \to \{\bot, \top\}$$

denote the function such that

$$f_i^s(\vec{a}_i) = G_i(s(\vec{a}/\vec{x})),$$

where $\vec{a}_i$ is the restriction of $\vec{a}$ to the variables $\vec{x}_i$. Note that f_i^s is well-defined, since by the first conjunct of (4.6), the function F_i and hence the function G_i is $V([\psi, \varphi]) \cup \{\vec{x}_i\}$-determined. Furthermore, we define that

$$f_0^s(\vec{a}_0) := \top,$$

for every $s \in X$ and $\vec{a}_0 \in A^{n_0}$. If $\vec{a}_i \in A^{n_i}$, $i \leq m$, are tuples such that $\vec{a}_0\, \vec{a}_1 \ldots \vec{a}_m$ is of pattern π then clearly, for every $s \in X$, the modified assignment

$$s' := s(\vec{a}_0/\vec{x}_0, \ldots, \vec{a}_m/\vec{x}_m, f_1^s(\vec{a}_1)/\alpha_1, \ldots, f_m^s(\vec{a}_m)/\alpha_m)$$

is in Y. Now since $\mathfrak{A} \models_Y \vartheta$, by induction hypothesis, we have that $\mathfrak{A}, s' \models \vartheta^*$, for every $s \in X$. Therefore, since α_0 does not occur in ϑ^*, we have that

$$\mathfrak{A}, s'(\top/\alpha_0) \models \vartheta^*,$$

for every $s \in X$. Hence, the functions f_i^s, $i \leq m$, are as required in the truth condition of N_π, and we conclude that

$$\mathfrak{A}, s \models N_\pi\, \vec{x}_0 \alpha_0 \vec{x}_1 \alpha_1 \ldots \vec{x}_m \alpha_m\, \vartheta^*,$$

for every $s \in X$.

Assume then that

$$\mathfrak{A}, s \models N_\pi \vec{x}_0 \alpha_0 \vec{x}_1 \alpha_1 \dots \vec{x}_m \alpha_m \vartheta^*$$

holds for every $s \in X$. Hence, for every $s \in X$ and $i \leq m$, there exists a function

$$f_i^s : A^{n_i} \to \{\bot, \top\},$$

such that if $\vec{a}_0, \dots, \vec{a}_m$ is of pattern π then

$$\mathfrak{A}, s(\vec{a}_0/\vec{x}_0, \dots, \vec{a}_m/\vec{x}_m, f_0^s(\vec{a}_0)/\alpha_0, \dots, f_m^s(\vec{a}_m)/\alpha_m) \models \vartheta^*.$$

Since α_0 does not occur in ϑ^* we conclude that

$$\mathfrak{A}, s(\vec{a}_0/\vec{x}_0, \dots, \vec{a}_m/\vec{x}_m, f_1^s(\vec{a}_1)/\alpha_1, \dots, f_m^s(\vec{a}_m)/\alpha_m) \models \vartheta^*.$$

Now, for each i, $1 \leq i \leq m$, define the function

$$F_i : X(A^n/\vec{x}, F_1/\alpha_1, \dots, F_{i-1}/\alpha_{i-1}) \to \{\bot, \top\}$$

by setting that

$$F_i\Big(s(\vec{a}/\vec{x}, f_1^s(\vec{a}_1)/\alpha_1, \dots, f_{i-1}^s(\vec{a}_{i-1})/\alpha_{i-1})\Big) := f_i^s(\vec{a}_i),$$

where $\vec{a}_j \in A^{n_j}$ is the restriction of $\vec{a}$ to the variables in $\vec{x}_j$, $1 \leq j \leq i$. The functions F_i are obviously $V([\psi, \varphi]) \cup \{\vec{x}_i\}$-determined and hence

$$\mathfrak{A} \models_{X(A^n/\vec{x}, \vec{F}/\vec{\alpha})} \bigwedge_{1 \leq i \leq m} =(V([\psi, \varphi]) \cup \{\vec{x}_i\}, \alpha_i). \tag{4.7}$$

Furthermore, if $s' \in X(A^n/\vec{x}, \vec{F}/\vec{\alpha})$, then in fact

$$s' = s(\vec{a}_0/\vec{x}_0, \dots, \vec{a}_m/\vec{x}_m, f_1^s(\vec{a}_1)/\alpha_1, \dots, f_m^s(\vec{a}_m)/\alpha_m),$$

for some $s \in X$ and $\vec{a}_i \in A^{n_i}$, $i \leq n$, such that $\vec{a}_1, \dots, \vec{a}_n$ is of pattern π. Hence

$$\mathfrak{A}, s' \models \vartheta^*,$$

for each $s' \in X(A^n/\vec{x}, \vec{F}/\vec{\alpha})$. Thus, by induction hypothesis,

$$\mathfrak{A} \models_{X(A^n/\vec{x}, \vec{F}/\vec{\alpha})} \vartheta.$$

By this and (4.7), we conclude that (4.6) holds and thus that $\mathfrak{A} \models_X \psi$.

The cases (iii) – (vi) are trivial.

For the direction $\mathcal{FO}(\mathcal{POC}^+) \leq \mathcal{BBD}$, let $\varphi \in \mathcal{FO}(\mathcal{POC}^+)$ be a sentence. Without loss of generality, we may assume that each variable quantified in φ is quantified exactly once. We define recursively a translation $\psi \mapsto \psi^+$ for all subformulae ψ of φ as follows. If ψ is a literal, we define that $\psi \mapsto \psi$. For first-order connectives and quantifiers, we define that

$$\begin{aligned} (\vartheta \wedge \eta) &\mapsto (\vartheta^+ \wedge \eta^+), \\ (\vartheta \vee \eta) &\mapsto (\vartheta^+ \vee \eta^+), \\ \exists x\eta &\mapsto \exists x\eta^+, \\ \forall x\eta &\mapsto \forall x\eta^+. \end{aligned}$$

Finally, if $\psi = N_\pi \vec{x}_1\alpha_1 \dots \vec{x}_m\alpha_m\,\vartheta$, we define that

$$\psi^+ := \forall\vec{x}\,\exists\alpha_1 \dots \exists\alpha_m\Big(\big(\bigwedge_{1\leq i\leq m} {=}(V([\psi,\varphi]) \cup \{\vec{x}_i\}, \alpha_i)\big) \wedge \vartheta^+\Big),$$

where $\vec{x}$ is a tuple of exactly those variables that are in at least one of the tuples $\vec{x}_1, \dots, \vec{x}_m$. Clearly φ^+ is a $\mathcal{BBD}$ sentence.

It is now easy to prove by induction that for every subformula ψ of φ

$$\mathfrak{A} \models_X \psi^+ \;\Leftrightarrow\; \mathfrak{A}, s \models \psi \text{ for all } s \in X$$

holds for every model $\mathfrak{A}$ and team X on $\mathfrak{A}$ such that $\mathrm{dom}(X) = \mathrm{fr}(\psi^+)$. The proof is completely analogous to the inductive proof of the direction $\mathcal{BBD} \leq \mathcal{FO}(\mathcal{POC}^+)$ shown earlier. □

By using the same methods as in the proofs of Propositions 4.5.6, 4.5.11 and Theorem 4.6.2, we obtain the following theorem.

Theorem 4.6.3. *$\mathcal{RBD} \equiv \mathcal{POC}[\mathcal{FO}]$ and $\forall$-$\mathcal{BD} \equiv \mathcal{POC}[\mathcal{QF}]$.*

Proof. The inclusions $\mathcal{POC}[\mathcal{FO}] \leq \mathcal{RBD}$ and $\mathcal{POC}[\mathcal{QF}] \leq \forall$-$\mathcal{BD}$ follow directly by the translation $\varphi \mapsto \varphi^+$ defined in the proof of Theorem 4.6.2.

For the inclusion $\forall$-$\mathcal{BD} \leq \mathcal{POC}[\mathcal{QF}]$, notice that the methods used in the proofs of Propositions 4.5.6 and 4.5.11 produce, for each $\forall$-$\mathcal{BD}$-sentence φ, an equivalent $\forall$-$\mathcal{BD}$-sentence of the form

$$\forall\vec{x}\,\exists\vec{\alpha}\,\psi,$$

where ψ is a quantifier-free formula. Now, from the translation $\varphi \mapsto \varphi^*$, defined in the proof of Theorem 4.6.2, it follows that the sentence $(\forall\vec{x}\,\exists\vec{\alpha}\,\psi)^*$ is a $\mathcal{POC}[\mathcal{QF}]$-sentence equivalent to $\forall\vec{x}\,\exists\vec{\alpha}\,\psi$, and hence equivalent to φ.

We still need to show that $\mathcal{RBD} \leq \mathcal{POC}[\mathcal{FO}]$. Recall that $\mathcal{RBD}$ is the syntactic fragment of $\mathcal{BD}$ in which there is no Boolean dependence atoms in scope of existential first-order quantifiers. We will first establish that for each $\mathcal{RBD}$-sentence φ there exists an equivalent $\mathcal{RBD}$-sentence φ^- in which there is no quantification of Boolean variables in the scope of existential first-order quantifiers. Without loss of generality, we consider only models of cardinality at least 2. For each $\mathcal{BD}$ formula $\exists\alpha\psi$ without Boolean dependence atoms, we define that

$$(\exists\alpha\psi)' := \exists x\exists y\psi((x = y)/\alpha),$$

where x and y are fresh first-order quantifiers not occurring in ψ and the formula $\psi((x = y)/\alpha)$ is the formula obtained from ψ by substituting each free occurrence of α in ψ by $x = y$. Clearly, the formulae $\exists\alpha\psi$ and $(\exists\alpha\psi)'$ are equivalent in the class of all structures of cardinality at least 2. Hence, by Theorem 4.2.3, we conclude that for each $\mathcal{RBD}$-sentence φ there exists an equivalent $\mathcal{RBD}$-sentence φ^- in which there is no quantification of Boolean variables in the scope of existential first-order quantifiers. When the procedures used in the proofs of Propositions 4.5.6 and 4.5.11 are applied to φ^-, an equivalent sentence of the form

$$\forall\vec{x}\,\exists\vec{\alpha}\,\psi,$$

where ψ is a first-order formula, is obtained. Now from the translation $\varphi \mapsto \varphi^*$ defined in the proof of Theorem 4.6.2, it follows that the sentence $(\forall\vec{x}\,\exists\vec{\alpha}\,\psi)^*$ is an $\mathcal{POC}[\mathcal{FO}]$-sentence equivalent to φ^- and hence equivalent to φ. □

Corollary 4.6.4. *$\mathcal{BBD}$, $\mathcal{RBD}$ and $\forall$-$\mathcal{BD}$ have the zero-one law.*

Proof. By Theorem 4.3.7, $\mathcal{FO}(\mathcal{POC}^+)$ has the zero-one law. Therefore, by Theorem 4.6.2 $\mathcal{BBD}$ has the zero-one law. Hence the fragments $\mathcal{RBD}$ and $\forall$-$\mathcal{BD}$ of $\mathcal{BBD}$ have the zero-one law. □

Theorem 4.6.5. *$\mathcal{BD} \equiv \mathcal{D}$.*

Proof. $\mathcal{BD} \leq \mathcal{D}$ holds by Proposition 4.4.4. For the other direction we will give a translation $\varphi \mapsto \varphi^*$ from sentences of dependence logic to sentences of Boolean dependence logic. Let φ be an arbitrary $\mathcal{D}$-sentence in the normal form for $\mathcal{D}$ given in [Vää07, p. 98], i.e.,

$$\varphi := \forall\vec{x}\,\exists\vec{y}\,(\bigwedge_{1\leq i\leq n} =(\vec{x}_i, y_i) \;\wedge\; \psi),$$

where ψ is a quantifier free first-order formula, $\vec{x}_i$ is a vector of variables from $\vec{x}$ and y_i a variable from $\vec{y}$, $1 \leq i \leq n$. The translation φ^* of φ is the $\mathcal{BD}$-sentence

$$\forall\vec{x}\,\exists\vec{y}\Big(\psi \wedge \forall\vec{z}\,\exists\vec{\alpha} \bigwedge_{1\leq i\leq n} (=(\vec{x}_i, z_i, \alpha_i) \wedge (z_i = y_i \leftrightarrow \alpha_i))\Big),$$

where $\vec{z}$ and $\vec{\alpha}$ are tuples of fresh variables of length n, and z_i and α_i are variables from the corresponding tuple, $1 \leq i \leq m$. We will show that for every model $\mathfrak{A}$

$$\mathfrak{A} \models_{\{\emptyset\}} \varphi \quad \text{iff} \quad \mathfrak{A} \models_{\{\emptyset\}} \varphi^*.$$

Assume first that $\mathfrak{A} \models_{\{\emptyset\}} \varphi$. Hence

$$\mathfrak{A} \models_X \bigwedge_{1\leq i\leq n} =(\vec{x}_i, y_i) \ \wedge \ \psi,$$

for some team X that can be obtained from $\{\emptyset\}$ by evaluating the quantifier prefix of φ, i.e., $X = \{\emptyset\}(A^{|\vec{x}|}/\vec{x}, \vec{G}/\vec{y})$ for some functions

$$G_i : \{\emptyset\}(A^{|\vec{x}|}/\vec{x}, G_1/y_i, \ldots, G_{i-1}/y_{i-1}) \to A,$$

$1 \leq i \leq |y|$. Now since

$$\mathfrak{A} \models_X \bigwedge_{1\leq i\leq n} =(\vec{x}_i, y_i),$$

there exists, for every $1 \leq i \leq n$, a function

$$F_i : A^{|\vec{x}_i|} \to A$$

that maps the values of the variables $\vec{x}_i$ to the value of the variable y_i in the team X. Let Y denote the team

$$X(A^{|\vec{z}|}/\vec{z}, H_1/\alpha_1, \ldots, H_n/\alpha_n),$$

where $H_i : X(A^{|\vec{z}|}/\vec{z}, H_1/\alpha_1, \ldots H_{i-1}/\alpha_{i-1}) \to \{\bot, \top\}$ is obtained from F_i in the obvious way, i.e., such that that

$$H_i(s) := \begin{cases} \top & \text{if } F_i\big(s(\vec{x}_i)\big) = s(z_i) \\ \bot & \text{if } F_i\big(s(\vec{x}_i)\big) \neq s(z_i). \end{cases}$$

Notice that

$$\mathfrak{A} \models_Y \bigwedge_{1\leq i\leq n} (=(\vec{x}_i, z_i, \alpha_i) \wedge (z_i = y_i \leftrightarrow \alpha_i)).$$

Hence we have that

$$\mathfrak{A} \models_X \forall \vec{z} \exists \vec{\alpha} \bigwedge_{1 \leq i \leq n} (=(\vec{x}_i, z_i, \alpha_i) \wedge (z_i = y_i \leftrightarrow \alpha_i)).$$

Therefore, since $\mathfrak{A} \models_X \psi$, and since X was obtained from $\{\emptyset\}$ by evaluating the quantifier prefix $\forall \vec{x} \exists \vec{y}$, we conclude that $\mathfrak{A} \models_{\{\emptyset\}} \varphi^*$.

Assume then that $\mathfrak{A} \models_{\{\emptyset\}} \varphi^*$ holds. Hence

$$\mathfrak{A} \models_X \psi \wedge \forall \vec{z} \exists \vec{\alpha} \bigwedge_{1 \leq i \leq n} (=(\vec{x}_i, z_i, \alpha_i) \wedge (z_i = y_i \leftrightarrow \alpha_i)),$$

for some team X that can be obtained from $\{\emptyset\}$ by evaluating the quantifier prefix of φ^*. Furthermore

$$\mathfrak{A} \models_Y \bigwedge_{1 \leq i \leq n} (=(\vec{x}_i, z_i, \alpha_i) \wedge (z_i = y_i \leftrightarrow \alpha_i)),$$

for some team Y that can be obtained from X by evaluating the quantifiers $\forall \vec{z} \exists \vec{\alpha}$. We will show that for each i, $1 \leq i \leq n$,

$$\mathfrak{A} \models_X =(\vec{x}_i, y_i).$$

This together with the fact that $\mathfrak{A} \models_X \psi$ is enough to prove that $\mathfrak{A} \models_{\{\emptyset\}} \varphi$.

Fix i, $1 \leq i \leq n$, and let $s, t \in X$ be any two assignments such that $s(\vec{x}_i) = t(\vec{x}_i)$. Clearly there exist assignments $s', t' \in Y$ such that

$$s' \upharpoonright \mathrm{dom}(X) = s, \qquad t' \upharpoonright \mathrm{dom}(X) = t \quad \text{and} \quad s'(z_i) = t'(z_i) = s(y_i).$$

Now since

$$\mathfrak{A} \models_Y z_i = y_i \leftrightarrow \alpha_i,$$

we have that $s'(\alpha_i) = 1$. Furthermore, since

$$\mathfrak{A} \models_Y =(\vec{x}_i, z_i, \alpha_i),$$

we have that $s'(\alpha) = t'(\alpha)$. Hence $t'(\alpha) = 1$ and furthermore $t'(y_i) = t'(z_i)$. Thus, we have that

$$t(y_i) = t'(y_i) = t'(z_i) = s(y_i),$$

and can conclude that $t(y_i) = s(y_i)$. Therefore

$$\mathfrak{A} \models_X =(\vec{x}_i, y_i).$$

□

By Theorem 2.5.4 we know that $\mathcal{D} \equiv \mathcal{ESO}$. Hence we obtain the following corollary.

Corollary 4.6.6. *$\mathcal{BD} \equiv \mathcal{ESO}$.*

Corollary 4.6.7. *$\mathcal{BD}$ does not have the zero-one law.*

Proof. Remember that the zero-one law fails already for $\mathcal{D}^2$ (Proposition 3.4.6). Hence, by Theorem 4.6.5, $\mathcal{BD}$ does not have the zero-one law. □

4.7 Hierarchy of expressive power

In Section 4.6 we showed that the expressive power of the fragments $\mathcal{BBD}$, $\mathcal{RBD}$ and $\forall$-$\mathcal{BD}$ of Boolean dependence logic coincide with the expressive power of the fragments $\mathcal{FO}(\mathcal{POC}^+)$, $\mathcal{POC}[\mathcal{FO}]$ and $\mathcal{POC}[\mathcal{QF}]$ of $\mathcal{FO}(\mathcal{POC})$, respectively. In this section we show that the fragments $\mathcal{BBD}$, $\mathcal{RBD}$ and $\forall$-$\mathcal{BD}$ of Boolean dependence logic form a hierarchy with respect to expressive power. We show that

$$\mathcal{POC}[\mathcal{QF}] < \mathcal{POC}[\mathcal{FO}] < \mathcal{FO}(\mathcal{POC}^+).$$

and hence that

$$\forall\text{-}\mathcal{BD} < \mathcal{RBD} < \mathcal{BBD}.$$

Moreover, we establish that $\mathcal{BBD} < \mathcal{BD}$ and that $\mathcal{BD} \not\leq \mathcal{FO}(\mathcal{POC})$.

Lemma 4.7.1. *Let $\mathfrak{A}$ be a model, $\mathfrak{B}$ a submodel of $\mathfrak{A}$ and φ a sentence of $\mathcal{POC}[\mathcal{QF}]$. If $\mathfrak{A} \models \varphi$ then $\mathfrak{B} \models \varphi$.*

Proof. By Proposition 4.3.6, $\mathcal{POC}[\mathcal{QF}]$ is equivalent to $\mathsf{D}[\mathcal{QF}]$ and, by [HST08, Lemma 6], $\mathsf{D}[\mathcal{QF}]$ is equivalent with strict Σ^1_1. For strict Σ^1_1 the claim follows from [Bar69, Lemma 1.2]. □

Proposition 4.7.2. *$\mathcal{POC}[\mathcal{QF}] < \mathcal{POC}[\mathcal{FO}]$.*

Proof. Clearly $\mathcal{POC}[\mathcal{QF}] \leq \mathcal{POC}[\mathcal{FO}]$. By Lemma 4.7.1, the truth of a $\mathcal{POC}[\mathcal{QF}]$-sentence is preserved from models to its submodels. Hence it is enough to give a $\mathcal{POC}[\mathcal{FO}]$-sentence which is not preserved under taking submodels. Clearly

$$\exists x \exists y \neg x = y$$

is such a sentence. □

Since, by Theorem 4.6.3, $\forall$-$\mathcal{BD} \equiv \mathcal{POC}[\mathcal{QF}]$ and $\mathcal{RBD} \equiv \mathcal{POC}[\mathcal{FO}]$, the following result follows from Proposition 4.7.2.

Corollary 4.7.3. $\forall$-$\mathcal{BD} < \mathcal{RBD}$.

Definition 4.7.4. *Let $\mathcal{L}$ be a logic (or a fragment of a logic), τ a vocabulary, and $\mathfrak{A}$ and $\mathfrak{B}$ first-order structures over τ. We write that $\mathfrak{A} \Rrightarrow_{\mathcal{L}} \mathfrak{B}$, if the implication*

$$\mathfrak{A} \models \varphi \Rightarrow \mathfrak{B} \models \varphi$$

holds for every sentence $\varphi \in \mathcal{L}$.

Let N_π be a partially-ordered connective. By $N_\pi[\mathcal{FO}_r]$, we denote the set of all sentences in $\mathcal{POC}[\mathcal{FO}]$ which are of the form

$$N_\pi \vec{x}_1 \alpha_1 \dots \vec{x}_m \alpha_m \, \varphi,$$

where φ is a first-order formula with quantifier rank at most r. We will next define an Ehrenfeucht-Fraïssé game that captures the truth preservation relation $\mathfrak{A} \Rrightarrow_{N_\pi[\mathcal{FO}_r]} \mathfrak{B}$. This game is a straightforward modification of the corresponding game for $\mathsf{D}[\mathcal{FO}]$ by Sevenster and Tulenheimo [ST06], which in turn is based on the game for $\mathcal{FO}(\mathsf{D})$ by Sandu and Väänänen [SV92].

Definition 4.7.5. *Let $\mathfrak{A}$ and $\mathfrak{B}$ be first-order structures over a vocabulary τ and $r \geq 0$. Let $\pi = (n_1, \dots, n_m, E)$ be a pattern. The $N_\pi[\mathcal{FO}_r]$-*EF game *$N_\pi \mathrm{EF}_r(\mathfrak{A}, \mathfrak{B})$ on $\mathfrak{A}$ and $\mathfrak{B}$ is played by two players, Spoiler and Duplicator. The game has two phases.*

Phase 1:

- *Spoiler picks a function $f_i : A^{n_i} \to \{\bot, \top\}$, for each i, $1 \leq i \leq m$.*
- *Duplicator answers by choosing a function $g_i : B^{n_i} \to \{\bot, \top\}$, for each i, $1 \leq i \leq m$.*
- *Spoiler chooses tuples $\vec{b}_i \in B^{n_i}$, $1 \leq i \leq m$, such that $\vec{b}_1 \dots \vec{b}_m$ is of pattern π.*
- *Duplicator answers by choosing tuples $\vec{a}_i \in A^{n_i}$, $1 \leq i \leq m$, such that $\vec{a}_1 \dots \vec{a}_m$ is of pattern π, and $f_i(\vec{a}_i) = g_i(\vec{b}_i)$ for each $1 \leq i \leq m$. If there are no such tuples $\vec{a}_1, \dots, \vec{a}_m$, then Duplicator loses the play of the game.*

Phase 2:

- *Spoiler and Duplicator play the usual first-order EF-game of r rounds on the structures $(\mathfrak{A}, \vec{a}_1 \dots \vec{a}_m)$ and $(\mathfrak{B}, \vec{b}_1 \dots \vec{b}_m)$: On each round j, $1 \leq j \leq r$, Spoiler picks an element $c_j \in A$ (or $d_j \in B$), and Duplicator answer by choosing an element $d_j \in B$ (or $c_j \in A$, respectively).*

Duplicator wins the play of the game if and only if the mapping

$$\vec{a}_1 \dots \vec{a}_m c_1 \dots c_r \mapsto \vec{b}_1 \dots \vec{b}_m d_1 \dots d_r$$

is a partial isomorphism from $\mathfrak{A}$ to $\mathfrak{B}$. We say that Duplicator has a winning strategy in the game if and only she has a systematic way of answering all possible moves of Spoiler such that using it she always wins the play.

We show next that the game $N_\pi \mathrm{EF}_r$ can be used for studying the truth preservation relation $\Rrightarrow_{N_\pi[\mathcal{FO}_r]}$. This result is essentially the same as Proposition 12 in [ST06], which in turn is a special case of Proposition 7 of [SV92].

Proposition 4.7.6. *Let $\mathfrak{A}$ and $\mathfrak{B}$ be τ-structures, π a pattern, and $r \geq 0$. If Duplicator has a winning strategy in the game $N_\pi \mathrm{EF}_r(\mathfrak{A}, \mathfrak{B})$ then $\mathfrak{A} \Rrightarrow_{N_\pi[\mathcal{FO}_r]} \mathfrak{B}$.*

Proof. Assume that Duplicator has a winning strategy in $N_\pi \mathrm{EF}_r(\mathfrak{A}, \mathfrak{B})$. To prove $\mathfrak{A} \Rrightarrow_{N_\pi[\mathcal{FO}_r]} \mathfrak{B}$, assume that

$$\varphi = N_\pi \, \vec{x}_1 \alpha_1 \dots \vec{x}_m \alpha_m \, \psi$$

is a sentence of $N_\pi[\mathcal{FO}_r]$ such that $\mathfrak{A} \models \varphi$. We need to show that $\mathfrak{B} \models \varphi$. Now since $\mathfrak{A} \models \varphi$, there exists functions

$$f_i : A^{n_i} \to \{\bot, \top\},$$

$1 \leq i \leq m$, such that if $\vec{a}_1 \dots \vec{a}_m$, where $\vec{a}_j \in A^{n_j}$ for $1 \leq j \leq m$, is of pattern π then

$$\mathfrak{A} \models_{s[\vec{a}, \vec{f}]} \psi,$$

where $s[\vec{a}, \vec{f}]$ denotes the assignment that maps $\vec{x}_i$ to $\vec{a}_i$ and α_i to $f_i(\vec{a}_i)$, for $1 \leq i \leq m$.

Assume that Spoiler chooses the functions $f_1, \dots, f_m$ as his first move in phase 1 of the game $N_\pi \mathrm{EF}_r(\mathfrak{A}, \mathfrak{B})$. Let

$$g_i : B^{n_i} \to \{\bot, \top\},$$

$1 \leq i \leq m$, be the answer given by the winning strategy of Duplicator. Remember that by $s[\vec{b}, \vec{g}]$ we mean the assignment that maps $\vec{x}_i$ to $\vec{b}_i$ and α_i to $g_i(\vec{b}_i)$, for $1 \leq i \leq m$. To show $\mathfrak{B} \models \varphi$, it suffices to show that

$$\mathfrak{B} \models_{s[\vec{b}, \vec{g}]} \psi,$$

for all tuples $\vec{b}_1 \dots \vec{b}_m$ of pattern π such that $\vec{b}_i \in B^{n_i}$, for each $i \leq m$. Thus, let $\vec{b}_i \in B^{n_i}$, $1 \leq i \leq m$, be arbitrary tuples such that $\vec{b}_1 \dots \vec{b}_m$ is of pattern π. Let $\vec{a}_j \in A^{n_j}$, $1 \leq j \leq m$, be the answer given by the winning strategy of Duplicator when the second move of Spoiler is $\vec{b}_1, \dots, \vec{b}_m$. By the definition of the game $N_\pi \mathrm{EF}_r$, Duplicator then has a winning strategy in the first-order EF game with r rounds between the structures

$$(\mathfrak{A}, \vec{a}_1, \dots, \vec{a}_m) \quad \text{and} \quad (\mathfrak{B}, \vec{b}_1, \dots, \vec{b}_m).$$

By the standard EF theorem, it follows that $(\mathfrak{A}, \vec{a}_1, \dots, \vec{a}_m)$ and $(\mathfrak{B}, \vec{b}_1, \dots, \vec{b}_m)$ satisfy the same $\mathcal{FO}_r$-sentences. Note further that

$$f_1(\vec{a}_1) = g_1(\vec{b}_1), \dots, f_m(\vec{a}_m) = g_m(\vec{b}_m),$$

by the condition governing the choice of $\vec{b}_1, \dots, \vec{b}_m$, whence

$$(\mathfrak{A}, \vec{a}_1, \dots, \vec{a}_m, f_1(\vec{a}_1), \dots, f_m(\vec{a}_m)) \quad \text{and} \quad (\mathfrak{B}, \vec{b}_1, \dots, \vec{b}_m, g_1(\vec{b}_1), \dots, g_m(\vec{b}_m))$$

are equivalent with respect to all sentences of $\mathcal{FO}_r$ extended with Boolean variables. In particular, since $\mathfrak{A} \models_{s[\vec{a},\vec{f}]} \psi$, we have $\mathfrak{B} \models_{s[\vec{b},\vec{g}]} \psi$, as desired. □

Corollary 4.7.7. *Let $\mathcal{K}$ be a class of τ-structures. If for every pattern π and every $r \geq 0$ there exist τ-structures $\mathfrak{A}$ and $\mathfrak{B}$ such that $\mathfrak{A} \in \mathcal{K}$, $\mathfrak{B} \notin \mathcal{K}$ and Duplicator has a winning strategy in the game $N_\pi \mathrm{EF}_r(\mathfrak{A}, \mathfrak{B})$, then K is not definable in $\mathcal{POC}[\mathcal{FO}]$.*

Proof. We prove the contraposition of the claim. Thus, assume that K is definable in $\mathcal{POC}[\mathcal{FO}]$. Then there is a sentence φ of the form $N_\pi \vec{x}_1 \alpha_1 \dots \vec{x}_m \alpha_m \psi$ such that

$$\mathcal{K} = \{\mathfrak{A} \in \mathrm{Str}(\tau) \mid \mathfrak{A} \models \varphi\}.$$

Let r be the quantifier rank of ψ. Then by Proposition 4.7.6, there are no structures $\mathfrak{A} \in \mathcal{K}$ and $\mathfrak{B} \notin \mathcal{K}$ such that Duplicator has a winning strategy in the game $N_\pi \mathrm{EF}_r(\mathfrak{A}, \mathfrak{B})$. □

Theorem 4.7.8. $\mathcal{RBD} < \mathcal{BBD}$

Proof. By Proposition 4.4.4, $\mathcal{RBD} \leq \mathcal{BBD}$. For the strict inclusion, we show that non-connectivity of graphs is definable in $\mathcal{BBD}$, but not in $\mathcal{RBD}$. Let $\mathcal{K}$ denote the class non-connected graphs. Note first that a graph $\mathfrak{A} = (A, E^{\mathfrak{A}})$ is not connected if and only if there is a subset $U \subseteq A$ such that U and $A \setminus U$ are

nonempty, and there are no edges $(a, b) \in E^{\mathfrak{A}}$ between U and $A \setminus U$. This can be expressed by the $\mathcal{BBD}$-sentence

$$
\begin{array}{ll}
\exists u \exists v \forall x \forall y \exists \alpha \exists \beta (& \\
\quad =(x, \alpha) \wedge =(y, \beta) & \textit{There are two relations...} \\
\quad \wedge (x = y \rightarrow (\alpha \leftrightarrow \beta)) & \textit{... which are equal...} \\
\quad \wedge (x = u \rightarrow \alpha) \wedge (x = v \rightarrow \neg\alpha) & \textit{... and contain } u \textit{ but not } v. \\
\quad \wedge (\alpha \wedge \neg\beta \rightarrow \neg Exy)). & \textit{If } x \textit{ is in the relations but } y \textit{ is not then} \\
 & \textit{there is no edge between } x \textit{ and } y.
\end{array}
$$

We use Corollary 4.7.7 to prove that non-connectivity is not definable in $\mathcal{POC}[\mathcal{FO}]$. By Theorem 4.6.3, it then follows that non-connectivity is not definable in $\mathcal{RBD}$. Let us fix the pattern π and the number of rounds $r \geq 0$, and consider the game $N_\pi \mathrm{EF}_r$. Let $\mathfrak{A} = (A, E^{\mathfrak{A}})$ and $\mathfrak{B} = (B, E^{\mathfrak{B}})$ be the graphs such that

- $B = \{u_1, \ldots, u_k\}$, and $A = B \cup \{v_1, \ldots, v_k\}$,
- $E^{\mathfrak{B}} = \{(u_i, u_j) \in B^2 \mid |i - j| = 1\} \cup \{(u_1, u_k), (u_k, u_1)\}$,
- $E^{\mathfrak{A}} = E^{\mathfrak{B}} \cup \{(v_i, v_j) \in A^2 \mid |i - j| = 1\} \cup \{(v_1, v_k), (v_k, v_1)\}$.

Thus, $\mathfrak{B}$ is a cycle of length k, and $\mathfrak{A}$ is the disjoint union of two cycles of length k. In particular, $\mathfrak{A} \in \mathcal{K}$ and $\mathfrak{B} \notin \mathcal{K}$. We will show that if k is large enough, then Duplicator has a winning strategy in the game $N_\pi \mathrm{EF}_r(\mathfrak{A}, \mathfrak{B})$. By Corollary 4.7.7 it then follows that $\mathcal{K}$, i.e., non-connectivity of graphs, is not definable in $\mathcal{POC}[\mathcal{FO}]$.

Let

$$f_i : A^{n_i} \to \{\bot, \top\},\ 1 \leq i \leq m,$$

be the functions that Spoiler picks on his first move in the game. Duplicator will then answer by picking the functions

$$g_i : B^{n_i} \to \{\bot, \top\}, 1 \leq i \leq m,$$

where $g_i := f_i \restriction B$, for each $1 \leq i \leq m$. For his next move, Spoiler picks a tuple

$$\vec{b}_i \in B^{n_i},\ \text{for each } i,\ 1 \leq i \leq m,$$

such that $\vec{b}_1 \ldots \vec{b}_m$ is of pattern π. Now, Duplicator can simply answer by choosing the same tuples: let

$$\vec{a}_i := \vec{b}_i, \text{ for each } 1 \leq i \leq m.$$

Clearly the requirement $f_i(\vec{a}_i) = g_i(\vec{b}_i)$ is then satisfied. The game continues after this as the first-order EF-game with r rounds on the structures $(\mathfrak{A}, \vec{a}_1 \dots \vec{a}_m)$ and $(\mathfrak{B}, \vec{b}_1 \dots \vec{b}_m)$. Since the mapping

$$\vec{a}_1 \dots \vec{a}_m \mapsto \vec{b}_1 \dots \vec{b}_m$$

respects distances between nodes in the graphs $\mathfrak{A}$ and $\mathfrak{B}$, a standard argument shows that Duplicator has a winning strategy in the rest of the game, provided that k is big enough. □

By Theorems 4.6.2 and 4.6.3, $\mathcal{BBD} \equiv \mathcal{FO}(\mathcal{POC}^+)$ and $\mathcal{RBD} \equiv \mathcal{POC}[\mathcal{FO}]$. Hence we obtain the following corollary.

Corollary 4.7.9. *$\mathcal{POC}[\mathcal{FO}] < \mathcal{FO}(\mathcal{POC}^+)$.*

As a byproduct of the results concerning the zero-one law, we obtain the following results concerning expressive power.

Proposition 4.7.10. *$\mathcal{BBD} < \mathcal{BD}$, and moreover $\mathcal{BD} \not\leq \mathcal{FO}(\mathcal{POC})$.*

Proof. Clearly $\mathcal{BBD} \leq \mathcal{BD}$. By Theorem 4.3.7 and Corollary 4.6.4, $\mathcal{FO}(\mathcal{POC})$ and $\mathcal{BBD}$ have the zero-one law. By Corollary 4.6.7, $\mathcal{BD}$ does not have the zero-one law. Therefore $\mathcal{BBD} < \mathcal{BD}$ and $\mathcal{BD} \not\leq \mathcal{FO}(\mathcal{POC})$. □

4.8 Conclusion

In this chapter we defined a new variant of dependence logic called Boolean dependence logic. Boolean dependence logic is an extension of first-order logic with dependence atoms of the form $=(\vec{x}, \alpha)$, where $\vec{x}$ is a tuple of first-order variables and α is a Boolean variable. We also introduced a notational variant of partially-ordered connectives based on the narrow Henkin quantifiers of Blass and Gurevich [BG86]. We showed that the expressive power of Boolean dependence logic and dependence logic coincide. We defined natural syntactic fragments of Boolean dependence logic and proved that the expressive power of these fragments coincide with corresponding logics based on partially-ordered connectives. More formally, we showed that

$$\mathcal{BBD} \equiv \mathcal{FO}(\mathcal{POC}^+), \mathcal{RBD} \equiv \mathcal{POC}[\mathcal{FO}] \text{ and } \forall\text{-}\mathcal{BD} \equiv \mathcal{POC}[\mathcal{QF}].$$

Moreover, we proved that the fragments of Boolean dependence logic form a strict hierarchy in terms of expressive power, i.e., we showed that

$$\forall\text{-}\mathcal{BD} < \mathcal{RBD} < \mathcal{BBD} < \mathcal{BD}.$$

Therefore, we also showed that

$$\mathcal{POC}[\mathcal{QF}] < \mathcal{POC}[\mathcal{FO}] < \mathcal{FO}(\mathcal{POC}^{+}).$$

In addition, we obtained that $\mathcal{BD}$ does not have the zero-one law, whereas the logics below $\mathcal{BBD}$ and $\mathcal{FO}(\mathcal{POC})$ have the zero-one law. Therefore we obtained that $\mathcal{BD} \not\leq \mathcal{FO}(\mathcal{POC})$.

Chapter 5

Concluding Remarks

In the above chapters we have investigated various extensions of first-order logic suitable for modeling dependences of various nature. We have considered fragments of dependence logic, Boolean dependence logic, independence-friendly logic and first-order logic enriched with partially-ordered connectives. Our perspective has been directed on the study of expressive power and computational complexity. In the broad sense, this thesis contributes to the research of fragments of second-order logic and extensions of finite variable first-order logics. Simultaneously, we contribute to the basic research of logical background theory for the concept of dependence.

In Chapter 3 we studied the finite variable fragments of independence-friendly logic and dependence logic. We compared the expressive powers of these logics to fragments of first-order and existential second-order logic, and presented a comprehensive study on the computational complexity of different variants of the satisfiability problem and the model checking problem for these logics. The main accomplishment in this chapter was the discovery of the undecidability barrier between the satisfiability problems of two-variable dependence logic and two-variable independence-friendly logic.

In Chapter 4 we studied fragments of the newly defined Boolean dependence logic and first-order logic enriched with partially-ordered connectives. We showed that the expressive power of Boolean dependence logic and dependence logic coincide, and that the expressive powers of certain fragments of Boolean dependence logic coincide with that of natural logics enriched with partially-ordered connectives, i.e., we showed that

$$\mathcal{BBD} \equiv \mathcal{FO}(\mathcal{POC}^+), \mathcal{RBD} \equiv \mathcal{POC}[\mathcal{FO}] \text{ and } \forall\text{-}\mathcal{BD} \equiv \mathcal{POC}[\mathcal{QF}].$$

Moreover, we established a strict hierarchy with respect to expressive power

within Boolean dependence logic, i.e., we established that

$$\forall\text{-}\mathcal{BD} < \mathcal{RBD} < \mathcal{BBD} < \mathcal{BD}.$$

We conclude by outlining some future research prospects. One interesting open problem to be addressed in the future is to solve whether the satisfiability problem for two-variable dependence logic with unrestricted number of Boolean variables and Boolean dependence atoms is still decidable. Another interesting area of future research is the study of finite variable fragments of first-order logic enriched with generalized atoms. Generalized atoms are atomic formulae that assert properties for teams, e.g dependence atoms are generalized atoms. For a detailed depiction see a preprint by Kuusisto [Kuu13]. In particular, we are interested in a classification of generalized atoms with regards to the complexity of the satisfiability problem for the related two-variable logic.

Bibliography

[Abr07] S. Abramsky, *A compositional game semantics for multi-agent logics of imperfect information*, J. van Benthem, D. Gabbay and B. Lowe, eds., Texts in Logic and Games **1** (2007), no. 6, 11–48.

[AKVV13] S. Abramsky, J. Kontinen, J. A. Väänänen, and H. Vollmer, *Dependence logic: Theory and applications (dagstuhl seminar 13071)*, Dagstuhl Reports **3** (2013), no. 2, 45–54.

[AV09] S. Abramsky and J. Väänänen, *From IF to BI*, Synthese **167** (2009), no. 2, 207–230 (English).

[Bar69] J. Barwise, *Applications of Strict* Π^1_1 *Predicates to Infinitary Logic*, J. Symb. Log. **34** (1969), no. 3, 409–423.

[Ber66] R. Berger, *The undecidability of the domino problem*, American Mathematical Society memoirs, American Mathematical Society, 1966.

[BG86] A. Blass and Y. Gurevich, *Henkin quantifiers and complete problems*, Annals of Pure and Applied Logic **32** (1986), 1 – 16.

[BGG97] E. Börger, E. Grädel, and Y. Gurevich, *The classical decision problem*, Perspectives in Mathematical Logic, Springer, 1997.

[BK05] J. C. Bradfield and S. Kreutzer, *The complexity of independence-friendly fixpoint logic*, CSL, 2005, pp. 355–368.

[Bra13] J. C. Bradfield, *Team building in dependence*, CSL, 2013, pp. 116–128.

[Chu36] A. Church, *A note on the entscheidungsproblem*, J. Symb. Log. **1** (1936), no. 1, 40–41.

[Coo71] S. A. Cook, *The complexity of theorem-proving procedures*, Proceedings of the third annual ACM symposium on Theory of computing (New York, NY, USA), STOC '71, ACM, 1971, pp. 151–158.

[DK12] A. Durand and J. Kontinen, *Hierarchies in dependence logic*, ACM Trans. Comput. Logic **13** (2012), no. 4, 31:1–31:21.

[Ebb14] J. Ebbing, *Complexity and expressivity of dependence logic extensions*, Ph.D. thesis, Leibniz Universität Hannover, 2014.

[EF99] H. Ebbinghaus and J. Flum, *Finite model theory*, Perspectives in Mathematical Logic, Springer-Verlag GmbH, 1999.

[EFT94] H.-D. Ebbinghaus, J. Flum, and W. Thomas, *Mathematical logic (2. ed.)*, Undergraduate texts in mathematics, Springer, 1994.

[EHLV13] J. Ebbing, L. Hella, P. Lohmann, and J. Virtema, *Boolean dependence logic and partially-ordered connectives*, WoLLIC (L. Libkin, U. Kohlenbach, and R. J. G. B. de Queiroz, eds.), Lecture Notes in Computer Science, vol. 8071, Springer, 2013, pp. 111–125.

[EHM+13] J. Ebbing, L. Hella, A. Meier, J.-S. Müller, J. Virtema, and H. Vollmer, *Extended modal dependence logic*, WoLLIC, 2013, pp. 126–137.

[EK13] F. Engström and J. Kontinen, *Characterizing quantifier extensions of dependence logic*, J. Symb. Log. **78** (2013), no. 1, 307–316.

[EKV13] F. Engström, J. Kontinen, and J. A. Väänänen, *Dependence logic with generalized quantifiers: Axiomatizations*, WoLLIC, 2013, pp. 138–152.

[End70] H. Enderton, *Finite Partially-Ordered Quantifiers*, Zeitschrift für Mathematische Logik und Grundlagen der Mathematik **16** (1970), 393–397.

[EVW02] K. Etessami, M. Y. Vardi, and T. Wilke, *First-order logic with two variables and unary temporal logic*, Inf. Comput. **179** (2002), no. 2, 279–295.

[Fag76] R. Fagin, *Probabilities on finite models*, J. Symb. Log. **41** (1976), no. 1, 50–58.

[Gal12] P. Galliani, *The dynamics of imperfect information*, Ph.D. thesis, University of Amsterdam, 2012.

[GJ90] M. R. Garey and D. S. Johnson, *Computers and Intractability; A Guide to the Theory of NP-Completeness*, W. H. Freeman & Co., New York, NY, USA, 1990.

[GKLT69] Y. Glebskii, D. Kogan, M. Liogon'kii, and V. Talanov, *Range and degree of realizability of formulas in the restricted predicate calculus*, Cybernetics **5** (1969), no. 2, 142–154 (English).

[GKV97] E. Grädel, P. G. Kolaitis, and M. Y. Vardi, *On the decision problem for two-variable first-order logic*, Bulletin of Symbolic Logic **3** (1997), no. 1, 53–69.

[GO99] E. Grädel and M. Otto, *On logics with two variables*, Theoretical Computer Science **224** (1999), 73–113.

[GOR97a] E. Grädel, M. Otto, and E. Rosen, *Two-variable logic with counting is decidable*, Proceedings of the 12th Annual IEEE Symposium on Logic in Computer Science (Washington, DC, USA), LICS '97, IEEE Computer Society, 1997, pp. 306–.

[GOR97b] ———, *Undecidability results on two-variable logics*, STACS (R. Reischuk and M. Morvan, eds.), Lecture Notes in Computer Science, vol. 1200, Springer, 1997, pp. 249–260.

[Grä13] E. Grädel, *Model-checking games for logics of incomplete information*, Theoretical Computer Science, Special Issue dedicated to GandALF 2011 (2013), 2–14.

[GV13] E. Grädel and J. A. Väänänen, *Dependence and independence*, Studia Logica **101** (2013), no. 2, 399–410.

[Hen61] L. Henkin, *Some remarks on infinitely long formulas*, Infinistic Methods, Pergamon Press, 1961, pp. 167–183.

[Hin96] J. Hintikka, *The principles of mathematics revisited*, Cambridge University Press, 1996.

[Hod97a] W. Hodges, *Compositional semantics for a language of imperfect information*, Logic Journal of the IGPL **5** (1997), no. 4, 539–563.

[Hod97b] W. Hodges, *Some strange quantifiers*, Structures in Logic and Computer Science, A Selection of Essays in Honor of Andrzej Ehrenfeucht (London, UK, UK), Springer-Verlag, 1997, pp. 51–65.

[HS89] J. Hintikka and G. Sandu, *Informational independence as a semantical phenomenon*, Logic, Methodology and Philosophy of Science (J. E. Fenstad, I. T. Frolov, and R. Hilpinen, eds.), vol. 8, Elsevier, Amsterdam, 1989, pp. 571–589.

[HST08] L. Hella, M. Sevenster, and T. Tulenheimo, *Partially Ordered Connectives and Monadic Monotone Strict NP*, J. of Logic, Lang. and Inf. **17** (2008), no. 3, 323–344.

[HU00] J. E. Hopcroft and J. D. Ullman, *Introduction to automata theory, languages and computation, second edition*, Addison-Wesley, 2000.

[KKLV11] J. Kontinen, A. Kuusisto, P. Lohmann, and J. Virtema, *Complexity of Two-Variable Dependence Logic and IF-Logic*, Proceedings of the 2011 IEEE 26th Annual Symposium on Logic in Computer Science (Washington, DC, USA), LICS '11, IEEE Computer Society, 2011, pp. 289–298.

[KO05] E. Kieronski and M. Otto, *Small substructures and decidability issues for first-order logic with two variables*, Proceedings of the Twentieth Annual IEEE Symp. on Logic in Computer Science, LICS 2005 (P. Panangaden, ed.), IEEE Computer Society Press, June 2005, pp. 448–457.

[Kon10] J. Kontinen, *Coherence and complexity in fragments of dependence logic*, Ph.D. thesis, 2010.

[Kuu13] A. Kuusisto, *A Double Team Semantics for Generalized Quantifiers*, ArXiv e-prints (2013).

[KV13] J. Kontinen and J. A. Väänänen, *Axiomatizing first-order consequences in dependence logic*, Ann. Pure Appl. Logic **164** (2013), no. 11, 1101–1117.

[Lib04] L. Libkin, *Elements of finite model theory (texts in theoretical computer science. an eatcs series)*, SpringerVerlag, 2004.

[Loh12] P. Lohmann, *Computational aspects of dependence logic*, Ph.D. thesis, Leibniz Universität Hannover, 2012.

[LPRT95] K. Lodaya, R. Parikh, R. Ramanujam, and P. S. Thiagarajan, *A logical study of distributed transition systems*, Inf. Comput. **119** (1995), no. 1, 91–118.

[LV13] P. Lohmann and H. Vollmer, *Complexity results for modal dependence logic*, Studia Logica **101** (2013), no. 2, 343–366.

[Mor75] M. Mortimer, *On languages with two variables*, Zeitschr. f. math. Logik u. Grundlagen d. Math **21** (1975), 135–140.

[MSS11] A. L. Mann, G. Sandu, and M. Sevenster, *Independence-friendly logic - a game-theoretic approach*, London Mathematical Society lecture note series, vol. 386, Cambridge University Press, 2011.

[Nur09] V. Nurmi, *Dependence logic : investigations into higher-order semantics defined on teams*, Ph.D. thesis, University of Helsinki, 2009.

[Pap94] C. H. Papadimitriou, *Computational complexity*, Addison-Wesley, 1994.

[PH05] I. Pratt-Hartmann, *Complexity of the two-variable fragment with counting quantifiers*, J. of Logic, Lang. and Inf. **14** (2005), no. 3, 369–395.

[PH08] ———, *On the computational complexity of the numerically definite syllogistic and related logics*, Bulletin of Symbolic Logic **14** (2008), no. 1, 1–28.

[PST97] L. Pacholski, W. Szwast, and L. Tendera, *Complexity of two-variable logic with counting*, Proceedings of the 12th Annual IEEE Symposium on Logic in Computer Science (Washington, DC, USA), LICS '97, IEEE Computer Society, 1997, pp. 318–.

[Sco62] D. Scott, *A decision method for validity of sentences in two variables*, Journal of Symbolic Logic **27** (1962), 377.

[Sev09] M. Sevenster, *Model-theoretic and computational properties of modal dependence logic*, J. Log. Comput. **19** (2009), no. 6, 1157–1173.

[ST06] M. Sevenster and T. Tulenheimo, *Partially ordered connectives and Σ^1_1 on finite models*, Proceedings of the 2nd Computability in Europe Conference (CiE 2006), Logical Approaches to Computational Barriers, volume LNCS 3988, 2006, pp. 516–525.

[SV92] G. Sandu and J. Väänänen, *Partially ordered connectives*, Zeitschrift für Mathematische Logik und Grundlagen der Mathematik **38** (1992), 361–372.

[Tur36] A. M. Turing, *On computable numbers, with an application to the entscheidungsproblem*, Proceedings of the London Mathematical Society **42** (1936), 230–265.

[Vää07] J. A. Väänänen, *Dependence logic - a new approach to independence friendly logic*, London Mathematical Society student texts, vol. 70, Cambridge University Press, 2007.

[Var82] M. Y. Vardi, *The complexity of relational query languages (extended abstract)*, Proceedings of the fourteenth annual ACM symposium on Theory of computing (New York, NY, USA), STOC '82, ACM, 1982, pp. 137–146.

[VH10] J. Väänänen and W. Hodges, *Dependence of variables construed as an atomic formula*, Ann. Pure Appl. Logic **161** (2010), no. 6, 817–828.

[Wal70] W. J. J. Walkoe, *Finite partially-ordered quantification*, J. Symb. Log. **35** (1970), no. 4, 535–555.

[Yan14] F. Yang, *On extensions and variants of dependence logic*, Ph.D. thesis, University of Helsinki, 2014.