Rheinisch-Westfälische Technische Hochschule Aachen
Lehrstuhl für Informatik 7

29. Dezember 2009

Logik erster Stufe mit Erreichbarkeitsprädikaten über unendlichen Systemen

Diplomarbeit

Stefan Schulz

Gutachter

Dr. habil. C. Löding, Prof. Dr. Dr.h.c. W. Thomas, Prof. Dr. E. Grädel

Zusammenfassung

Im Allgemeinen befasst sich Verifikation damit, zu prüfen, ob ein (möglicherweise unendliches) System bzw. eine Struktur eine gegebene Spezifikation erfüllt. Für gewöhnlich werden solche Spezifikationen durch Formeln über einer grundlegenden Logik ausgedrückt, wie zum Beispiel die Prädikatenlogik erster Stufe (FO) oder die monadische Logik zweiter Stufe (MSO). Ein Nachteil von FO-Logik ist, dass Aussagen über Erreichbarkeit, die in der Verifikation eine große Rolle spielen, nicht ausgedrückt werden können. Um diesem Problem entgegenzutreten, kann man in die mächtigere MSO-Logik wechseln oder eine Logik dazwischen benutzen, beispielsweise FO erweitert um Erreichbarkeitsprädikate. Letzterer Ansatz bietet den Vorteil, dass FO-Logik mit Erreichbarkeit auf mehr Strukturen entschieden werden kann.

Wir stellen in dieser Diplomarbeit eine Methode vor, mit der unendliche Strukturen generiert werden können, in denen diese Erweiterung von FO entscheidbar ist. Das Verfahren ist ähnlich zur Abwicklung, wobei eine Struktur transformiert wird und gleichzeitig die Entscheidbarkeit einiger Logiken erhalten bleibt. Dies erlaubt es, aus der Entscheidbarkeit von MSO-Logik zu folgern, dass in der transformierten Struktur FO-Logik mit Erreichbarkeitsprädikaten entscheidbar ist.

Im weiteren Verlauf der Arbeit untersuchen wir Klassen von unendlichen Gitterstrukturen, welche in der Verifikation, wegen ihrer einfachen Erscheinung aber gleichzeitig großen Ausdruckskraft in logischen Formeln, sehr wichtig sind. Interessanterweise reagiert die Entscheidbarkeit von Logiken über Gittern stark auf die Änderung einzelner Parameter, wie die Anzahl der Dimensionen, die vorhandenen Relationen oder die Ausdrucksstärke der Erreichbarkeit in der betrachteten Logik.

Der abschließende Schwerpunkt beschäftigt sich damit, ein Äquivalenzergebnis einer Strukturklasse zu verallgemeinern, auf der FO mit Erreichbarkeit entscheidbar ist. Die Strukturen dieser Klasse können entweder durch reguläre Grundtermersetzung (RGTR) oder durch ein endliches Gleichungssystem mit Graphoperationen dargestellt werden. Angetrieben von einer Erweiterung des zweiten Formalismus verallgemeinern wir RGTR, so dass beide Formalismen wieder die gleichen Strukturen repräsentieren.

Erklärung

Hiermit versichere ich, dass ich diese Diplomarbeit selbständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe. Die Stellen meiner Arbeit, die dem Wortlaut oder dem Sinn nach anderen Werken entnommen sind, habe ich in jedem Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht.

Aachen, den 29. Dezember 2009

RWTH Aachen University
Chair of Computer Science 7

December 29, 2009

First-Order Logic with Reachability Predicates over Infinite Systems

Diploma Thesis

Stefan Schulz

Supervisors

Dr. habil. C. Löding, Prof. Dr. Dr.h.c. W. Thomas, Prof. Dr. E. Grädel

Abstract

The general task in verification is to check whether a given (possibly infinite) system or structure satisfies a given specification. The specification is usually expressed by a logical formula in a fundamental logic like first-order (FO) predicate logic or monadic second-order (MSO) logic. A weakness of FO logic is that reachability properties, which are important in verification, are not definable. To overcome this problem, one can switch to the more powerful MSO logic or use some logic in between like the extension of FO logic by reachability predicates. The latter solution is better since FO logic with reachability tends to be decidable in more structures than MSO logic.

In this thesis we present a method to generate infinite structures on which such extensions of FO logic are decidable. The technique is similar to unfolding and transforms a structure by preserving the decidability of logical theories. This allows to transfer known decidability results of MSO logic to FO logic with reachability predicates on the class of transformed structures.

In the course of this work we focus on the class of structures of infinite grids which is very important in verification due to its simplicity and the expressiveness of logical formulas in it. The interesting thing is that the decidability of logics between FO and MSO over those simple structures turns out to be very sensitive to various parameters like the number of dimensions, the available relations and the power of reachability in the considered logics.

The final investigation of this thesis is concerned with generalizing an equivalence result of a structure class where FO logic with reachability is known to be decidable. The structures of that class can either be described by regular ground term rewriting (RGTR) or by an finite equation system with graph operations. Motivated by results for an extension of the latter formalism we give a generalization of RGTR such that both formalism again represent the same class of structures.

Contents

1 Introduction

Algorithmically checking logical formulas in a certain structure is a standard task in computer science. It is usually needed in computational model theory [BG2004] when the specification of some property can be expressed as a formula in some logic. That is to say that the formula holds in a structure if and only if the specification is met. Then the latter problem can be decided elegantly if there exists some algorithm to check formulas of the considered logic in the considered structure. For classical logics like first-order logic (FO) or monadic second-order logic (MSO) there has already been extensive research on identifying infinite structures which have a decidable theory in that logic, i.e. deciding if a formula belongs to the set of formulas holding in the considered structure.

A way of obtaining decidability of some logic in a structure or class of structures is representing the elements of the structure as words or trees and to define relations by means of automata, transducers or rewriting. By using certain closure properties of the underlying (mainly automaton-based) models one for instance obtains a decidable FO-theory for the classes of automatic structures [Hod1983, KN1995, BG2000] and tree-automatic structures [DT1990, BG2000]. Those classes contain a lot of important structures. For example the decidability of Presburger arithmetic [Pre1929, Pre1991] can also be shown by encoding it as an automatic structure; with a more advanced idea one can also decide Skolem arithmetic by an encoding as a tree-automatic structure (cf. [Blu1999]). MSO logic was first known to be decidable for the structure of the natural numbers with successor (the theory is also known as S1S) by reducing it to automata on infinite words [Büc1962]. At least as famous as that result is its generalization to the MSO-theory of the infinite binary tree (known as S2S) by using tree-automata over infinite trees [Rab1969]. Decades later structure classes have been identified where MSO logic is decidable and S1S and S2S are included. These are the classes of pushdown-graphs [MS1985] and prefix recognizable graphs [Cau1996]. Taking those structure classes as starting point we can obtain even bigger classes by structure transformation. But more on that in Section 1.1.

Reachability Something that can be expressed in MSO logic but not in FO logic is reachability. Properties that deal with reachability occur often in verification. Then FO logic is not applicable any longer and one has to switch to some temporal logic or MSO instead. Often the ability to express reachability is the only reason to make use of the increased power of MSO. Then there are fewer structures where model-checking for this logic still is decidable due to the increased power.

In those cases one can pick some logic like FO and extend it by reachability which results in better decidability of the model-checking problem. In this logic denoted by FO(R) one can express (next to the standard FO features) that one element reaches

another by a finite sequence of edges. For FO(R) we often still have positive decidability results [DT1990] on important structure classes even if MSO logic is undecidable. When using the set of edge relations as alphabet, finite sequences of relations as words and sets of them as languages then FO(R) can express reachability by languages which are the Kleene closure of some subalphabet. There is a way to extend the power of reachability expressions by allowing more complex language classes. A very natural one is the class of regular languages yielding FO with regular reachability: FO(Reg). While FO(R) logic can obviously be expressed in MSO, it is a bit more complicated, but possible as well, to simulate the regular reachability of FO(Reg) logic in MSO. Hence we get the chain of logics: FO, FO(R), FO(Reg) and MSO, i.e. each logic is a superlogic of the logic before. Logics between FO and MSO have a huge importance in verification because there exist properties which can be specified for example in FO(Reg) but not in FO(R) (cf. [LO2007]). On the other hand there exist structures where FO(Reg), or maybe even just FO(R), is the strongest logic which is decidable. So by considering logics between FO and MSO we can gain a finer graduation of their theories being decidable or undecidable.

We will light up that space between FO and MSO logic by three completely different investigations:

- Chapter 3 (introduced in Section 1.1): We will present a new kind of transformation on graph structures. This gives us a tool to systematically generate structures having a decidable FO(Reg)-theory.
- Chapter 4 (introduced in Section 1.2): Since the first approach yields many but of course not all graph structures having that very logic decidable, we focus on a relatively small but interesting class of graph structures here. The center of attention is on infinite grids and how much reachability is decidable there.
- Chapter 5 (introduced in Section 1.3): Finally we turn to another class of graph structures each element of which has a decidable FO(R)-theory. There exist two completely differently motivated formalisms, both of them creating that class of so called regular ground term rewriting graphs. For one formalism a generalization is known which still yields graphs with decidable FO(R) logic. The goal is to extend the other formalism such that both again create the same graph structures.

1.1 Structure Transformation

A very generic way of finding structures with some decidable logic is by structure transformation, i.e. creating a new structure from the original one by following some systematic principle. This works because ideal such principles furthermore allow it to transfer decidability results on logics of the original structure to the new structure which is exactly where the focus lies on in Chapter 3.

Interpretation One major tool of structure transformation is interpretation. There a new structure is described by formulas in some logic. To that end we take a formula

with a single free variable which defines the domain of the new structure as a subset of the original domain. The new relations are defined analogously each by a formula with a number of free variables according to the arity of the relation. When considering some structure then those formulas define the domain and relations of a new structure. Depending on the kind of interpretation we can more or less transfer decidability results of logical theories of the original structure to the new one. This is because formulas over the interpreted structure can be easily transformed to formulas over the original one by using the formulas which define the new domain and relations. The standard FO or MSO interpretation for example sustains the decidability of FO or MSO logic respectively.

Up to now the new elements were a subset of the original domain. This can be different in other variants of interpretations; like in finite set interpretation [CL2007] where each new element is a finite set of old elements. The domain and relations are defined by formulas on sets in a logic similar to MSO (to be more precise: Weak MSO) and decidability of this very logic in the old structure yields decidability of FO-logic in the new one.

A similar variant is tuple-interpretation where each new element is interpreted as a tuple of old elements of some fixed length. When specifying the new domain and relations in FO logic or higher and this very logic being decidable in the considered structure, we obtain the decidability of the FO-theory in the interpreted structure. The interesting thing about interpretation by finite sets or tuples is that they connect different logics like FO and MSO. We even extend the latter approach for FO logic with different levels of reachability.

Unfolding In general, tools for structure transformation do not have to be logically motivated even if some have excellent properties in transferring decidability of logical theories like just mentioned for interpretations. One main representative of this kind is the unfolding of graph structures. Here the new created structure consists of all (finite) paths starting in some specific node. The edge relations extend those paths step by step. For example the unfoldings of finite graphs are exactly the regular graphs. Both graph classes have a decidable MSO-theory. In general this holds not just for those simple structures: the MSO-theory of an unfolding of a graph structure is decidable if MSO logic is decidable in that structure [CW1998].

Caucal's Hierarchy By sequentially applying unfolding and standard MSO-interpretation to the class of finite graphs we end up with a hierarchy of graph structures (cf. [Cau2002]). The very useful property of this so called Caucal hierarchy is that each graph in it has an decidable MSO-theory since both transformations preserve the decidability of MSO logic over finite graphs.

Set-Based Unfolding Chapter 3 is concerned with an approach similar to unfolding. This transformation is called set-based unfolding in contrast to the normal (path-based) unfolding from above. The main idea stays the same but instead of having paths as new

elements we abstract from this and keep just the last element and the set of elements visited by such a path. The information about relations and the visiting order of the elements is omitted. However we get a weaker result for transferring the decidability of logics. Informally spoken Corollary 3.4 states that the FO(Reg)-theory of a set-based unfolding is decidable if in the original structure finiteness of sets is MSO-definable and MSO logic is decidable. This is a simplified version of our main result about set-based unfolding Theorem 3.3. It is shown there that transferring the decidability still works for quotients of set-based unfoldings for some MSO-definable congruence. The idea for set-based unfolding was motivated by a proof in [LO2007] showing that FO logic with regular reachability is decidable in free inverse monoids. Here we use this technique in a more general approach which is applicable to any kind of graph structure.

Since huge classes of graph structures are known to have a decidable MSO-theory this results in even bigger classes where FO(Reg) logic is decidable. Furthermore by set-based unfolding it is possible to produce structures for which the FO(Reg)-theory is decidable while the MSO-theory is not. Such structures are for example free inverse monoids [Cal1997, LO2007].

1.2 Classifying a Structure's Decidability

Besides classifying huge classes of structures in terms of some logic being decidable by using structure transformation. However, in computational model theory it is important as well to look at a fixed class of structures and to determine which logics are decidable or undecidable there and what their expressive power is. Chapter 4 is dealing with this very topic with a focus on infinite grid-like structures. Grids are very interesting structures since they look simple but one can reduce highly undecidable theoretical problems to the model-checking of powerful logics like MSO [Tho1990, WT2004]. On the other hand these structures have a decidable FO-theory. As already mentioned in the beginning FO with regular reachability is a logic between FO and MSO. In our case this means it is somewhere between decidable and highly undecidable. We furthermore consider reachability by language classes like context-free languages which goes beyond the expressiveness of MSO logic. The expressive power of FO formulas with context-free reachability is incomparable to MSO according to Remark 4.17 which is a consequence to the main results of Chapter 4.

Reachability in Grids The main results about decidability of FO logic with reachability in grid structures strongly depend on the number of dimensions and on which relations are available in the grid. The second aspect is of big importance here because the relations of a structure can be used in reachability expressions which can dramatically increase the expressive power. In contrast to that in plain first-order logic we can omit or add relations as long as they are definable by some formula. For grid structures of any dimension which just have successor relations for each dimension FO(CF) logic is decidable according to Theorem 4.7. It stays decidable even if we add unary predicates for testing whether dimension of being zero according to Corollary 4.10. On the other

hand things get complicated very fast if we introduce predecessor relations for each dimension. Together with the relations for zero-testing we get the undecidability of FO(Reg) logic in the 2-dimensional grid by Theorem 4.15. If we refrain from using zero-tests we have undecidability of FO(Reg) logic as well by Corollary 4.12 but this time just for at least 3 dimensions. The latter result also shows that FO(Reg) logic over 3-dimensional grid structures with backward edges is highly undecidable since in the proof of Corollary 4.12 we reduce the Peano arithmetic [Göd1931] to this problem. Most of those main results are connected to a number of similar results concerning FO with reachability by more exotic language classes in grid structures.

Transitive Closure Another extension FO(TC) of first-order logic allows transitive closure [CR1998, Imm1999, Avr2003]. In its simplest form $\mathrm{FO(TC)}_{(1)}$ we introduce a new expression which describes the transitive closure of the binary relation of a given formula with two free variables. This logic is interesting since transitive closure of binary relations defined by formulas can be expressed in MSO as well. Hence $\mathrm{FO(TC)}_{(1)}$ is another logic between FO and MSO. We continue the considerations of [Wöh2005] and show a logic being decidable on 2-dimensional grids but not in the 3-dimensional case (cf. Corollary 4.22). This result is interesting since it is the first time a difference in decidability of a fixed logic occurs not just between grids of dimension 1 and 2 but between 2 and 3 instead.

1.3 Equivalent Formalizations

As mentioned before the decidability of a logic in a structure is either a consequence of some structure transformation or the elements and relations of the structure can be encoded in an underlying formalism which is mostly motivated by automata.

Ground Term Rewriting One such formalism is ground term rewriting (GTR) as studied in [Bra1969, DT1990, Löd2002, Löd2003]. One can define a class of structures where the elements consist of terms over a given ranked alphabet. Two such terms are connected by a relation if the first one contains a ground term (i.e. a term with no variables) and there is a rewriting rule which allows to replace that ground term by another ground term such that the second term arises after the rewriting. The structures in that class (like for example the grid structures of Chapter 4) are called ground term rewriting graphs and have a decidable FO(R)-theory [DT1990]. This was proven by representing the terms as finite trees and encoding the relations as tree-automatic relations, i.e. as tree-automata recognizing the overlay of two trees which are in that relation. By this, one can show FO logic to be decidable. The simple reachability which is still missing is realized by the use of tree-transducers for rewriting which have the nice property to be closed under transitive closure [DT1985] and to describe tree-automatic relations [DT1990]. We do not lose any decidability if we allow regular sets of terms (or view them as trees) instead of just a single term on either side of a rewriting rule since tree-

automata are used in the proof anyway. This yields the slightly generalized regular ground term rewriting (RGTR) graphs where FO(R) logic is still decidable.

At the beginning of Chapter 5 we consider a restricted variant of GTR allowing to only rewrite ground terms without nesting, i.e. when seeing terms as trees then each left side of a rewriting rule is just a leaf. This approach is called ground term expansion (GTX) but even for the simple graphs generated by GTX we get undecidable FO(Reg) logic (cf. Remark 5.12). More precisely this has been shown for the product of two infinite binary trees which each is a configuration graph of a basic process algebra (BPA) [BW1990, May2000] and has a decidable MSO-theory. This means that products of BPA graphs do not preserve that decidability.

Vertex Replacement with Product A completely differently motivated approach of defining the very same graphs as by RGTR is by equation systems and graph operations. For vertex replacement with product (VRP) we use five such operations on colored graphs (for some finite set of colors) as defined in [Col2002, Col2004]: a constant singleton graph, recoloring of a graph, adding edges between vertices of some color to the graph, the disjoint union of two graphs and the asynchronous product of two graphs. A system of equations with those VRP operators can also be seen as a tree having those operators as vertices. We unfold the equation system which yields a tree of VRP operators: the so called VRP tree. Then a finite equation system is equal to a VRP tree which is regular, i.e. there are only finitely many subtrees. The graph which is represented by such a infinite VRP tree (or recursively defined equation system) is the least fixed point of the root. The main result of Colcombet's work is that RGTR graphs are isomorphic to the graphs which can be described by the least fixed point of a regular VRP tree or finite VRP equation system (without considering colors). Furthermore he explained how to decide the FO(R)-theory of a graph represented by VRP tree by reducing it to the MSO-theory of the tree itself. This means that in case of regular VRP trees we have that FO(R) logic is decidable in the graph represented by the tree since MSO is decidable in regular trees. For non-regular VRP trees which have a decidable MSO-theory we still get the decidability of FO(R) logic in its graph but here we have no equivalent representation. The main investigation of Chapter 5 is to find a formalism to fill this gap.

A Generalization of Ground Term Rewriting Since VRP trees with a decidable MSO-theory are a superset of regular VRP trees we use the equivalence of this weaker approach to RGTR and extend it to generalized regular ground term rewriting (GRGTR). The idea is to introduce some preferably infinite tree with decidable MSO-theory in RGTR as well which will function as a skeleton for the trees and rewriting rules of the RGTR. Then like for tree-automatic relations we overlay the RGTR trees with the skeleton tree which enables rewriting rules to describe trees not just by the ground tree at some position but instead by an overlay of that ground tree and the skeleton tree at the same position. The main result of Chapter 5 is Corollary 5.22 stating that GRGTR graphs are effectively equivalent to the graphs represented by VRP trees (up to isomorphism and

color removal). The MSO-theory of the skeleton tree is of course decidable if and only if the MSO-theory of the VRP tree is decidable. A direct consequence is that FO(R) logic is decidable in graphs of GRGTR if the skeleton tree has a decidable MSO-theory (cf. Corollary 5.20). Furthermore we get Colcombet's equivalence as a special case, since the skeleton tree is regular if and only if the VRP tree is, and GRGTR with a regular skeleton tree can be simulated by plain RGTR already.

2 Definitions

2.1 Sets, Functions, etc.

We will use the following sets of numbers: the *Boolean values* $\mathbb{B} := \{0, 1\}$, the *natural numbers* $\mathbb{N} := \{0, 1, 2, \ldots\}$ and the set of all *integers* $\mathbb{Z} := \{\ldots, -2, -1, 0, 1, 2, \ldots\}$.

The *power set* of a set S will be denoted by $\mathcal{P}(S) := \{ S' \mid S' \subseteq S \}$. The *product* of sets $S_1, \ldots, S_n$ is defined as $S_1 \times \ldots \times S_n := \left\{ (s_1, \ldots, s_n) \;\middle|\; s_i \in S_i \right\}$. If $S = S_1 = \ldots = S_n$ we abbreviate $S^n = S_1 \times \ldots \times S_n$. The set of all *finite sequences* of a set S is denoted by $S^* := \bigcup_{n \in \mathbb{N}} S^n$.

Functions

Given sets A, B, a *function* $f : A \to B$ associates each entry $a \in A$ with an entry $f(a) \in B$. The set $\mathrm{dom}(f) := A$ is called *domain* and B is called the *codomain* of f. Consider functions $f_i : S_{i-1} \to S_i$ with $i \in \{1, \ldots, n\}$ for some $n \in \mathbb{N}$. The *composition* of them is the function $f_1 \circ \ldots \circ f_n : S_0 \to S_n$ with $\big(f_1 \circ \ldots \circ f_n\big)(s_0) := f_n\big(\ldots f_2\big(f_1(s_0)\big)\ldots\big)$. If $S = S_0 = \ldots = S_n$ and $f = f_1 = \ldots = f_n : S \to S$ we abbreviate $f^n := f_1 \circ \ldots \circ f_n : S \to S$. Note that for the case $n = 0$ it follows that $f^0 =: \mathsf{id}_S$ is the *identity on* S with $\mathsf{id}_S(s) := s$.

With $[a_1 \mapsto b_1, \ldots, a_n \mapsto b_n]$ or $\begin{bmatrix} a_1 \mapsto b_1 \\ \vdots \\ a_n \mapsto b_n \end{bmatrix}$ we denote a function $f : \{a_1, \ldots, a_n\} \to B$ with $f(a_i) := b_i$ for some codomain $B \supseteq \{b_1, \ldots, b_n\}$. Note that in case of $n = 0$ we get an *empty function* $[] : \varnothing \to B$. Given a function $f : A \to B$ we construct a new function $f[a_1 \mapsto b_1, \ldots, a_n \mapsto b_n] : A \to B$ with $a_i \in A$ and $b_i \in B$ such that

$$f[a_1 \mapsto b_1, \ldots, a_n \mapsto b_n](a) \quad := \quad \begin{cases} b_i & \text{if } a = a_i \text{ for some } i \in \{1, \ldots, n\}, \\ f(a) & \text{otherwise.} \end{cases}$$

2.2 Structures

Signatures and Structures

A *signature* τ is a set of symbols for functions and relations. With $F_n(\tau)$ and $R_n(\tau)$ we denote the function and relation symbols respectively of signature τ having arity n. A *τ-structure* $\mathfrak{A} = \big(A, (f^{\mathfrak{A}})_{f \in F_n(\tau), n \in \mathbb{N}}, (P^{\mathfrak{A}})_{P \in R_n(\tau), n \in \mathbb{N}}\big)$ consists of some set A as *domain*, an n-ary function $f^{\mathfrak{A}} : A^n \to A$ for each function symbol $f \in F_n(\tau)$ of arity n and an n-ary relation $P^{\mathfrak{A}} \subseteq A^n$ for each relation symbol $P \in R_n(\tau)$ of arity n. We will consider only finite signatures in this thesis. Furthermore we will not explicitly distinguish between a function or relation symbol and the actual function or relation if the meaning is clear

from context. On the other hand if the signature is not given explicitly we use the natural names of the functions and relations of the structure as their symbols.

Equivalences, Congruences, Quotients

Let A be some set. A binary relation $\sim \subseteq A \times A$ is called an *equivalence relation* if it is:

- *Reflexive*: for all $a \in A$ holds $a \sim a$,
- *Symmetric*: for all $a, b \in A$ holds $b \sim a$ if $a \sim b$,
- *Transitive*: for all $a, b, c \in A$ holds $a \sim c$ if $a \sim b$ and $b \sim c$.

For some $a \in A$ the set $[a]_\sim := \{ b \in A \mid a \sim b \}$ (or simply $[a]$ if $\sim$ is understood) is called the *equivalence class* of element a which called *representative* of its class $[a]$. The set of all equivalence classes is $A/\sim := \big\{ [a] \;\big|\; a \in A \big\}$.

Let $\mathfrak{A} = \big(A, (f^{\mathfrak{A}})_{f \in F}, (P^{\mathfrak{A}})_{P \in R}\big)$ be a structure. An equivalence relation $\sim \subseteq A \times A$ on the domain of the structure is called *congruence on* $\mathfrak{A}$ if it is compliant with the functions and relations of the structure, i.e. that for all $n \in \mathbb{N}$ and $a_1, \ldots, a_n, b_1, \ldots, b_n \in A$ with $a_i \sim b_i$ for all $i \in \{1, \ldots, n\}$:

- for each n-ary function $f^{\mathfrak{A}}$ of $\mathfrak{A}$ holds $f^{\mathfrak{A}}(a_1, \ldots, a_n) \sim f^{\mathfrak{A}}(b_1, \ldots, b_n)$,
- for each n-ary relation $P^{\mathfrak{A}}$ of $\mathfrak{A}$ holds $(a_1, \ldots, a_n) \in P^{\mathfrak{A}} \Leftrightarrow (b_1, \ldots, b_n) \in P^{\mathfrak{A}}$.

Let $\sim$ be a congruence on a structure $\mathfrak{A} = \big(A, (f^{\mathfrak{A}})_{f \in F}, (P^{\mathfrak{A}})_{P \in R}\big)$ again. Then the *quotient structure* or *factor structure* $\mathfrak{A}/\sim = \big(A/\sim, (f^{\mathfrak{A}/\sim})_{f \in F}, (P^{\mathfrak{A}/\sim})_{P \in R}\big)$ is a structure having the same signature as $\mathfrak{A}$ with:

- for each n-ary function $f^{\mathfrak{A}}$ of $\mathfrak{A}$: $f^{\mathfrak{A}/\sim}([a_1], \ldots, [a_n]) := f^{\mathfrak{A}}(a_1, \ldots, a_n)$,
- for each n-ary relation $P^{\mathfrak{A}}$ of $\mathfrak{A}$: $([a_1], \ldots, [a_n]) \in P^{\mathfrak{A}/\sim} :\Leftrightarrow (a_1, \ldots, a_n) \in P^{\mathfrak{A}}$.

These definition is independent from the choice of the representatives $a_1, \ldots, a_n$: all elements $b_1, \ldots, b_n$ with $a_i \sim b_i$ yield the same result since $\sim$ is a congruence.

2.3 Words, Languages, Automata, etc.

Consider the free monoid generated by some finite set Σ. The elements of the monoid are exactly the finite strings of symbols over the *alphabet* Σ and called *words*. The neutral element is the *empty word* ϵ and the operation $\cdot$ is called *concatenation*. The *length* $|w|$ of a word $w \in \Sigma^*$ is n iff $w \in \Sigma^n$. The number of occurrences of a symbol $a \in \Sigma$ in a word $w \in \Sigma^*$ is denoted by $|w|_a$. A set $L \subseteq \Sigma^*$ of words is called *language*. The concatenation can be extended to languages $L_1, \ldots, L_n \subseteq \Sigma^*$ as well: $L_1 \cdot \ldots \cdot L_n = \{ w_1 \cdot \ldots \cdot w_n \mid w_i \in L_i \}$. Concatenation can be understood as the product operation on words or languages. Thus for $L = L_1 = \ldots = L_n$ we abbreviate $L^n := L_1 \cdot \ldots \cdot L_n$ and the *Kleene closure* of L is defined as $L^* = \bigcup_{n \in \mathbb{N}} L^n$. Note that $L^0 = \{\epsilon\} \subseteq L^*$.

2.3.1 Regular Languages

The *regular languages* $\mathcal{L}_{\text{Reg}}(\Sigma)$ over some alphabet Σ (or simply $\mathcal{L}_{\text{Reg}}$ if the alphabet is clear from context) are defined inductively:

- The *empty language* and the *singleton languages*:
  $\varnothing \in \mathcal{L}_{\text{Reg}}(\Sigma)$, $\{\epsilon\} \in \mathcal{L}_{\text{Reg}}(\Sigma)$ and $\{a\} \in \mathcal{L}_{\text{Reg}}(\Sigma)$ for each $a \in \Sigma$,

- The *union*, the *concatenation* and the *Kleene closure* of regular languages:
  $L \cup L' \in \mathcal{L}_{\text{Reg}}(\Sigma)$, $L{\cdot}L' \in \mathcal{L}_{\text{Reg}}(\Sigma)$ and $L^* \in \mathcal{L}_{\text{Reg}}(\Sigma)$ where $L, L' \in \mathcal{L}_{\text{Reg}}(\Sigma)$.

Regular languages can be represented by *regular expressions* which are the expressions $\varnothing$, ϵ, a for each $a \in \Sigma$ and $r+r'$, $r{\cdot}r'$, r^* where r, r' are regular expressions. The semantics of these operators correspond to the language operations defined above.

Finite Automata

Another way of representing regular languages is by a *deterministic finite automaton* $\mathcal{A} = (Q, \Sigma, \delta, q_0, F)$ with finite set Q of *states*, alphabet Σ, *transition function* $\delta : Q \times \Sigma \to Q$, *initial state* $q_0 \in Q$ and *final states* $F \subseteq Q$. We extend the transition function δ to work on words instead of just single symbols: $\hat{\delta} : Q \times \Sigma^* \to Q$ where $\hat{\delta}(q, \epsilon) = q$ and $\hat{\delta}(q, w{\cdot}a) = \delta\big(\hat{\delta}(q, w), a\big)$. The language *accepted* by automaton $\mathcal{A}$ is $L(\mathcal{A}) := \big\{\, w \in \Sigma^* \;\big|\; \hat{\delta}(q_0, w) \in F \,\big\}$.

A *non-deterministic finite automaton* $\mathcal{A} = (Q, \Sigma, \Delta, I, F)$ with finite *state set* Q, alphabet Σ, *transition relation* $\Delta \subseteq Q \times \Sigma^* \times Q$, *initial states* $I \subseteq Q$ and *final states* $F \subseteq Q$. Again we extend the transition relation Δ to work on concatenations of words: $\hat{\Delta} : Q \times \Sigma^* \times Q$ where $(q, \epsilon, q) \in \hat{\Delta}$ and $(p, u{\cdot}v, q) \in \hat{\Delta}$ iff $(p, u, p') \in \hat{\Delta}$ and $(p', v, q) \in \Delta$. The language *accepted* by the automaton $\mathcal{A}$ is $L(\mathcal{A}) := \big\{\, w \in \Sigma^* \;\big|\; (p, w, q) \in \hat{\Delta},\ p \in I,\ q \in F \,\big\}$.

The following well known result connects the previous three approaches:

Proposition 2.1 ([HU1979]) *The following formalisms describe exactly the regular languages $\mathcal{L}_{Reg}$:*

a) Regular expressions,

b) Deterministic finite automata,

c) Non-deterministic finite automata.

Furthermore one can switch between these formalisms effectively. □

Deterministic finite automata have the big advantage that they are easy to complement. Given a deterministic finite automaton $\mathcal{A} = (Q, \Sigma, \delta, q_0, F)$, the deterministic finite automaton $\bar{\mathcal{A}} = (Q, \Sigma, \delta, q_0, Q \setminus F)$ accepts the *complement*: $L(\bar{\mathcal{A}}) = \overline{L(\mathcal{A})} := \Sigma^* \setminus L(\mathcal{A})$. Thus regular languages are closed under complement.

Star-Free Languages

The *star-free languages* $\mathcal{L}_{\mathrm{SF}}(\Sigma)$ over some alphabet Σ (or simply $\mathcal{L}_{\mathrm{SF}}$) are a subset of the regular languages $\mathcal{L}_{\mathrm{Reg}}$. To that end we introduce an operator for the complement which does not increase the expressiveness. On the other hand we remove the Kleene closure from the set of operators. Formally the star-free languages are defined as follows:

- The *empty language* and the *singleton languages*:
  $\varnothing \in \mathcal{L}_{\mathrm{SF}}(\Sigma)$, $\{\epsilon\} \in \mathcal{L}_{\mathrm{SF}}(\Sigma)$ and $\{a\} \in \mathcal{L}_{\mathrm{SF}}(\Sigma)$ for each $a \in \Sigma$,
- The *union*, the *concatenation* and the *complement* of star-free languages:
  $L \cup L' \in \mathcal{L}_{\mathrm{SF}}(\Sigma)$, $L{\cdot}L' \in \mathcal{L}_{\mathrm{SF}}(\Sigma)$ and $\overline{L} \in \mathcal{L}_{\mathrm{SF}}(\Sigma)$ where $L, L' \in \mathcal{L}_{\mathrm{SF}}(\Sigma)$.

Again we have expressions — the *star-free expressions* — to describe star-free languages: $\varnothing$, ϵ, a for each $a \in \Sigma$ and $r + r'$, $r{\cdot}r'$, ${\sim}r$ where r, r' are star-free expressions. The semantics of these operators correspond to the language operations defined above.

2.3.2 Context-Free Languages

A *context-free grammar* $G = (N, \Sigma, \to, S)$ consists of a finite set N of *non-terminal* symbols, an alphabet Σ, a finite set $\to \subseteq N \times (N \cup \Sigma)^*$ of *production rules* and a *start symbol* $S \in N$. We define a *reduction relation* $\Rightarrow \subseteq (N \cup \Sigma)^* \times (N \cup \Sigma)^*$ as an extension of the production rules $\to$: $\alpha{\cdot}A{\cdot}\alpha' \Rightarrow \alpha{\cdot}\beta{\cdot}\alpha'$ iff $A \to \beta$ for some $A \in N$, $\alpha, \alpha' \in (N \cup \Sigma)^*$. The language represented by G consists of the Σ-words reachable from the start symbol S: $L(G) := \{\, w \in \Sigma^* \mid S \Rightarrow^* w \,\}$. The *Context-free languages* $\mathcal{L}_{\mathrm{CF}}(\Sigma)$ (or simply $\mathcal{L}_{\mathrm{CF}}$) are the languages accepted by context-free grammars.

The regular languages $\mathcal{L}_{\mathrm{Reg}}$ are completely included in the context-free languages $\mathcal{L}_{\mathrm{CF}}$. Given a non-deterministic finite automaton $\mathcal{A} = (Q, \Sigma, \Delta, q_0, F)$ we construct a context-free grammar $G_{\mathcal{A}} = (Q, \Sigma, \to, q_0)$ with $p \to a{\cdot}q$ for each $(p, a, q) \in \Delta$ and $q \to \epsilon$ iff $q \in F$. Then it is easy to see that the accepted languages are equal: $L(\mathcal{A}) = L(G_{\mathcal{A}})$.

Normal Forms

A context-free grammar $G = (N, \Sigma, \to, S)$ is in *Chomsky normal form* iff $\to$ only contains rules of the form $A \to B{\cdot}C$, $A \to a$ and $S \to \epsilon$ with $A, B, C \in N$, $a \in \Sigma$ and the start symbol $S \in N$. The grammar G is in *Greibach normal form* iff $\to$ only contains rules $A \to a{\cdot}B_1{\cdot}\ldots{\cdot}B_k$ or $S \to \epsilon$ with $k \in \mathbb{N}$, $A, B_1, \ldots, B_k \in N$, $a \in \Sigma$ and the start symbol $S \in N$.

Pushdown Automata

A *non-deterministic pushdown automaton* $\mathcal{P}$ is a generalization of the non-deterministic finite automaton. $\mathcal{P} = (Q, \Sigma, \Gamma, \Delta, \gamma_0, I, F)$ consists of a finite state set Q, alphabet Σ, *stack alphabet* Γ, a finite *pushdown transition relation* $\Delta \subseteq Q \times \Sigma^* \times \Gamma \times Q \times \Gamma^*$, an *initial stack symbol* $\gamma_0 \in \Gamma$ and sets of initial and final states $I, F \subseteq Q$. A pushdown automaton can store information not just in its state but in a *stack* as well. We define

an extension $\hat{\Delta} \subseteq Q \times \Sigma^* \times \Gamma^* \times Q \times \Gamma^*$ of the transition relation Δ by $(q, \epsilon, \omega, q, \omega) \in \hat{\Delta}$ and $(p, u{\cdot}v, \mu, q, \omega{\cdot}\nu) \in \hat{\Delta}$ iff $(p, u, \mu, p', \omega{\cdot}\gamma) \in \hat{\Delta}$ and $(p', v, \gamma, q, \nu) \in \Delta$. The language accepted by $\mathcal{P}$ is $L(\mathcal{P}) := \{ w \in \Sigma^* \mid (p, w, \gamma_0, q, v) \in \hat{\Delta},\ p \in I,\ q \in F,\ v \in \Gamma^* \}$.

Like Proposition 2.1 for regular languages we have an equivalence of the formalisms just introduced:

Proposition 2.2 *Each of the following formalisms describes exactly the context-free languages $\mathcal{L}_{CF}$:*

a) Context-free grammars,

b) Context-free grammars in Chomsky normal form,

c) Context-free grammars in Greibach normal form,

d) Non-deterministic pushdown automata.

Furthermore one can switch between these formalisms effectively. □

Counter Languages

Let us consider a restriction of context-free languages. A *non-deterministic counter automaton* $\mathcal{P} = (Q, \Sigma, \mathbb{B}, \Delta, 0, I, F)$ is a non-deterministic pushdown automaton where the stack alphabet may contain just one symbol (here: 1) next to the initial stack symbol (here: 0). Furthermore the transition relation $\Delta \subseteq \big(Q \times \Sigma^* \times \{0\} \times Q \times (0{\cdot}1^*)\big) \cup \big(Q \times \Sigma^* \times \{1\} \times Q \times \{1\}^*\big)$ must leave the initial stack symbol at the bottom of the stack. According to this the stack content is always of the form $0{\cdot}1^n$ for some $n \in \mathbb{N}$. Thus the stack can be identified with a counter holding some value $n \in \mathbb{N}$. We will use this to represent a non-deterministic counter automaton $\mathcal{P}$ equivalently by $\mathcal{C} = (Q, \Sigma, \Delta', I, F)$ with *counter transition relation* $\Delta' \subseteq Q \times \Sigma^* \times \mathbb{B} \times Q \times \mathbb{N}$. A transition rule $(p, w, b, q, m) \in \Delta'$ means that from state p with counter value n one can change to state q and counter value $n - b + m$ by reading input symbols w iff $\mathrm{sgn}(n) = b$. Thus for each rule $(p, w, 0, q, 0{\cdot}1^n) \in \Delta$ we have $(p, w, 0, q, n) \in \Delta'$ and for $(p, w, 1, q, 1^n) \in \Delta$ we get $(p, w, 1, q, n) \in \Delta'$ to specify the same language. The languages defined by non-deterministic counter automata are called *counter languages* $\mathcal{L}_{\mathrm{CL}}(\Sigma)$ or $\mathcal{L}_{\mathrm{CL}}$ for short and are obviously a subset of the context-free languages $\mathcal{L}_{\mathrm{CF}}$.

2.3.3 Petri Net Languages

A *Petri net* $N = (P, T, E)$ consists of finite disjoint sets P, T of *places* and *transitions* respectively and an edge relation $E \subseteq (P \times T) \cup (T \times P)$. A *marking* $m : P \to \mathbb{N}$ assigns $m(p)$ of *tokens* to each place $p \in P$. A transition $t \in T$ can *fire* or *get executed* iff there is at least one token in each of its *input places* ${}^\bullet t := \{ p \in P \mid (p, t) \in E \}$. To *fire* or *execute* a transition $t \in T$ for some marking $m_0 : P \to \mathbb{N}$ we remove exactly one token from each *input place* ${}^\bullet t := \{ p \in P \mid (p, t) \in E \}$ and add exactly one to each *output place*

$t^\bullet := \{ p \in P \mid (t,p) \in E \}$ leading to a new marking $m : P \to \mathbb{N}$. We write:

$$m_0 \xrightarrow{t}_N m \quad \text{with} \quad 0 \notin m_0({}^\bullet t), \quad m(p) \;:=\; m_0(p) + \begin{cases} -1 & \text{iff } p \in {}^\bullet t, \\ 0 & \text{otherwise} \end{cases} + \begin{cases} 1 & \text{iff } p \in t^\bullet, \\ 0 & \text{otherwise.} \end{cases}$$

We extend this notation to transition sequences: $m_0 \xrightarrow{t_1,\ldots,t_n}_N m$ if there exist markings $m_1, \ldots, m_n$ such that $m_{i-1} \to_N^{t_i} m_i$ for each single transition and $m = m_n$. We say then that m is *reachable* from m_0, or $m_0 \to_N m$ for short. A place p is *bounded* for a given marking m_0 iff $m(p)$ is bounded for all markings m reachable from m_0, otherwise it is *unbounded*.

To connect Petri nets with formal languages we introduce some labeling: a Σ-*labeled Petri net* $\mathcal{P} = (N, \Sigma, \lambda, m_0, m_f)$ consists of a Petri net $N = (P, T, E)$, an alphabet Σ, a *labeling function* $\lambda : T \to \Sigma^*$ and the *initial* and *final markings* $m_0, m_f : P \to \mathbb{N}$. We extend the labeling to transition sequences. We write $m \xrightarrow{w_1 \cdot \ldots \cdot w_n}_{\mathcal{P}} m'$ if $m \xrightarrow{t_1,\ldots,t_n}_N m'$ for some transitions $t_1, \ldots, t_n \in T$ with $\lambda(t_i) = w_i$. The labeled Petri net $\mathcal{P}$ accepts the language $L(\mathcal{P}) := \{ w \in \Sigma^* \mid m_0 \xrightarrow{w}_{\mathcal{P}} m_f \}$. The *n-unbounded Petri net languages* $\mathcal{L}_{\mathrm{PN}^n}(\Sigma)$, or $\mathcal{L}_{\mathrm{PN}^n}$, are the languages accepted by labeled Petri nets with at most n unbounded places for some $n \in \mathbb{N}$. If the number of unbounded places does not matter we get the *Petri net languages* $\mathcal{L}_{\mathrm{PN}}(\Sigma) := \bigcup_{n \in \mathbb{N}} \mathcal{L}_{\mathrm{PN}^n}(\Sigma)$, abbreviated as $\mathcal{L}_{\mathrm{PN}}$. Obviously we have the inclusions $\mathcal{L}_{\mathrm{PN}^n} \subseteq \mathcal{L}_{\mathrm{PN}^{n+1}} \subseteq \mathcal{L}_{\mathrm{PN}}$ for all $n \in \mathbb{N}$.

We now want to compare Petri net languages to the language classes defined previously in terms of expressiveness. The easiest case are 0-unbounded Petri net languages. Consider a labeled Petri net $\mathcal{P} = \big(N = (P,T,E), \Sigma, \lambda, m_0, m_f\big)$ which defines such a language. Since all places are bounded we can just reach a finite number of markings from the initial marking m_0. This can easily be encoded by a finite automaton. Let $\mathcal{A}_{\mathcal{P}} = (Q, \Sigma, \Delta, I, F)$ be an extended non-deterministic finite automaton with states $Q := \{ m : P \to \mathbb{N} \mid m_0 \to_N^* m \} \cup \{m_f\}$, initial states $I := \{m_0\}$, finial states $F := \{m_f\}$ and transitions $\big(m, \lambda(t), m'\big) \in \Delta$ if $m \xrightarrow{t}_N m'$ for some $t \in T$ and $m, m' \in Q$. By construction the automaton accepts the same language as the Petri net: $L(\mathcal{A}_{\mathcal{P}}) = L(\mathcal{P})$.

Remark 2.3 0-unbounded Petri nets languages are regular: $\mathcal{L}_{\mathrm{PN}^0} \subseteq \mathcal{L}_{\mathrm{Reg}}$. □

The converse holds as well. Given an extended non-deterministic finite automaton $\mathcal{A} = (Q, \Sigma, \Delta, I, F)$ we consider a labeled Petri net $\mathcal{P}_{\mathcal{A}} = (N, \Sigma, \lambda, m_0, m_f)$ with Petri net $N = (P, T, E)$ where:

- $P := Q \,\dot\cup\, \{q_0', q_f'\}, \qquad T := \{ t_{I,q} \mid q \in I \} \cup \Delta \cup \{ t_{F,q} \mid q \in F \},$
- $E := \bigcup\limits_{q \in I} \big\{(q_0', t_{I,q}), (t_{I,q}, q)\big\} \cup \bigcup\limits_{t=(p,w,q) \in \Delta} \big\{(p,t),(t,q)\big\} \cup \bigcup\limits_{q \in F} \big\{(q, t_{F,q}), (t_{F,q}, q_f')\big\},$
- $\lambda(t) := \begin{cases} w & \text{if } t = (p,w,q) \in \Delta, \\ \epsilon & \text{otherwise,} \end{cases} \quad m_0(p) := \begin{cases} 1 & \text{if } p = q_0', \\ 0 & \text{otherwise,} \end{cases} \quad m_f(p) := \begin{cases} 1 & \text{if } p = q_f', \\ 0 & \text{otherwise.} \end{cases}$

The idea is to have one place for each state of the automaton and each transition is representing an entry of the transition relation of the automaton. Additionally we need an extra start and final place which allow to jump to an initial state or from an final state by an ϵ-transition. Each marking reachable from the initial marking m_0 has in total exactly one token. This token represents the current state of $\mathcal{A}$. Each place of $\mathcal{P}$ has at most one token and is thus bounded. By construction the Petri net accepts the same language as the automaton: $L(\mathcal{A}) = L(\mathcal{P}_\mathcal{A})$.

Remark 2.4 All regular languages are included in the 0-unbounded Petri nets languages: $\mathcal{L}_{\text{Reg}} \subseteq \mathcal{L}_{\text{PN}^0}$. □

Finally let us classify the next more complex class of Petri net languages: 1-unbounded Petri net languages. Consider a labeled Petri net $\mathcal{P} = \big(N = (P, T, E), \Sigma, \lambda, m_0, m_f\big)$ with exactly one unbounded place $p_\infty \in P$, i.e. the remaining places $P_0 := P \setminus \{p_\infty\} \subseteq P$ are bounded. The set $M := \{\, m : P \to \mathbb{N} \mid m_0 \to_N m \,\}$ of markings reachable from the initial marking m_0 is infinite but when restricting these markings to the bounded places we get a finite set: $M_0 := \{\, m|_{P_0} : P_0 \to \mathbb{N} \mid m_0 \to_N m \,\}$. We can construct a non-deterministic counter automaton $\mathcal{C}$ which simulates the bounded places by using its state space M_0 and the single unbounded place using its counter. Additionally $\mathcal{C}$ has to initialize the counter with the value of $m_0(p_\infty)$ and has to check if the counter has value $m_f(p_\infty)$ an the end. The rest is pretty much the same as in the proof of Remark 2.3 plus simulating the operations on the unbounded place by the counter.

Remark 2.5 1-unbounded Petri nets languages are counter languages: $\mathcal{L}_{\text{PN}^1} \subseteq \mathcal{L}_{\text{CL}}$. □

2.3.4 Simple Languages

For the sake of completeness we want to introduce a very simple class of languages: the *simple languages* $\mathcal{L}_{\text{R}}(\Sigma) := \{\, \Sigma_0^* \mid \Sigma_0 \subseteq \Sigma \,\}$, or $\mathcal{L}_{\text{R}}$ for short. They are not really of interest in language theoretical study but will occur in this work every now and then for simple reachability expressions in structures (this is where the R in $\mathcal{L}_{\text{R}}$ comes from). Obviously the simple languages are regular but they are so simple that they are even star-free:

$$\Sigma_0^* = \bigcap_{a \in \Sigma \setminus \Sigma_0} \overline{\Sigma^* \cdot a \cdot \Sigma^*} = \overline{\bigcup_{a \in \Sigma \setminus \Sigma_0} \Sigma^* \cdot a \cdot \Sigma^*} = \overline{\bigcup_{a \in \Sigma \setminus \Sigma_0} \overline{\varnothing} \cdot a \cdot \overline{\varnothing}}.$$

At the end of this subsection we want to give a summary of the language classes and the relations between them which we just introduced:

Remark 2.6 (Summary) Language inclusions:

$$\varnothing \subseteq \mathcal{L}_{\text{R}} \subseteq \mathcal{L}_{\text{SF}} \subseteq \mathcal{L}_{\text{Reg}} = \mathcal{L}_{\text{PN}^0} \subseteq \mathcal{L}_{\text{PN}^1} \subseteq \begin{cases} \mathcal{L}_{\text{CL}} \subseteq \mathcal{L}_{\text{CF}} \\ \mathcal{L}_{\text{PN}^2} \subseteq \mathcal{L}_{\text{PN}} \end{cases}$$

□

2.4 Logic

2.4.1 First-Order

The *first-order* logic or *FO* logic for short allows to specify formulas for some structure that can compare domain elements, check predicates for some elements, use the functions of the structure and quantify over elements. Of course we also build up formulas by Boolean connections.

Syntax

Let τ be a signature and V a countably infinite set of variables. The set $T(\tau)$ of *τ-terms* is defined inductively as: $V \subseteq T(\tau)$ and $f(t_1, \ldots, t_n) \in T(\tau)$ for each $f \in F_n(\tau)$ and $t_1, \ldots, t_n \in T(\tau)$. With $\mathrm{var}(t)$ we denote the set of variables in term $t \in T(\tau)$. The set $\Phi_{\mathrm{FO}}(\tau)$ of *first-order formulas over τ* is defined inductively:

- $(t = t') \in \Phi_{\mathrm{FO}}(\tau)$ for each term $t, t' \in T(\tau)$,
- $P(t_1, \ldots, t_n) \in \Phi_{\mathrm{FO}}(\tau)$ for each relation symbol $P \in R_n(\tau)$ and terms $t_1, \ldots, t_n \in T(\tau)$,
- $\neg\varphi \in \Phi_{\mathrm{FO}}(\tau)$, $(\varphi \vee \psi) \in \Phi_{\mathrm{FO}}(\tau)$ and $(\exists x : \varphi) \in \Phi_{\mathrm{FO}}(\tau)$ for each $\varphi, \psi \in \Phi_{\mathrm{FO}}(\tau)$, $x \in V$.

Furthermore we introduce the abbreviations:

$$\begin{aligned}
\varphi \wedge \psi &:= \neg(\neg\varphi \vee \neg\psi) \\
\varphi \to \psi &:= \neg\varphi \vee \psi \\
\varphi \leftrightarrow \psi &:= (\varphi \to \psi) \wedge (\psi \to \varphi) \\
\forall x : \varphi &:= \neg\exists x : \neg\varphi \\
\exists x_1, \ldots, x_n : \varphi &:= \exists x_1 : \ldots \exists x_n : \varphi \\
\forall x_1, \ldots, x_n : \varphi &:= \forall x_1 : \ldots \forall x_n : \varphi
\end{aligned}$$

The set $\mathrm{free}(\varphi)$ of *free variables* of a first-order formula φ is defined as follows:

$$\begin{aligned}
\mathrm{free}(t = t') &:= \mathrm{var}(t) \cup \mathrm{var}(t') \\
\mathrm{free}\big(P(t_1, \ldots, t_n)\big) &:= \mathrm{var}(t_1) \cup \ldots \cup \mathrm{var}(t_n) \\
\mathrm{free}(\neg\varphi) &:= \mathrm{free}(\varphi) \\
\mathrm{free}(\varphi \vee \psi) &:= \mathrm{free}(\varphi) \cup \mathrm{free}(\psi) \\
\mathrm{free}(\exists x : \varphi) &:= \mathrm{free}(\varphi) \setminus \{x\}
\end{aligned}$$

To denote that formula φ has free variables $\mathrm{free}(\varphi) = \{x_1, \ldots, x_n\}$ we write $\varphi(x_1, \ldots, x_n)$.

Semantics

Let $\mathfrak{A} = \big(A, (f^{\mathfrak{A}})_{f \in F_n(\tau), n \in \mathbb{N}}, (P^{\mathfrak{A}})_{P \in R_n(\tau), n \in \mathbb{N}}\big)$ be a τ-structure. An *interpretation* is a pair $(\mathfrak{A}, \beta)$ with $\beta : V_0 \to A$ for some subset $V_0 \subseteq V$ of variables. Using this we can *interpret* terms by $[\![\cdot]\!]^{(\mathfrak{A},\beta)} : \big\{ t \in T(\tau) \;\big|\; \mathrm{var}(t) \subseteq \mathrm{dom}(\beta) \big\} \to A$ with:

- $[\![v]\!]^{(\mathfrak{A},\beta)} := \beta(v)$ for $v \in \mathrm{dom}(\beta)$,
- $[\![f(t_1,\ldots,t_n)]\!]^{(\mathfrak{A},\beta)} := f^{\mathfrak{A}}\big([\![t_1]\!]^{(\mathfrak{A},\beta)},\ldots,[\![t_n]\!]^{(\mathfrak{A},\beta)}\big)$ for each $f \in F_n(\tau)$,

and *interpret* formulas as well by $[\![\cdot]\!]^{(\mathfrak{A},\beta)} : \big\{\, \varphi \in \Phi_{\mathrm{FO}}(\tau) \;\big|\; \mathrm{free}(\varphi) \subseteq \mathrm{dom}(\beta) \,\big\} \to \mathbb{B}$ with:

- $[\![t = t']\!]^{(\mathfrak{A},\beta)} := \begin{cases} 1 & \text{if } [\![t]\!]^{(\mathfrak{A},\beta)} = [\![t']\!]^{(\mathfrak{A},\beta)}, \\ 0 & \text{otherwise,} \end{cases}$
- $[\![P(t_1,\ldots,t_n)]\!]^{(\mathfrak{A},\beta)} := \begin{cases} 1 & \text{if } \big([\![t_1]\!]^{(\mathfrak{A},\beta)},\ldots,[\![t_n]\!]^{(\mathfrak{A},\beta)}\big) \in P^{\mathfrak{A}}, \\ 0 & \text{otherwise,} \end{cases}$
- $[\![\neg\varphi]\!]^{(\mathfrak{A},\beta)} := 1 - [\![\varphi]\!]^{(\mathfrak{A},\beta)}$,
- $[\![\varphi \vee \psi]\!]^{(\mathfrak{A},\beta)} := \max\big([\![\varphi]\!]^{(\mathfrak{A},\beta)}, [\![\psi]\!]^{(\mathfrak{A},\beta)}\big)$,
- $[\![\exists x : \varphi]\!]^{(\mathfrak{A},\beta)} := \begin{cases} 1 & \text{if } [\![\varphi]\!]^{(\mathfrak{A},\beta[x\mapsto a])} = 1 \text{ for some } a \in A, \\ 0 & \text{otherwise.} \end{cases}$

We say that a formula $\varphi \in \Phi_{\mathrm{FO}}(\tau)$ *holds for* $(\mathfrak{A},\beta)$ or $(\mathfrak{A},\beta) \vDash \varphi$ for short if $[\![\varphi]\!]^{(\mathfrak{A},\beta)} = 1$. In the case that $\varphi(x_1,\ldots,x_n)$ has free variables $x_1,\ldots,x_n$ we write $\mathfrak{A} \vDash \varphi(a_1,\ldots,a_n)$ to abbreviate $(\mathfrak{A},[x_1 \mapsto a_1,\ldots,x_n \mapsto a_n]) \vDash \varphi(x_1,\ldots,x_n)$. The relation *defined* by a formula $\varphi(x_1,\ldots,x_n) \in \Phi_{\mathrm{FO}}(\tau)$ with n free variables is:

$$[\![\varphi(x_1,\ldots,x_n)]\!]^{\mathfrak{A}} \;:=\; \Big\{\, (a_1,\ldots,a_n) \in A^n \;\Big|\; \mathfrak{A} \vDash \varphi(a_1,\ldots,a_n) \,\Big\} \;\subseteq\; A^n.$$

The *FO-theory* $\mathrm{Th}_{\mathrm{FO}}(\mathfrak{A})$ of $\mathfrak{A}$ is the set of first-order formulas with no free variables that hold in $\mathfrak{A}$: $\mathrm{Th}_{\mathrm{FO}}(\mathfrak{A}) := \big\{\, \varphi \in \Phi_{\mathrm{FO}}(\tau) \;\big|\; \mathrm{free}(\varphi) = \varnothing,\ \mathfrak{A} \vDash \varphi \,\big\}$.

2.4.2 First-Order with Reachability

We will introduce an extension of the first-order logic which is dealing with one of the weaknesses of FO: given some graph-like structure it cannot be expressed by a plain first-order formula that some element is reachable from another. We will fix this here.

Syntax

Let τ be a signature just containing relation symbols of arity 1 and 2 (i.e. the signature of a graph structure with node predicates and possibly multiple edge relations), V a countably infinite set of variables and $\mathcal{L} \subseteq \mathcal{P}(\Sigma^*)$ some language class on the alphabet $\Sigma := R_1(\tau) \cup R_2(\tau)$. The set $\Phi_{\mathrm{FO}(\mathcal{L})}(\tau)$ of *first-order formulas with $\mathcal{L}$-reachability* is defined analogously to the first-order formulas but extended by a new atomic formula called a *reachability expression*: $\mathsf{reach}_L(x,y) \in \Phi_{\mathrm{FO}(\mathcal{L})}(\tau)$ for each $x,y \in V$, $L \in \mathcal{L}$. Note that we use just variables x,y instead of arbitrary terms. This is because there are no functions in the structure and hence it does not make any difference here (i.e.

$T(\tau) = V$). Other than the abbreviations seen so far we introduce an additional one if $\Sigma^* \in \mathcal{L}$: $\mathsf{reach}(x,y) := \mathsf{reach}_{\Sigma^*}(x,y)$. The set $\mathrm{free}(\varphi)$ of free variables of a reachability expression is $\mathrm{free}\big(\mathsf{reach}_L(x,y)\big) := \{x,y\}$.

For the language classes defined in Section 2.3 we will write FO(Reg), FO(CF), etc. instead of $\mathrm{FO}(\mathcal{L}_{\mathrm{Reg}})$, $\mathrm{FO}(\mathcal{L}_{\mathrm{CF}})$, etc. respectively.

Semantics

Let $\mathfrak{A} = \big(A, (P^{\mathfrak{A}})_{P \in R_1(\tau) \cup R_2(\tau)}\big)$ be a graph structure with a signature τ as above. We just give the extension of the definition of first-order interpretation to deal with the new reachability expressions since the rest remains the same. We set $[\![\mathsf{reach}_L(x,y)]\!]^{(\mathfrak{A},\beta)} := 1$ if and only if there exists a sequence $P_1 \cdot \ldots \cdot P_n \in L$ of relation symbols (i.e. $P_i \in R_1(\tau) \cup R_2(\tau)$) such that $[\![\psi_{P_1 \cdot \ldots \cdot P_n}(x,y)]\!]^{(\mathfrak{A},\beta)} = 1$ where

$$\psi_{P_1 \cdot \ldots \cdot P_n}(x,y) \;:=\; \exists z_0, \ldots, z_n : (x = z_0) \wedge (y = z_n) \wedge \bigwedge_{i \in \{1,\ldots,n\}} \psi'_{P_i}(z_{i-1}, z_i)$$

$$\text{and} \quad \psi'_P(z,z') \;:=\; \begin{cases} P(z) \wedge z = z' & \text{if } P \in R_1(\tau), \\ P(z,z') & \text{if } P \in R_2(\tau). \end{cases}$$

Otherwise we set $[\![\mathsf{reach}_L(x,y)]\!]^{(\mathfrak{A},\beta)} := 0$. Informally spoken this means that there is a path from x to y (actually from $\beta(x)$ to $\beta(y)$) such that the sequence of labels on that path is in L where one can identify the unary predicates with a binary edge relation consisting of just loops. Analogously we extend the definition of the first-order theory: the *FO($\mathcal{L}$)-theory* $\mathrm{Th}_{\mathrm{FO}(\mathcal{L})}(\mathfrak{A}) := \big\{ \varphi \in \Phi_{\mathrm{FO}(\mathcal{L})}(\tau) \;\big|\; \mathrm{free}(\varphi) = \varnothing,\ \mathfrak{A} \vDash \varphi \big\}$ of $\mathfrak{A}$.

Inclusion

Since FO with reachability is an extension of plain FO we created kind of a hierarchy. For two logics $\mathfrak{L}, \mathfrak{L}'$ we say that $\mathfrak{L}$ is a sublogic of $\mathfrak{L}'$ or $\mathfrak{L} \leq \mathfrak{L}'$ for short if for each τ-structure $\mathfrak{A}$: $\Phi_{\mathfrak{L}}(\tau) \subseteq \Phi_{\mathfrak{L}'}(\tau)$ and $\mathrm{Th}_{\mathfrak{L}}(\mathfrak{A}) = \mathrm{Th}_{\mathfrak{L}'}(\mathfrak{A}) \cap \Phi_{\mathfrak{L}}(\tau)$.

Consider two language classes $\mathcal{L} \subseteq \mathcal{L}' \subseteq \mathcal{P}(\Sigma^*)$ over some alphabet Σ. For each $\mathsf{reach}_L(x,y) \in \Phi_{\mathrm{FO}(\mathcal{L})}(\tau)$ we obviously have $\mathsf{reach}_L(x,y) \in \Phi_{\mathrm{FO}(\mathcal{L}')}(\tau)$ as well because of $L \in \mathcal{L} \subseteq \mathcal{L}'$. Since $\Phi_{\mathrm{FO}(\mathcal{L}')}(\tau)$ differs from $\Phi_{\mathrm{FO}(\mathcal{L})}(\tau)$ just by allowing more complex languages in its reachability expressions we can conclude that $\Phi_{\mathrm{FO}(\mathcal{L})}(\tau) \subseteq \Phi_{\mathrm{FO}(\mathcal{L}')}(\tau)$ and even more important:

Remark 2.7 Let $\mathcal{L} \subseteq \mathcal{L}'$ be two language classes. Then $\mathrm{FO}(\mathcal{L}) \leq \mathrm{FO}(\mathcal{L}')$. □

We can apply this result to Remark 2.6 yielding an ascending order of FO-theories with reachability for the languages classes considered there. But more on this later. Note that $\mathrm{FO}(\varnothing)$ can be identified with plain FO:

Remark 2.8 Let $\mathcal{L}$ be a language class. Then $\mathrm{FO} = \mathrm{FO}(\varnothing)$, i.e. $\mathrm{FO} \leq \mathrm{FO}(\varnothing) \leq \mathrm{FO}$. □

2.4.3 Monadic Second-Order

The *monadic second-order* logic or *MSO* for short is another extension of first-order logic. In MSO it is possible to quantify not just over single elements but over sets of elements instead. This is where the name comes from: predicates are of second-order type and monadic means a restriction to unary predicates. MSO logic is much more powerful than FO while the theory still is decidable for some important structures.

Syntax

Let τ be a signature and V, V' countably infinite sets of variables. Next to the old set $V = \{x, y, z, \ldots\}$ of first-order variables denoted by small letters we have another disjoint set $V' = \{X, Y, Z, \ldots\}$ of monadic second-order variables denoted by capital letters. The set $\Phi_{\mathrm{MSO}}(\tau)$ of *monadic second-order formulas* is defined inductively (the first two types of formulas are the same as for first-order formulas while the last two types are new):

- $(t_1 = t_2), P(t_1, \ldots, t_n) \in \Phi_{\mathrm{MSO}}(\tau)$ where $t_1, t_2, \ldots, t_n \in T(\tau)$ and $P \in R_n(\tau)$,
- $(\neg\varphi), (\varphi \vee \psi), (\exists x : \varphi) \in \Phi_{\mathrm{MSO}}(\tau)$ for each $\varphi, \psi \in \Phi_{\mathrm{MSO}}(\tau)$, $x \in V$.
- $(\exists X : \varphi) \in \Phi_{\mathrm{MSO}}(\tau)$ for each $\varphi \in \Phi_{\mathrm{MSO}}(\tau)$, $X \in V'$.
- $X(t) \in \Phi_{\mathrm{MSO}}(\tau)$ for each term $t \in T(\tau)$ and $X \in V'$,

We extend some of the old abbreviations to monadic second-order quantification:

$$\begin{aligned} \forall X : \varphi &:= \neg\exists X : \neg\varphi \\ \exists X_1, \ldots, X_n : \varphi &:= \exists X_1 : \ldots \exists X_n : \varphi \\ \forall X_1, \ldots, X_n : \varphi &:= \forall X_1 : \ldots \forall X_n : \varphi \end{aligned}$$

Next to the set $\mathrm{free}(\varphi) \subseteq V$ of free first-order variables of a MSO-formula φ we also need the set $\mathrm{free}'(\varphi) \subseteq V'$ of free monadic second-order variables which is defined as follows:

$\varphi = (t = t') :$	$\mathrm{free}(\varphi) := \mathrm{var}(t) \cup \mathrm{var}(t')$	$\mathrm{free}'(\varphi) := \varnothing$
$\varphi = P(t_1, \ldots, t_n) :$	$\mathrm{free}(\varphi) := \mathrm{var}(t_1) \cup \ldots \cup \mathrm{var}(t_n)$	$\mathrm{free}'(\varphi) := \varnothing$
$\varphi = \neg\psi :$	$\mathrm{free}(\varphi) := \mathrm{free}(\psi)$	$\mathrm{free}'(\varphi) := \mathrm{free}'(\psi)$
$\varphi = \psi \wedge \vartheta :$	$\mathrm{free}(\varphi) := \mathrm{free}(\psi) \cup \mathrm{free}(\vartheta)$	$\mathrm{free}'(\varphi) := \mathrm{free}'(\psi) \cup \mathrm{free}'(\vartheta)$
$\varphi = \exists x : \psi :$	$\mathrm{free}(\varphi) := \mathrm{free}(\psi) \setminus \{x\}$	$\mathrm{free}'(\varphi) := \mathrm{free}'(\psi)$
$\varphi = \exists X : \psi :$	$\mathrm{free}(\varphi) := \mathrm{free}(\psi)$	$\mathrm{free}'(\varphi) := \mathrm{free}'(\psi) \setminus \{X\}$
$\varphi = X(t) :$	$\mathrm{free}(\varphi) := \mathrm{var}(t)$	$\mathrm{free}'(\varphi) := \{X\}$

To denote that an MSO-formula φ has free FO-variables $\mathrm{free}(\varphi) = \{x_1, \ldots, x_n\}$ and free MSO-variables $\mathrm{free}'(\varphi) = \{X_1, \ldots, X_m\}$ we write $\varphi(x_1, \ldots, x_n, X_1, \ldots, X_m)$.

Semantics

Let $\mathfrak{A} = \left(A, (f^{\mathfrak{A}})_{f \in F_n(\tau), n \in \mathbb{N}}, (P^{\mathfrak{A}})_{P \in R_n(\tau), n \in \mathbb{N}}\right)$ be a τ-structure. An *interpretation* is a pair $(\mathfrak{A}, \beta, \beta')$ with $\beta : V_0 \to A$, $\beta' : V_0' \to \mathcal{P}(A)$ for some subsets $V_0 \subseteq V$, $V_0' \subseteq V'$ of first-order and second-order variables respectively. Using this we can *interpret* formulas by

$$[\![\cdot]\!]^{(\mathfrak{A},\beta,\beta')} : \left\{ \varphi \in \Phi_{\mathrm{MSO}}(\tau) \;\middle|\; \mathrm{free}(\varphi) \subseteq \mathrm{dom}(\beta),\ \mathrm{free}'(\varphi) \subseteq \mathrm{dom}(\beta') \right\} \to \mathbb{B}$$

with:

- $[\![t = t']\!]^{(\mathfrak{A},\beta,\beta')} := \begin{cases} 1 & \text{if } [\![t]\!]^{(\mathfrak{A},\beta)} = [\![t']\!]^{(\mathfrak{A},\beta)}, \\ 0 & \text{otherwise,} \end{cases}$
- $[\![P(t_1, \ldots, t_n)]\!]^{(\mathfrak{A},\beta,\beta')} := \begin{cases} 1 & \text{if } \left([\![t_1]\!]^{(\mathfrak{A},\beta)}, \ldots, [\![t_n]\!]^{(\mathfrak{A},\beta)}\right) \in P^{\mathfrak{A}}, \\ 0 & \text{otherwise,} \end{cases}$
- $[\![\neg\varphi]\!]^{(\mathfrak{A},\beta,\beta')} := 1 - [\![\varphi]\!]^{(\mathfrak{A},\beta,\beta')}$,
- $[\![\varphi \vee \psi]\!]^{(\mathfrak{A},\beta,\beta')} := \max\left([\![\varphi]\!]^{(\mathfrak{A},\beta,\beta')}, [\![\psi]\!]^{(\mathfrak{A},\beta,\beta')}\right)$,
- $[\![\exists x : \varphi]\!]^{(\mathfrak{A},\beta,\beta')} := \begin{cases} 1 & \text{if } [\![\varphi]\!]^{(\mathfrak{A},\beta[x \mapsto a],\beta')} = 1 \text{ for some } a \in A, \\ 0 & \text{otherwise,} \end{cases}$
- $[\![\exists X : \varphi]\!]^{(\mathfrak{A},\beta,\beta')} := \begin{cases} 1 & \text{if } [\![\varphi]\!]^{(\mathfrak{A},\beta,\beta'[X \mapsto A'])} = 1 \text{ for some } A' \subseteq A, \\ 0 & \text{otherwise,} \end{cases}$
- $[\![X(t)]\!]^{(\mathfrak{A},\beta,\beta')} := \begin{cases} 1 & \text{if } [\![t]\!]^{(\mathfrak{A},\beta)} \in \beta'(X), \\ 0 & \text{otherwise.} \end{cases}$

We say that a formula $\varphi \in \Phi_{\mathrm{MSO}}(\tau)$ *holds for* $(\mathfrak{A}, \beta, \beta')$ or $(\mathfrak{A}, \beta, \beta') \vDash \varphi$ for short if $[\![\varphi]\!]^{(\mathfrak{A},\beta,\beta')} = 1$. If $\varphi(x_1, \ldots, x_n, X_1, \ldots, X_m)$ has free variables $x_1, \ldots, x_n, X_1, \ldots, X_m$ we write $\mathfrak{A} \vDash \varphi(a_1, \ldots, a_n, A_1, \ldots, A_m)$ to abbreviate $(\mathfrak{A}, [x_1 \mapsto a_1, \ldots, x_n \mapsto a_n], [X_1 \mapsto A_1, \ldots, X_m \mapsto A_m]) \vDash \varphi(x_1, \ldots, x_n, X_1, \ldots, X_m)$. This can be done analogously if the order of FO and MSO-variables is arbitrary. The relation *defined* by a formula $\varphi(x_1, \ldots, x_n) \in \Phi_{\mathrm{MSO}}(\tau)$ with free variables of just first-order is defined as in the first-order case:

$$[\![\varphi(x_1, \ldots, x_n)]\!]^{\mathfrak{A}} := \left\{ (a_1, \ldots, a_n) \in A^n \;\middle|\; \mathfrak{A} \vDash \varphi(a_1, \ldots, a_n) \right\} \subseteq A^n.$$

The *MSO-theory* $\mathrm{Th}_{\mathrm{MSO}}(\mathfrak{A})$ of $\mathfrak{A}$ is the set of monadic second-order formulas with no free variables that hold in $\mathfrak{A}$: $\mathrm{Th}_{\mathrm{MSO}}(\mathfrak{A}) := \left\{ \varphi \in \Phi_{\mathrm{MSO}}(\tau) \;\middle|\; \mathrm{free}(\varphi) = \mathrm{free}'(\varphi) = \varnothing,\ \mathfrak{A} \vDash \varphi \right\}$.

Reachability

As already mentioned MSO is strictly more powerful than FO. We will use this additional expressive power to simulate reachability expressions from the previous subsection for some language classes.

The most simple ones are the simple languages $\mathcal{L}_{\mathrm{R}}$ from Subsection 2.3.4. This kind of reachability can be expressed by an MSO-formula. This is due to the fact that the positions reachable by the Kleene closure of some subalphabet can be described as the transitive closure of a binary relation which represents that subalphabet and can be defined by some formula. When considering the more powerful language class of the regular languages from Subsection 2.3.1 which is represented by regular expressions, then the Kleene closure we have just considered is the most difficult of the operators of regular expressions. Indeed we can inductively define MSO-formulas simulating regular reachability for some arbitrary graph structure $\mathfrak{A} = \left(A, (P^{\mathfrak{A}})_{P \in R_1(\tau) \cup R_2(\tau)}\right)$ of signature τ by:

$$
\begin{aligned}
\mathsf{reach}_{\varnothing}(x,y) &:= \mathsf{false} \\
\mathsf{reach}_{\epsilon}(x,y) &:= x = y \\
\mathsf{reach}_{P}(x,y) &:= \begin{cases} P(x) \wedge (x = y) & \text{if } P \in R_1(\tau) \\ P(x,y) & \text{if } P \in R_2(\tau) \end{cases} \\
\mathsf{reach}_{r_1 + r_2}(x,y) &:= \mathsf{reach}_{r_1}(x,y) \vee \mathsf{reach}_{r_2}(x,y) \\
\mathsf{reach}_{r_1 \cdot r_2}(x,y) &:= \exists z : \Big(\mathsf{reach}_{r_1}(x,z) \wedge \mathsf{reach}_{r_2}(z,y)\Big) \\
\mathsf{reach}_{r^*}(x,y) &:= \forall Z : \Big(Z(x) \wedge \Big(\forall z, z' : \big(Z(z) \wedge \mathsf{reach}_{r}(z,z')\big) \to Z(z')\Big) \to Z(y)\Big)
\end{aligned}
$$

Since such regular reachability expressions can be expressed in plain MSO we will allow reach-expressions with regular languages as abbreviation in MSO-formulas. Thus regular reachability is already a feature of MSO.

Remark 2.9 $\mathrm{FO}(\mathrm{Reg}) \leq \mathrm{MSO}$. □

When demanding a small precondition we can get an even more advanced result.

Proposition 2.10 ([LO2007, Proposition 24]) *Let τ be a signature consisting of relation symbols of arity 1 and 2. For every regular language L over the alphabet $R_1(\tau) \cup R_2(\tau)$ the* extended reachability expression $\mathsf{Reach}_L(x,y,Z)$ *is effectively MSO-definable such that for each τ-structure $\mathfrak{A} = \left(A, (P^{\mathfrak{A}})_{P \in R_1(\tau) \cup R_2(\tau)}\right)$, elements $a, a' \in A$ and finite subset $A_0 \subseteq A$ we have $\mathfrak{A} \vDash \mathsf{Reach}_L(a, a', A_0)$ iff a reaches a' by a path with a label sequence in L that visits exactly the elements of A_0.* □

This extended reachability expression Reach is stronger than the expression reach used for reachability in first-order logics.

Finally we want to summarize the results of Remark 2.7, 2.8 and 2.9 with respect to the language hierarchy of Remark 2.6:

Remark 2.11 (Summary) Logic hierarchy:

$$\mathrm{FO} = \mathrm{FO}(\varnothing) \leq \mathrm{FO}(\mathrm{R}) \leq \mathrm{FO}(\mathrm{SF}) \leq \mathrm{FO}(\mathrm{Reg}) \leq \begin{cases} \mathrm{FO}(\mathrm{PN}^1) \leq \begin{cases} \mathrm{FO}(\mathrm{CL}) \leq \mathrm{FO}(\mathrm{CF}) \\ \mathrm{FO}(\mathrm{PN}^2) \leq \mathrm{FO}(\mathrm{PN}) \end{cases} \\ \mathrm{MSO} \end{cases}$$

□

2.4.4 Structure Transformation

MSO-Interpretation

Interpretations are a standard technique in logic to define a structure by formulas over another structure. Using this it is very easy to transfer decidability results from one structure to the other. We will start with a very simple but powerful interpretation completely based on MSO-formulas. Later on we also introduce variants for FO with reachability.

Let $\mathfrak{A} = \left(A, (P^{\mathfrak{A}})_{P \in R_n(\tau),\, n \in \mathbb{N}}\right)$ be a structure with a signature τ of just relation symbols and $\mathfrak{B} = \left(B, (f^{\mathfrak{B}})_{f \in F_n(\pi),\, n \in \mathbb{N}}, (P^{\mathfrak{B}})_{P \in R_n(\pi),\, n \in \mathbb{N}}\right)$ be a π-structure. We say that $\mathfrak{A}$ is *MSO-interpretable* in $\mathfrak{B}$ if $\mathfrak{A}$ is isomorphic to the τ-structure

$$\left(\llbracket \psi_{\mathrm{dom}}(x) \rrbracket^{\mathfrak{B}}, \left(\llbracket \psi_P(x_1, \ldots, x_n) \rrbracket^{\mathfrak{B}}\right)_{P \in R_n(\tau),\, n \in \mathbb{N}}\right)$$

for some MSO-formulas $\psi_{\mathrm{dom}}(x) \in \Phi_{\mathrm{MSO}}(\pi)$ and $\psi_P(x_1, \ldots, x_n) \in \Phi_{\mathrm{MSO}}(\pi)$ for each relation symbol $P \in R_n(\tau)$. We call the tuple $\left(\psi_{\mathrm{dom}}(x), \left(\psi_P(x_1, \ldots, x_n)\right)_{P \in R_n(\tau),\, n \in \mathbb{N}}\right)$ an *MSO-interpretation*.

Remark 2.12 If a structure $\mathfrak{A}$ is MSO-interpretable in a structure $\mathfrak{B}$ with decidable MSO-theory then the one of $\mathfrak{A}$ is decidable as well. □

The idea of the proof of the above remark is to take a given formula $\varphi \in \Phi_{\mathrm{MSO}}(\tau)$ for the τ-structure $\mathfrak{A}$ and to transform it to a formula $\hat{\varphi} \in \Phi_{\mathrm{MSO}}(\pi)$ for the π-structure $\mathfrak{B}$ such that $\mathfrak{A} \models \varphi$ iff $\mathfrak{B} \models \hat{\varphi}$. We can use the formulas of a suitable MSO-interpretation $\left(\psi_{\mathrm{dom}}(x), \left(\psi_P(x_1, \ldots, x_n)\right)_{P \in R_n(\tau),\, n \in \mathbb{N}}\right)$ to inductively define the transformation $\hat{\cdot} : \Phi_{\mathrm{MSO}}(\tau) \to \Phi_{\mathrm{MSO}}(\pi)$ fairly easy since the formulas tell us how to define the domain and predicates of $\mathfrak{A}$ in $\mathfrak{B}$:

$$\begin{aligned}
\widehat{x = y} &:= x = y \\
\widehat{P(x_1, \ldots, x_n)} &:= \psi_P(x_1, \ldots, x_n) \\
\widehat{\neg\varphi} &:= \neg\hat{\varphi} \\
\widehat{\varphi \vee \psi} &:= \hat{\varphi} \vee \hat{\psi} \\
\widehat{\exists x : \varphi} &:= \exists x : \left(\hat{\varphi} \wedge \psi_{\mathrm{dom}}(x)\right) \\
\widehat{\exists X : \varphi} &:= \exists X : \left(\hat{\varphi} \wedge \forall x : \left(X(x) \to \psi_{\mathrm{dom}}(x)\right)\right) \\
\widehat{X(x)} &:= X(x)
\end{aligned}$$

Unfolding

Another type of transformation that preserves the decidability of the MSO-theory is unfolding. Let $\mathfrak{A} = \left(A, (P^{\mathfrak{A}})_{P \in R_1(\tau) \cup R_2(\tau)}\right)$ be a graph structure of signature τ. The *unfolding* $\mathsf{Unf}_{a_0}(\mathfrak{A})$ of $\mathfrak{A}$ from a given start element $a_0 \in A$ is a tree structure $\mathsf{Unf}_{a_0}(\mathfrak{A}) := \mathfrak{B} = \left(B, (P^{\mathfrak{B}})_{P \in R_1(\tau) \cup R_2(\tau)}\right)$ of the same signature τ with:

- $B := \left\{ a_0 P_1 a_1 \dots P_n a_n \;\middle|\; \text{for all } i \in \{1, \dots, n\} :\ a_i \in A,\ P_i \in R_2(\tau),\ (a_{i-1}, a_i) \in P_i^{\mathfrak{A}} \right\}$,
- for $P \in R_1(\tau)$: $P^{\mathfrak{B}} := \left\{ a_0 P_1 a_1 \dots P_n a_n \;\middle|\; a_n \in P^{\mathfrak{A}} \right\}$,
- for $P \in R_2(\tau)$: $P^{\mathfrak{B}} := \left\{ (a_0 P_1 a_1 \dots P_n a_n,\ a_0 P_1 a_1 \dots P_n a_n P a) \;\middle|\; (a_n, a) \in P^{\mathfrak{A}} \right\}$.

The unfolding is always a tree.

Proposition 2.13 ([CW1998]) *The unfolding $\mathsf{Unf}_{a_0}(\mathfrak{A})$ of a graph structure $\mathfrak{A}$ with decidable MSO-theory from any MSO-definable start element a_0 has again a decidable MSO-theory.* □

Caucal's Hierarchy

With MSO-interpretation and unfolding we just have seen two different approaches of transforming structures such that the decidability of the MSO-theory is preserved. Caucal used both of these transformations to build up an infinite hierarchy [Cau2002][1] of graph structures. The *Caucal hierarchy* is the class $\bigcup_{n \in \mathbb{N}} (\mathcal{T}_n \cup \mathcal{G}_n)$ of graph structures defined inductively as follows:

- $\mathcal{T}_0 :=$ the class of finite trees,
- $\mathcal{G}_n :=$ the class of graph structures MSO-interpretable in a tree of $\mathcal{T}_n$,
- $\mathcal{T}_{n+1} :=$ the class of unfoldings of trees in $\mathcal{T}_n$.

Proposition 2.14 *Each graph structure of the Caucal hierarchy has a decidable MSO-theory.* □

Isomorphic-Definability

To interpret a unary relation we use the relation $[\![\varphi(x)]\!]^{\mathfrak{A}} \subseteq A$ defined by a formula $\varphi(x)$. Unfortunately in some cases it is not possible to define a unary relation this way. In that case it would be nice to at least define such a relation up to isomorphism. If we have a formula $\varphi(X)$ with one free MSO-variable it suffices to take any unary relation $A_0 \subseteq A$ with $\mathfrak{A} \models \varphi(A_0)$. It still remains to fix one such relation.

Consider a π-structure $\mathfrak{B} = \left(B, (f^{\mathfrak{B}})_{f \in F_n(\pi), n \in \mathbb{N}}, (P^{\mathfrak{B}})_{P \in R_n(\pi), n \in \mathbb{N}}\right)$ and structure $\mathfrak{A} = \left(A, (f^{\mathfrak{A}})_{f \in F_n(\pi), n \in \mathbb{N}}, (P^{\mathfrak{A}})_{P \in R_n(\pi), n \in \mathbb{N}}, C^{\mathfrak{A}}\right)$ over signature τ which is π plus a new unary

[1] In fact Caucal used a different definition for this hierarchy. Anyhow the structures stay at the same level even when using MSO-interpretations instead of the weaker inverse rational mappings.

predicate symbol C. We say that the predicate $C^{\mathfrak{A}}$ of $\mathfrak{A}$ is *MSO-isomorphic-definable* in $\mathfrak{B}$ if there is an MSO-formula $\psi_C(X) \in \Phi_{\mathrm{MSO}}(\pi)$ such that each τ-structure $\big(B, (f^{\mathfrak{B}})_{f \in F_n(\pi),\, n \in \mathbb{N}}, (P^{\mathfrak{B}})_{P \in R_n(\pi),\, n \in \mathbb{N}}, \tilde{C}\big)$ with $\mathfrak{B} \models \psi_C(\tilde{C})$ is isomorphic to $\mathfrak{A}$.

Theorem 2.15 *If a unary predicate C of a structure $\mathfrak{A}$ is MSO-definable up to isomorphism in a structure $\mathfrak{B}$ with decidable MSO-theory then the one of $\mathfrak{A}$ is decidable as well.* □

Proof Let $\psi_C(X) \in \Phi_{\mathrm{MSO}}(\pi)$ be a formula that defines the predicate C of the τ-structure $\mathfrak{A}$ in π-structure $\mathfrak{B}$. Again we transform a given formula $\varphi \in \Phi_{\mathrm{MSO}}(\tau)$ for $\mathfrak{A}$ into a formula $\varphi' := \exists X : \big(\psi_C(X) \wedge \varphi\big) \in \Phi_{\mathrm{MSO}}(\pi)$ for $\mathfrak{B}$ and get $\mathfrak{A} \models \varphi$ iff $\mathfrak{B} \models \varphi'$.

The formula φ is actually over the signature τ which is π extended by a unary predicate symbol C. The trick is that φ can be viewed as a formula over signature π possibly having $C \in \mathrm{free}'(\varphi)$. We can existentially quantify this second-order variable and force it to meet the specification of $\psi_C(X)$ which yields the MSO-formula φ' over the signature π of $\mathfrak{B}$. For the correctness of this construction it does not matter which relation is actually used for C as long as it satisfies the specification $\psi_C(X)$ since by definition each resulting structure is isomorphic to $\mathfrak{A}$. ■

This simple but nice idea to interpret the MSO-theory of a structure in the MSO-theory of another gives us the motivation to try similar approaches. Often we need to represent the elements of the new structure not just by one single element of the old structure but by a tuple of old elements. By doing so we will lose the MSO-features in the new structure since tuples can be handled in MSO but sets of tuples cannot. We accept this drawback and go on without sets which still allows us to transfer the decidability of the theory of the old structure to the FO-theory of the new structure:

Tuple-Interpretation

Let $\mathfrak{A} = \big(A, (P^{\mathfrak{A}})_{P \in R_n(\tau),\, n \in \mathbb{N}}\big)$ be a structure with signature τ of just relation symbols, $\mathfrak{B} = \big(B, (f^{\mathfrak{B}})_{f \in F_n(\pi),\, n \in \mathbb{N}}, (P^{\mathfrak{B}})_{P \in R_n(\pi),\, n \in \mathbb{N}}\big)$ be a π-structure again and let the logic $\mathfrak{L}$ either be MSO or FO($\mathcal{L}$) for some language class $\mathcal{L}$. We say that $\mathfrak{A}$ is *m-tuple-$\mathfrak{L}$-interpretable* in $\mathfrak{B}$ if $\mathfrak{A}$ is isomorphic to

$$\Big(\big[\![\psi_{\mathrm{dom}}(\bar{x})\big]\!]^{\mathfrak{B}}, \big(\big[\![\psi_P(\bar{x}_1, \ldots, \bar{x}_n)\big]\!]^{\mathfrak{B}}\big)_{P \in R_n(\tau),\, n \in \mathbb{N}}\Big)$$

for $\mathfrak{L}$-formulas over signature π with m-ary tuples of variables (i.e. $\bar{x}, \bar{x}_1, \ldots \in V^m$):

- $\psi_{\mathrm{dom}}(\bar{x}) \in \Phi_{\mathfrak{L}}(\pi)$ and
- $\psi_P(\bar{x}_1, \ldots, \bar{x}_n) \in \Phi_{\mathfrak{L}}(\pi)$ for each relation symbol $P \in R_n(\tau)$,

where the entries of the relation $\big[\![\psi_P(\bar{x}_1, \ldots, \bar{x}_n)\big]\!]^{\mathfrak{B}} \subseteq B^{n \cdot m}$, which are formally $(n \cdot m)$-tuples, have to be seen as n-tuples of m-tuples: $\big[\![\psi_P(\bar{x}_1, \ldots, \bar{x}_n)\big]\!]^{\mathfrak{B}} \subseteq (B^m)^n$. We call $\Big(\psi_{\mathrm{dom}}(\bar{x}), \big(\psi_P(\bar{x}_1, \ldots, \bar{x}_n)\big)_{P \in R_n(\tau),\, n \in \mathbb{N}}\Big)$ an *m-tuple-$\mathfrak{L}$-interpretation.*

Theorem 2.16 *Let logic $\mathfrak{L}$ either be MSO or FO($\mathcal{L}$) for some language class $\mathcal{L}$. If a structure $\mathfrak{A}$ is m-tuple-$\mathfrak{L}$-interpretable in a structure $\mathfrak{B}$ that has a decidable theory in that $\mathfrak{L}$ then the FO-theory of $\mathfrak{A}$ is decidable as well.* □

PROOF Actually this can be proven analogously to Remark 2.12. Given a suitable m-tuple-$\mathfrak{L}$-interpretation $\Big(\llbracket\psi_{\mathrm{dom}}(\bar{x})\rrbracket^{\mathfrak{B}}, \big(\llbracket\psi_P(\bar{x}_1,\ldots,\bar{x}_n)\rrbracket^{\mathfrak{B}}\big)_{P\in R_n(\tau),\, n\in\mathbb{N}}\Big)$ with m-tuples of variables we define a transformation $\hat{\cdot}: \Phi_{\mathrm{FO}}(\tau) \to \Phi_{\mathfrak{L}}(\pi)$ of formulas as follows:

$$\begin{aligned}
\widehat{x = y} &:= x_1 = y_1 \wedge \ldots \wedge x_m = y_m \\
\widehat{P(x_1,\ldots,x_n)} &:= \psi_P(\ x_{1,1},\ldots,x_{1,m}\ ,\ \ldots\ ,\ x_{n,1},\ldots,x_{n,m}\ ) \\
\widehat{\neg\varphi} &:= \neg\hat{\varphi} \\
\widehat{\varphi \vee \psi} &:= \hat{\varphi} \vee \hat{\psi} \\
\widehat{\exists x : \varphi} &:= \exists x_1,\ldots,x_m : \big(\hat{\varphi} \wedge \psi_{\mathrm{dom}}(x_1,\ldots,x_m)\big)
\end{aligned}$$

By construction we get again $\mathfrak{A} \models \varphi$ iff $\mathfrak{B} \models \hat{\varphi}$. Since the transformation is effective it also transfers the decidability of the $\mathfrak{L}$-theory of $\mathfrak{B}$ to the FO-theory of $\mathfrak{A}$. ∎

Tuple-Interpretation with Reachability

Since this work is focused on FO with reachability we have to extend tuple-interpretation to handle reachability expressions as well. Let τ be a signature containing just relation symbols of arity 1 or 2, $\mathfrak{B} = \big(B, (f^{\mathfrak{B}})_{f\in F_n(\pi),\, n\in\mathbb{N}}, (P^{\mathfrak{B}})_{P\in R_n(\pi),\, n\in\mathbb{N}}\big)$ be a π-structure and $\Big(\psi_{\mathrm{dom}}(\bar{x}), \big(\psi_P(\bar{x}_1,\ldots,\bar{x}_n)\big)_{P\in R_n(\tau),\, n\in\mathbb{N}}\Big)$ a m-tuple-$\mathfrak{L}$-interpretation. If for some language class $\mathcal{L}$ over alphabet $\Sigma := R_1(\tau) \cup R_2(\tau)$ there exist formulas $\psi^L_{\mathsf{reach}}(x_1,\ldots,x_m,y_1,\ldots,y_m) \in \Phi_{\mathfrak{L}}(\pi)$ such that

$$\begin{aligned}
\Big(\llbracket\psi_{\mathrm{dom}}(\bar{x})\rrbracket^{\mathfrak{B}}, \big(\llbracket\psi_P(\bar{x}_1,\ldots,\bar{x}_n)\rrbracket^{\mathfrak{B}}\big)_{P\in R_n(\tau),\, n\in\mathbb{N}}\Big) &\models \mathsf{reach}_L\big((b_1,\ldots,b_m),(b'_1,\ldots,b'_m)\big) \\
\text{iff}\quad \mathfrak{B} &\models \psi^L_{\mathsf{reach}}(b_1,\ldots,b_m,b'_1,\ldots,b'_m)
\end{aligned}$$

for each $L \in \mathcal{L}$ and $(b_1,\ldots,b_m),(b'_1,\ldots,b'_m) \in \llbracket\psi_{\mathrm{dom}}(\bar{x})\rrbracket^{\mathfrak{B}}$ then we call the extended tuple $\Big(\psi_{\mathrm{dom}}(\bar{x}), \big(\psi_P(\bar{x}_1,\ldots,\bar{x}_n)\big)_{P\in R_n(\tau),\, n\in\mathbb{N}}, \big(\psi^L_{\mathsf{reach}}(\bar{x},\bar{y})\big)_{L\in\mathcal{L}}\Big)$ an *m-tuple-$\mathfrak{L}$-interpretation with $\mathcal{L}$-reachability*. Any τ-structure $\mathfrak{A}$ that is m-tuple-$\mathfrak{L}$-interpretable by this interpretation is said to be *m-tuple-$\mathfrak{L}$-interpretable with $\mathcal{L}$-reachability*.

Theorem 2.17 *Let $\mathcal{L}$ be some language class such that each language $L \in \mathcal{L}$ can be finitely represented according to some model formalism (e.g. by expressions, automata, etc.) and let logic $\mathfrak{L}$ either be MSO or FO($\mathcal{L}'$) for some language class $\mathcal{L}'$. If a structure $\mathfrak{A}$ is m-tuple-$\mathfrak{L}$-interpretable with $\mathcal{L}$-reachability in a structure $\mathfrak{B}$ with decidable $\mathfrak{L}$-theory by an interpretation with effective transformation $L \mapsto \psi^L_{\mathsf{reach}}(\bar{x},\bar{y})$ for $L \in \mathcal{L}$ then the FO($\mathcal{L}$)-theory of $\mathfrak{A}$ is decidable as well.* □

PROOF Let $\Big(\llbracket\psi_{\mathrm{dom}}(\bar{x})\rrbracket^{\mathfrak{B}}, \big(\llbracket\psi_P(\bar{x}_1,\ldots,\bar{x}_n)\rrbracket^{\mathfrak{B}}\big)_{P\in R_n(\tau),\, n\in\mathbb{N}}, \big(\psi^L_{\mathsf{reach}}(\bar{x},\bar{y})\big)_{L\in\mathcal{L}}\Big)$ be a m-tuple-$\mathfrak{L}$-interpretation with $\mathcal{L}$-reachability meeting all of the preconditions of the theorem. We reuse the proof of Theorem 2.16 and extend the formula-transformation to a domain with reachability expressions by specifying the transformation for that very type of formula:

$$\hat{\cdot}: \Phi_{\mathrm{FO}(\mathcal{L})}(\tau) \to \Phi_{\mathfrak{L}}(\pi) \quad \text{with} \quad \widehat{\mathsf{reach}_L(x,y)} := \psi^L_{\mathsf{reach}}(x_1,\ldots,x_m,y_1,\ldots,y_m).$$

Since transforming reachability expressions is effective by precondition the whole transformation stays effective. And again we get by construction that $\mathfrak{A} \models \varphi$ iff $\mathfrak{B} \models \hat{\varphi}$. Thus the decidability of the $\mathfrak{L}$-theory of $\mathfrak{B}$ yields a decidable FO($\mathcal{L}$)-theory of $\mathfrak{A}$. ∎

2.5 Trees

A *tree* is a special type of a graph structure which is acyclic and where each but one special vertex has exactly one predecessor. The special vertex is called the *root*. Formally consider a tree as a function $t : L \to \Sigma$ which is defined on a domain $L \subseteq \mathbb{N}^*$ and a codomain Σ which is a *ranked* finite alphabet, i.e. each symbol $f \in \Sigma$ has some *rank* $|f| \in \mathbb{N}$. To meet the requirements of a tree we furthermore demand that $\epsilon \in L$ (the root) and that for each $n \in \mathbb{N}$: $w \cdot n \in L$ iff $w \in L$ and $0 \le n < |t(w)|$. A *subtree* of tree t *rooted* at some position $r \in L$ is the tree $t|_r : \{\, w \mid r{\cdot}w \in L \,\} \to \Sigma$ with $t|_r(w) := t(r{\cdot}w)$. If the root of tree t has symbol $f = t(\epsilon) \in \Sigma$ then we also write $t = f(t|_1,\ldots,t|_{|f|})$.

We say that t is *finite* if the domain L is finite, otherwise it is *infinite*. A tree is said to be *regular* if it has finitely many different subtrees (up to isomorphism). Thus finite trees are always regular and regular infinite trees can be represented finitely as well. The set T_Σ consists of all trees over the ranked alphabet Σ, while the subset $T^{\mathsf{fin}}_\Sigma \subseteq T_\Sigma$ contains just the finite ones.

Automata We will extend the ideas of Section 2.3 about languages of words to trees by generalizing the definition of automata. A set $T \subseteq T^{\mathsf{fin}}_\Sigma$ is called a *language of finite trees* or *tree-language* for short.

A *deterministic finite (bottom-up) tree-automaton* $\mathcal{A} = \big(Q, \Sigma, (\delta_f)_{f\in\Sigma}, F\big)$ has a finite state set Q with a subset of final states $F \subseteq Q$, a ranked alphabet Σ and *transition functions* $\delta_f : Q^{|f|} \to Q$ for each $f \in \Sigma$. The automaton $\mathcal{A}$ evaluates a tree $t = f(t_1,\ldots,t_{|f|}) \in T^{\mathsf{fin}}_\Sigma$ inductively to a state $\hat{\delta}(t) := \delta_f\big(\hat{\delta}(t_1),\ldots,\hat{\delta}(t_{|f|})\big)$. The tree-language recognized by $\mathcal{A}$ is $T^{\mathsf{fin}}(\mathcal{A}) := \{\, t \in T^{\mathsf{fin}}_\Sigma \mid \hat{\delta}(t) \in F \,\}$. The tree-languages recognizable by deterministic finite tree-automata are called *regular tree-languages*.

Transducer A simple way to transform words of trees is by transducers. A *deterministic finite (top-down) tree-transducer* $\mathcal{T} = \big(Q, \Sigma, \Gamma, (\delta_q)_{q\in Q}, q_0\big)$ consists of a finite state set $Q = Q_P \dot\cup Q_C$ of *production* states Q_P and *consumption* states Q_C, an initial state $q_0 \in Q$, two ranked alphabets Σ, Γ for input and output and *transduction functions* δ_q such that

- for each production state $q \in Q_P$: δ_q is a constant tuple $(g, q_1,\ldots,q_{|g|})$ for some $g \in \Gamma$,

- for each consumption state $q \in Q_C$: δ_q is a function mapping each $f \in \Sigma$ to some tuple (i, q') with $0 \le i \le |f|$ and $q' \in Q$.

A tree $t \in T_\Sigma$ gets *transformed* by the transducer $\mathcal{T}$ to the tree $\mathcal{T}(t) := \hat{\delta}_{q_0}(t) \in T_\Gamma$ with $\hat{\delta}$ being inductively defined as

$$\hat{\delta}_q\big(t = f(t_1, \ldots, t_{|f|})\big) \quad = \quad \begin{cases} g\big(\hat{\delta}_{q_1}(t), \ldots, \hat{\delta}_{q_{|g|}}(t)\big) & \text{if } q \in Q_P \text{ and } \delta_q = (g, q_1, \ldots, q_{|g|}), \\ \hat{\delta}_{q'}(t_i) & \text{if } q \in Q_C \text{ and } \delta_q(f) = (i, q'). \end{cases}$$

Those transducers have two interesting properties. The first one is on preserving representability of the transformed tree.

Remark 2.18 Deterministic finite tree-transducers preserve regularity of trees. □

The idea of the proof uses the fact that regular trees can be represented finitely, say by some set Q of states. The transducer has another finite set Q' of states. When we consider a state set which is the product $Q \times Q'$ of the old ones then we have a finite representation for the transformed tree. Thus it is regular.

The second result is on preserving decidability of logics. Therefore we identify a tree $t : L \to \Sigma$ with a graph structure $\big(L, E, R, (P_f)_{f\in\Sigma}\big)$ with unlabeled edges $E = \big\{ (w, w \cdot n) \;\big|\; w \cdot n \in L \big\}$, $R = \big\{ (w \cdot (n-1), w \cdot n) \;\big|\; w \cdot n \in L,\ n \ge 1 \big\}$ and unary predicates $P_f = \big\{ w \;\big|\; t(w) = f \big\}$.

Remark 2.19 Deterministic finite tree-transducers preserve the decidability of MSO logic of trees. □

This can be proven be transforming the given tree structure by the means of structure transformations which preserve the decidability of MSO logic. For instance MSO-interpretation and unfolding.

3 Set-Based Unfolding

In this section we want to present a new kind of model-theoretic structure transformation. It all started with ideas of [LO2007] about the decidability of the FO(Reg)-theory of (the graphs of) free inverse monoids. We dumped all the algebraic background from the proof such that the remaining core yields the following transformation which works fine not just for free inverse monoids.

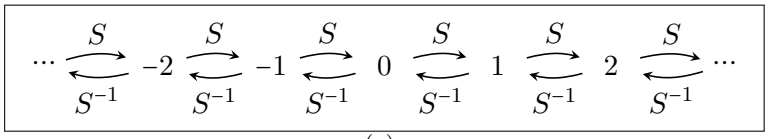
The set-based unfolding is an abstraction of the normal (path-based) unfolding. We do not take complete paths as elements of our new structure; instead we consider the elements of the original structure we have visited on such a path and additionally store its last element. Thus instead of taking a path $v_0 P_1 v_1 \dots P_n v_n$ as an element we abstract from it and just keep the tuple $\big(v_n, \{v_0, v_1, \dots, v_n\}\big)$. This is a so called a *trace*. In the general definition multiple start traces are allowed.

Definition 3.1 (Set-based unfolding) Let $\mathfrak{A} = \big(A, (P^{\mathfrak{A}}_\gamma)_{\gamma\in\Gamma}, (E^{\mathfrak{A}}_\sigma)_{\sigma\in\Sigma}\big)$ be a relational structure with Γ-labeled unary predicates $P^{\mathfrak{A}}_\gamma$ and Σ-labeled binary edge relations $E^{\mathfrak{A}}_\sigma$. The set-based unfolding $\mathsf{Unf}^{\mathsf{Set}}_I(\mathfrak{A})$ of structure $\mathfrak{A}$ from some set of start traces $I \subseteq A \times \mathcal{P}(A)$ (i.e. for each trace $(a_0, A_0) \in I$: A_0 is finite and $a_0 \in A_0$) is a structure $\mathsf{Unf}^{\mathsf{Set}}_{A_0}(\mathfrak{A}) := \mathfrak{B} = \big(B, (P^{\mathfrak{B}}_\gamma)_{\gamma\in\Gamma}, (E^{\mathfrak{B}}_\sigma)_{\sigma\in\Sigma}\big)$ of the same signature with:

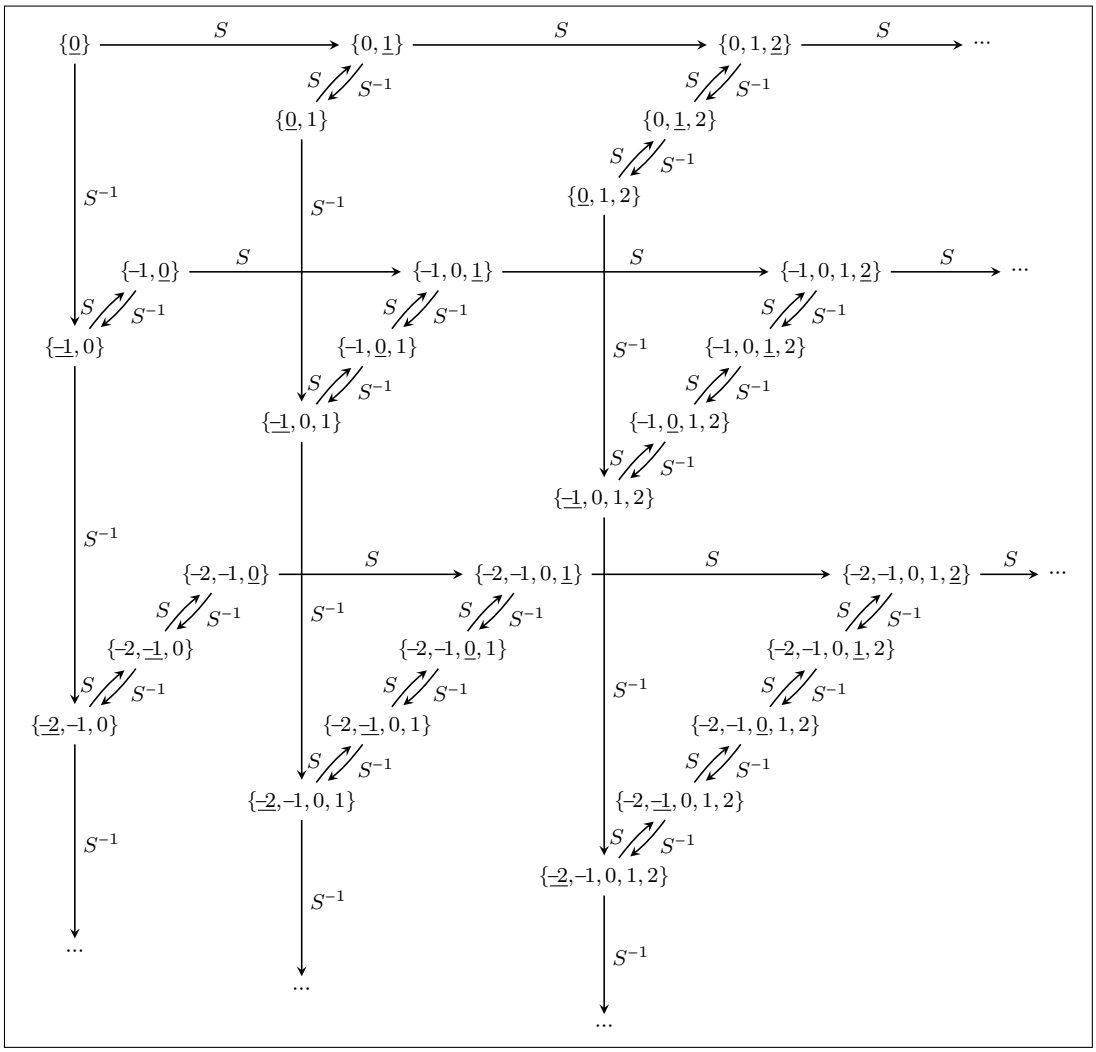
- B is the smallest set of traces which
  - contains the start traces $I \subseteq B \subseteq \big(A \times \mathcal{P}(A)\big)$,
  - and is closed under $E^{\mathfrak{B}}_\sigma$, i.e. that if trace $(v, V) \in B$ and $(v, v') \in E^{\mathfrak{A}}_\sigma$ then trace $\big(v', V \cup \{v'\}\big) \in B$,
- $P^{\mathfrak{B}}_\gamma := \Big\{ (v, V) \in B \;\Big|\; v \in P^{\mathfrak{A}}_\gamma \Big\}$ and
- $E^{\mathfrak{B}}_\sigma := \Big\{ \big((v, V), (v', V \cup \{v'\})\big) \in B^2 \;\Big|\; (v, v') \in E^{\mathfrak{A}}_\sigma \Big\}$. □

Informally spoken the structure $\mathsf{Unf}^{\mathsf{Set}}_I(\mathfrak{A})$ consists of all traces reachable from the start traces I while the distinguished elements simply inherit the unary predicates and edge relations from the original structure $\mathfrak{A}$. In contrast to the normal (path-based) unfolding from Subsection 2.4.4 the set-based unfolding does not have to be a tree.

Example 3.2 (Free inverse monoid) Consider the structure of the free group of the singleton alphabet $\{S\}$ which is isomorphic to the structure of the integers $\mathfrak{Z} = (\mathbb{Z}, 0, S, S^{-1})$ with zero, successor relation S and predecessor relation S^{-1} as in Figure 3.1(a). With the set of start traces I just containing the single trace $\big(0, \{0\}\big)$ set-based unfolding yields a structure as depicted in Figure 3.1(b). From an algebraic point of view this structure is isomorphic to the free inverse monoid $\mathrm{FIM}(\{S\})$ of the singleton alphabet $\{S\}$. □



(a)



(b)

Figure 3.1: (a) Structure $\mathfrak{Z} = (\mathbb{Z}, 0, S, S^{-1})$ of integers and (b) its set-based unfolding $\mathsf{Unf}_I^{\mathsf{Set}}(\mathfrak{Z})$ from $I := \{(0, \{0\})\}$. Unary predicates have been omitted and the last element of each trace has been underlined for a more compact notation. See Example 3.2.

Like for the normal (path-based) unfolding we have a result that allows us to transfer decidability of some logic. The proof uses a mixture of the standard MSO-interpretation from Subsection 2.4.4 and finite set interpretation [CL2007], i.e. a new structure consists of finite sets of the old one. While MSO-interpretation preserves the decidability of MSO logic, the finite set interpretations transfers the decidability of MSO logic only to FO logic in the interpreted structure. The result we achieve is a mixture of the properties of both interpretations as well: starting with a structure where finiteness is MSO-definable and MSO-logic is decidable we end up having FO(Reg) logic decidable, which is between MSO and FO logic. This is the main result about set-based unfoldings:

Theorem 3.3 *For any graph structure $\mathfrak{A}$ having a decidable MSO-theory the FO(Reg)-theory of a quotient of the set-based unfolding $\mathsf{Unf}_I^{\mathsf{Set}}(\mathfrak{A})/{\sim}$ is decidable if the following specifications are MSO-definable in $\mathfrak{A}$:*

- *the set of start traces I,*
- *the congruence $\sim$ and*
- *finiteness.* □

Proof The proof is a mixture of standard MSO-interpretation and finite set interpretation. Some element x of the new structure is represented in the old structure as the same single element x by the first interpretation and as a finite set X of elements by the second interpretation. Here we combine both approaches and represent it as the tuple (x, X). The obvious idea is to encode a trace as a tuple of an FO and an MSO-variable for its last element and the set of visited elements respectively. An FO(Reg)-formula φ will then be transformed into an MSO-formula $\hat{\varphi}$ such that $\mathsf{Unf}_I^{\mathsf{Set}}(\mathfrak{A})/{\sim} \vDash \varphi$ iff $\mathfrak{A} \vDash \hat{\varphi}$. This way we are able to decide whether a given FO(Reg)-formula φ holds in set-based unfolding $\mathsf{Unf}_I^{\mathsf{Set}}(\mathfrak{A})/{\sim}$ by transforming it to $\hat{\varphi}$ and checking that MSO-formula on $\mathfrak{A}$ which can be done by precondition. The formula transformation $\hat{\cdot} : \Phi_{\mathrm{FO(Reg)}}(\tau) \to \Phi_{\mathrm{MSO}}(\tau)$ is defined inductively as follows:

$$
\begin{aligned}
\widehat{x = y} &:= \psi_{\sim}(x, X, y, Y) \\
\widehat{P_\gamma(x)} &:= P_\gamma(x) \\
\widehat{E_\sigma(x, y)} &:= \widehat{\mathsf{reach}_\sigma(x, y)} \\
\widehat{\neg\varphi} &:= \neg\hat{\varphi} \\
\widehat{\varphi \vee \psi} &:= \hat{\varphi} \vee \hat{\psi} \\
\widehat{\exists x : \varphi} &:= \exists x : \exists X : \Big(\hat{\varphi} \wedge \psi_{\mathrm{dom}}(x, X)\Big) \\
\widehat{\mathsf{reach}_L(x, y)} &:= \exists y' : \exists Y', Z : \Big(\mathsf{Reach}_L(x, y', Z) \wedge (Y' = X \cup Z) \\
&\qquad\qquad \wedge \psi_{\sim}(y, Y, y', Y') \wedge \psi_{\mathrm{dom}}(y', Y')\Big)
\end{aligned}
$$

where $\psi_{\sim}(x, X, y, Y) \in \Phi_{\mathrm{MSO}}(\tau)$ defines the congruence $\sim$, $\psi_I(x, X)$ defines the set of start traces I and $\mathsf{Reach}_L(x, y, Z)$ is MSO-definable in $\mathfrak{B}$ according to Proposition 2.10.

The domain formula $\psi_{\mathrm{dom}}(x,X) = \exists y \exists Y,Z : \big(\psi_I(y,Y) \wedge \mathsf{Reach}_L(y,x,Z) \wedge (X = Y \cup Z)\big)$ we use here allows only traces that are reachable from some initial trace specified by $\psi_I(x,X)$. The set comparison $X = Y \cup Z$ for MSO-variable X,Y,Z is MSO-definable by the formula $\forall x : X(x) \leftrightarrow \big(Y(x) \vee Z(x)\big)$.

The idea behind the transformation of $\mathsf{reach}_L(x,y)$ is to start with trace (x,X) that reaches some trace (y',Y') that is congruent to (y,Y) by a path in L. To proof the correctness we have to recall the definition of an $\mathsf{Unf}_I^{\mathsf{Set}}(\mathfrak{A})$-congruence: for each n-ary relation R of $\mathsf{Unf}_I^{\mathsf{Set}}(\mathfrak{A})$ (here only $n \in \{1,2\}$) and traces $(b_1,B_1) \sim (b_1',B_1'),\ldots,(b_n,B_n) \sim (b_n',B_n')$:

$$\Big((b_1,B_1),\ldots,(b_n,B_n)\Big) \in R \quad \Leftrightarrow \quad \Big((b_1',B_1'),\ldots,(b_n',B_n')\Big) \in R.$$

By induction this can be extended to finite paths as well since it results from a finite usage of edge relations. Thus for congruent traces $(b_1,B_1) \sim (b_1',B_1'),(b_2,B_2) \sim (b_2',B_2')$ and any word $w \in (\Sigma \cup \Gamma)^*$ we get:

$$\mathsf{Unf}_I^{\mathsf{Set}}(\mathfrak{A}) \vDash \mathsf{reach}_w\Big((b_1,B_1),(b_2,B_2)\Big) \quad \Leftrightarrow \quad \mathsf{Unf}_I^{\mathsf{Set}}(\mathfrak{A}) \vDash \mathsf{reach}_w\Big((b_1',B_1'),(b_2',B_2')\Big).$$

Since we know this now for each single word it does holds for languages as well. ∎

Since the trivial congruence $=$ (i.e. $x \sim y \Leftrightarrow x = y$) is MSO-definable we can state a simplified version of Theorem 3.3:

Corollary 3.4 *For any graph structure $\mathfrak{A}$ having a decidable MSO-theory the FO(Reg)-theory of the set-based unfolding $\mathsf{Unf}_I^{\mathsf{Set}}(\mathfrak{A})$ is decidable if the set of start traces I and finiteness are MSO-definable in $\mathfrak{A}$.* □

When asking for the power of set-based unfolding one might wonder why Theorem 3.3 ends up with just FO(Reg) instead of full MSO like for normal (path-based) unfolding. This result is sharp and we cannot achieve full MSO. To show this reconsider the structure FIM($\{S\}$) of the free inverse monoid of a singleton alphabet from Example 3.2. This structure was shown to have no decidable MSO-theory in [Cal1997]. We produced it as a very simple set-based unfolding (i.e. with the trivial congruence $x \sim y \Leftrightarrow x = y$) of the structure $\mathfrak{Z}$ of integers, which itself matches all the premises of Theorem 3.3. So when applying this theorem we conclude that the FO(Reg)-theory of FIM($\{S\}$) is decidable. This fact has already been worked out in [LO2007] where the idea for set-based unfolding came from. There a very limited version of it was used for a proof. Nevertheless we can reuse this result to underline the qualities of set-based unfolding as a general method of structure transformation. We have not yet investigated other structures with a similar property.

Remark 3.5 Free inverse monoids have undecidable MSO-theories. Nevertheless their FO(Reg)-theories can be shown to be decidable by set-based unfolding. □

The first part of this remark can be shown by taking the structure FIM($\{S\}$) of the free inverse monoid of alphabet $\{S\}$ (see Example 3.2) and MSO-interpreting the

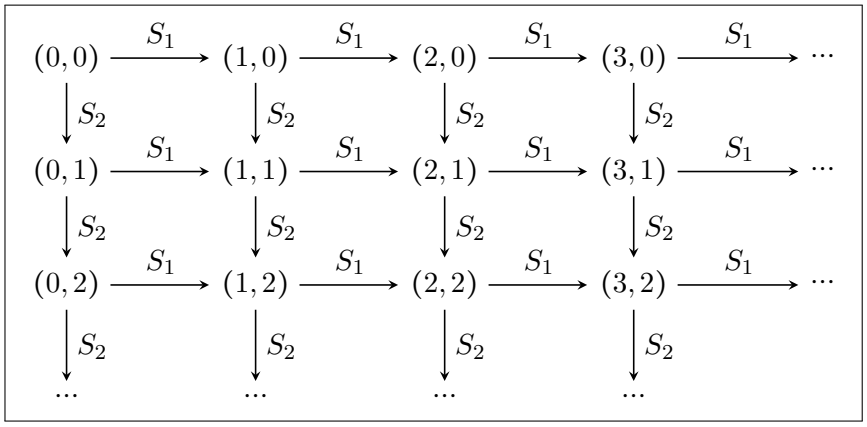


Figure 3.2: 2-dimensional infinite grid $\mathfrak{N}^2$.

infinite grid $\mathfrak{N}^2 = (\mathbb{N}^2, S_1, S_2)$ in it with $S_1 = \{((x,y),(x{+}1,y)) \mid x,y \in \mathbb{N}\}$ and $S_2 = \{((x,y),(x,y{+}1)) \mid x,y \in \mathbb{N}\}$ as depicted in Figure 3.2. An MSO-interpretation $(\psi_{\mathrm{dom}}(x), \psi_{S_1}(x,y), \psi_{S_2}(x,y))$ to produce the grid from the free inverse monoid could look like this:

$$\begin{aligned}
\psi_{\mathrm{dom}}(x) &:= \exists y : \big(S^{-1}(x,y) \wedge \neg S(y,x)\big) \\
\psi_{S_1}(x,y) &:= \exists z, z' : \big(\mathsf{reach}(x,z) \wedge \mathsf{reach}(z,x) \wedge S(z,z') \wedge \\
&\qquad\qquad \mathsf{reach}(y,z') \wedge \mathsf{reach}(z',y) \wedge \psi_{\mathrm{dom}}(y)\big) \\
\psi_{S_2}(x,y) &:= S^{-1}(x,y)
\end{aligned}$$

The similarity of the two structures can be seen when comparing Figure 3.1(b) and 3.2. This interpretation represents each element $(n_1, n_2) \in \mathbb{N}^2$ of the infinite grid as a trace with visited set $\{-n_2, \ldots, n_1\}$ of the free inverse monoid. There can be many of those traces so we fix the S^{-1}-maximal of them which is $\big(-n_2, \{-n_2, \ldots, n_1\}\big)$.

Since the grid $\mathfrak{N}^2$ is MSO-interpretable in the monoid $\mathrm{FIM}(\{S\})$ the following fact allows us to directly transfer the undecidability of the MSO-theory from the grid to the monoid structure according to Remark 2.12:

Proposition 3.6 ([Tho1990]) *The MSO-theory of the infinite grid $\mathfrak{N}^2$ is undecidable.* □

So far we have seen the structure $\mathrm{FIM}(\{S\})$ and that its FO(Reg)-theory is decidable but its MSO-theory is not. This is due to its very close connection to the infinite grid $\mathfrak{N}^2$ which has an undecidable MSO-theory. An interesting question would be if the FO(Reg)-theory of the grid $\mathfrak{N}^2$ is decidable. Perhaps it is possible to achieve this structure by a quotient structure of some set-based unfolding? That latter question is still open but likely to be negative.

Using the idea from the proof above we get a relation $\sim$ on $\mathrm{FIM}(\{S\})$ such that the equivalence classes form the domain of $\mathfrak{N}^2$ in a very natural way: for any two traces $(n,N),(m,M)$ we define $(n,N) \sim (m,M)$ iff $n = m$. Obviously this is an equivalence

relation but we cannot use it for Theorem 3.3 since it is no congruence. This truly is a pity since creating the grid by trace-interpretation would attest a lot of power to this interpretation. So the first question has to be answered in a different way. Chapter 4 will address this and other related problems and comes up with results for most of them. Among others the FO(Reg)-theory of the infinite grid $\mathfrak{N}^2$ is decidable (Theorem 4.7).

4 Reachability in Infinite Grids

In this section we will study infinite grid structures and the decidability of their first order theories with respect to reachability. In our case the n-dimensional infinite grid $\mathfrak{N}^n$ is the product of the n structures of natural numbers $\mathfrak{N}$ with successor relation S:

$$\mathfrak{N} = (\mathbb{N}, S) \quad \Rightarrow \quad \mathfrak{N}^n = \big(\mathbb{N}^n, (S_i)_{1\le i\le n}\big).$$

See Figure 3.2 for the 2-dimensional grid $\mathfrak{N}^2$. When considering logical formulas in FO or MSO it does not matter if we define extra predecessor relations S_i^{-1} or border predicates 0_i because those are already definable in FO and MSO. But in this case it does matter since we are dealing with FO plus reachability expressions: in for instance $\mathfrak{N}^2$ the node $(1,1)$ is reachable from $(0,0)$ but not the other way round. So let us define some variants of grids:

Definition 4.1 Infinite grids ($n \ge 1$):

- 1-way grid: $\mathfrak{N}^n := \big(\mathbb{N}^n, (S_i)_{1\le i\le n}\big)$.
- 2-way grid: $\mathfrak{N}^{\pm n} := \big(\mathbb{N}^n, (S_i)_{1\le i\le n}, (S_i^{-1})_{1\le i\le n}\big)$.
- 1-way grid with 0-test: $\mathfrak{N}_0^n := \big(\mathbb{N}^n, (S_i)_{1\le i\le n}, (0_i)_{1\le i\le n}\big)$.
- 2-way grid with 0-test: $\mathfrak{N}_0^{\pm n} := \big(\mathbb{N}^n, (S_i)_{1\le i\le n}, (S_i^{-1})_{1\le i\le n}, (0_i)_{1\le i\le n}\big)$.

where $\begin{cases} \big((x_1,\dots,x_n),(y_1,\dots,y_n)\big) \in S_i \text{ iff } x_j = y_j \text{ for all } j \ne i \text{ and } x_i + 1 = y_i, \\ \big((x_1,\dots,x_n),(y_1,\dots,y_n)\big) \in S_i^{-1} \text{ iff } x_j = y_j \text{ for all } j \ne i \text{ and } x_i = y_i + 1, \\ (x_1,\dots,x_n) \in 0_i \text{ iff } x_i = 0. \end{cases}$ □

The idea of introducing the unary 0-test predicates 0_i is that these are useful in reachability expressions for logics higher than FO(R). In a reachability expression $\mathsf{reach}_L(x,y)$ for some language L elements of the unary relations 0_i can be seen as having loops labeled with 0_i in order to think of them as edge as well (see Figure 4.1). Later on we will see that there are huge differences in the decidability of the theories of these variants of logics.

4.1 Decidability Results

We will start with a small remark about the origin of all grids: the natural numbers with successor $\mathfrak{N} = (\mathbb{N}, S)$. Fortunately $\mathfrak{N}$ has good decidability properties since it is not really a product structure like higher-dimensional grids.

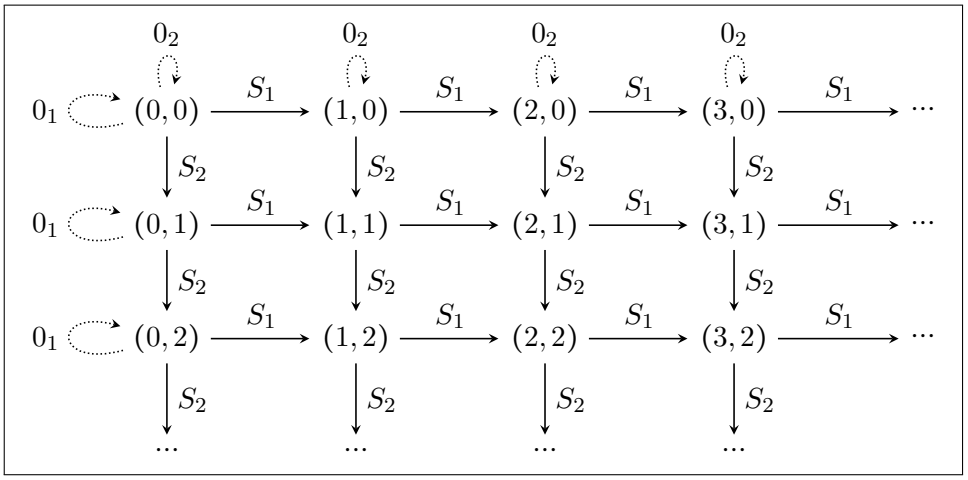


Figure 4.1: 2-dimensional 1-way grid with 0-test $\mathfrak{N}_0^2$.

Proposition 4.2 ([Büc1962]) *The MSO-theory of* $\mathfrak{N} = (\mathbb{N}, S)$ *is decidable.* □

Formally this structure $\mathfrak{N}$ is equal to $\mathfrak{N}^1$ and in case of MSO it is basically the same as any 1-dimensional grid.

Remark 4.3 The 1-dimensional grids $\mathfrak{N}^1$, $\mathfrak{N}_0^1$, $\mathfrak{N}^{\pm 1}$, $\mathfrak{N}_0^{\pm 1}$ have decidable MSO-theories. □

This can help us to gain positive decidability results on higher dimensional grids when considering weak logics like FO(R) or below. This is because it is very easy to interpret n-dimensional grids in the 1-dimensional one, since the higher ones simply are products of the 1-dimensional grid. The following n-tuple-MSO-interpretation with R-reachability reduces the FO(R)-theories of $\mathfrak{N}^n$, $\mathfrak{N}_0^n$, $\mathfrak{N}^{\pm n}$ and $\mathfrak{N}_0^{\pm n}$ to the decidable MSO-theories of $\mathfrak{N}^1$, $\mathfrak{N}_0^1$, $\mathfrak{N}^{\pm 1}$ and $\mathfrak{N}_0^{\pm 1}$ respectively according to Theorem 2.17:

$$
\begin{aligned}
\psi_{\mathrm{dom}}(x_1,\ldots,x_n) &:= \mathsf{true} \\
\psi_{0_i}(x_1,\ldots,x_n) &:= 0(x_i) && \text{(only for } \mathfrak{N}_0^n \text{ and } \mathfrak{N}_0^{\pm n}\text{)} \\
\psi_{S_i}(x_1,\ldots,x_n,y_1,\ldots,y_n) &:= \bigwedge_{j\neq i} x_j = y_j \ \wedge\ S(x_i,y_i) \\
\psi_{S_i^{-1}}(x_1,\ldots,x_n,y_1,\ldots,y_n) &:= \bigwedge_{j\neq i} x_j = y_j \ \wedge\ S^{-1}(x_i,y_i) && \text{(only for } \mathfrak{N}^{\pm n} \text{ and } \mathfrak{N}_0^{\pm n}\text{)} \\
\psi^{\Sigma_0^*}_{\mathsf{reach}}(x_1,\ldots,x_n,y_1,\ldots,y_n) &:= \bigwedge_i \mathsf{reach}_{(\Sigma_0\cap\{S_i,S_i^{-1}\})^*}(x_i,y_i)
\end{aligned}
$$

Remark 4.4 The FO(R)-theories of $\mathfrak{N}^n$, $\mathfrak{N}_0^n$, $\mathfrak{N}^{\pm n}$ and $\mathfrak{N}_0^{\pm n}$ are decidable for all $n \geq 1$. □

For FO with more powerful reachability let us first study 1-way grids without 0-test: $\mathfrak{N}^n$. Consider a reachability expression $\mathsf{reach}_L(x,y)$ for some language $L \subseteq \{S_1,\ldots,S_n\}^*$. The movement in each dimension is independent from the other dimensions and each dimension can move just forward. Thus for the reachability expression $\mathsf{reach}_L(x,y)$ the

exact order of symbols in the words of L does not matter. It is just important how often each symbol occurs in a word. This is exactly what the *Parikh mapping* $\Psi(L)$ is talking about:

$$\Psi(L) \;:=\; \left\{ \left(|w|_{S_1}, \ldots, |w|_{S_n} \right) \;\middle|\; w \in L \right\} \;\subseteq\; \mathbb{N}^n.$$

Fortunately the Parikh mapping $\Psi(L)$ will be of a very simple kind if L is not too complex:

Proposition 4.5 ([Par1966, Theorem 2]) *The Parikh mapping $\Psi(L) \subseteq \mathbb{N}^n$ of a context-free language L over an (ordered) alphabet Σ of size n is semilinear. Furthermore this semilinear representation is effective.* □

A set $N \subseteq \mathbb{N}^n$ is *linear* if $N = \bar{x}_0 + \langle \bar{x}_1, \ldots, \bar{x}_m \rangle$ for some $\bar{x}_i \in \mathbb{N}^n$ where $\langle \bar{x}_1, \ldots, \bar{x}_m \rangle := \left\{ k_1 \cdot \bar{x}_1 + \ldots + k_n \cdot \bar{x}_n \;\middle|\; k_i \in \mathbb{N} \right\}$ is the submonoid generated by $\bar{x}_1, \ldots, \bar{x}_m$. A *semilinear* set is a finite union of linear sets.

Example 4.6 (Parikh mappings) The regular language $L := (S_1 + S_1 \cdot S_2)^* \cdot S_2$ has Parikh mapping $\Psi(L) = (0,1) + \langle (1,0), (1,1) \rangle$. □

By use of the semilinear representation of languages in reachability expressions we are able to simplify them and to decide the theories of 1-way grids without 0-test, which is one of the main results of this section:

Theorem 4.7 *The FO(CF)-theory of $\mathfrak{N}^n$ is decidable for all $n \geq 1$.* □

Proof We will reduce this problem to *Presburger arithmetic*, i.e. the FO-theory on natural numbers with addition $(\mathbb{N}, 0, 1, +)$ which is decidable [Pre1929, Pre1991]. This will be done by n-tuple-FO-interpretation with CF-reachability:

$$\begin{aligned}
\psi_{\mathrm{dom}}(x_1, \ldots, x_n) &:= \mathsf{true} \\
\psi_{S_i}(x_1, \ldots, x_n, y_1, \ldots, y_n) &:= \bigwedge_{j \neq i} x_j = y_j \;\wedge\; x_i + 1 = y_i \\
\psi^L_{\mathsf{reach}}(x_1, \ldots, x_n, y_1, \ldots, y_n) &:= \bigvee_k \exists \ell_1, \ldots, \ell_{m_k} : \\
&\qquad \bigwedge_i y_i = x_i + \underbrace{(1 + \ldots + 1)}_{z_{0,i}} + \Big(\underbrace{(\ell_1 + \ldots + \ell_1)}_{z_{1,i}} + \ldots + \underbrace{(\ell_{m_k} + \ldots + \ell_{m_k})}_{z_{m_k,i}} \Big) \\
&\qquad \text{where } \Psi(L) = \bigcup_k \bar{z}_0 + \langle \bar{z}_1, \ldots, \bar{z}_{m_k} \rangle
\end{aligned}$$

The trick for reachability expressions is taking the Parikh mapping (Proposition 4.5) and checking for each of its linear sets $\bar{z}_0 + \langle \bar{z}_1, \ldots, \bar{z}_{m_k} \rangle$ if the difference tuple between $\bar{x}$ and $\bar{y}$ is in this set: $\bigwedge_i y_i = x_i + z_{0,i} + (\ell_1 \cdot z_{1,i} + \ldots + \ell_{m_k} \cdot z_{m_k,i})$. The transformation from a context-free language L to its reachability formula $\psi^L_{\mathsf{reach}}(x, y)$ is obviously effective hence Theorem 2.17 allows us to transfer the decidability of Presburger arithmetic to the FO(CF)-theory of the 1-way grids. ∎

It is even possible to turn the proof around by reducing Presburger arithmetic back to the FO(SF)-theory of $\mathfrak{N}^2$. This can be done by the following 1-tuple-FO(SF)-interpretation (without loss of generality we will treat the functions of Presburger arithmetic as relations, i.e. $0 \coloneqq \{0\}, 1 \coloneqq \{1\} \subseteq \mathbb{N}^1$ and $+ \coloneqq \left\{ (x,y,z) \in \mathbb{N}^3 \;\middle|\; x+y=z \right\} \subseteq \mathbb{N}^3$) according to Theorem 2.16:

$$\begin{aligned}
\psi_{\mathrm{dom}}(x) &\coloneqq \neg\exists y : S_2(y,x) \\
\psi_0(x) &\coloneqq \neg\exists y : S_1(y,x) \\
\psi_+(x,y,z) &\coloneqq \exists y', z' : \mathsf{reach}_{(S_1 \cdot S_2)^*}(0,y') \wedge \mathsf{reach}_{S_2^*}(y,y') \wedge \\
&\qquad\qquad \mathsf{reach}_{(S_1 \cdot S_2)^*}(x,z') \wedge \mathsf{reach}_{S_1^*}(y',z') \wedge \mathsf{reach}_{S_2^*}(z,z')
\end{aligned}$$

where 0 is the element origin $(0,0)$ of the grid which is FO-definable easily. Any natural number n is represented in the grid by $(n,0)$ according to the above definition. The addition is calculated in a geometric way. When checking $x+y=z$ we have to verify that $(x,0)+(y,0)=(z,0)$ in the grid. The first step is defining the point $y'=(y,y)$ by finding the point on the diagonal which has the same first component as $(y,0)$. Then we do the same trick but start with the diagonal in the point $(x,0)$ instead of $(0,0)$ and take the point that has the same second component as $y'=(y,y)$. By this we get point $z'=(x+y,y)$. The last reachability expression ensures that $(z,0)$ has the same first component as z'. Thus we end up with $z=x+y$. All languages used in reachability expressions in this proof are star-free (without proof).

Remark 4.8 Presburger arithmetic, i.e. FO-theory of $(\mathbb{N},0,1,+)$, can be reduced to the FO(SF)-theory of $\mathfrak{N}^2$. □

Next we want to extend the result of Theorem 4.7 to infinite 1-way grids with 0-test. A 0-test $0_i(x)$ outside of a reachability expression is easy to express by using just plain FO: $0_i(x) \coloneqq \neg\exists y : S_i(y,x)$. Thus the problematic occurrences are those resting in the language L of a reachability expression $\mathsf{reach}_L(x,y)$. But since we can only move forward in 1-way grids we can probably alter the language L in such a way that explicit 0-tests are not needed and done implicitly instead.

Lemma 4.9 *Each reachability expression* $\mathsf{reach}_L(x,y)$ *with a context-free language* $L \subseteq \{S_1,\ldots,S_n,0_1,\ldots,0_n\}^*$ *can be replaced by an FO(CF)-formula without 0-tests.* □

Proof Consider $\mathsf{reach}_L(x,y)$. Without loss of generality $\epsilon \notin L$, otherwise replace $\mathsf{reach}_L(x,y)$ by $x=y \vee \mathsf{reach}_{L\setminus\{\epsilon\}}(x,y)$. Let $G=(N,\Sigma=\{S_1,\ldots,S_n,0_1,\ldots,0_n\},P,S)$ be a context-free grammar for $L \subseteq \Sigma^*$ in *Chomsky normal form*, i.e. P just contains productions of the form $A \to B{\cdot}C$ or $A \to a$ with $A,B,C \in N$ and $a \in \Sigma$.

We will transform the productions of G in such a way that each nonterminal symbol $A \in N$ gets several copies $A_{\bar{x},\bar{y}}$ for $\bar{x},\bar{y} \in \mathbb{B}^n$ and $\bar{x} \leq \bar{y}$. The *input vector* $\bar{x}$ indicates which dimensions have to be treated as non-0 before the production A is used. Analogously the *output vector* $\bar{y}$ indicates which dimensions are non-0 afterward. Using this we can remove the 0-test symbols $0_i \in \Sigma$ since the test can be done implicitly.

Formally we define a context-free grammar $G_{\bar{x}} = (N', \Sigma' = \{S_1, \ldots, S_n\}, P', S')$ for each $\bar{x} \in \mathbb{B}^n$ with nonterminals $N' := \{S'\} \cup \left\{ A_{\bar{x},\bar{y}} \;\middle|\; A \in N,\ \bar{x}, \bar{y} \in \mathbb{B}^n,\ \bar{x} \leq \bar{y} \right\}$, and the following productions in P' (for each $\bar{y}, \bar{y}', \bar{z} \in \mathbb{B}^n$ with $\bar{x} \leq \bar{y} \leq \bar{y}' \leq \bar{z}$):

- $A_{\bar{y},\bar{z}} \to B_{\bar{y},\bar{y}'} \cdot C_{\bar{y}',\bar{z}}$ iff $A \to B \cdot C$ in P,
- $A_{\bar{y},\bar{z}} \to S_i$ with $\bar{z} = (y_1, \ldots, y_{i-1}, 1, y_{i+1}, \ldots, y_n)$ iff $A \to S_i$ in P,
- $A_{\bar{y},\bar{y}} \to \epsilon$ iff $y_i = 0$ and $A \to 0_i$ in P,
- $S' \to S_{\bar{x},\bar{y}}$.

Notice that these rules only allow to skip an 0-test in some component i iff the input vector guarantees that this component is still 0.

If we know the input vector $\bar{z}$ at the position of the considered reachability expression $\mathsf{reach}_L(x, y)$ in a FO(CF)-formula then we can replace this expression by $\mathsf{reach}_{L(G_{\bar{z}})}(x, y)$ which has no 0-test in it. It remains to find the correct input vector for each situation. This can be done by choosing z_i according to $\exists y : S_i(y, x)$. In total the replacement expression for $\mathsf{reach}_L(x, y)$ is

$$\bigvee_{\bar{z} \in \mathbb{B}^n} \Big(\big(\bigwedge_i \psi_{i,z_i}(x) \big) \to \mathsf{reach}_{L(G_{\bar{z}})}(x, y) \Big)$$

with $\psi_{i,1}(x) := \exists y : S_i(y, x)$ and $\psi_{i,0}(x) := \neg \psi_{i,1}(x)$. By construction this formula holds in the grid without 0-test $\mathfrak{N}^n$ iff $\mathsf{reach}_L(x, y)$ holds in the grid with 0-test $\mathfrak{N}_0^n$. ∎

Thus we can formally define a 1-tuple-FO(CF)-interpretation with CF-reachability of $\mathfrak{N}_0^n$ in $\mathfrak{N}^n$ by

$$\begin{aligned}
\psi_{\mathrm{dom}}(x) &:= \mathsf{true} \\
\psi_{0_i}(x) &:= \neg \exists y : S_i(y, x) \\
\psi_{S_i}(x, y) &:= S_i(x, y) \\
\psi^{L(G)}_{\mathsf{reach}}(x, y) &:= \bigvee_{\bar{z} \in \mathbb{B}^n} \Big(\big(\bigwedge_i \psi_{i,z_i}(x) \big) \to \mathsf{reach}_{L(G_{\bar{z}})}(x, y) \Big)
\end{aligned}$$

with the replacement for reachability expression from the proof above. Hence Lemma 4.9 and Theorem 4.7 yield:

Corollary 4.10 *The FO(CF)-theory of $\mathfrak{N}_0^n$ is decidable for all $n \geq 1$.* □

4.2 Undecidability Results

Unfortunately we cannot give more positive results on grids. This is because 2-way grids in which moving back and forth by reachability expressions is possible appear to allow more tricks. In those grids we can extend the result of Remark 4.8 from Presburger arithmetic to *Peano arithmetic*, i.e. the FO-theory of $(\mathbb{N}, 0, 1, +, \cdot)$ instead of just $(\mathbb{N}, 0, 1, +)$.

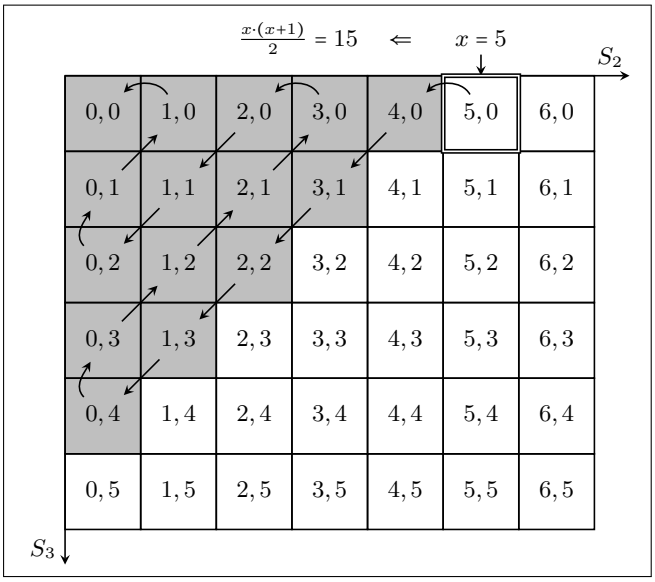


Figure 4.2: Counting the number of blocks in the triangle below $(x, 0)$.

Lemma 4.11 *Peano arithmetic, i.e. the FO-theory of* $(\mathbb{N}, 0, 1, +, \cdot)$*, can be reduced to the FO(SF)-theory of* $\mathfrak{N}^{\pm 3}$*.* □

Proof We have already seen how to reduce Presburger arithmetic to the FO(SF)-theory of $\mathfrak{N}^2$ in Remark 4.8. It is just left to interpret multiplication. To do so we first reduce multiplication to the square operation: $x \cdot y = z$ iff $(x + y)^2 = z + z + x^2 + y^2$. Next we reduce the square to another similar operation $\frac{x\cdot(x+1)}{2}$: $x^2 = y$ iff $\frac{x\cdot(x+1)}{2} + \frac{x\cdot(x+1)}{2} = y + x$. Geometrically motivated $\frac{x\cdot(x+1)}{2}$ is exactly the number of blocks in the triangle numerically below $(x, 0)$ as illustrated in Figure 4.2. We will count the blocks of a triangle in the second and third dimension by a reachability expression. The first dimension is used to count the blocks or to be more precise the number of edges on the way through the triangle as shown in Figure 4.2. Again without loss of generality we are working with relations instead of functions, i.e. $0 := \{0\}, 1 := \{1\} \subseteq \mathbb{N}^1$, $+ := \left\{ (x, y, z) \in \mathbb{N}^3 \;\middle|\; x + y = z \right\} \subseteq \mathbb{N}^3$ and $\triangle := \left\{ (x, y) \in \mathbb{N}^2 \;\middle|\; \frac{x\cdot(x+1)}{2} = y \right\} \subseteq \mathbb{N}^2$. We define a 1-tuple-FO(SF)-interpretation of $(\mathbb{N}, 0, 1, +, \triangle)$ in $\mathfrak{N}^{\pm 3}$ analogously to the one of Remark 4.8 but adapted to 3 dimensions and extended by the new relation $\triangle$:

$$
\begin{aligned}
\psi_{\mathrm{dom}}(x) &:= \neg\exists y : S_2(y, x) \vee S_3(y, x) \\
\psi_0(x) &:= \neg\exists y : S_1(y, x) \\
\psi_+(x, y, z) &:= \exists y', z' : \mathsf{reach}_{(S_1\cdot S_2)^*}(0, y') \wedge \mathsf{reach}_{S_2^*}(y, y') \wedge \\
&\qquad \mathsf{reach}_{(S_1\cdot S_2)^*}(x, z') \wedge \mathsf{reach}_{S_1^*}(y', z') \wedge \mathsf{reach}_{S_2^*}(z, z') \\
\psi_\triangle(x, y) &:= \mathsf{reach}_L(x, y) \;\wedge\; \forall z : \big(\mathsf{reach}_L(x, z) \rightarrow \mathsf{reach}_{S_1^*}(z, y)\big)
\end{aligned}
$$

where 0 again is the element origin $(0,0,0)$ of the grid which is FO-definable and

$$L \;:=\; (S_1^{-1}\cdot S_2)^* \cdot \Big((S_2^{-1}\cdot S_1)\cdot(S_2^{-1}\cdot S_3\cdot S_1)^* + (S_3^{-1}\cdot S_1)\cdot(S_3^{-1}\cdot S_2\cdot S_1)^*\Big)^*.$$

The first part of L simply prepares $x = (x_1,0,0)$ by swapping its components to $(0,x_1,0)$. The second part does the actual job of walking through the triangle and counting the hops according to Figure 4.2. The advanced reader might have noticed that the first and second part of L do not perform 0-tests since they are not available in $\mathfrak{N}^{\pm 3}$. We solve this by not just accepting $\mathsf{reach}_L(x,y)$ but by forcing additionally that y is maximal in its first component. With this extra restriction the path described by the first part of L has to result in $(0,x_1,0)$ and the path of the second part may only change its direction on the edges of the grid in order to get the maximal y. Furthermore the language L is star-free as well (without proof). ∎

The fact that Peano arithmetic is undecidable [Göd1931] yields our main result for undecidability:

Corollary 4.12 *The FO(SF)-theory of $\mathfrak{N}^{\pm 3}$ is undecidable.* □

By altering the proof of Lemma 4.11 we can show a similar result. Since we use the third dimension only in reachability expressions and require it to be 0 otherwise we can simulate this dimension by a Petri net with one unbounded place:

Remark 4.13 The FO(PN^1)-theory of $\mathfrak{N}^{\pm 2}$ is undecidable. □

Another variant of the proof of Lemma 4.11 uses a Petri net with two unbounded places to count the blocks. All other operations are done in $\mathfrak{N}^2$ as described in Remark 4.8:

Remark 4.14 The FO(PN^2)-theory of $\mathfrak{N}^2$ is undecidable. □

Next we want to demonstrate that 0-tests make decidability even worse. Proposition 3.6 already stated the undecidability of the MSO-theories of 2-dimensional grids. This can be proven by reducing the halting problem of Turing machines which is one of the first decision problems ever proven to be undecidable [Tur1936]: an MSO-formula is constructed which assigns one configuration of the Turing machine to each row of the grid and then checks if there is a valid sequence of configurations beginning with the start configuration and reaching some accepting configuration.

Let us consider a closely related problem: the halting problem for Minsky machines with two registers to model checking an FO(Reg)-formula over $\mathfrak{N}_0^{\pm 2}$. A *program* of a *Minsky machine* with two registers is a finite list of n instructions. Each content $[\ell]$ of each line ℓ for $1 \le \ell < n$ can be either of the following instructions (where $r \in \{1,2\}$ is one of the two registers and program lines $1 \le \ell', \ell'' \le n$):

- $\mathsf{incr}(r,\ell')$, which increases the value of register r and continues the computation at line ℓ', or

- $\mathsf{decr}(r, \ell', \ell'')$, which decreases the value of register r if it was positive and continues the computation at line ℓ'; otherwise the computation continues at line ℓ''.

The instruction of the last line n is the halt instruction $[n] \coloneqq \mathsf{halt}$.

Minsky machines with two registers are known to be Turing complete [Min1961, Min1967], thus the halting problem is undecidable as well.

Theorem 4.15 *The FO(Reg)-theory of* $\mathfrak{N}_0^{\pm 2}$ *is undecidable.* □

Proof We will reduce the halting problem for Minsky machines with two registers to model checking an FO(Reg)-formula over $\mathfrak{N}_0^{\pm 2}$.

The idea of the proof is that the values of two registers are actually the same as the state space of the 2-dimensional grid. The register operations for increasing and decreasing with 0-test are features of the grid as well. To simulate the lines of the program we will use the states of a finite automaton for a language of a reachability expression. Thus reaching a halt instruction can be reduced to reachability in the grid.

Consider a program as above for a Minsky machine with two registers. We define a deterministic finite automaton $\mathcal{A} = (Q, \Sigma, \delta, q_0, F)$ to simulate the program with:

- state set $Q \coloneqq \{0, \dots, n\}$ for the lines of the program $1, \dots, n$ and 0 for the bad state,
- alphabet $\Sigma \coloneqq \{S_1, S_2, S_1^{-1}, S_2^{-1}, 0_1, 0_2\}$ of the grid structure $\mathfrak{N}_0^{\pm 2}$,
- transition function δ defines as

$$\delta(\ell, a) \coloneqq \begin{cases} \ell' & \text{if } 1 \le \ell < n,\ [\ell] = \mathsf{incr}(r, \ell') \text{ and } a = S_r, \\ \ell' & \text{if } 1 \le \ell < n,\ [\ell] = \mathsf{decr}(r, \ell', \ell'') \text{ and } a = S_r^{-1}, \\ \ell'' & \text{if } 1 \le \ell < n,\ [\ell] = \mathsf{decr}(r, \ell', \ell'') \text{ and } a = 0_r, \\ 0 & \text{otherwise} \end{cases}$$

  Note that each vertex for each dimension r in this grid has either an S_r^{-1}-edge or and 0_r-edge.

- and initial state $q_0 \coloneqq 1$ and final states $F \coloneqq \{n\}$.

Then the regular language $L(\mathcal{A}) \subseteq \{S_1, S_2, S_1^{-1}, S_2^{-1}, 0_1, 0_2\}^*$ of this automaton simulates the computation of the program in the grid, i.e. the program halts iff $\mathfrak{N}_0^{\pm 2} \vDash \exists x : \mathsf{reach}_{L(\mathcal{A})}(0, x)$. ∎

For this result we can get a slight variation too by simulating one dimension of the grid by means of more powerful languages in reachability expressions. Since we need the 0-test we have to use counter languages this time instead of Petri nets languages as in Remark 4.13. The counter simply simulates the simulation of one register in the proof of Theorem 4.15:

Remark 4.16 The FO(CL)-theory of $\mathfrak{N}_0^{\pm 1}$ is undecidable. □

So in the grid $\mathfrak{N}_0^{\pm 1}$ MSO logic is decidable (Remark 4.3) and FO(CL) logic is not. In the grid $\mathfrak{N}^2$ it is exactly the other way around: MSO logic is undecidable (Proposition 3.6) and FO(CL) logic is decidable (Theorem 4.7). Hence we can separate the two logics:

Remark 4.17 The logics FO(CL) and MSO are incomparable, i.e. FO(CL) $\not\leq$ MSO and MSO $\not\leq$ FO(CL). □

A summary of the above decidability results can be found in Table 4.1.

4.3 Transitive Closure

The bad decidability properties of product structures have been studied before. Stefan Wöhrle has done some investigation on this topic in his PhD thesis [Wöh2005]. Among others he gave some results about decidability and undecidability of FO plus transitive closure over the 2-dimensional infinite grid $\mathfrak{N}^2$.

In detail the logic $\mathrm{FO}(\mathrm{TC})_{(k)}$ he studied is basically FO extended by a new operator $\big[TC_{\bar{x},\bar{y}}\,\varphi(\bar{x},\bar{y},\bar{z})\big](\bar{s},\bar{t})$ for the transitive closure of the formula φ where $\bar{x},\bar{y}$ are disjoint ℓ-tuples of variables for some $1 \leq \ell \leq k$, $\bar{z}$ is a tuple of free variables of the TC expression and $\bar{s},\bar{t}$ are ℓ-tuples of terms. This formula holds iff there exists a finite sequence $\bar{s}_0, \bar{s}_1, \ldots, \bar{s}_m$ of ℓ-tuples of variables such that the following holds:

$$(\bar{s} = \bar{s}_0) \;\wedge\; \varphi(\bar{s}_0, \bar{s}_1, \bar{z}) \;\wedge\; \varphi(\bar{s}_1, \bar{s}_2, \bar{z}) \;\wedge\; \ldots \;\wedge\; \varphi(\bar{s}_{m-1}, \bar{s}_m, \bar{z}) \;\wedge\; (\bar{s}_m = \bar{t}).$$

Thus the operator exactly defines the transitive closure of the relation defined by φ. Since the new operator can be very strong we will work with the restricted version $\mathrm{FO}(\mathrm{TC})^n_{(k)}$ where the transitive closure operator may be nested at most n times.

When restricting to $\mathrm{FO}(\mathrm{TC})_{(1)}$ we have a logic between FO and MSO again since the transitive closure of formulas describing binary relations can be expressed in MSO. An inclusion diagram like in Remark 2.11 for infinite grids can be extended by $\mathrm{FO}(\mathrm{TC})_{(1)}$ and restricted variants to the following (without proof):

Remark 4.18 Logic hierarchy (on infinite grid structures):

$$\mathrm{FO} \;\leq\; \mathrm{FO(R)} \;\leq\; \left\{ \begin{array}{c} \mathrm{FO(SF)} \;\leq\; \mathrm{FO(Reg)} \\ \mathrm{FO(TC)}^1_{(1)} \end{array} \right\} \;\leq\; \left\{ \begin{array}{l} \mathrm{FO(PN^1)} \;\leq\; \left\{ \begin{array}{l} \mathrm{FO(CL)} \;\leq\; \mathrm{FO(CF)} \\ \mathrm{FO(PN^2)} \;\leq\; \mathrm{FO(PN)} \end{array} \right. \\ \mathrm{FO(TC)}^2_{(1)} \;\leq\; \mathrm{FO(TC)}_{(1)} \;\leq\; \mathrm{MSO} \end{array} \right.$$ □

Wöhrle studied FO logic with transitive closure (cf. [Wöh2005]) and came up with a positive and a negative decidability result for the 2-dimensional infinite grid:

Proposition 4.19 ([Wöh2005, Theorem 4.1]) *The $FO(TC)^2_{(1)}$-theory of the 2-dimensional infinite grid $\mathfrak{N}^2$ is undecidable.* □

Proposition 4.20 ([Wöh2005, Theorem 4.4]) *The $FO(TC)^1_{(1)}$-theory of the 2-dimensional infinite grid $\mathfrak{N}^2$ is decidable.* □

1-way grids		
$\mathfrak{N}^1$	$\mathbf{FO(R)} \leq \mathbf{FO(SF)} \leq \mathbf{FO(Reg)} \leq$	$\left\{ \begin{array}{l} \mathbf{FO(PN^1)} \leq \left\{ \begin{array}{l} \mathbf{FO(CL)} \leq \mathbf{FO(CF)} \\ \mathbf{FO(PN^2)} \leq \mathbf{FO(PN)} \end{array} \right. \\ \mathbf{MSO} \end{array} \right.$
$\mathfrak{N}^2$	$\mathbf{FO(R)} \leq \mathbf{FO(SF)} \leq \mathbf{FO(Reg)} \leq$	$\left\{ \begin{array}{l} \mathbf{FO(PN^1)} \leq \left\{ \begin{array}{l} \mathbf{FO(CL)} \leq \mathbf{FO(CF)} \\ \mathbf{FO(PN^2)} \leq \mathbf{FO(PN)} \end{array} \right. \\ \mathbf{MSO} \end{array} \right.$
...		...
1-way grids with 0-test		
$\mathfrak{N}_0^1$	$\mathbf{FO(R)} \leq \mathbf{FO(SF)} \leq \mathbf{FO(Reg)} \leq$	$\left\{ \begin{array}{l} \mathbf{FO(PN^1)} \leq \left\{ \begin{array}{l} \mathbf{FO(CL)} \leq \mathbf{FO(CF)} \\ \mathbf{FO(PN^2)} \leq \mathbf{FO(PN)} \end{array} \right. \\ \mathbf{MSO} \end{array} \right.$
$\mathfrak{N}_0^2$	$\mathbf{FO(R)} \leq \mathbf{FO(SF)} \leq \mathbf{FO(Reg)} \leq$	$\left\{ \begin{array}{l} \mathbf{FO(PN^1)} \leq \left\{ \begin{array}{l} \mathbf{FO(CL)} \leq \mathbf{FO(CF)} \\ \mathbf{FO(PN^2)} \leq \mathbf{FO(PN)} \end{array} \right. \\ \mathbf{MSO} \end{array} \right.$
...		...
2-way grids		
$\mathfrak{N}^{\pm 1}$	$\mathbf{FO(R)} \leq \mathbf{FO(SF)} \leq \mathbf{FO(Reg)} \leq$	$\left\{ \begin{array}{l} \mathbf{FO(PN^1)} \leq \left\{ \begin{array}{l} \mathbf{FO(CL)} \leq \mathbf{FO(CF)} \\ \mathbf{FO(PN^2)} \leq \mathbf{FO(PN)} \end{array} \right. \\ \mathbf{MSO} \end{array} \right.$
$\mathfrak{N}^{\pm 2}$	$\mathbf{FO(R)} \leq \mathbf{FO(SF)} \leq \mathbf{FO(Reg)} \leq$	$\left\{ \begin{array}{l} \mathbf{FO(PN^1)} \leq \left\{ \begin{array}{l} \mathbf{FO(CL)} \leq \mathbf{FO(CF)} \\ \mathbf{FO(PN^2)} \leq \mathbf{FO(PN)} \end{array} \right. \\ \mathbf{MSO} \end{array} \right.$
$\mathfrak{N}^{\pm 3}$	$\mathbf{FO(R)} \leq \mathbf{FO(SF)} \leq \mathbf{FO(Reg)} \leq$	$\left\{ \begin{array}{l} \mathbf{FO(PN^1)} \leq \left\{ \begin{array}{l} \mathbf{FO(CL)} \leq \mathbf{FO(CF)} \\ \mathbf{FO(PN^2)} \leq \mathbf{FO(PN)} \end{array} \right. \\ \mathbf{MSO} \end{array} \right.$
...		...
2-way grids with 0-test		
$\mathfrak{N}_0^{\pm 1}$	$\mathbf{FO(R)} \leq \mathbf{FO(SF)} \leq \mathbf{FO(Reg)} \leq$	$\left\{ \begin{array}{l} \mathbf{FO(PN^1)} \leq \left\{ \begin{array}{l} \mathbf{FO(CL)} \leq \mathbf{FO(CF)} \\ \mathbf{FO(PN^2)} \leq \mathbf{FO(PN)} \end{array} \right. \\ \mathbf{MSO} \end{array} \right.$
$\mathfrak{N}_0^{\pm 2}$	$\mathbf{FO(R)} \leq \mathbf{FO(SF)} \leq \mathbf{FO(Reg)} \leq$	$\left\{ \begin{array}{l} \mathbf{FO(PN^1)} \leq \left\{ \begin{array}{l} \mathbf{FO(CL)} \leq \mathbf{FO(CF)} \\ \mathbf{FO(PN^2)} \leq \mathbf{FO(PN)} \end{array} \right. \\ \mathbf{MSO} \end{array} \right.$
$\mathfrak{N}_0^{\pm 3}$	$\mathbf{FO(R)} \leq \mathbf{FO(SF)} \leq \mathbf{FO(Reg)} \leq$	$\left\{ \begin{array}{l} \mathbf{FO(PN^1)} \leq \left\{ \begin{array}{l} \mathbf{FO(CL)} \leq \mathbf{FO(CF)} \\ \mathbf{FO(PN^2)} \leq \mathbf{FO(PN)} \end{array} \right. \\ \mathbf{MSO} \end{array} \right.$
...		...

Color legend: **decidable**, **unknown** and **undecidable**.

Table 4.1: Summary of decidability of logics in grid structures.

We will add another negative result concerning the 3-dimensional grid.

Lemma 4.21 *The FO(Reg)-theory of $\mathfrak{N}_0^{\pm n}$ can be reduced to the FO(TC)$^1_{(1)}$-theory of $\mathfrak{N}^{n+1}$.* □

Proof The idea is to simulate reachability expressions by transitive closure operations. The additional dimension of the target grid is needed to keep track of the states of an automaton for the reachability language.

Thus we will give a 1-tuple-FO(TC)$^1_{(1)}$-interpretation with regular reachability of $\mathfrak{N}_0^{\pm n}$ in $\mathfrak{N}^{n+1}$. Tuple-interpretation has actually not yet been formally defined for FO(TC) but we will skip this since it is defined straight forward as for MSO or FO with reachability. The interpretation is as follows:

$$\begin{aligned}
\psi_{\mathrm{dom}}(x) &:= \neg\exists y S_{n+1}(y,x)\\
\psi_{0_i}(x) &:= \neg\exists y : S_i(y,x)\\
\psi_{S_i}(x,y) &:= S_i(x,y)\\
\psi_{S_i^{-1}}(x,y) &:= S_i(y,x)\\
\psi^{L(\mathcal{A})}_{\mathsf{reach}}(x,y) &:= \exists x',y'\Big(\vartheta^{\mathcal{A}}_I(x,x') \wedge \big[TC_{u,v}\,\vartheta^{\mathcal{A}}_\Delta(u,v)\big](x',y') \wedge \vartheta^{\mathcal{A}}_F(y,y')\Big)
\end{aligned}$$

where for some given finite automaton $\mathcal{A} = (Q,\Sigma,\Delta,I,F)$ with states $Q = \{0,\dots,m\}$ and alphabet $\Sigma = \{S_1,\dots,S_n,S_1^{-1},\dots,S_n^{-1},0_1,\dots,0_n\}$ the following formulas are definable in FO(TC)$^1_{(1)}$:

- $\vartheta^{\mathcal{A}}_I(x,x')$, defining that $x = (x_1,\dots,x_n,0)$ and $x' = (x_1,\dots,x_n,q)$ for some $q \in I$,
- $\vartheta^{\mathcal{A}}_F(y,y')$, defining that $y = (y_1,\dots,y_n,0)$ and $y' = (y_1,\dots,y_n,q)$ for some $q \in F$,
- $\vartheta^{\mathcal{A}}_\Delta(x,y)$, defining that there is $(q,a,q') \in \Delta$ and $x = (x_1,\dots,x_n,q), y = (y_1,\dots,y_n,q')$ such that:
  - if $a = S_i$ then $x_i + 1 = y_i$ and $x_j = y_j$ for all $j \neq i$,
  - if $a = S_i^{-1}$ then $x_i = y_i + 1$ and $x_j = y_j$ for all $j \neq i$,
  - if 0_i then $x_i = 0$ and $x_j = y_j$ for all j. ∎

Analogous to Theorem 2.17 the decidability of the interpreting structure can be transferred to the interpreted one. Thus we can combine Lemma 4.21 with Theorem 4.15 to state the following result:

Corollary 4.22 *The FO(TC)$^1_{(1)}$-theory of the 3-dimensional infinite grid $\mathfrak{N}^3$ is undecidable.* □

This is a very interesting consequence since it is the first result for infinite grids in this work demonstrating a difference in decidability for dimension 2 and 3. Of course it is not as big as between dimension 1 and 2, but anyhow one might not expect that difference for higher dimensions intuitively.

4.4 Blind Grids

In a last investigation on grids we want to give a result about a less powerful variant. The so called *blind grids* are products of the natural numbers with successor $\mathfrak{N} = (\mathbb{N}, S)$ again but the successor relations of the different dimensions cannot be distinguished here. Thus we end up with a structure $\mathfrak{N}^{(n)} := (\mathbb{N}^n, S)$ which is similar to $\mathfrak{N}^n = \left(\mathbb{N}^n, (S_i)_{1 \le i \le n}\right)$ with $S := \bigcup_i S_i$.

It is easy to see that blind grids $\mathfrak{N}^{(n)}$ can be interpreted into their non-blind counterparts $\mathfrak{N}^n$. We use 1-tuple-$\mathfrak{L}$-interpretation with reachability in case of logic FO($\mathcal{L}$) for any language class $\mathcal{L}$; for MSO we simply use MSO-interpretation. Both types of interpretation look as follows:

$$\begin{aligned} \psi_{\mathrm{dom}}(x) &:= \mathsf{true} \\ \psi_S(x) &:= \bigvee_{1 \le i \le n} S_i(x,y) \\ \psi^L_{\mathsf{reach}}(x,y) &:= \mathsf{reach}_L(x,y) \qquad \text{(not for MSO-interpretation)} \end{aligned}$$

Remark 4.23 The $\mathfrak{L}$-theory of $\mathfrak{N}^{(n)}$ can be reduced to the $\mathfrak{L}$-theory of $\mathfrak{N}^n$ for all $n \ge 1$ and logic $\mathfrak{L}$ being MSO or FO($\mathcal{L}$) for some language class $\mathcal{L}$. □

Now the decidability result of Theorem 4.7 can be applied for blind grids as well:

Corollary 4.24 *The FO(CF)-theory of $\mathfrak{N}^{(n)}$ is decidable for all $n \ge 1$.* □

The more interesting question is how undecidable these grids are. We can answer this question for the MSO-theory.

Lemma 4.25 *The MSO-theory of $\mathfrak{N}^{(n)} \times \mathfrak{N} = (\mathbb{N}^n \times \mathbb{N}, S_{\mathfrak{N}^{(n)}}, S_{\mathfrak{N}})$ can be reduced to the MSO-theory of $\mathfrak{N}^{(n+1)}$.* □

Proof We will show how to uniquely describe one dimension of the blind grid $\mathfrak{N}^{(n+1)}$. This dimension will form the $\mathfrak{N}$-component while the remaining n dimensions form the $\mathfrak{N}^{(n)}$-component.

First we have to characterize one dimension. This cannot be done by interpretation since all dimensions behave in the same way. The formula

$$\psi_1(X) := \exists 0, 1 : \Big(S(0,1) \wedge X(1) \wedge \forall y : \neg S(y,0) \wedge \big(X(y) \to y = 1 \big) \Big)$$

defines up to isomorphism a relation just containing one element $(0, \ldots, 0, 1)$ as the successor of the grids origin in the last dimension. We actually cannot guarantee that really the successor of the last dimension was chosen. But the dimension cannot be distinguished, thus it is all isomorphic. Since the relation 1 contains just that single element we will identify it with a constant 1. Up to now Theorem 2.15 allows us to reduce the MSO-theory of $(\mathbb{N}^{n+1}, 1, S)$ to the one of $\mathfrak{N}^{(n+1)} = (\mathbb{N}^{n+1}, S)$.

In a second step we MSO-interpret $\mathfrak{N}^{(n)} \times \mathfrak{N} = (\mathbb{N}^n \times \mathbb{N}, S_{\mathfrak{N}^{(n)}}, S_{\mathfrak{N}})$ in our structure $(\mathbb{N}^{n+1}, 1, S)$ just created. The interpretation looks as follows:

$$\begin{aligned}
\psi_{\mathrm{dom}}(x) &:= \exists x_0 : \big(\vartheta^*_{\mathrm{diag}}(x_0, x) \wedge \neg \mathsf{reach}(1, x_0)\big) \\
\psi_{S_{\mathfrak{N}}}(x, y) &:= \vartheta_{\mathrm{diag}}(x, y) \\
\psi_{S_{\mathfrak{N}^{(n)}}}(x, y) &:= S(x, y) \wedge \exists x_0, y_0 : \big(\vartheta^*_{\mathrm{diag}}(x_0, x) \wedge \vartheta^*_{\mathrm{diag}}(y_0, y) \wedge S(x_0, y_0)\big)
\end{aligned}$$

where $\vartheta_{\mathrm{diag}}(x, y) := \forall z : \big(S(x, z) \to S(z, y)\big)$ specifies that y is the direct diagonal successor of x and $\vartheta^*_{\mathrm{diag}}(x_0, x)$ states that the diagonal through x starts in x_0 on the border:

$$\begin{aligned}
\vartheta^*_{\mathrm{diag}}(x_0, x) &:= \neg\exists y : \vartheta_{\mathrm{diag}}(y, x_0) \wedge \\
&\quad \forall D : \Big(D(x_0) \wedge \big(\forall d, d' : D(d) \wedge \vartheta_{\mathrm{diag}}(d, d') \to D(d')\big) \to D(x)\Big).
\end{aligned}$$

By this interpretation we restrict the domain to grid elements $(x_1, \ldots, x_n, x_{n+1})$ such that no entry is smaller than x_{n+1} by forcing that the diagonal through each grid position starts in some $x_0 = (x_{0,1}, \ldots, x_{0,n}, 0)$ which simply means that they have to be below the main diagonal when viewed from the last dimension. The successor relation $S_{\mathfrak{N}}$ for the $\mathfrak{N}$-component is simply mapped to the diagonal successor which is unique:

$$(\mathbb{N}^{n+1}, 1, S) \vDash \psi_{S_{\mathfrak{N}}}(\bar{x}, \bar{y}) \quad :\Leftrightarrow \quad \forall i : \; x_i + 1 = y_i.$$

For the successor relation $S_{\mathfrak{N}^{(n)}}$ of the $\mathfrak{N}^{(n)}$-component we want to allow the successor in any dimension but the last one. We force this by checking that $y = (y_1, \ldots, y_{n+1})$ is a successor of $x = (x_1, \ldots, x_{n+1})$ such that the starting point $y_0 = (y_1 - y_{n+1}, \ldots, y_n - y_{n+1}, 0)$ of the main diagonal through y is a successor of the starting point $x_0 = (x_1 - x_{n+1}, \ldots, x_n - x_{n+1}, 0)$ of the main diagonal through x, i.e.:

$$(\mathbb{N}^{n+1}, 1, S) \vDash \psi_{S_{\mathfrak{N}^{(n)}}}(\bar{x}, \bar{y}) \quad :\Leftrightarrow \quad \exists i \leq n : \; x_i + 1 = y_i \;\wedge\; \forall j \neq i : \; x_j = y_j.$$

If y would be the successor of x in the last dimension, i.e. that $x_i = y_i$ for all $i \leq n$ and $x_{n+1} + 1 = y_{n+1}$, then y_0 is a successor of x_0 iff x_{n+1} is the largest number in x. But this case can never occur since y_{n+1} would be larger than any other number in y which means that y is not part of the domain according to $\psi_{\mathrm{dom}}(y)$.

The lemma now follows by Remark 2.12. ∎

Lemma 4.26 *The MSO-theory of $\mathfrak{N}^n$ can be reduced to the MSO-theory of $\mathfrak{N}^{(n)}$.* □

Proof Follows Lemma 4.25 by induction. For the base case $n = 1$ both structures are isomorphic: $\mathfrak{N}^1 = \mathfrak{N}^{(1)}$. In the induction step we can reduce the MSO-theory of $\mathfrak{N}^{(n)} \times \mathfrak{N}$ to the one of $\mathfrak{N}^{(n+1)}$ by Lemma 4.25. By induction hypotheses the MSO-theory of $\mathfrak{N}^n$ can be reduced to the one of the first component $\mathfrak{N}^{(n)}$. Hence in total the MSO-theory of $\mathfrak{N}^n \times \mathfrak{N}$ which is isomorphic to $\mathfrak{N}^{n+1}$ can be reduced to the one of $\mathfrak{N}^{(n+1)}$. ∎

We now can reduce the MSO-theory of $\mathfrak{N}^{(2)}$ to the one of $\mathfrak{N}^2$ which is known to be undecidable by Proposition 3.6:

Corollary 4.27 *The MSO-theory of the 2-dimensional blind grid $\mathfrak{N}^{(2)}$ is undecidable.* □

5 Ground Term Rewriting

Let us start with an informal introduction to term rewriting. We are working with *terms* over function symbols Σ, each symbol $f \in \Sigma$ having some arity $|f| \in \mathbb{N}$, and *variables* V, where each variable has rank 0. Thus terms can be viewed as trees over a ranked alphabet $\Sigma \dot\cup V$ and the set of all finite terms over functions Σ and variables V is the same as $T^{\mathsf{fin}}_{\Sigma \dot\cup V}$: the set of finite trees over that ranked alphabet. A term or tree in $T^{\mathsf{fin}}_{\Sigma} = T^{\mathsf{fin}}_{\Sigma \dot\cup \varnothing}$ has no variables and is called *ground term* or *ground tree*.

In general a *term rewriting system* $(T, \Sigma, V, A, \to)$ consists of a set $T \subseteq T^{\mathsf{fin}}_{\Sigma}$ of terms over the ranked alphabet Σ. Furthermore there are relations defined between those terms by finitely many *rewriting rules* $l \xrightarrow{a} r$, each labeled by some *action label* $a \in A$. In this context $l, r \in T^{\mathsf{fin}}_{\Sigma \dot\cup V}$ are terms with variables V such that all of the variables of r already occur in l. Given some term, such a rules allow one to replace any sub-term that matches the pattern l by r where ground terms are bound to variables and are substituted for the variables in r. The rewriting will be obvious after considering Example 5.1.

We are dealing with graph structures described by a term rewriting system, which in case of the above rewriting system is the structure $\big(T, (E_a)_{a \in A}\big)$ of Σ-terms without variables. They are connected according to the rewriting rules, i.e. there is an edge $(t, t') \in E_a$ iff there exists a rewriting rule labeled by $a \in A$ which reduces term t to t' as described above. Those graphs are called *term rewriting graphs*.

Example 5.1 (Infinite grid) Let us consider the term rewriting system with ranked alphabet $\Sigma = \{2, 1_L, 1_R, 0\}$, each symbol having a rank according to its number, variables $V := \{X\}$ and a domain set of terms that are reachable from $2\big(1_L(0), 1_R(0)\big)$. For the relations let $A = \{S_1, S_2\}$ be the set of labels and use the rewriting rules $1_L(X) \xrightarrow{S_1} 1_L(1_L(X))$ and $1_R(X) \xrightarrow{S_2} 1_R(1_R(X))$. The graph of this term rewriting system is depicted in Figure 5.1(a) and isomorphic to the infinite grid $\mathfrak{N}^2$. □

In general term rewriting can result in undecidable logical theories easily (even for a restricted variant of term rewriting):

Proposition 5.2 ([Tre1998]) *The FO-theory of unlabeled linear term rewriting graphs is undecidable.* □

Root Term Rewriting Since it is our point of interest to study FO logic enriched by some reachability predicates we have to find more restricted variants in order to get positive results. Let us examine *root term rewriting*, which is term rewriting where it is only allowed to rewrite the root of the term.

Example 5.3 (Infinite grid) Consider the root term rewriting system with ranked alphabet $\Sigma = \{2, 1, 0\}$, each symbol having the rank according to its number, variables $V := \{X, Y\}$, labels $A = \{S_1, S_2\}$ and rewriting rules $2(X, Y) \xrightarrow{S_1} 2(1(X), Y)$ and $2(X, Y) \xrightarrow{S_2} 2(X, 1(Y))$ over the set of terms that are reachable from $2(0, 0)$. The graph of this root term rewriting system is depicted in Figure 5.1(b) and isomorphic to the infinite grid $\mathfrak{N}^2$ again. □

But even this approach quickly leads to a negative result for the desired logic:

Lemma 5.4 *The FO(R)-theory of root term rewriting graphs is undecidable.* □

Proof We can easily show this by reducing the undecidable halting problem over Turing machines [Tur1936] to the check of a simple reachability expression in a (linear and unlabeled) root term rewriting graph. The idea is to encode the configurations of a given Turing machine M as terms $q(l, r)$, where q is a state of M and l, r are lists representing the tape content left and right of the position of the head in these configurations. The rewriting rules transform one configuration into the successor configuration. Now it is obvious that the Turing machine M halts on an empty tape iff an accepting configuration is reachable from the start configuration. ■

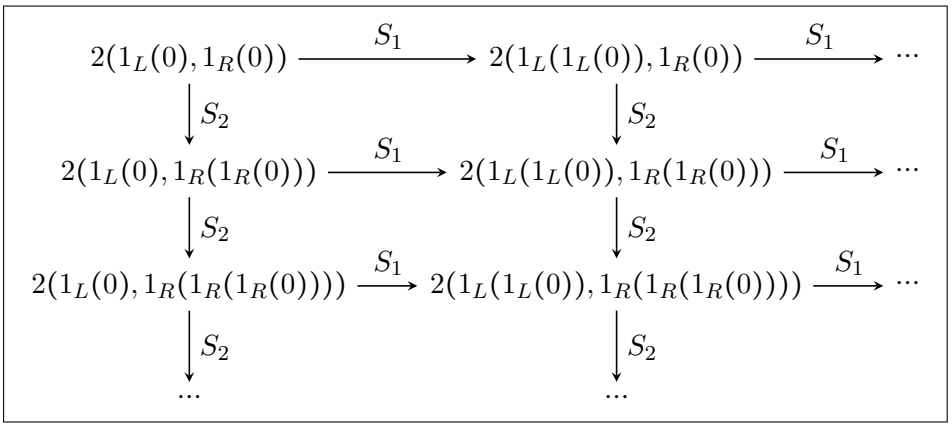
Ground Term Rewriting It is surprising that term rewriting becomes a lot more decidable when restricting it to the rewriting of ground terms instead of the root term: in a *ground term rewriting* (or *GTR* for short) system $(T, \Sigma, A, \rightarrow)$ the terms l, r of the left and right side of each rewriting rule $l \xrightarrow{a} r$ must not have variables, i.e. $V = \varnothing$ and $l, r \in T_\Sigma^{\mathsf{fin}}$. With ground term rewriting we can model our old friend (see Example 5.1) as well:

Example 5.5 (Infinite grid) Consider the ground term rewriting system $(T, \Sigma, A, \rightarrow)$ with ranked alphabet $\Sigma = \{2, 1, 0_L, 0_R\}$, each symbol having the arity according to its number, labels $A = \{S_1, S_2\}$ and rewriting rules $0_L \xrightarrow{S_1} 1(0_L)$, $0_R \xrightarrow{S_2} 1(0_R)$ over the set of terms that are reachable from $2(0_L, 0_R)$. The graph of this ground term rewriting system is depicted in Figure 5.1(c) and isomorphic to the infinite grid $\mathfrak{N}^2$ as well. □

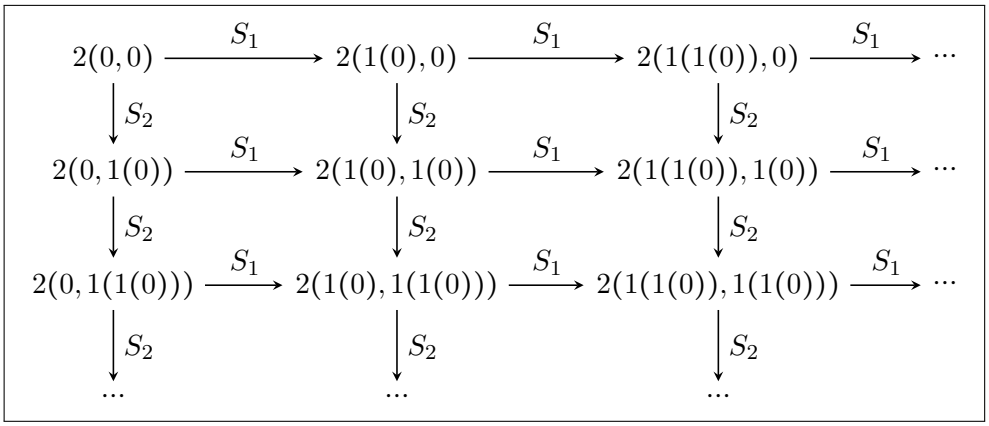
With the ideas of this example it is furthermore possible to specify all grids from Chapter 4 by ground term rewriting including the 3-dimensional 2-way grid $\mathfrak{N}^{\pm 3}$. For this very grid we know that the FO(R)-theory is the strongest theory that is still decidable (Remark 4.4) while the next stronger theory of FO(SF) is undecidable (Corollary 4.12). Nevertheless there is a relatively strong decidability results known for GTR graphs:

Proposition 5.6 ([DT1990]) *Consider a GTR system with a domain consisting of terms reachable from some start term. The FO(R)-theory of its graph is decidable.* □

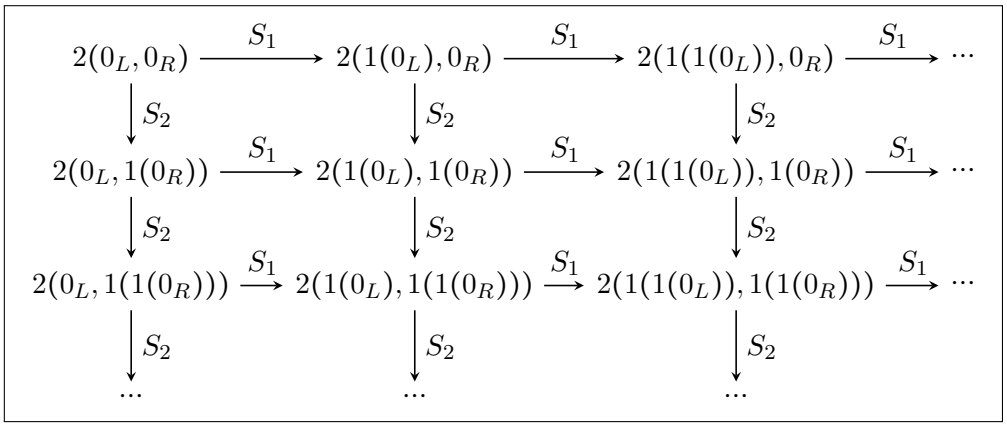
In this sense GTR seems to be a very sharp result since the gap between the decidable FO(R)-theory and the lowest logic which is undecidable can be very small as in the case of $\mathfrak{N}^{\pm 3}$.



(a)



(b)



(c)

Figure 5.1: Infinite grid $\mathfrak{N}^2$ as term rewriting graph: (a) by plain term rewriting, see Example 5.1; (b) by root term rewriting, see Example 5.3; (c) by ground term rewriting, see Example 5.5.

Regular Ground Term Rewriting The proof of Proposition 5.6 according to [DT1990] represents terms as trees and rewriting is realized by tree-transducers and tree-automatic relations. Since the terms of a rewriting rule are recognized by finite tree-automata we can easily extended this proof to allow regular sets of terms $L, R \subseteq T_\Sigma^{\mathsf{fin}}$ in rewriting rules instead of single ground terms l, r. A rule $L \xrightarrow{a} R$ with regular sets L, R of trees can be understood as an abbreviation for (possibly infinitely many) rules $l \xrightarrow{a} r$ for each $l \in L$ and $r \in R$. Thus by finitely many regular rewriting rules we can actually represent infinitely many rewriting rules of the old format. This extension of GTR is called *regular ground term rewriting* (or *RGTR* for short).

Remark 5.7 Consider a RGTR system with a domain consisting of terms reachable from some start term. The FO(R)-theory of its graph is decidable. □

5.1 Ground Term Expansion

In this first approach we want to restrict ground term rewriting in such a way that we end up with a model which has better decidability. Since infinite grids, as studied in Chapter 4, can be represented by ground term rewriting but do not all have a decidable FO(Reg)-theory, the GTR variant introduced tries to cover some of the grids that can handle FO(Reg).

Definition 5.8 *Ground term expansion* is ground term rewriting restricted to rules $l \to r$ such that l is a constant, i.e. $l = c \in \Sigma$ with arity $|c| = 0$. □

Example 5.9 (Infinite grid with 0-test) Consider the ground term expansion system with ranked alphabet $\Sigma = \{2, 1, 0_{L0}, 0_{R0}, 0_L, 0_R\}$, each symbol having the arity according to its number, labels $A = \{0_1, 0_2, S_1, S_2\}$ rewriting rules $0_{L0} \xrightarrow{0_1} 0_{L0}$, $0_{R0} \xrightarrow{0_2} 0_{R0}$, $0_{L0} \xrightarrow{S_1} 1(0_L)$, $0_L \xrightarrow{S_1} 1(0_L)$, $0_{R0} \xrightarrow{S_2} 1(0_R)$, $0_R \xrightarrow{S_2} 1(0_R)$ over the set of terms that are reachable from $2(0_{L0}, 0_{R0})$. Then its graph is isomorphic to the infinite grid $\mathfrak{N}_0^2$.

Actually any 1-way grid $\mathfrak{N}^n$, $\mathfrak{N}^{(n)}$, $\mathfrak{N}_0^n$, $\mathfrak{N}_0^{(n)}$ can be generated this way. □

Example 5.10 (Binary trees) Consider the ground term expansion system with ranked alphabet $\Sigma = \{2, 1_L, 1_R, 0_1, 0_2\}$, each symbol having the arity according to its number, labels $A = \{L_0, R_0, L_1, R_1\}$ and rewriting rules $0_i \xrightarrow{L_i} 1_L(0_i)$, $0_i \xrightarrow{R_i} 1_R(0_i)$ for $i \in \{0, 1\}$ over the set of terms that are reachable from $2(0_1, 0_2)$. Then its graph is isomorphic to the product $\mathcal{T}_2 \times \mathcal{T}_2$ of the binary tree $\mathcal{T}_2$. □

Unfortunately this product leads to the undecidability result that we actually hoped to avoid:

Lemma 5.11 *The FO(Reg)-theory of $\mathcal{T}_2 \times \mathcal{T}_2$ is not decidable.* □

Proof Given an instance $(u_1, v_1), \ldots, (u_n, v_n) \in (\Sigma^*)^2$ of the Post correspondence problem [Pos1946] we will define an FO(Reg)-formula which is satisfied in $\mathcal{T}_2 \times \mathcal{T}_2 = (\mathbb{B}^* \times$

$\mathbb{B}^*, L_0, R_0, L_1, R_1$) iff the instance has a solution, i.e. there exist $k_1, \ldots, k_m \in \{1, \ldots, n\}$ such that $u_{k_1} \cdot \ldots \cdot u_{k_m} = v_{k_1} \cdot \ldots \cdot v_{k_m}$.

The first step is to uniquely encode the alphabet Σ as words of length $|\Sigma|$ over the binary alphabet $\{L, R\}$. Now we obtain encoded words $u_i', v_i' \in \{L, R\}^*$ by replacing the Σ-symbols in u_i, v_i by their encoded words. Furthermore in u_i' we replace each symbol L by L_0 and R by R_0 and obtain the word $u_i'' \in \{L_0, R_0\}^*$. Analogously we replace in v_i' the symbols L, R by L_1, R_1 and obtain the word $v_i'' \in \{L_1, R_1\}^*$.

The instance has a solution iff over $\mathcal{T}_2 \times \mathcal{T}_2$ the formula

$$\varphi \quad := \quad \exists x : \Big(reach_{(\bigcup_{1 \le i \le n} u_i'' \cdot v_i'')^*}(0, x) \ \wedge \ reach_{(L_0 \cdot L_1 + R_0 \cdot R_1)^*}(0, x) \Big)$$

is satisfied where 0 is an abbreviation for the origin (product of the roots of the trees) which is FO-definable.

The idea is that each sequence of u_i'' corresponds to a path in the first tree and the v_i''-sequences correspond to paths in the other tree respectively. With this formula we ask for the existence of sequences in both trees such that u_i'' and v_i'' are taken in parallel while the second part simply checks if both sequences are the same. They are the same iff the sequence of the according u_i yields the same word as the sequence of v_i. ∎

However, Lemma 5.11 shows that even products of very simple structures like the binary tree, which is the configuration graph of a basic process algebra [BW1990, May2000] and has a decidable MSO-theory, can have an undecidable FO(Reg)-theory:

Remark 5.12 The FO(Reg)-theory of ground term expansion graphs can be undecidable. □

5.2 VRP-Equations

Finding restrictions of ground term rewriting that do have a decidable FO(Reg)-theory on their graphs actually turned out to be not that easy. The additional expressive power compared to FO(R) is probably too strong. But there is another aspect which has not been considered up to now: finding more general variants of (regular) ground term rewriting that still come up with a decidable FO(R)-theory.

In the research of Thomas Colcombet a differently motivated way of describing graph structures similar to those of RGTR was considered. This formalism is called *vertex replacement with product* or *VRP* for short [Col2002].

The idea is to describe graphs by equations with graph operations. The VRP-operations are dealing with *colored graphs*, i.e. structures $G = \big(V, E, \lambda\big)$ with a possibly infinite set V of nodes, edge relations $E \subseteq V \times A \times V$ for a finite set A of *action labels* again and a *coloring function* $\lambda : V \to C$ for a finite set C of *colors*. Let us fix some sets A and C. The five VRP-operators on colored graphs are defined as:

- *Single vertex constant*: $\dot{c} \ := \ \big(\{0\}, \varnothing, \{0 \mapsto c\}\big)$ for some color $c \in C$,
- *Recoloring*: $[\phi]\big(V, E, \lambda\big) \ := \ \big(V, E, \phi \circ \lambda\big)$ for some recolor function $\phi : C \to C$,

- *Adding edges*: $[c \overset{a}{\bowtie} d]\big(V, E, \lambda\big) := \big(V, E', \lambda\big)$ for some label $a \in A$ and colors $c, d \in C$, where $E' = E \cup \Big(\lambda^{-1}(c) \times \{a\} \times \lambda^{-1}(d)\Big)$.

- *Disjoint union*: $\big(V_0, E_0, \lambda_0\big) \oplus \big(V_1, E_1, \lambda_1\big) := \big(V, E, \lambda\big)$
$$\text{where } \begin{cases} V = V_0 \times \{0\} \ \cup \ V_1 \times \{1\} \\ E = \Big\{ \big((u,i), a, (v,i)\big) \;\Big|\; (u,a,v) \in E_i, \ i \in \{0,1\} \Big\} \\ \lambda\big((v,i)\big) = \lambda_i(v) \end{cases}$$

- *Asynchronous product*: $\big(V_0, E_0, \lambda_0\big) \otimes_\eta \big(V_1, E_1, \lambda_1\big) := \big(V, E, \lambda\big)$ for $\eta : C^2 \to C$,
$$\text{where } \begin{cases} V = V_0 \times V_1 \\ E = \Big\{ \big((v_0, v_1), a, (v_0', v_1)\big) \;\Big|\; (v_0, a, v_0') \in E_0, \ v_1 \in V_1 \Big\} \\ \qquad \cup \Big\{ \big((v_0, v_1), a, (v_0, v_1')\big) \;\Big|\; v_0 \in V_0, \ (v_1, a, v_1') \in E_1 \Big\} \\ \lambda\big((v_0, v_1)\big) = \eta\big(\lambda_0(v_0), \lambda_n(v_1)\big) \end{cases}$$

Furthermore we introduce the natural subgraph relation $\subseteq$: $\big(V_0, E_0, \lambda_0\big) \subseteq \big(V_1, E_1, \lambda_1\big)$ iff $V_0 \subseteq V_1$, $E_0 \subseteq E_1$ and $\lambda_0 = \lambda_1|_{V_0}$. Bearing in mind that this relation leads to a *complete partial order* (cpo) with the empty graph as least element we can finally define the interpretation of a VRP-equation system: it simply is the least fixed point.

There are a lot of ways to give the description of an VRP-equation system. The one we prefer is by *VRP-trees* as the unfolding of the equations. This yields possibly infinite trees over the ranked alphabet $\Omega_{A,C}$ of VRP-operators: $\dot{c}$ has arity 0, $[\phi]$ and $[c \overset{a}{\bowtie} d]$ arity 1, $\oplus$ and $\otimes_\eta$ arity 2 (for some $a \in A$, $c, d \in C$, $\phi : C \to C$, $\eta : C^2 \to C$). Note that the alphabet $\Omega_{A,C}$ is finite since so are the sets A and C. The *VRP-interpretation* of such a tree $T_{\Omega_{A,C}}$ is the graph defined by the root operator.

Example 5.13 (Infinite grid) Let $A = \{S_1, S_2\}$, $C = \{0, 1, 2\}$. The interpretation of G in the following VRP-equations is isomorphic to $\mathfrak{N}^2$ when ignoring the colors:

$$G := G_0 \otimes_{[(c,d)\mapsto 0]} G_1, \qquad G_i := [0 \overset{S_i}{\bowtie} 1]\left(\dot{0} \oplus \left[\begin{smallmatrix} 0\mapsto 1 \\ 1\mapsto 2 \\ 2\mapsto 2 \end{smallmatrix}\right] G_i\right) \text{ for } i \in \{0,1\}.$$

The VRP-tree for G is depicted in Figure 5.2. □

The VRP-tree of this example is of a special kind since it is infinite but can be represented finitely: it is a regular tree. The regular trees are exactly the unfoldings of finite graphs or of finite equation systems which is more interesting in our case. For those types of VRP-trees is the following result:

Proposition 5.14 ([Col2002]) *The interpretations of regular VRP-trees are equivalent to graphs of RGTR systems with regular sets of terms as domains up to isomorphism and color removal. This equivalence is effective.* □

We connect this with another result of Colcombet about the decidability of FO(R) logic in those graph structures:

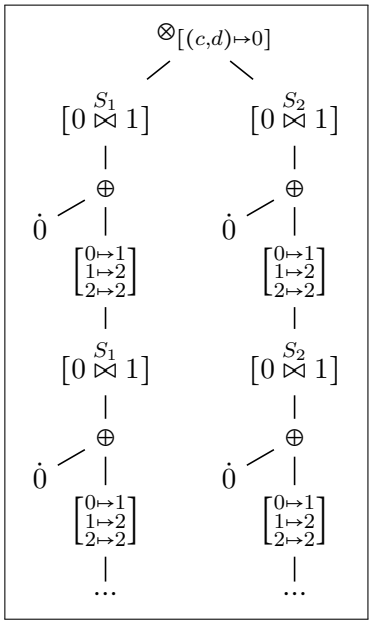


Figure 5.2: VRP-tree defining the infinite grid $\mathfrak{N}^2$ (see Example 5.13).

Proposition 5.15 ([Col2002]) *The FO(R)-theory of the VRP-interpretation of a VRP-tree with decidable MSO-theory is decidable.* □

Since regular trees have a decidable MSO-theory, both propositions together yield a decidability result for RGTR which is stronger than Remark 5.7:

Corollary 5.16 *Consider a RGTR system with a regular set of terms as domain. The FO(R)-theory of its graph is decidable.* □

VRP is an extension of *vertex replacement* (VR) [BC1987, Cou1990] by just adding the fifth operator for synchronous products. With the reduced power of VR one can represent exactly the *prefix recognizable graphs* [Bar1998]. Those graphs have a decidable MSO-theory [Cau1996]. Thus the drawback of the additional power of the product operation are weaker decidability properties.

5.2.1 Generalized RGTR

When we look back to the last results then we have seen the equivalence of RGTR-graphs and interpretations of regular VRP-trees in Proposition 5.14. But Proposition 5.15 predicts a decidable logic for the more general class of interpretations of VRP-trees having a decidable MSO-theory instead of just being regular. We want to find a suitable extension of RGTR such that we end up again with an equivalence but this time to the interpretations of VRP-trees where MSO logic is decidable.

The idea is to introduce a infinite tree structure for RGTR as well which is functioning as skeleton in this case. In this extension of RGTR the regular sets of trees are defined

on an overlay of this tree with the skeleton tree, i.e. both trees, the finite one and the possibly infinite skeleton tree get merged like in the case of tree-automatic relations. In detail the overlay looks as follows:

Definition 5.17 The *overlay* of a tree $t : \mathrm{dom}(t) \to \Sigma$ and $s : \mathrm{dom}(s) \to \Gamma$ is the tree

$$t\|_s : \mathrm{dom}(t) \to \Sigma\|_\Gamma, \qquad t\|_s(w) := \begin{cases} \big(t(w), s(w)\big) & \text{if } w \in \mathrm{dom}(s), \\ \big(t(w), \bot\big) & \text{otherwise} \end{cases}$$

with the same domain as t but over the ranked *overlay alphabet* $\Sigma\|_\Gamma := \Sigma \times \big(\Gamma \dot\cup \{\bot\}\big)$ where each symbol inherits the rank of its first component from Σ. The extension of the overlay to sets $T \subseteq T_\Sigma$ and a tree $s \in T_\Gamma$ is the set $T\|_s := \big\{ t\|_s \;\big|\; t \in T \big\} \subseteq T_{\Sigma\|_\Gamma}$. □

Definition 5.18 (Generalized RGTR) A *generalized regular ground term rewriting system* (or *GRGTR* for short) formally is a tuple $(s, \Gamma, T, \Sigma, A, \Delta)$ consisting of a possibly infinite skeleton tree s over the ranked alphabet Γ, a regular set T of terms over the ranked overlay alphabet $\Sigma\|_\Gamma$ where Σ is another ranked alphabet, a finite set A of actions and a finite set Δ of rewriting rules of the form $L \xrightarrow{a} R$ with label $a \in A$ and regular sets L, R of finite trees over the ranked overlay alphabet $\Sigma\|_\Gamma$.

The graph represented by $\mathcal{A}$ has as domain the set $T \cap (T_\Sigma^{\mathsf{fin}}\|_s)$, i.e. consist of all trees of T such that the overlaid part is a prefix of the skeleton tree $s \in T_\Gamma$. There is an a-labeled edge between two trees t, t' of the domain if there exists a rewriting rule $L \xrightarrow{a} R$ and a position in both trees such that for the subtrees l, r of t, t' from that position holds that $l \in L$ and $r \in R$ and the rest of t and t' are equal. □

Thus the definition of the graph of a GRGTR system $(s, \Gamma, T, \Sigma, A, \Delta)$ corresponds to the graph of the RGTR system $(T, \Sigma\|_\Gamma, A, \Delta)$ except that the domain of the graph is restricted to trees of $T_\Sigma^{\mathsf{fin}}\|_s$, i.e. trees where the overlaid component is a prefix of the skeleton tree $s \in T_\Gamma$.

Next we want to establish a equivalence result like Proposition 5.14 but this time for GRGTR graphs and graphs of VRP-trees with decidable MSO-logic. We split the proof in the following two lemmas each stating one direction of the equivalence result.

Lemma 5.19 *For each GRGTR system exists a deterministic finite tree-transducer which transforms the skeleton to a VRP-tree representing the same graph (up to isomorphism and color removal). The transducer can be constructed effectively.* □

Proof To prove this lemma we give the definition of a deterministic finite tree-transducer that merges the skeleton tree of the given GRGTR system and the automata, which define its domain an relations, to a single VRP-tree that simulates the GRGTR system.

Consider a given GRGTR system $(s, \Gamma, T, \Sigma, A, \Delta)$ with regular set T and regular sets $L, R \in T_{\Sigma\|_\Gamma}$ for each rewriting rule $L \xrightarrow{a} R$ in Δ. We first introduce one single deterministic finite tree-automaton $\mathcal{A}_F = (Q, \Sigma, \delta, F)$ for those languages of finite trees over alphabet $\Sigma\|_\Gamma$, i.e. that

- there exists a set $F_T \subseteq Q$ with $T = T(\mathcal{A}_{F_T})$ and
- there exists a relation $\Delta' \subseteq Q \times A \times Q$ such that for each $a \in A$:

$$\bigcup_{\substack{L,R \in T_{\Sigma\|\Gamma} \\ L \xrightarrow{a} R \text{ in } \Delta}} L = \bigcup_{\substack{p,q \in Q \\ (p,a,q) \in \Delta'}} T(\mathcal{A}_{\{p\}}) \qquad \text{and} \qquad \bigcup_{\substack{L,R \in T_{\Sigma\|\Gamma} \\ L \xrightarrow{a} R \text{ in } \Delta}} R = \bigcup_{\substack{p,q \in Q \\ (p,a,q) \in \Delta'}} T(\mathcal{A}_{\{q\}}).$$

This automaton can be constructed for instance by a product of automata each representing one of the languages.

For simplicity we next make the bottom symbol $\bot$ in the skeleton tree $s \in T_\Gamma$ explicit. Let the ranked alphabet $\Gamma' := \Gamma \dot\cup \{\bot\}$ with each symbol of rank $m := \max\{|f| \mid f \in \Sigma\}$ which is the maximal rank of a symbol in Σ. Then define the tree $s' \in T_{\Gamma'}$ as the complete infinite m-ary tree as extension of s by the new bottom symbol:

$$s' \colon \{0,\dots,m\} \to \Gamma' \qquad \text{with} \qquad s'(w) := \begin{cases} s(w) & \text{if } w \in \operatorname{dom}(s), \\ \bot & \text{otherwise.} \end{cases}$$

Then $t\|_s = t\|_{s'}$ for each $t \in T_\Sigma$.

Now we can define the deterministic finite transducer $\mathcal{T} = \big(Q', \Gamma', \Omega_{A,Q}, (\delta'_q)_{q\in Q'}, q'_0\big)$ from the ranked alphabet Γ' of the extended skeleton tree s' to the ranked alphabet of VRP-operators with actions A (from the given GRGTR) and colors Q which are set to the states of the automaton above. The states $Q' := Q \dot\cup \{q'_0\}$ of the transducer are the states of the automaton too plus a new initial state q_0. The transduction function δ' can be obviously defined formally in such a way to get the following transduction:

$$\begin{aligned} \hat\delta'_{q_0}(s) &:= \bigoplus_{q\in F_T} \hat\delta'_q(s) \\ \hat\delta_q\Big(g(s'_1,\dots,s'_m)\Big) &:= \underbrace{\cdots [p \overset{a}{\bowtie} q] \cdots}_{\substack{\text{for each} \\ (p,a,q)\in\Delta'}} \bigoplus_{\substack{f\in\Sigma,\ q_1,\dots,q_{|f|}\in Q, \\ \delta_{(f,g)}(q_1,\dots,q_{|f|})=q}} \left(\otimes^{|f|}_{\delta_{(f,g)}} \Big(\hat\delta'_{q_1}(s'_1),\dots,\hat\delta'_{q_{|f|}}(s'_{|f|})\Big)\right) \end{aligned}$$

where F_T, Δ', Q, δ are from the definition of the automaton $\mathcal{A}_F$ above.

In a last step the extended operators $\bigoplus$ and $\otimes^n_\eta$ with arbitrary arity $n \in \mathbb{N}$ can be easily reduced to a sequence of the normal binary operators $\oplus$ and $\otimes_\eta$ since we use them with a bounded arity. This leads to little differences in δ' and to more colors (for the product) but the semantics stays the same.

The transformation $\mathcal{T}(s)$ of the skeleton tree s indeed simulates the GRGTR system since the structure of the transformed tree corresponds to the structure of the skeleton and the automaton for the domain and relations of the GRGTR system is simulated by the colors of that VRP-tree. ∎

According to Remark 2.19 do deterministic finite tree-transducer preserve the decidability of MSO logic of tree. Thus by applying Proposition 5.15 to Lemma 5.19 we get the same decidability for GRGTR graphs as for VRP-interpretations:

Corollary 5.20 *The FO(R)-theory of a GRGTR graph is decidable if the skeleton tree has a decidable MSO-theory.* □

The converse of Lemma 5.19 is a bit easier to show:

Lemma 5.21 *For each VRP-tree there exists a GRGTR system having that tree as skeleton and representing the same graph (up to isomorphism and color removal). The GRGTR system can be constructed effectively.* □

Proof Consider a given (usually infinite) VRP-tree s with finite action alphabet A, finite set C of colors over the ranked alphabet $\Omega_{A,C}$ of VRP-operators (consisting of constant symbols $\dot{c}$, unary symbols $[\phi], [c \overset{a}{\bowtie} d]$ and binary symbols $\oplus, \otimes_\eta$ for each $a \in A$, $c, d \in C$, $\phi : C \to C$ and $\eta : C^2 \to C$).

We will simulate its VRP-interpretation by a GRGTR system with s as skeleton. This can be done by representing each vertex of the interpretation as a finite subtree of the VRP-tree s where for each vertex in the subtree we have that

- if the vertex is labeled by some $[\phi], [c \overset{a}{\bowtie} d]$ then the child vertex is in the subtree,
- if the vertex is labeled by $\oplus$ then either the left or the right child is in the subtree,
- if the vertex is labeled by some $\otimes_\eta$ then both children are in the subtree.

Then each vertex of the VRP-interpretation corresponds to exactly one of such subtrees. This will be the domain of the GRGTR graph.

To simulate the coloring and edges between the vertices we have to use the rewriting rules of the GRGTR-system. We can define a deterministic finite tree-automaton with the colors C as state set which computes the color at each internal node of a tree of the domain in a bottom-up manner. Then an edge operator $[c \overset{a}{\bowtie} d]$ in such a tree of the domain can be used to switch from a tree that had color c at this very vertex to another one having color d.

The idea of representing the vertices of the graph of a VRP-tree as its subtrees is depicted in Figure 5.3. We took the VRP-tree of Example 5.13 which is defining the infinite grid. The subtrees in Figure 5.3(a) and 5.3(b) represent the vertices $(1, 0)$ and $(1, 1)$ respectively. Additionally we denoted the color that the tree-automaton would assign to each vertex of the subtree as explained above. The vertices represented by the two subtrees are furthermore connected by an S_2-edge. This edge is represented by the vertex right of the root of the subtrees. The edge can be applied since the tree-automaton assign to that vertex the color 0 in the first subtree and color 1 in the second subtree, which are exactly the colors that the $[0 \overset{S_2}{\bowtie} 1]$-vertex of this edge requires.

For the rewriting rules we define an automaton scheme which can be used for the domain as well. Let $\mathcal{A}_F = \left(Q, \Sigma\|_{\Omega_{A,C}}, (\delta_{(f,g)})_{f \in \Sigma, g \in \Omega_{A,C}}, F\right)$ be a deterministic finite tree-automata with variable set F of final states and

- state set $Q := C \times \Omega_{A,C} \dot\cup \{\#, \bot\}$ representing the color of the subtree below a vertex and the VRP-operator at this very vertex (plus two extra symbols $\#$ for the empty tree and $\bot$ as bad state),

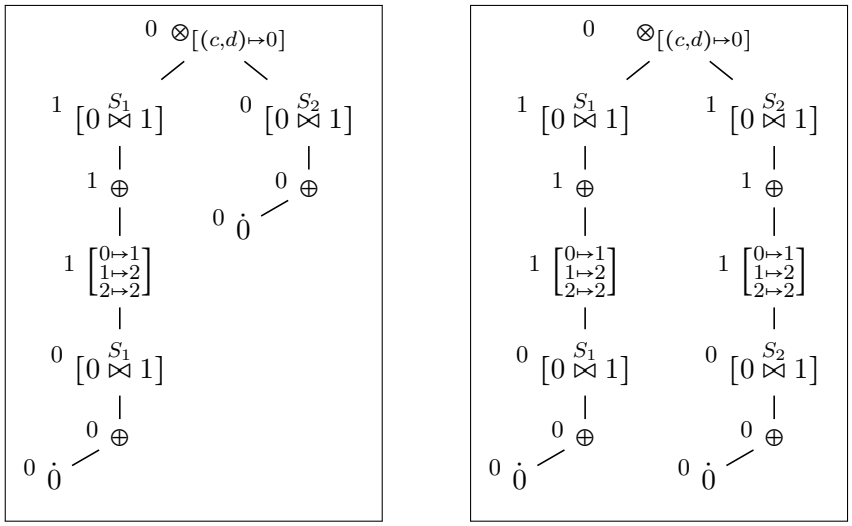


Figure 5.3: Two finite subtrees of the VRP-tree of Example 5.13 each representing the grid vertices $(1,0)$ in (a) and $(1,1)$ in (b).

- ranked overlay alphabet $\Sigma\|_{\Omega_{A,C}}$ with the simple ranked alphabet $\Sigma \coloneqq \{0,1,2\}$ where each symbol corresponds to its rank,
- transition functions δ_f defined as:
  - $\delta_{(0,g)} \coloneqq \#$
  - $\delta_{(1,\dot{c})}(\#) \coloneqq (c,\dot{c})$
  - $\delta_{(1,[\phi])}\big((c,g)\big) \coloneqq \big(\phi(c),[\phi]\big)$
  - $\delta_{(1,[d \overset{a}{\bowtie} d'])}\big((c,g)\big) \coloneqq \big(c,[d \overset{a}{\bowtie} d']\big)$
  - $\delta_{(2,\oplus)}\big((c,g),\#\big) \coloneqq \big(c,\oplus\big)$
  - $\delta_{(2,\oplus)}\big(\#,(c,g)\big) \coloneqq \big(c,\oplus\big)$
  - $\delta_{(2,\otimes_\eta)}\big((c_1,g_1),(c_2,g_2)\big) \coloneqq \big(\eta(c_1,c_2),\otimes_\eta\big)$

  for each $g \in \Omega_{A,C}$, $c,d,d' \in C$, $a \in A$, $\phi : C \to C$ and $\eta : C^2 \to C$. For all other cases we set $\delta_{(n,g)}(q_1,\dots,q_n) \coloneqq \bot$.

Then a subtree with state (c,g) corresponds to a vertex of color c and by means of $g \in \Omega_{A,C}$ we can specify where the edges are.

All together we get a GRGTR system $(s,\Omega_{A,C},T,\Sigma,A,\Delta)$ with skeleton tree s, action set A, ranked alphabets $\Omega_{A,C}$ and Σ as introduced above. Finally we set the domain to $T \coloneqq T\big(\mathcal{A}_{C\times\Omega_{A,C}}\big)$ and the rewriting rules to the set Δ consists of the following rules:

$$T\left(\mathcal{A}_{\left\{\left(c,[c \overset{a}{\bowtie} d]\right)\right\}}\right) \xrightarrow{a} T\left(\mathcal{A}_{\left\{\left(d,[c \overset{a}{\bowtie} d]\right)\right\}}\right) \qquad \text{for each } a \in A \text{ and } c,d \in C.$$

This GRGTR system is simulating the operations of the given VRP-tree and thus specifies the same graph. ■

Putting Lemma 5.21 and 5.19 together we get the desired equivalence:

Corollary 5.22 *The structure class of VRP-interpretations is equivalent to the class of GRGTR-graphs up to isomorphism and color removal.* □

When restricting to trees with a decidable MSO-theory we generate graph structures having a decidable FO(R) logic according to Proposition 5.15 and Corollary 5.16. When furthermore restricting to regular trees then we get the result of Proposition 5.14 as a special case since in that case the skeleton tree can already be simulated by the limited power of a RGTR-system.

6 Conclusion

6.1 Summary

In this thesis we considered first-order logic with reachability predicates over infinite systems. There are many important classes of graph structures that have a decidable first-order logic but monadic second-order logic is undecidable. The extensions of first-order logic with reachability we focused on in this work are much stronger than plain first-order logic and mostly weaker than monadic second-order logic. This means for that important structure classes that some of those extended logics are probably still decidable. We had a look at the following different aspects of first-order logic with reachability predicates over infinite systems:

- by set-based unfolding we gave a structure transformation which yields structures where first-order logic with regular reachability is decidable,
- then we considered several variants of reachability in first-order logic over infinite grids, since the decidability in those structures turned out to be very sensitive to small changes in the grid or the expressive power of the logic and
- finally we dealt with graph structures representable by ground term rewriting, where first-order logic with simple reachability is decidable, and presented a generalization which is equivalent to the class of graphs representable by vertex replacement and product operations.

First we introduced the basics in Chapter 2 about infinite graph structures and logics. The important logics here were first-order (FO) logic and extensions of it like monadic second-order (MSO) logic and FO($\mathcal{L}$) which is FO logic with reachability by relation sequences of some language class $\mathcal{L}$. After defining some language classes and showing the relations between them in Remark 2.6, we got an similar inclusion diagram for the logics above in Remark 2.11. The most important logics besides FO and MSO are FO(R) and FO(Reg) which is FO with simple reachability and with regular reachability respectively.

By set-based unfolding in Chapter 3 we created a new type of structure transformation which is very similar to the normal (path-based) unfolding. The transformed graph structures are completely different since we abstracted the paths of the normal unfolding. But on the other hand this abstraction does not preserve the decidability of MSO logic any longer as the normal unfolding did. Anyway we showed in Corollary 3.4 that this transformation does transfer the decidability of MSO logic to the decidability of FO(Reg) if finiteness is MSO-definable in the given graph structure. In Theorem 3.3, which is

the more complex version of this result, we even get the same decidability in a quotient structure of the set-based unfolding for any congruence that is MSO-definable in the given structure. At the end of that chapter about set-based unfolding we showed that the above results are sharp, i.e. that there are simple structures with decidable MSO logic such that their set-based unfoldings do not have that very logic decidable (cf. Remark 3.5).

In Chapter 4 we are determining how much reachability is decidable in infinite grid-like structures. We first defined a number of slightly different infinite grids since for logics like FO(R) and FO(Reg) it can be important to have more relations available even if they are definable in FO. So we studied the decidability of the logics introduced in Chapter 2 in n-dimensional infinite grids $\mathfrak{N}^n$ with just forward edges, $\mathfrak{N}^{\pm n}$ with just forward and backward edges and $\mathfrak{N}_0^n, \mathfrak{N}_0^{\pm n}$ as before but additionally with relations testing for border positions (cf. Definition 4.1). The most important results are, that FO(Reg)-logic is decidable in the so called one-way grids $\mathfrak{N}^n, \mathfrak{N}_0^n$ for any dimension $n \in \mathbb{N}$ (Theorem 4.7 and Corollary 4.10) but is undecidable in the two-way grids $\mathfrak{N}^{\pm 3}$ (Corollary 4.12) and $\mathfrak{N}_0^{\pm 2}$ (Theorem 4.15) for dimension 3 and 2 respectively or higher. Furthermore we gave various results concerning FO with reachability by more powerful language classes. The decidability of grids is summarized in Table 4.1. Two of those results allowed to conclude that FO logic with context-free reachability cannot be simulated by MSO logic (Remark 4.17).

Chapter 5 was about a very interesting class of graph structures having FO(R) logic decidable: the graphs of regular ground term rewriting (GTR) systems. First we tried to find a restricted version of GTR such that its graphs have even FO(Reg) logic decidable. Our approach of just extending terms unfortunately still yields graphs having that very logic undecidable. In the rest of that chapter we generalized an equivalence result which states that GTR graphs are the same as graphs of finite equation systems of vertex replacement and product (VRP) operations. We introduced generalized regular GTR (GRGTR) which is capable to represent the graphs of even infinite VRP equation systems (Lemma 5.21). On the other hand those GRGTR graphs can be expressed by an infinite VRP equation system (Lemma 5.19). If the unfolding of such a possibly infinite equation system, which is a tree of VRP operators, has a decidable MSO logic then FO(R) logic is decidable in the represented graph. Thus by GRGTR we found a new automaton-based representation of that very class of graph structures having FO(R) logic decidable.

6.2 Further Research

Besides the new results there also remain a couple of open questions. The set-based unfolding for instance is a generalization of a construction for a proof in [LO2007] which was used there to show the decidability of a certain algebraic property. Since this construction was tailored for just this single problem, the question is where else the expressive power of FO(Reg) logic over set-based unfoldings can be used. When looking at the decidability of FO with reachability in infinite grid structures then there are just about half of the questions answered. Anyway for the important logics FO, FO(R),

FO(Reg) and MSO all but one grid has been classified: it is still unknown if FO(Reg) logic is decidable in the 2-dimensional 2-way grid $\mathfrak{N}^{\pm 2}$. The problem is supposed to be decidable since regular reachability in this 2-dimensional structure can be expressed by simple reachability in 5-dimensional vector addition systems [HP1979] which is known to be decidable. The problematic thing here is the combination of reachability and FO logic.

Bibliography

[Avr2003] Avron, A. Transitive closure and the mechanization of mathematics. In *Thirty Five Years of Automating Mathematics*, pages 149–171. Kluwer Academic Publishers, 2003. 5

[Bar1998] Barthelmann, K. When Can An Equational Simple Graph Be Generated By Hyperedge Replacement? In *In MFCS*, pages 543–552, 1998. 52

[BC1987] Bauderon, M. and Courcelle, B. Graph Expressions and Graph Rewritings. *Mathematical Systems Theory*, 20(2-3):83–127, 1987. 52

[BG2000] Blumensath, A. and Grädel, E. Automatic Structures. In *Proceedings of the 15th IEEE Symposium on Logic in Computer Science, LICS 2000*, pages 51–62. IEEE Computer Society Press, 2000. 1

[BG2004] Blumensath, A. and Grädel, E. Finite Presentations of Infinite Structures: Automata and Interpretations. *Theory of Computing Systems*, 37:641–674, 2004. 1

[Blu1999] Blumensath, A. Automatic Structures. Diploma thesis, RWTH-Aachen, 1999. 1

[Bra1969] Brainerd, W. S. Tree Generating Regular Systems. *Information and Control*, 14:217–231, 1969. 5

[Büc1962] Büchi, J. R. On a decision method in restricted second order arithmetic. In *International Congress on Logic, Methodology and Philosophy of Science*, pages 1–11. Stanford University Press, 1962. 1, 34

[BW1990] Baeten, J. C. M. and Weijland, W. P. *Process algebra*. Cambridge University Press, New York, NY, USA, 1990. 6, 50

[Cal1997] Calbrix, H. La théorie monadique du second ordre du monoïde inversif libre est indécidable. *Bulletin of the Belgian Mathematical Society*, 4(1):53–65, 1997. 4, 30

[Cau2002] Caucal, D. On Infinite Terms Having a Decidable Monadic Theory. In Diks, K. and Rytter, W., editors, *MFCS*, volume 2420 of *Lecture Notes in Computer Science*, pages 165–176. Springer, 2002. 3, 22

[Cau1996] Caucal, D. On infinite transition graphs having a decidable monadic theory. In *Proceedings of the 23rd International Colloquium on Automata, Languages and Programming, ICALP '96*, volume 1099 of *Lecture Notes in Computer Science*, pages 194–205. Springer, 1996. 1, 52

[CL2007] Colcombet, T. and Löding, C. Transforming structures by set interpretations. *Logical Methods in Computer Science*, 3(2), 2007. 3, 29

[Col2002] Colcombet, T. On Families of Graphs Having a Decidable First Order Theory with Reachability. In *29th ICALP*, number 2380 in Lecture Notes in Computer Science, pages 98–109, Malaga, July 2002. Springer. Best student paper award, track B. 6, 50, 51, 52

[Col2004] Colcombet, T. *Propriétés et représentation de structures infinies (properties and representation of infinite structures)*. PhD thesis, Université Rennes I, March 2004. 6

[Cou1990] Courcelle, B. Graphs as Relational Structures: An Algebraic an Logical Approach. In Ehrig, H., Kreowski, H.-J., and Rozenberg, G., editors, *Graph-Grammars and Their Application to Computer Science*, volume 532 of *Lecture Notes in Computer Science*, pages 238–252. Springer, 1990. 52

[CR1998] Cornell, T. and Rogers, J. Model Theoretic Syntax. In *The Glot International State of the Article Book 1, Studies in Generative Grammar 48, Mouton de Gruyter*, pages 101–125, 1998. 5

[CW1998] Courcelle, B. and Walukiewicz, I. Monadic Second-Order Logic, Graph Coverings and Unfoldings of Transition Systems. *Annals of Pure and Applied Logic*, 92(1):35–62, 1998. 3, 22

[DT1985] Dauchet, M. and Tison, S. Decidability of confluence for ground term rewriting systems. In Budach, L., editor, *FCT*, volume 199 of *Lecture Notes in Computer Science*, pages 80–89. Springer, 1985. 5

[DT1990] Dauchet, M. and Tison, S. The Theory of Ground Rewrite Systems is Decidable. In *LICS*, pages 242–248. IEEE Computer Society, 1990. 1, 2, 5, 47, 49

[Göd1931] Gödel, K. Über formal unentscheidbare Sätze der Principia Mathematica und verwandter Systeme. *Monatshefte für Mathematik und Physik*, 38(1):173–198, 1931. 5, 39

[Hod1983] Hodgson, B. R. Décidabilité par automate fini. *Ann. Sci. Math. Québec*, 7(3):39–57, 1983. 1

[HP1979] Hopcroft, J. E. and Pansiot, J.-J. On the Reachability Problem for 5-Dimensional Vector Addition Systems. *Theor. Comput. Sci.*, 8:135–159, 1979. 60

[HU1979] Hopcroft, J. E. and Ullman, J. D. *Introduction to Automata Theory, Languages, and Computation.* Addison Wesley, 1979. 10

[Imm1999] Immerman, N. *Descriptive Complexity.* Springer, 1999. 5

[KN1995] Khoussainov, B. and Nerode, A. Automatic Presentations of Structures. In *Logical and Computational Complexity. Selected Papers. Logic and Computational Complexity, International Workshop LCC '94, Indianapolis, Indiana, USA, 13-16 October 1994*, volume 960 of *Lecture Notes in Computer Science*, pages 367–392. Springer, 1995. 1

[LO2007] Lohrey, M. and Ondrusch, N. Inverse monoids: Decidability and complexity of algebraic questions. *Inf. Comput.*, 205(8):1212–1234, 2007. 2, 4, 20, 27, 30, 59

[Löd2002] Löding, C. Model-Checking Infinite Systems Generated by Ground Tree Rewriting. In *Proceedings of Foundations of Software Science and Computation Structures, FoSSaCS 2002*, volume 2303 of *Lecture Notes in Computer Science*, pages 280–294. Springer, 2002. 5

[Löd2003] Löding, C. *Infinite Graphs Generated by Tree Rewriting.* PhD thesis, RWTH Aachen, Germany, 2003. 5

[May2000] Mayr, R. Process Rewrite Systems. *Information and Computation*, 156(1–2):264–286, 2000. 6, 50

[Min1961] Minsky, M. L. Recursive Unsolvability of Post's Problem of 'Tag' and Other Topics in Theory of Turing Machines, 1961. 40

[Min1967] Minsky, M. L. *Computation: Finite and Infinite Machines.* Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1967. 40

[MS1985] Muller, D. E. and Schupp, P. E. The theory of ends, pushdown automata, and second-order logic. *Theoretical Computer Science*, 37:51–75, 1985. 1

[Par1966] Parikh, R. J. On Context-Free Languages. *J. ACM*, 13(4):570–581, 1966. 35

[Pos1946] Post, E. L. A variant of a recursively unsolvable problem. *Bull. of the Am. Math. Soc*, (52), 1946. 49

[Pre1929] Presburger, M. Über die Vollständigkeit eines gewissen Systems der Arithmetik ganzer Zahlen, in welchem die Addition als einzige Operation hervortritt. *Comptes Rendus du Premier Congrès des Mathématiciens des Pays Slaves, Warszawa*, pages 92–101, 1929. 1, 35

[Pre1991] Presburger, M. On the completeness of a certain system of arithmetic of whole numbers in which addition occurs as the only operation. *Hist. Philos. Logic*, (12):225–233, 1991. 1, 35

[Rab1969] Rabin, M. O. Decidability of Second-Order Theories and Automata on Infinite Trees. *Transactions of the American Mathematical Society*, 141:1–35, July 1969. 1

[Tho1990] Thomas, W. Automata on infinite objects. pages 133–191, 1990. 4, 31

[Tre1998] Treinen, R. The First-Order Theory of Linear One-Step Rewriting is Undecidable. *Theor. Comput. Sci.*, 208(1-2):179–190, 1998. 46

[Tur1936] Turing, A. M. On Computable Numbers, with an application to the Entscheidungsproblem. *Proc. London Math. Soc.*, 2(42):230–265, 1936. 39, 47

[Wöh2005] Wöhrle, S. *Decision Problems over Infinite Graphs: Higher-order Pushdown Systems and Synchronized Products*. PhD thesis, RWTH Aachen, 2005. 5, 41

[WT2004] Wöhrle, S. and Thomas, W. Model Checking Synchronized Products of Infinite Transition Systems. In *LICS*, pages 2–11. IEEE Computer Society, 2004. 4