UNIVERSITAT DE BARCELONA
Facultat de Filosofia

MÀSTER EN LÒGICA PURA I APLICADA

Treball final de màster

Multimodal logics for musical grammars

Autor: Roger Asensi Arranz

Director: Dr. David Fernández-Duque
Realitzat a: Departament de Filosofia

Barcelona, 29 de juliol de 2024

Contents

Introduction

Historically, the study of musical form and harmony through mathematical tools has heavily relied on the premises and assumptions from the music-theoretic formalization for the context of the object of analysis. Many of the latter usually involve some sort of mathematical construction, regardless of how explicit it might be.

A more recent approach, applied mainly to the derivation of harmonic progressions, stems from the usage of context-free grammars and other more general sorts of linguistic structures. We focus on some examples proposed to parse chord sequences in the domain of early jazz standards ([Ro20], [Gr14]), although some considerations are borrowed from those dedicated to occidental classical music from the late nineteenth century ([MR21]).

Behind each of the visited models lie certain musical hypotheses that shape the scope of the used objects, as well as the distinctive features of the rules which they operate with. Thus, it is natural for some questions to arise: what do these have in common? Can they be thought of as instances of a more general system? How strong should the assumptions behind such a frame be?

We turn to modal logic to formalize such common grounds for several grammars to be inspected. In particular, we use a fusion of lineal temporal logic (LTL) and K modal logic endowed with the transitive closure of its modal operator. The proposed models can evaluate any formula in the language and inherit the considered grammar through reflecting it the relation corresponding to the modal operator of K.

Our goal is to provide a sound formalization of such structures and study the behavior of well-known decision problems in the several restrictions and fragments of the logic that appear naturally. Concurrently, we will base some of the results on and interpret them in the key of actual musical grammars, while observing the categorization of overall classes of models.

Despite us not being aware of this construction being applied to musical analysis anywhere else at the moment of writing this, several of the methods used are well-known and extended for other kinds of multimodal logics. We highlight reducibility relations between decision problems ([Ho01]), canonical models and filtrations ([Go87]) and other notions from computational complexity theory ([LP81]).

This thesis is structured as follows:

- Chapter 1 motivates and presents the basic machinery to handle our constructions from the harmonic and logical perspectives. While no previous musical knowledge is required, the reader familiar with some basic theory

might appreciate some intuitions behind the proposals.

- Following this, chapter 2 proposes the structures we will study henceforth and underlines some of their main properties.
- Chapter 3 introduces and discusses model checking. Undecidability is reasoned in terms of reduction to the Post Correspondence Problem, and decidable fragments are based on restrictions for the accessibility relation in particular grammars.
- Another problem, decidability, is explored in two of its variants, subject to whether we fix a grammar or not. Filtration and fragment analysis are noteworthy aspects of chapter 4.
- Finally, chapter 5 deals with some concrete musical examples and summarizes the obtained results.

I would like to acknowledge the determined patience and dedication of David Fernández-Duque, my thesis advisor. Any positive aspect of this work can only be seen as a result of his constant guidance and commitment to materialize a not so typical proposal for a project.

I also want to express my appreciation for the comments and inputs from colleague musicians and music theory researchers. Of course, I am forever grateful to my friends and family for their unwavering support, without which it would be hard to imagine having closed this stage of my life.

Abstract

Musical theory has often employed multiple grammars to formalize harmonic languages. We reinterpret a particular model in terms of a fusion of temporal and transitive modal logics. This work focuses on the analysis of typical decision problems of the field, while exploring how might the results vary according to the applied restrictions. In order to do so, we recur to well-known techniques and methods from computability theory and the field of modal logic. Some examples from the music-theoretic literature are presented and analyzed through the lenses of the considered results and observations.

2020 Mathematics Subject Classification. 03B62, 03B45, 03D05.

Chapter 1

Preliminary notions

This preparatory chapter provides the elementary tools and techniques to understand and develop the structures upon which our logic will be based. Several music-theoretic references are contrasted and summarized through the exposition of grammars, and a particular rule set from [Ro20] is taken as a readily applicable example. Most of the definitions of grammars and modal logics can be found in widely available manuals, such as [LP81] and [Ch80]. Some basic knowledge of general logic and complexity theory is required to comprehend the upcoming chapters – however, this introductory chapter attempts to provide the bases to operate with modal logics and grammars without the reader needing to have studied them previously.

1.1 Motivation

After the formal segregation of 'the arts' in the *quadrivium*, the relationship between harmony and mathematics has still been an object of study, albeit from different standpoints. In general, the attempts by European musical academic institutions to formalize certain aspects of classical music (theory) have only manifested the inevitable tension between a series of proposed schemata and the escaping from rules by actual musical practice.

Undoubtedly, when it comes to relating form and micro-scale, harmony has mostly been the predominant object of analysis in this musical tradition. Within the styles where the functional relation between chords prevails, we can distinguish several ways to view harmonic phenomena: on the one hand, the scale at which the conjunction of sounds that we come to understand as harmony is most relevant, be it in a voice-leading sense or in a more structural viewpoint; on the other hand, we might discuss how these entities interact with each other, to what extent is an alleged macro-form representative or source of the local development

of chord sequences, when certain ways of composing and understanding music in the Baroque are based in the juxtaposition of semioticly meaningful resources.[1]

In any case, the recent trend in data-based models such as [HR11] not only establishes a compromise between such complementary systems, but also requires us to understand music as a lax, evolving conglomerate of languages whose conception also stems from different understandings of form and sound.

In one respect, combinatorial theories of tonal/modal areas are preceded by Guido de Arezzo's hexachordal scales, often appearing in the context of Carey and Clampitt's seminal publications on well-formed scales ([CC89]). Through the interaction with schemata theory, documented recently by [HR11], has given rise to systems such as the ones presented by [DN18] and [DN19]. Another word-theoretic formalization is explored by [CN11].

Besides those, but not completely unrelated, the sort of constructions we will base our work on have their origins in the tradition of Chomsky generative grammars and, in particular, in Lerdahl and Jackendoff's Generative Theory of Tonal Music ([JL83], also [HHT06]). Improving on existing attempts to formalize chord sequences of given styles in tree-like derivations (such as [NR15]), several notions from Neo-Riemannian theory and additional assumptions assist the author in refining some of the previous milestones. Incidentally, similar arrangements for melodic progressions can already be found in Schenker's writings decades before.[2]

From the mathematical standpoint, both sorts of systems may be thought of as lacking due to issues with the completeness of the characterized material or the representativity of the obtained models. In some sense, a possible cause for this is apparent simplicity of the systems, which can be traced back to the minimal but profound assumptions the authors make regarding the represented corpus, such as fifth-based movement or right-heading harmony, respectively. Nonetheless, it is this primitiveness which prevents the models from becoming completely saturated and trivial, and which leaves room for a musical interpretation outside the mathematical formalization.

Multiple grammars

Through the history of the development of generative models for a given musical quality, it has become evident that the choice of foundations for a certain system has deep consequences in the outcome of the enterprise. It can often be observed that models constructed with different mathematical machinery, but which

[1] See [Gj07] for a thorough presentation.

[2] For example, as seen by [Be19].

are coherent or complementary in terms of its elementary assumptions, can give rise to combinations which improve some of the weaknesses from the initial ones.

For instance, our initial aim was to view in terms of modal logic Rohrmeier's proposal of a grammar for classic jazz harmony, through a direct translation or an equivalent formulation as of soundness and completeness. One of the core principles is the identification of the tunes as all-encompassing harmonic units, that is, chords large in temporal scale which can be decomposed in its derivations. Such transformations are assumed to rely in the notion that each chord can either be substituted, or prepared and extended to the left.

However, the outcome might leave unsatisfied the reader expecting to find more complete information in an aseptic, formal environment. If they were to refuse the belief that musical interpretation is indispensable, a possible solution would be to consider context-dependent rules, such as those allowing for turnarounds (see [HOR18]) or other structural connections. But there would still exist problems of the same nature even by regarding modal harmony (see [MR21]) and common-note modulation, or by trying to replicate some of the transformations seen for octatonic symmetries and Euler's Tonnetz space ([Ty12]).

One of the most prominent approaches for trying to circumvent these issues comes as some variation of probabilistic models. In [HOR18], the authors manage to address the double focus on global form and local endings, essentially thanks to process information in a more detailed manner with tags. Some generalization of context-free grammars (abstract CFGs and probabilistic ACFGs) is introduced, along with the application of parsing and Bayesian inference techniques. Even though they involve data sets, the derivations occur in a rather local scale, although the doctoral dissertation [Ha20] presents a more thorough study with generalizations.

Another direction is taken in [Ha12] and [HR09], where a measure of similarity between musical segments is presented in order to establish a grammar which assigns tags to chords and searches, given two sequences and their generation trees, the largest commonly embedabble tree. Furthermore, [Gr14] supplements this by considering Markov constructions. On a related note, other parsers can also contemplate psychological or proper linguistic aspects, as seen in [KOOTU20] and [AIMT97].

Finally, some efforts have been made to gather data for probabilistic models of particular styles or composers, as it is the case of [BS13], [BS19] and [AW04]. For more akin statistical processing of corpora, [CR08] assigns probabilities to certain transitions in Bach chorales (normalized or flattened in terms of musical information), and [HMNR19] studies the proportion of certain transitions within Beethoven string quartets, aiming to contextualize some progressions (in different

levels of globality) and quantify tonal direction.

So even if our focus is not turning to probabilistic automata (despite already committing classes larger than PP), we switch perspectives in hopes of providing a more general framework to explore decision problems for grammars related to harmony. Some aspects of operations on topological and metric spaces can be interpreted in terms of dynamical systems, and [KM05] [KKWZ06] opt for a modal logic as a tool to express and utilize this correspondence.

We will adopt a combination of temporal logic and an accessibility logic to express in general the possible unrestricted musical grammars, hoping to systematize the study of sequence parsing and complexity of relevant problems. For instance, one could propose a formula φ to summarize the notion of a harmonic progression resolving in a certain tonal context:[3] then, φ could be manipulated in another expression with the purpose of checking if a resolution occurs in a given sequence, or if some variation or reprise can take place in any grammar complying with given limitations. These two kinds of questions correspond to two central decision problems, *model checking* and *satisfiability*, which will be analyzed in the following chapters.

1.2 Musical machinery

A significant amount of suggested models for analyzing tonal harmonic sequences relies on the notion of grammar. In particular, given the nature of the derivation rules and the ambiguity of some musical structures, some of the existing systems focus on context-free grammars, but we will use more general tools. We provide the sufficient background to understand the constructions that motivated this work and to reproduce their intricacies onto our logic and formalization. Recall that:

Definition 1.1. *A(n* unrestricted*)* grammar *is a 4-tuple* $G = \langle V, \Sigma, P, S \rangle$*, where* V *is a set of* variables*,* Σ *is an alphabet of* terminal symbols *usually disjoint from* V*,* $P \subseteq ((V \cup \Sigma)^* \setminus \{\varepsilon\}) \times (V \cup \Sigma)^*$ *is a collection of* production rules*, and* $S \subseteq V$ *is the set of* starting variables[4]*.*

We operate with words on $V \cup \Sigma$ in the following manner: we define the relation $\Rightarrow_G$, given words u, u', as $u \Rightarrow_G u'$ if there exist $v, v', w, w', \alpha, t \in (V \cup \Sigma)^*$ and $\alpha \neq \varepsilon$ such that $u = v\alpha w$, $u' = v'tw'$, and $\alpha \rightsquigarrow_G t$ is a production rule, i.e. $(\alpha, t) \in P$.

[3] Personal communication with Bryan Perilla.

[4] Usually, only a single starting variable is regarded. For the sake of meaning coherence of our variables, we consider a set of initial variables, but a workaround would be to introduce an auxiliary starting variable s_0 and adding the rules $s_0 \to s$, for every $s \in S$.

We write $\alpha \rightsquigarrow t$ if there is no risk of ambiguity, but we reserve the non-subscripted $\Rightarrow$ for a future modal relation. We do denote by $\Rightarrow_G^*$ the (reflexive) transitive closure of $\Rightarrow_G$, and call the set $L(G) = \{u \in \Sigma^* \mid \exists s \in S\ (s \Rightarrow_G^* u)\}$ the *language* of the grammar G.

In musical terms, our field of study are harmonic sequences of classical tonal jazz, so it is natural to question which representation of chords serves our purpose best. From the musician standpoint, "surface-level" chords are enough for their practice and understanding of a tune. However, provided that e.g. Gmin[7] can either be a first degree of G minor, or a second degree of F major, it is necessary to somehow include this information in the syntactical formalization. So let $D := \{I, II, \ldots, VII\}$ be the set of degrees and $K := \{\text{C}, \text{c}, \text{C\#}, \text{c\#}, \ldots, \text{B}, \text{b}\}$ the set of keys/tonalities.[5] We opt to express variables as pairs (d,k) of degree and key, usually written as d_k, so that the actual chord can be inferred (and actually materialized) by some unary production rule: for instance, i_{g} and ii_{F} correspond to the aforementioned Gmin[7] in the respective contexts. Note that, for the sake of simplicity, we identify all extensions, inversions and modalities[6] (maj/min) of any given chord, since they are functionally equivalent in this context.

Having presented and refined the building blocks of our sequences, the goal of a system should be to formalize the rule set that encapsulates the possible sequences; that is, the rules for which the language of their grammar is precisely the desired collection of sequences. We assume our set of initial variables to be $\{I_k \mid k \in K\}$, the set of first degrees, based on the assumption that, in our corpus, any chord beyond the final I_k is a *da capo*/preparation of the beginning of the sequence.[7]

In [Ro20], first motivated by [Ro11], Rohrmeier distinguishes three types of rules:

- unary rules: chord substitutions, whether they preserve tonal function (e.g. tritone substitutions) or not (tonicizations).
- (binary) prolongational rules, which add another instance of the variable in question ($I \rightsquigarrow I\ \ I$).
- (binary) preparatory rules, which derive the harmony that leads to a particular chord ($V \rightsquigarrow V/V\ \ V$).

In particular, only left-branching rules are considered among the latter (i.e. for

[5] Notice that, unlike keys, we do not split degrees in their major (uppercase) and minor (lowercase) variants. This is because we *generally* assume they are unequivocally determined by the key the chord is found in, thus –for our purposes– we can assume $I = i$, $II = ii$,..., and employ each in their context.

[6] Understood as the major, minor, diminished, augmented, sus, etc. variants.

[7] See [HOR18] for a discussion in the case of turnarounds.

every $X \in V$ in the left hand side of a rule, the rule is of the form $X \rightsquigarrow YX$, for some Y): this materializes the assumption that functional harmony is right-headed, in the sense that every chord can only be understood as the predecessor of some other further along the tune. We outline the common-practice rule schemata, followed by a description of their notation:

Diatonic fifth	$X \rightsquigarrow \Delta/X\ X$	Substitution	$X \rightsquigarrow Sub/X$
2^{ry} dominant	$X \rightsquigarrow V/X\ X$		
Leading tone	$X \rightsquigarrow vii^{\circ}/X\ X$	Tonicization	$X_{key=Y} \rightsquigarrow I_{key=X/Y}$
Half cadence 1	$V \rightsquigarrow \flat VI\ V$		
Half cadence 2	$V \rightsquigarrow IV\ V$	Modulation	$Z/X_{key=Y} \rightsquigarrow Z_{key=X/Y}$
Diminished 1	$X \rightsquigarrow X^{\circ}\ X$	Tritone subs.	$V/X_{key=Y} \rightsquigarrow V_{key=\flat V/X/Y}$
Diminished 2	$X \rightsquigarrow \flat ii^{\circ}/X\ X$	Backdoor V	$V/X_{key=Y} \rightsquigarrow V_{key=\flat III/X/Y}$
Half-diminished	$X \rightsquigarrow ii^{7\flat 5}/X\ X$	Mode inversion	$X_{key=Y} \rightsquigarrow X_{key=inv(Y)}$
Plagal cadence	$I \rightsquigarrow IV\ I$	Terminal rules	$X \rightsquigarrow \mathrm{Chord}(X)$

Table 1.1: Summary of the possible rules for a generative syntactic model of harmonic sequences, in the context of jazz standards. Instances of terminal rules replace a variable by their corresponding chord as a symbol in Σ. By Δ we refer to the diatonic fifth, and *inv* switches major and minor modes.

The forward dash notation X/Y serves to refer to a particular chord which occupies the position X in a scale over Y: for example, Δ/vi in D major is an F minor chord, that is, $(d,k) = (iii,\mathrm{D})$. Depending on the rule and its musical context, we may choose to remain in the same key or modulate to one where the alluded chord actually occurs, such as expressing $\flat ii^{\circ}/ii$ in E$\flat$ major (F$\sharp$ diminished) as (vii,g). Barring some exceptions (like tritone substitutions), every X/Y just corresponds to shorthand for some chord, which allows us to handle them as ordinary elements of the domain of our rules. Recall that we separate major and minor keys as different objects, but not scale degrees, whose representation does not affect the way we apply rules. Similarly, we use the superscripts ^{7}, $^{\Delta 7}$, $^{-7}$, $^{\circ}$ and $^{+7}$ to denote (seventh) dominant, major, minor, diminished and augmented chords, respectively.

When dealing with particular frames and grammars, we will choose equally expressive subsets of this list, sometimes even simpler collections. It is important to note that, unless specified, every rule schema is applicable to any degree and key, even though some of the instances are not common (musical) practice. We will

not make such distinctions, as they will not affect the expressive or computational power of the examples we will present. The following is a possible interpretation of the harmonic sequence of a tune:

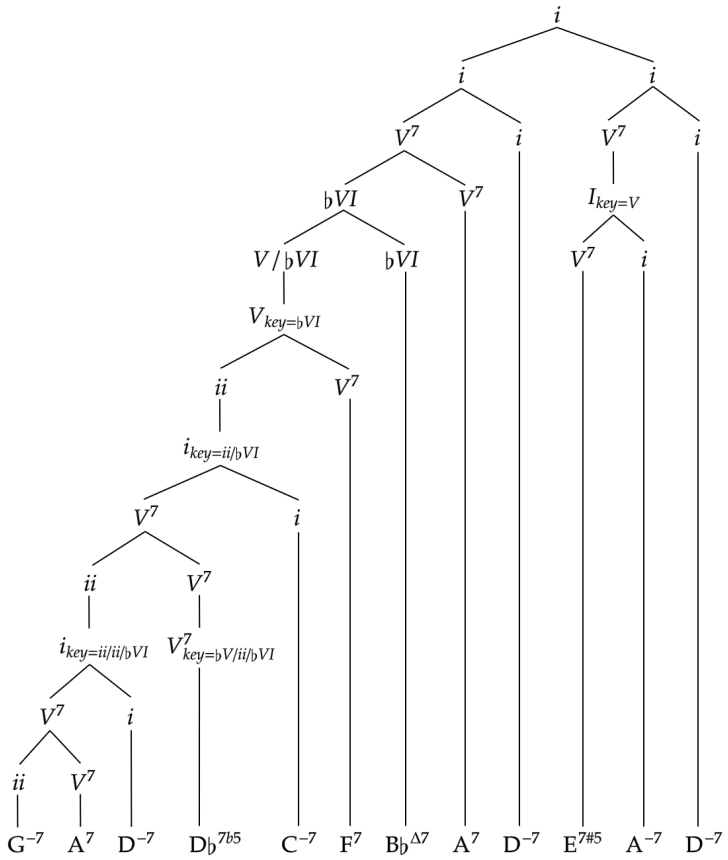

Figure 1.1: Blue in green

Example 1.2. This analysis of *Blue in green* focuses on the right-headed drive of the harmony, since the first instance of D^{-7} could have been thought of as an independent branch overall. Thus, the "change" and the tritone substitution are integrated within the general outline. Note that the *key* notation is dropped for a neater result.

Such opposing interpretations of a same sequence are common, though this capacity to recreate complex sequences leaves aside the frequency analysis of certain changes – there might not be any rule to explain certain transition (especially by chromatic proximity), but it is precisely the gap in the tree (the disruption of continuity) that highlights its relevance, as some of the people who worked on it argue.

Context-free grammars grant the systems of chord analysis the possibility to reproduce a particular set of rules in virtually any layer within the harmonic architecture of the piece. By acknowledging chords or scale degrees as candidates for both surface-level elements and modal regions in a larger structure, we are capable of applying the same sort of "logic" universally. On the other hand, it is fairly more common to see certain rules near the surface, like diminished passages and approximations, although those which appear in both levels of complexity have their meaning differ (like a modulation as a whole tonal area or just as a coloration). A case may be made for introducing restrictions which could differentiate certain structural levels, so as to reflect some notion of metric or not to allow certain odd resolution of harmonic sequences – with our tools, an arrangement can be proposed at the cost of perhaps sacrificing the aforementioned reproducibility to some extent.

What should we retain from this section, when suggesting a modal logic framework for harmonic material? Our subject of study will be chord sequences which can be transformed into others: this motivates a possible accessibility relation for our modal logic, besides the temporal one, which is already implicit in the fact that sequences live in a particular timeline. In terms of the building blocks to be considered, we can either view elements of the sequences as "generators" chords + (degree in scale), or scale + (degree of chord). For the sake of efficiency, we will not be regarding inversions, major/minor variants and other information beyond structural harmony.

1.3 Modal logic

We can view modal logic as extension of classical logic which lets us express, depending on the context, a certain modality:[8] in our case, we are interested in the notion of reachability. Syntactically, the recursive definition for the construction of formulas can now allow for a new unary connective $\Box$ and its dual $\Diamond := \neg\Box\neg$. Semantically, they let us argue about the worlds where a formula holds – in particular, whenever a formula holds in all (necessarily) or some (possibly) worlds connected to the one we are evaluating the formula in.

Definition 1.3. *We consider structures (models) of the form* $\mathcal{M} = \langle W, R, V \rangle$*, where* W *is a set of* worlds/states*,* R *a binary* relation *between worlds (responding to the accessibility between them) and* $V : W \to \text{Prop}$ *a mapping which assigns to every world the propositional variables satisfied in it, called* valuation*. We write* $\mathcal{M}, s \vDash \Box\varphi$ *when* φ *holds in every state* t *accessible from* s *(*sRt*).*

[8] See [BB10] for more developed interpretations.

It is also plausible to consider several relations in one model, and label the modal operators in accordance to the relation they reference. From a theoretical perspective, we are interested in studying the decidability and complexity of some problems related to given formulas:

- Model checking (MC): given a formula φ, a model $\mathcal{M} = \langle W, R, V \rangle$ and a state $s \in W$, output whether $\mathcal{M}, s \vDash \varphi$ or not.
- Satisfiability: given a formula φ, output whether there exists a model $\mathcal{M}$ and a state s for which $\mathcal{M}, s \vDash \varphi$. We distinguish:
  - Satisfiability without a fixed grammar (SAT): there exist no restrictions on the grammar underlying the models.
  - Satisfiability for a given grammar (SAT_G)[9]: given a formula φ and a grammar G, output whether there exists a model $\mathcal{M}$ based on G and a state s for which $\mathcal{M}, s \vDash \varphi$.
- Validity (VAL): the dual of Satisfiability[10] – given a formula φ, output whether $\mathcal{M}, s \vDash \varphi$ for every model $\mathcal{M}$ and every state s. We replicate the previous distinction here as well.

Given a logic, the interest of studying each of these decision problems relies on finding out if there exists a procedure (an algorithm, a Turing machine, etc.) which can correctly answer the query for every possible given input. Moreover, if so, one can discuss the complexity of the suggested algorithm. Accounting for a single modal operator, model checking for basic modal logic is decidable and runs in polynomial time: this uses the notion of the *characteristic formula* of a state (of degree n), which identifies its state by the formulas that hold in it up to modal depth n, and is defined recursively as

$$\chi^0_{\mathcal{M},s} = \bigwedge_{p \in V(s)} p \wedge \bigwedge_{p \notin V(s)} \neg p, \quad \text{and} \quad \chi^{n+1}_{\mathcal{M},s} = \chi^0_{\mathcal{M},s} \wedge \Big(\bigwedge_{t | sRt} \Diamond \chi^n_{\mathcal{M},t} \Big) \wedge \Big(\Box \bigvee_{t | sRt} \chi^n_{\mathcal{M},t} \Big).$$

Notice as well how, in case that satisfiability of the logic is decidable, model checking may be reducible to it: within a model $\mathcal{M}$ and a state s, it is precisely the substructure determined by s ($\mathcal{M}$ restricted to the set of states reachable from s by some R^n) that determines the truth value of some formula evaluated in $\mathcal{M}, s$, $\Box$ is the only modality. For logics like BML, it is possible to encapsulate such information in a formula by using characteristic $\chi^n_{\mathcal{M},s}$, although this is not ensured for other combinations of logics.

These are well studied properties ([Ng05]). Satisfiability of basic modal logic is also decidable, as granted by the filtration method. Given a set of modal formulas

[9] Note the *non-italic* 'G' is not meant to correspond to a grammar.

[10] In extensions of classical logic.

Γ and a model $\mathcal{M}$, we can identify states which satisfy the same formulas in Γ through an equivalence relation $\tilde{R}_\Gamma$. Letting a structure $\tilde{\mathcal{M}}$ whose states are the equivalence classes of $\tilde{R}_\Gamma$ and its valuation the valuation of $\mathcal{M}$ over any representative of the class, we call $\tilde{\mathcal{M}}$ a *filtration* of $\mathcal{M}$ with respect to Γ if, for every state s in $\mathcal{M}$ and any formula φ, $\mathcal{M}, s \vDash \varphi$ iff $\tilde{\mathcal{M}}, [s]_\Gamma \vDash \varphi$. It is enough to require that sRt implies $[s]_\Gamma \tilde{R}_\Gamma [t]_\Gamma$ to obtain a *minimal* filtration. The decidability of the satisfiability problem for some φ can then be reduced to checking the satisfiability of φ in the possible filtrations of our structures: since we are interested in a formula with a certain (finite) modal depth, the size of such filtrations will be bounded, and it will be enough to perform model checking in each of them. Note, though, that the complexity of this method will be at least exponential in general.

Similarly to how we have regarded states as the objects which formulas make reference to, one can also follow [DGL16] in introducing a syntax for states and paths altogether. Later on, at the cost of formally referring to states as initial positions of some path, we will interpret formulas for states as formulas referring to paths, thus arguments and proofs will be greatly simplified.

Definition 1.4. *Recall that a path π is a sequence of states $\pi(0) = s_0, \pi(1) = s_1, \pi(2) = s_2, \ldots$ for which $(s_i, s_{i+1}) \in R$. We define* state formulas *φ and* path formulas *ϑ by simultaneous recursion as:*

$$\begin{array}{lllllllllll} \varphi := & p \in \mathrm{Prop} & | & \neg\varphi & | & \varphi \wedge \varphi & | & \mathsf{A}\vartheta & | & \mathsf{E}\vartheta \\ \vartheta := & \varphi & | & \neg\vartheta & | & \vartheta \wedge \vartheta & | & \mathsf{X}\vartheta & | & \vartheta \mathsf{U} \vartheta \end{array}$$

Since A (and its dual, E) stand for the basic modal operators, we will interpret A as $\mathcal{M}, s \vDash \mathsf{A}\vartheta$ iff every path π starting on s ($\pi(0) = s$) satisfies ϑ. State formulas are evaluated on paths by checking their truth value on $\pi(0)$. On the other hand, $\mathsf{X}\vartheta$ is satisfied by some path π iff ϑ is satisfied by $\pi[1, +\infty)$. Finally, $\vartheta \mathsf{U} \vartheta'$ holds in π iff there exists some natural n for which $\mathcal{M}, \pi[n, +\infty) \vDash \vartheta'$ and ϑ holds for every $\pi[i, +\infty)$, $i < n$. It can be illustrative to observe that the names for the operators are short for *all* (paths), *every* (paths), *next* (state) and *until* (state).

The previous construction is usually named the Full Computation Tree Logic (CTL*). Theories in CTL* are capable of modeling structures which represent branching-time, understanding paths as possible linear sequences of time flows. Thus, this kind of temporal logic is a generalization of LTL (linear temporal logic) and CTL, which subsumes reachability logic (TLR) and, clearly, basic modal logic. The fact that we can view state formulas as path formulas will allow us to work exclusively with path formulas, as $\mathcal{M}, \pi$ will satisfy some $p \in \mathrm{Prop}$ iff $p \in \pi(0)$, and $A\vartheta$ iff every path which shares the first state with π satisfies ϑ.

Remark 1.5. CTL* has nice properties with respect to linear temporal logic. First, due to bisimilarity, any CTL* state formula which is satisfiable is so too in a tree model. Every valid formula in LTL is also valid in CTL* (otherwise, there would exist a path not satisfying it, a counterexample in LTL). In addition, model-checking for CTL* can be reduced to that of LTL, and due to the fact that the latter is PSPACE and so is the reduction, MC(CTL*) is PSPACE too. However, despite being decidable, satisfiability is more complex (2EXPTIME), and model-checking (in fact, global too) in CTL can be done in polynomial time.

We are interested in studying the attributes of several systems, and for that it will be common that we consider *fragments* of our logic, not only derived from the restriction of rule sets, but also of the modal operators used – this will allow us to have stronger, more specific results. On the other hand, we may also consider strengthened versions of modal operators; for instance, in Section 2.1, we will present $\Diamond_1$ to declare accessibility from one path onto another by the application a single rule, but we will also write $\Diamond$ for the transitive closure of $\Diamond_1$: adding this operator to a linear temporal logic effectively integrates an "orthogonal" modality, which could be viewed as some reduct of CTL*, since $\Diamond\psi$ functions as $\top \mathsf{U}' \psi$ for some until operator U'.

Once we have chosen our particular logic and context, our next step will be to formulate the conditions for the models that validate certain conditions to be musically relevant. We can either opt to propose a general axiomatic for which any model of the theory is a desired sequence, or pinpoint the existence of certain paths inside the model which make sense as acceptable harmonic sequences. Nevertheless, it will be wise to introduce some metalanguage conditions (such as finitude of models, etc.), besides any particular requirement that we *are* able to express through our logics.

Chapter 2

Theoretical construction

Throughout the development of modal logic, several situations to be modelled have given rise to a multitude of systems and refinements of the basic formulations in Section 1.3. These include logics whose frames and operators satisfy additional properties, and combinations of logics in the form of products, fusions, etc. In this chapter, we set out to justify our choice of models for a suitable reinterpretation of grammars for musical sequences. We also present some results related to derivability and complexity.

2.1 Our model

A first naive translation of the workings of the introduced grammars in terms of modal logic could be the following: given a fixed set of derivation rules, we can interpret as a modal operator the relation which associates chord sequences and any of their transformations upon a single rule application. Thus, the language of our grammar could be seen as the set of all leaves from the trees which are models of the logic and are rooted in a state with degree I.

We will first consider schemata of arbitrary rules and progressively focus on those which satisfy certain properties: for instance, that they apply to a single chord, do not remove chords, and are at most binary and left-branching (any added chord appears on the left of the one the rule was applied to). So, eventually, our rules may be of the form $vxu \to v(yx)u$, but for now let us suppose they are just relations between chord sequences.

An attempt

It is also possible to approach the reinterpretation of our grammar as a direct, shoehorned translation into some expansion of CTL*. However, this results in a

burdensome endeavor which uses elements much more "technical" than natural in the context of modal logic. Let the set of propositional variables represent scale degrees (*Deg*) together with some reserved characters, and consider a collection of modality relations labeled by keys (in the set *Key*). In the context of Full CTL, we may seek to construct a theory satisfied only by models whose paths are associated with valid chord sequences in the grammar – that is, for example, we can allow infinite branches of states with empty valuations.

Example 2.1. Let $\varepsilon := \bigwedge_{d \in Deg} \neg d$. Given some key k, the operator AX_k is semantically interpreted as "for every immediate successor in the modality k", while EX_k is its existential dual. Some desirable axioms of our theory would be (1) that every state has at least a chord or is a *stopping state* (an empty state, hence it does not have any non-empty successor); and (2) that every immediate predecessor of a stopping state is a first degree, i.e. that every non-empty state $\pi(0)$ connected to a stopping state via a key k must have $V(\pi(0)) = \{I\}$ and not have any successor in the context of another key.

$$(1)\ \ \forall \pi\, \mathcal{M}, \pi \vDash \varepsilon \rightarrow \bigwedge_{k \in Key} \mathsf{AX}_k \varepsilon$$

$$(2)\ \ \forall k \forall \pi\, \mathcal{M}, \pi \vDash \bigvee_{d \in Deg} d \wedge \mathsf{EX}_k \varepsilon \rightarrow I \wedge \bigwedge_{d \neq I} \neg d \wedge \bigwedge_{k' \neq k} \mathsf{AX}_{k'} \bot$$

It is conceivable that this rudimentary formalization makes things difficult insomuch as it relegates the translation of rule applications to the axiomatic of the theory instead of to a modal relation. This is especially true for the (attempted) remaining axioms, which either subsume the existence of special "transition states" which prevent unary rules from duplicating chords, or introduce a variant of the *until* operator in order to express that, if a chord appears, it is because a binary rule has already been applied somewhere ahead.

The actual logic

We ultimately forfeit our objective to translate the grammar in terms of an "elementary" modal logic, having our focus shifted from states to paths and relying on a formulation which involves two accessibility relations. For this purpose, every time we address a concrete problem or an example, we will consider a fixed set of derivation rules (which determine the newer relation) and specify, within the corresponding logic, which sort of models offer a meaningful answer to our inquiry (e.g. we could restrict our search to structures which consist exclusively of paths derived from a certain initial state).

In that sense, hoping to provide a more natural interpretation of the logic which makes it easier to work with, we dismiss having any (meta)condition force all of our models to contain every valid sequence of the grammar, or even to contain only valid sequences. However, it is legitimate (and feasible) to question whether a model is complete with respect to the grammar, and under which conditions that happens. Notice that, in any case, our following definition for the relations will already reflect in the models some of the properties of the derivation rules and of the underlying mechanisms.

The flavor of our logics will resemble that of products, and they will be reducible to a certain extent. In order present our formulas, let us first introduce the structures which will model our logic.

Definition 2.2. *Let R a fixed set of derivation rules of a grammar G. Our models are 4-ples $\mathcal{M} = \langle W, \to, \Rightarrow, V \rangle$, where W is the set of states, $\to \subseteq W \times W$ acts as the* (immediate) successor *relation of paths corresponding to harmonic sequences, and $V : W \to Deg \times Key$ is a valuation. Let Π be the set of $\to$-paths in $\mathcal{M}$; then we define $\Rightarrow \subseteq \Pi \times \Pi$ as the collection of pairs (π, π') for which $(V(\pi(i)))_{0 \leq i < |\pi|} \Rightarrow_G (V(\pi'(j)))_{0 \leq j < |\pi'|}$ for some rule in R.*[1]

Note that, so far, valuations assign no more than a chord to each state. We might refer as *valuation of a path* to the sequence of valuations of its states. Our formulas will refer to paths in the structures we handle. The syntax of our (path-)formulas is summarized by the following recursive definition:

$$\varphi ::= \ p \in \text{Prop} \mid \neg\varphi \mid \varphi \wedge \varphi \mid \mathsf{X}\varphi \mid \varphi \mathsf{U} \varphi \mid \Diamond_1 \varphi \mid \Diamond \varphi$$

To introduce the semantics for each of the operators, recall that the *next* and *until* operators are inherited from LTL, and that $\Diamond$ expands through transitivity an otherwise basic logic related to $\Diamond_1$. Let $\mathcal{M} = (W, \to, \Rightarrow, V)$ and $\pi \in \Pi$. We write:

- $\mathcal{M}, \pi \vDash p$ (for $p \in \text{Prop}$) iff $p \in V(\pi(0))$.
- $\mathcal{M}, \pi \vDash \neg\varphi$ iff $\mathcal{M}, \pi \nvDash \varphi$.
- $\mathcal{M}, \pi \vDash \varphi \wedge \psi$ iff $\mathcal{M}, \pi \vDash \varphi$ and $\mathcal{M}, \pi \vDash \psi$.
- $\mathcal{M}, \pi \vDash \mathsf{X}\varphi$ iff $\mathcal{M}, \pi[1, +\infty) \vDash \varphi$.
- $\mathcal{M}, \pi \vDash \varphi \mathsf{U} \psi$ iff there exists some $n \geq 0$ such that $\mathcal{M}, \pi[n, +\infty) \vDash \psi$ and, for every $i < n$, $\pi[i, +\infty) \vDash \varphi$.
- $\mathcal{M}, \pi \vDash \Diamond_1 \varphi$ iff there exists some $\pi' \in \Pi$ such that $\pi \Rightarrow \pi'$ and $\mathcal{M}, \pi' \vDash \varphi$.
- $\mathcal{M}, \pi \vDash \Diamond \varphi$ iff there exist $r \geq 0$, $\pi_0 = \pi, \pi_1, \ldots, \pi_r \in \Pi$ such that $\pi_j \Rightarrow \pi_{j+1}$ and $\mathcal{M}, \pi_r \vDash \varphi$.

[1] Notice $\Rightarrow_G$ instead of its transitive closure $\Rightarrow_G^*$ in the definition.

Note that the $\Diamond$ operator acts as the (reflexive) transitive closure of $\Diamond_1$, and that it is not enough that the goal path is in the model, but also every intermediate one. We denote our full logic by $\mathcal{L}_{\mathsf{U}}$, and the fragment without formulas containing the U operator by $\mathcal{L}_{\mathsf{X}}$. Let us discuss a brief example:

Example 2.3. Let $W = (w_1^1, w_2^1, w_2^2, w_3^1, w_3^2, w_3^3)$, $\rightarrow = \{(w_2^1, w_2^2), (w_3^1, w_3^2), (w_3^2, w_3^3)\}$ and $V = \{(w_1^1, I_{\mathrm{C}}), (w_2^1, IV_{\mathrm{C}}), (w_2^2, I_{\mathrm{C}}), (w_3^1, IV_{\mathrm{C}}), (w_3^2, V_{\mathrm{C}}), (w_3^3, I_{\mathrm{C}})\}$. Thus, the set of paths is $\Pi = \{(w_1^1), (w_2^1), (w_2^2), (w_3^1), (w_3^2), (w_3^3),\ (w_2^1, w_2^2), (w_3^1, w_3^2), (w_3^2, w_3^3),\ (w_3^1, w_3^2, w_3^3)\}$. Suppose R consists of the plagal cadence rule ($I \rightsquigarrow IV\ \ I$) and the schema of diatonic fifths ($X \rightsquigarrow \Delta/X\ \ X$): then, $\Rightarrow$ has the pairs:

- $(w_1^1) \Rightarrow (w_2^1, w_2^2)$, granted by $I \rightsquigarrow IV\ \ I$.
- $(w_2^2) \Rightarrow (w_3^2, w_3^3)$, given by $I \rightsquigarrow V\ \ I$.
- $(w_2^1, w_2^2) \Rightarrow (w_3^1, w_3^2, w_3^3)$, as witnessed by the inclusions of the paths above in the current ones.
- $(w_1^1) \Rightarrow (w_3^2, w_3^3)$, again by $I \rightsquigarrow V\ \ I$.

In $\langle W, \rightarrow, \Rightarrow, V\rangle$, the formula $\Diamond\psi := \Diamond(IV_{\mathrm{C}} \wedge \mathsf{X}(V_{\mathrm{C}} \wedge \mathsf{X}I_{\mathrm{C}}))$ holds in the paths (w_1^1) and (w_2^1, w_2^2), while $\Diamond_1\psi$ only holds in (w_2^1, w_2^2). However, $\Diamond I_{\mathrm{C}}$ is unsatisfiable in this structure.

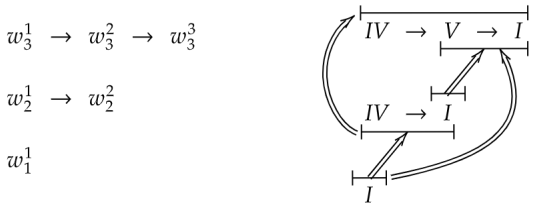

Figure 2.1: Paths and $\Diamond_1$ accessibility relation from the previous example.

Observe the particularity of this model: it allows us to visualize the actual transformations between paths, which is useful to comprehend how chord sequences are derived in the original grammar. Nevertheless, the translation between parsing trees and modal structures is seldom literal, as $(w_1^1) \Rightarrow (w_3^2, w_3^3)$ does not take place in the grammatical derivation of $(IV_{\mathrm{C}}, V_{\mathrm{C}}, I_{\mathrm{C}})$. In general, despite this kind of models being useful for our theoretical purposes and for their applications, we must account for the existence of much more convoluted structures.

Remark 2.4. Given a grammar G with its fixed rule set (and initial variables corresponding to first degrees), its language $L(G)$ corresponds to the collection of valuations of the paths accessible from (w) in some model where $V(w) = I_k$, for

any $k \in Key$. That is, the valuations of goal paths which witness that, in some structure, the formula $I_k \wedge \mathsf{X}\bot \wedge \Diamond\psi$, for some $k \in Key$ and some ψ.[2]

Thus, while our grammar might not express every sequence, it is possible that they are included in a model of $\mathcal{L}_{\mathsf{U}}$ or $\mathcal{L}_{\mathsf{X}}$. But this motivates some decision problems, such as ask examining if a given sequence can be transformed into another. In a general context, this belongs to an instance of the satisfiability problem (for a fixed grammar): given an initial sequence $S = (d_{1k_1}, \ldots, d_{nk_n})$ and a goal sequence $S' = (d'_{1k'_1}, \ldots, d'_{mk'_m})$, it is enough to check if there exists some model in which the the following formula is satisfied (let us use $\chi_{S \Rightarrow^* S'} := \chi_S \wedge \Diamond\chi_{S'}$ as shorthands):

$$d_{1k_1} \wedge \mathsf{X}(d_{2k_2} \wedge \mathsf{X}(\ldots \wedge \mathsf{X}d_{nk_n})) \wedge \Diamond(d'_{1k'_1} \wedge \mathsf{X}(d'_{2k'_2} \wedge \mathsf{X}(\ldots \wedge \mathsf{X}d'_{mk'_m}))).$$

We now introduce a basic model which has exactly a representative for all possible diatonic chords – in other words, where every possible valuation appears in one and only one state. It lacks the illustrative traits of the previous structure, in which the repeated valuations in distinct states allowed for a clearer interpretation of the notion of derivability. Nevertheless, being a finite model which is as connected as it can be, it is an optimal structure in which we can enquire general questions about valid sequences.

Definition 2.5. *Let G be a grammar. Consider $W = \{w_{d_k} \mid d \in Deg, k \in Key\}$, $\rightarrow = W \times W$ and $V : w_{d_k} \mapsto (d,k)$,[3] with $\Rightarrow$ defined accordingly. We call $\mathcal{M}_0 = \langle W, \rightarrow, \Rightarrow, V\rangle$ the* basic product model.

Proposition 2.6. *Let G a fixed grammar, $\mathcal{M}_0$ its basic product model. The set of accessible paths from some (w_{I_k}) is complete with respect to $L(G)$, in the sense that for every finite chord sequence S, S is in the language of G if and only if S witnesses that $\mathcal{M}_0, (w_{I_k}) \vDash \chi_{(I_k) \Rightarrow^* S}$ for some $k \in Key$.*

Proof. On the one hand, if S belongs to $L(G)$, then there are some $r \geq 0$, $k \in Key$ and $S_0 = (I_k), S_1, \ldots, S_r = S$, such that $S_i \Rightarrow_G S_{i+1}$. Since $(W, \rightarrow)$ is a complete digraph, every finite sequence of states of $\mathcal{M}_0$ exists as a path in the model, so we can find $\pi_0, \ldots, \pi_r$ whose valuations are $S_0, \ldots, S_r$. Note that the definition of $\Rightarrow$ entails that $\pi_i \Rightarrow \pi_{i+1}$, since the valuation of π_{i+1} is derivable in G from the valuation of π_i. Hence, $\mathcal{M}_0, (w_{I_k}) \vDash \Diamond\chi_S$. For the converse direction, if $(w_{I_k}), \pi_1, \ldots, \pi_r = S$ witness that $\chi_{(I_k) \Rightarrow^* S}$ holds, simply observe that (I_k), $V(\pi_1)$,..., $V(\pi_r)$ is precisely the derivation of S in G. □

[2] Here we interpret $\mathsf{X}\bot$ as $\pi(1)$ being empty, i.e., our path π not having a second state in which any chord exists.

[3] Although we will often identify states with their valuations.

Normally, $\mathcal{L}_X$ will serve for most purposes in grammatical common practice, such as parsing of sequences and interderivability. In some of these cases, there will be general algorithms which operate in PSPACE – but we will favor $\mathcal{L}_X$ over $\mathcal{L}_U$ when possible due to its better decidability conditions. Moreover, since some cases of model checking may be reduced to satisfiability, it will be convenient to work on fixed models such as $\mathcal{M}_0$.

2.2 Properties

We exhibit some results and descriptions which will help us contextualize part of the sections in the main chapters of this thesis. While they are not strictly applied in any of the presented proofs, they can be thought of as a first exploration of some of the techniques used, be it on how to decide the satisfaction of a formula in a given context, or on the refinement of some approaches to adjust them to a certain category of grammar, structure, etc.

Checking properties in certain models

It is often convenient to study the satisfiability of a formula in a certain class of models, so that more optimal methods can be applied. One of these is the *finite model property*, which asserts that any formula that is not a theorem of a logic is actually not valid in some finite model (mentioned in several contexts, such as [Ga06]). Given that our definition the logic was made in semantic terms, we can think of similar properties which imply every satisfiable formula being satisfied in a structure finite in some way.[4]

Usually, it is not enough to ensure that some logic has the FMP in order to show that the satisfiability problem is decidable ([CGP01]), as it is the case of the following result. Instead, we often need some bound on the size of the obtained models, which depends on the given formula, so that we may generate and test every one of them.

Definition 2.7. *Let $\varphi \in \mathcal{L}_X$ and $Sub'(\varphi) = \{(0,\varphi),\ldots\}$ an indexation of its subformulas. For every $(k,\phi) \in Sub'(\varphi)$ (abridged as ϕ_k), denote:*

- *If $\phi = p \in \mathrm{Prop}$, let $\Pi^+(\phi_k) = \{(\{p\})\}$ and $\Pi^-(\phi_k) = \{(\{\overline{p}\})\}$.*
- *If $\phi = \neg\psi \neq \neg\Diamond\psi'$ with index i, let $\Pi^+(\phi_k) = \Pi^-(\psi_i)$ and $\Pi^-(\phi_k) = \Pi^+(\psi_i)$.*

[4] In Chapter 3, we will come across models which are presented in a finite manner, but can hide infinite relations with finite schemata or infinite relations which cannot be represented without some unbounded operator.

– *If* $\phi = \psi \wedge \chi$ *with indices* i *and* j, *let* $\Pi' = \Pi^+(\psi_i) \cup \Pi^-(\psi_i) \cup \Pi^+(\chi_j) \cup \Pi^-(\chi_j)$, $m = \max\{|s| \mid s \in \Pi'\}$ *and* $V = \mathcal{P}(Var(\Pi') \cup \overline{Var}(\Pi'))$. *Define* $\Pi^+(\phi_k) =$

$$\{s \in V^m \mid \exists s_0 \in \Pi^+(\psi_i)\, \exists s_1 \in \Pi^+(\chi_j)\, \forall n \\ (s(n) = s_0(n) \cup s_1(n) \ \wedge\ \forall p \in \mathrm{Prop}\ \neg(p, \overline{p} \in s_0(n) \cup s_1(n)))\}$$

and $\Pi^-(\phi_k) =$

$$\{s \in V^m \mid \forall n \forall p \in \mathrm{Prop}\ \neg(p, \overline{p} \in s(n))\} \setminus \Pi^+(\phi_k).$$

– *If* $\phi = \mathsf{X}\psi$ *with index* i, *let* $\Pi' = \Pi^+(\psi_i) \cup \Pi^-(\psi_i)$ *and* $V = \{v \in \mathcal{P}(Var(\Pi') \cup \overline{Var}(\Pi')) \mid \forall p \in \mathrm{Prop}\ \neg(p, \overline{p} \in v)\}$. *Define* $\Pi^+(\phi_k) = \{(v)^\frown s' \mid v \in V \wedge s' \in \Pi^+(\psi_i)\}$ *and* $\Pi^-(\phi_k) = \{(v)^\frown s' \mid v \in V \wedge s' \in \Pi^-(\psi_i)\}$.
– *If* $\phi = \Diamond\psi$ *with index* i, *such that* ϕ_k *is not the subformula of some* $\phi'_j = \neg\Diamond\psi_i$, *introduce a variable* p_i *and let* $\Pi^+(\phi_k) = \{(\{p_i\})\}$ *and* $\Pi^-(\phi_k) = \{(\{\overline{p_i}\})\}$.
– *If* $\phi = \neg\Diamond\psi$ *with index* i, *let* $\Pi^+(\phi_k) = \{(\{\overline{p_j}\}) \mid \psi_j \in Sub'(\varphi)\}$ *and* $\Pi^-(\phi_k) = \{(\{p_j\}) \mid \psi_j \in Sub'(\varphi)\}$.

Each Π^+[5] (respectively, Π^-) contains the set of representation of paths which satisfy (resp. do not satisfy) a formula. In the case of $\Diamond$, it suffices to note if a path can be accessed, so we can just introduce a fresh variable representing every instance ψ_i of a ψ. We have left out the case $\Diamond_1$, because the purpose of this definition is to illustrate the inductive nature of some of the arguments below. The next proposition ensures that we can check the satisfiability of a formula for a fixed grammar within a finite set of well-behaved models.[6]

Proposition 2.8. *Given a grammar* G *with rule set*[7] R *and a formula* φ *for* $\mathcal{L}_{\mathsf{X}}$, *there exist* $m_\varphi \in \mathbb{N}$ *and* $\{\mathcal{M}_{\varphi,i} \mid i \leq m_\varphi\}$ *such that* φ *is satisfiable if and only if it holds in a path of some* $\mathcal{M}_{\varphi_i}$.

Proof. We prove by induction on $d_\Diamond(\varphi)$ that there is some $l(\varphi) \in \mathbb{N}$ for which there exist some $\mathcal{M}, \pi_0$ having $\mathcal{M}, \pi_0 \vDash \varphi$ if and only if there exist $\mathcal{M}, \pi_0$, with $\mathcal{M}, \pi_0 \vDash \varphi$, for which every path in $\Pi_\mathcal{M}$ has length at most $l(\varphi)$. Given that it is actually the valuations of paths what determine the satisfaction of formulas, for each formula φ there will exist $k = \sum_{i=0}^{l(\varphi)} |\mathrm{Prop}|^i$ and $\pi_1, \ldots, \pi_k$ such that the structures we consider only contain such paths; that is, there are up to 2^k possible models to be checked.

We start with the case $d_\Diamond(\varphi) = 0$, and show the claim by (secondary) induction on $d_{\Diamond_1}(\varphi)$:

[5] Not to be confused with the irreflexive transitive closure of a relation, R^+.

[6] In the sense that no infinite paths are allowed (even those presented as a circular graph), so model checking can be performed, as seen in the upcoming chapter.

[7] Recall that we only regarded grammars with finite rule schemata.

- For $d_{\Diamond_1}(\varphi) = 0$, the only possible modal operator in φ is X, whence the problem is reducible to SAT(LTL), which is PSPACE and involves exactly the first $d_{\mathsf{X}}(\varphi)$ positions of any path to check.
- Assume the claim has been shown for every $0 \leq d < d_{\Diamond_1}(\varphi)$, and let $\varphi = \theta(\Diamond_1\psi_1, \ldots, \Diamond_1\psi_k)$ for some $\theta(x_1, \ldots, x_k)$ with null $\Diamond_1$-depth. By IH2, for each of the ψ_i, it is enough to test models with paths up to length $l(\psi_i)$. Observe the following commutativity properties for formulas ψ, ψ':

$$\mathsf{X}(\psi \wedge \psi') \equiv \mathsf{X}\psi \wedge \mathsf{X}\psi', \quad \mathsf{X}\neg\psi \equiv \neg\mathsf{X}\psi.$$

Hence, there exists a disjunctive formula $\theta' := \bigvee_{a \in A} \bigwedge_{b \in B_a} \mathsf{X}^{x(a,b)}\theta_{a,b}$ equivalent to θ, such that $\theta_{a,b}$ is either $p \in \mathrm{Prop}$, $\neg p$, x_i or $\neg x_i$. So $\varphi \equiv \theta'[x_i \leftarrow \Diamond_1\psi_i]$ and it suffices to verify if any of the $\bigwedge_{b \in B_A} \mathsf{X}^{x(a,b)}\theta_{a,b}[x_i \leftarrow \Diamond_1\psi_i]$ is satisfiable.

Fix some $a \in A$. We work under the premise[8] that rule sets R are finite in the sense that, for some finite $R_0 \subseteq R$, every $(s, s') \in R$ is of the form $uvw \rightsquigarrow uv'w$, for some $(v, v') \in R_0$. Then, whenever we apply some rule $V(\pi) = uvw \rightsquigarrow uv'w = V(\pi')$ with $|u| \geq \max\{x(a,b) \mid b \in B_a\} + \max\{l(\psi_i) \mid i \leq k\}$, we obtain that $\mathsf{X}^{x(a,b)}\theta_{a,b}[x_i \leftarrow \Diamond_1\psi_i]$ holds in π exactly when it holds in π'. Therefore, our result holds for paths of length $l(\varphi) = \max\{l_a \mid a \in A\}$, where

$$l_a := \max\{x(a,b) \mid b \in B_a\} + \max\{l(\psi_i) \mid i \leq k\} + \max\{|v| \mid \exists v' \, ((v, v') \in R_0)\}.$$

Assume that the initial claim holds for every formula of $\Diamond$-depth less than $d_{\Diamond}(\varphi) > 0$, and suppose again that $\varphi = \theta(\Diamond\psi_1, \ldots, \Diamond\psi_k)$ for some $d_{\Diamond}(\theta) = 0$. Let $l(\psi_i)$ the maximum path length allowed for each ψ_i, and θ be equivalent to some disjunctive $\theta' = \bigvee_{a \in A} \bigwedge_{b \in B_a} \mathsf{X}^{x(a,b)}\theta_{a,b}$, where $\varphi \equiv \theta'[x_i \leftarrow \Diamond\psi_i]$ and each $\theta_{a,b} \notin \{x_i \mid i \leq k\} \cup \{\neg x_i \mid i \leq k\}$ is of null $\Diamond$-depth. We proceed by secondary induction on $d_{\Diamond_1}(\theta')$, and focus on the base case, since the induction step is analogous to the previous one.

So let $d_{\Diamond_1}(\theta) = d_{\Diamond_1}(\theta') = d_{\Diamond_1}(\theta_{a,b}) = 0$. Our goal is to show the simultaneous accessibility or non-accessibility of the paths determined by some given formulas ψ'_α, ψ''_β. That is, letting $\psi := \bigwedge_\alpha \Diamond\psi'_\alpha \wedge \bigwedge_\beta \neg\Diamond\psi''_\beta$, we want to prove the existence of some $p \in \mathbb{N}$ for which there are $\mathcal{M}, \pi_0 \vDash \psi$ iff there are $\mathcal{M}, \pi_0 \vDash \psi$ such that $|\pi| \leq p$ for all $\pi \in \Pi_{\mathcal{M}}$.

Recall the notation χ_u for characteristic formulas of sequences and consider

[8] Discussed in detail at the beginning of the next chapter.

the set of tuples of paths

$$L_{l,m;l',m'}(G,\pi_0,p) := \{(v_1,\dots,v_m;v'_1,\dots,v'_{m'}) := (u_1 \restriction [0,l_1),\dots,u_m \restriction [0,l_m);\\ u'_1 \restriction [0,l'_1),\dots,u_{m'} \restriction [0,l'_{m'})) \mid (\forall j\, l_j \leq l \wedge \forall j'\, l'_{j'} \leq l') \wedge \\ \exists \mathcal{M}\, (\pi_0 \in \Pi_{\mathcal{M}} \wedge \forall \pi \in \Pi_{\mathcal{M}}\, |\pi| \leq p \wedge \mathcal{M},\pi_0 \vDash \bigwedge_{j\leq m} \Diamond \chi_{v_j} \wedge \bigwedge_{j'\leq m'} \Diamond \chi_{v'_{j'}})\}.$$

Because there only exist finitely many combinations of m (resp. m') sequences of length less than l (l'), the set $\bigcup_{|\pi_0|\leq p_0} \bigcup_{p\leq p_0} L_{l,m,l',m'}(G,\pi_0,p)$ stabilizes for some natural p_0. Thus, for $\bigwedge_{b\in B_a} \mathsf{X}^{x(a,b)}\theta_{a,b}[x_i \leftarrow \Diamond\psi_i]$, our property holds at $l_a := p_0(a) + \max\{x(a,b) \mid b \in B_a\}$, for some $p_0(a)$ determined by $l = l' = \max\{l(\psi_i) \mid i \leq k\}$ and $m + m' = |B_a|$. Therefore, $l(\varphi) = \max\{l_a \mid a \in A\}$. □

Observe that the fact that a sequence of set stabilizes does not provide information on when it does so, so this result is not immediately useful in terms of computation, i.e. this proposition does not grant a proof of satisfiability. However, some of the procedures appear in the next chapters (in particular, model checking for a particular segment of $\mathcal{L}_{\mathsf{X}}$, which does feature some more initial information to work on).

Classifications

It is of our interest to narrow down where some given decision problems belong to a certain complexity, or are decidable at all. Several restrictions on the grammars and structures used will affect this performance. Commonly, the more general a considered collection is, the harder the problems become in terms of complexity, although it is possible that enforcing less conditions to a decision problem makes it computable.

For instance, although we mentioned that some of the most influential grammars for harmonic sequences are context-free, we will describe other classes which yield varied results ([Hu20]).

Definition 2.9. *Let $G = \langle V,\Sigma,P,S\rangle$ be a grammar. The following descending classification in terms of the nature of the rule schemata is often know as* Chomsky hierarchy*:*

- *G is* unrestricted *if every rule is of the form $\alpha \rightsquigarrow \beta$, where $\alpha,\beta \in (V\cup\Sigma)^*$ and $\alpha \neq \varepsilon$.*
- *G is* context-sensitive *if every rule is of the form $\alpha A\alpha' \rightsquigarrow \alpha\beta\alpha$, where $A \in V$ and $\alpha,\alpha',\beta \in (V\cup\Sigma)^* \setminus \{\varepsilon\}$.*
- *G is* context-free *if every rule is of the form $A \rightsquigarrow \beta$, where $A \in V$ and $\beta \in (V\cup\Sigma)^* \setminus \{\varepsilon\}$.*

– *G is* left-regular *(resp.* right-regular*) if every rule is of the form $A \rightsquigarrow Ba$ (resp. $A \rightsquigarrow aB$), where $A, B \in V$ and $a \in \Sigma$.*

In some references, the rule $S \rightsquigarrow \varepsilon$ is accepted in every type.

Unrestricted grammars correspond to computable languages, that is, those which are recognized as the set of solutions of a decision problem coded by a Turing machine or any equivalent computation model ([CPP14]). This is not to say that we cannot use an unrestricted grammar as a means to represent undecidable problems, as we will usually describe them as a membership problem (often undecidable) of a certain sequence for a given grammar.

Some of the reductions to undecidable problems will employ context-sensitive grammars, although not every one of them: the rules of the form $\alpha'\alpha\alpha'' \rightsquigarrow \alpha'\varepsilon\alpha''$ are not in the grammar when $\alpha \neq S$. However, since the sensitivity to context implies that the derivation of a variable depends on its surroundings, the swapping of contiguous letters is allowed ([Si12]). The rule "$AB \rightsquigarrow BA$" can be obtained with the set of production rules

$$A\mathbf{B} \rightsquigarrow A\mathbf{X}, \quad \mathbf{A}X \rightsquigarrow \mathbf{Y}X, \quad Y\mathbf{X} \rightsquigarrow Y\mathbf{A}, \quad \mathbf{Y}A \rightsquigarrow \mathbf{B}A,$$

where the bold variables represent the one to which a rule is applied, and the derivation is the following:

$$AB \Rightarrow_G AX \Rightarrow_G YX \Rightarrow_G YA \Rightarrow_G BA$$

Context-sensitivity also applies to the situation of substituted words, and in fact, rules defined as the substitution $\alpha \rightsquigarrow \beta$ of a word by another can sill belong to context-sensitive grammars as long as $0 < |\alpha| \leq |\beta|$. In general, both strict order inversions and overall substitutions of words cannot take place in context-free grammars ([MP71]).

Another possible set of restrictions arises from the fact that the accessibility operator $\Diamond$ can become difficult to handle in the context of complexity. Given that the sequences generated through a grammar G from a given word may be arbitrarily large, we might restrict our problems to classes of grammars where the manipulation of $\Diamond$ does not blow up the amount of information needed for a checking.

We can also restrict the operators used in formulas for a particular decision problem. From the musical standpoint, it usually is convenient to consider a class of formulas of a given form, such as $p \wedge \Diamond\psi$ with $\psi \in \mathrm{LTL} \restriction \mathrm{X}$, which refers to the problem of deciding if a certain sequence described by a formula ψ is derivable from another which starts with p.

The latter works well in conjunction with particular fragments of a logic which satisfies a certain property. For example, suppose that an operator $*$ commutes well with $\Diamond_1$ ($\Diamond_1 * \phi \leftrightarrow *\Diamond_1\phi$), $\Diamond$ and $\neg$, and it also distributes with respect to $\wedge$ ($*(\phi \wedge \psi) \leftrightarrow *\phi \wedge *\psi$). Then, it seems natural to enquire if the satisfiability of the fragment can be reduced to that of the case without $*$, given that the branches of the derivation tree of any given formula will always end in some sequence of the form $(X^i p, \dots X^i p, p)$.

At any rate, most of the properties described are not general enough, as in the case of the *Church-Rosser property* for multi-modal logics ([GKWZ03]) does not hold for structures based on certain grammars. Consider $G = \langle V, \Sigma, P, I \rangle$, with

$$V = \{I, ii, \dots, vii\}, \qquad \Sigma = \{\mathrm{C}, \mathrm{d}, \mathrm{e}, \mathrm{F}, \mathrm{G}, \mathrm{a}, \mathrm{b}\}, \qquad P = \{X \rightsquigarrow X\,\Delta / X\}.$$

This "reverse" representation of a traditionally left-branching grammar is relevant in some reductions from undecidable problems, as seen in the next chapter. With our modal operators, the property reads $\mathsf{X}\neg\Diamond_1\neg p \rightarrow \neg\Diamond_1\neg\mathsf{X}p$. Notice that, for a given sequence FC,

$$\mathrm{FA} \vDash \mathsf{X}\neg\Diamond_1\neg\mathrm{A}, \quad \text{as} \quad \mathrm{A} \nvDash \Diamond_1\neg\mathrm{A},$$

because any rule applied to a character preserves it in the same position. On the other hand, by applying the only possible rule to F, we obtain

$$\mathrm{FA} \Rightarrow_G \mathrm{FCA}, \quad \text{hence} \quad \mathrm{FA} \vDash \Diamond_1\neg\mathsf{X}\mathrm{A}.$$

This example occurs because we chose to involve a sequence which does not belong to the language of G, but paths having such valuations can take place in the context of the models of our logic. The validity of these properties determines if the logics we consider can be viewed as some specific construction of more basic logics, such as a product or a fusion.

Chapter 3

Decision problems: model checking

When approaching practical questions, fixing sets of derivation rules and operating within them makes any investigation more accessible. However, we may inquire if it is reasonable to expect an efficient (or even decidable) outcome. In the current chapters, we will inspect two recurrent decision problems in the field of modal logic: model checking (given $\mathcal{M}, \pi, \varphi$, is $\mathcal{M}, \pi \vDash \varphi$ true?) and satisfiability (given φ, does it hold in some path of a model?). We will employ general rule sets whenever it is feasible, and narrow down commonplace features otherwise. Some of the arguments have their roots in classical notions in computability theory, such as the PCP or Savitch's theorem.

3.1 Model Checking of $\mathcal{L}_{\mathsf{X}}$

Let us start with, arguably, the most essential result. Whenever the problems we lay out throughout this section are decidable and we can avoid deducing them as reductions from decision problems on other logics, our proofs will mostly be based on induction on the construction of the formulas to be checked. Some of the ideas stem from the well-known *labelling algorithm* ([BB06]), which demands we assign to each considerable state the subformulas which hold – in practice, we will devise recursive algorithms that carry implicitly this bottom-up implementation.

Dealing with the transitive closure $\Diamond$ magnifies the complexity with respect to the fragment of $\Diamond_1$ and its akin BML, whose model checking and satisfiability problems belong to the classes PTIME and PSPACE, respectively. This is because it usually involves monitoring an amount of derivations which cannot be bound as efficiently. We mostly turn to algorithms which keep track of how many rules

we have applied or how many different propositional values each state has taken, thus the relevance of space complexity classes.

It is precisely by minding the former that we will show that model checking is decidable for some fragments of $\mathcal{L}_{\mathsf{X}}$. Forthcoming descriptions will clarify the peculiarities of certain rules and why it is not effective to delimit space by supervising states in the context of general rule sets. As will be customary for the remainder of the chapter, any discussion regarding complexity will be preceded by that of decidability. We begin by introducing a metric for formulas which will condition the complexity of our results.

Definition 3.1. *Let φ be a formula for $\mathcal{L}_{\mathsf{X}}$ or $\mathcal{L}_{\mathsf{U}}$ and $* \in \{\mathsf{X}, \mathsf{U}, \Diamond_1, \Diamond\}$ a modal operator. Given the derivation tree of φ, we define $d_*(\varphi)$, the* modal $*$-depth *of φ, as the largest number of instances of $*$ in a single branch. If there is no risk of confusion, let $d(\varphi)$ represent the (total)* modal depth*, i.e. the largest number of instances of any combined $*$ in a single branch.*

Observation 3.2. *In most general terms, we define/frame rule sets as binary relations between harmonic sequences, i.e. between paths of some collection of (arbitrarily large) structures. Thus, they can be infinite. However, insomuch as the underlying transformations are bound to a finite schema, we assert a rule set R is finite when there exists some finite $R_0 \subseteq R$ for which any $(u, u') \in R$ is of the form $v_1 w v_2 \rightsquigarrow v_1' w' v_2$, with $(w, w') \in R_0$. By finite enumerations of R, we refer to an enumeration of any such R_0.*

From now on, let us assume that rule sets are finite in the sense of the observation; in the case that seemingly finite representations are infinite, we will discuss an alternative proposal if necessary.[1]

General case

In order to scrutinize $\mathrm{MC}(\mathcal{L}_{\mathsf{X}})$, one could fix some rule set R of a grammar G and an initial sequence S, and set out to delimit some $n(R, S, l)$ such that every initial segment of length l from some S' obtained as $S \Rightarrow_G^* S'$ is already in $L_l^{n(R,S,l)}(G, S)$, where $L_l^n(G, S) := \{u \restriction [0, l) \mid \exists k < n\ (S \Rightarrow_G^k u)\}$, for any natural n. Nevertheless, despite the fact that there are only finitely many different sequences of length l and the union $\bigcup_i^n L_l^i(G, s)$ stabilizes at some n, such an $n(R, S, l)$ will turn out to be not computable:[2] as the next example shows, chord sequences of arbitrary length may need to be explored to determine whether a shorter string is accessible.

[1] Moreover, in the current setting, it only makes sense to assume the finitude of inputted models and paths in algorithms of which we discuss complexity.

[2] This is actually due to the undecidability of $\mathrm{MC}(\mathcal{L}_{\mathsf{X}})$ shown later, but the example illustrates it for now.

Example 3.3. Let G_n be a grammar with ruleset $R_n = \{(C^{2\uparrow\uparrow i}, C^{2\uparrow\uparrow(i+1)}) \mid i < n\} \cup \{(C^{2\uparrow\uparrow n}, G)\}$, where $\uparrow\uparrow$ represents Knuth's up-arrow notation for iterated exponentiation. The chord sequence (G) can be derived through linearly-many rule applications with respect to n, but not in exponential space.

Having an upper bound for the derivation of sequences of a certain length would establish an upper bound on the amount of rule applications to be checked in any instance of $\Diamond$: we would show via a recursive argument that this quantity depends on the length of the considered paths, hence of the modal depth of our formula too. We will soon arrive to the conclusion that $\mathrm{MC}(\mathcal{L}_{\mathsf{X}})$ is not decidable, but we shall bear in mind the functioning of the labelling algorithm and, in particular, the $\Diamond$ step:

Algorithm 1 Case $\Diamond$ from a hypothetical algorithm $\mathrm{MC}(\mathcal{L}_{\mathsf{X}})(\mathcal{M}, \pi, \phi)$

```
...
case ◇ψ:
let m := m(M, π, ψ)                                  ▷ Non-computable bound
for r̄ = (0, ⋯ (m), 0); r̄ ≤ (k, ⋯ (m), k); r̄++_lexicographically do
    for j = 0; j < m; j++ do
        if MC(L_X)(M, π, ψ) then
            return true
        end if
        if R_{r(j)} = (π, π′) for some π′ then
            π ← π′
        else
            break                                    ▷ Only the j-loop
        end if
    end for                                          ▷ Reset value of π
end for
return false
...
```

As mentioned in section 1.3, another possibility would consist in reducing model checking to satisfiability. However, unlike in BML, given a path π in some $\mathcal{M}$, it is generally not possible to reconstruct by means of characteristic formulas of π the substructure of $\mathcal{M}$ on which the truth value of a formula depends. In other words: for $\mathcal{L}_{\mathsf{X}}$, satisfiability is not stronger than model checking. Certainly, we can claim that these methods fail because there exists a proof of the undecidability of $\mathrm{MC}(\mathcal{L}_{\mathsf{X}})$.

Definition 3.4. *The* Post Correspondence Problem *(PCP) is the decision problem given*

by the following:

- ***Input:*** *an alphabet Σ containing at least two characters, and two m-tuples of words on Σ, $U = (u_i)_{i\leq m}$ and $V = (v_i)_{i\leq m}$.*
- ***Output:*** *is there a sequence of indices $(i_1,\dots,i_r)$, with $1 \leq i_j \leq m$, for which $u_{i_1}\dots u_{i_r} = v_{i_1}\dots v_{i_r}$?*

The PCP was shown to be undecidable by [Po46], so it suffices to reduce it to $\mathrm{MC}(\mathcal{L}_X)$ to establish the undecidability of the latter. We proceed by defining a grammar which recreates the process of juxtaposing pairs of words, from which we can construct a model of $\mathcal{L}_X$ whose reachability relation $\Rightarrow$ corresponds to the rule set.

Proposition 3.5. *PCP is reducible to* $\mathrm{MC}(\mathcal{L}_X)$.

Proof. Let Σ, U and V be the input for an instance of the PCP. For any pair of words (w,w'), wlog $n = |w| \leq |w'| = m$, let $[w,w'] = ((w(i),w'(i)))_{i\leq q}{}^\frown((\lambda,w'(i)))_{n<i\leq m}$, where λ is a character used to represent an empty space (different from the empty word ε). That is, the actual words and paths consist of pairs in $\Sigma \cup \{\lambda, S_0, S\}$. Define the grammar G as $\langle\{(S_0,S_0),(S,S)\} \cup (\Sigma \cup \{\lambda\}) \times (\Sigma \cup \{\lambda\}),\emptyset,P,[S_0,S_0]\rangle$, where S_0 and S are auxiliary variables and (given words w,w' on the set of variables) P consists of the rules:

- $[S_0,S_0] \rightsquigarrow [u_iS,v_iS]$, for $1 \leq i \leq m$: to iniciate the derivation by choosing some initial pair of elements of $U \times V$.[3]
- $[wS,w'S] \rightsquigarrow [wu_iS,w'v_iS]$, for $1 \leq i \leq m$: to expand the current pair of words.
- $[xw,xw'] \rightsquigarrow [w,w']$, where $x \in \Sigma$: to delete coinciding characters in the starting position.
- $[wS,w'S] \rightsquigarrow [w,w']$: to terminate the derivation, with $w,w' \in Var^*$. [Alternatively, let $[wS,w'S] \rightsquigarrow [w\lambda,w'\lambda]$].

Despite the discrepancy between the meaning of the elements of this grammar and our first approach to $\mathcal{L}_X$, they operate identically on the basis of syntax, as they still refer to strings of pairs. Notice that it only makes sense to apply the last rule if our current word is precisely $[wS',w'S']$ for $w = w'$, but at some previous point of the derivation the lengths of w and w' may differ.

It is clear that (Σ,U,V) is a valid instance of the PCP if and only if $L(G) = \{\varepsilon\} \neq \emptyset$:[4] for any witness $u_{i_1}\dots u_{i_r} = v_{i_1}\dots v_{i_r} =: w$, the pair $[u_{i_1}\dots u_{i_r}S,v_{i_1}\dots v_{i_r}S]$

[3] Notice how, in the following rules, the empty characters in the possible pairs (λ,x) and (x,λ) are not respected: any newly attached word will take the place of the lopsided half. This issue will be explored after the first attempt at proposing a reduction.

[4] Or $\varepsilon \in L(G)$, if we define the language to contain every derived word (even with variables) or introduce terminal rules $A_x \rightsquigarrow x$ which assigns to each variable the chord character to which it corresponds.

can be derived from $[S_0, S_0]$, and we can proceed as $[wS, wS] \rightsquigarrow [w \restriction [1, |w|)S, w \restriction [1, |w|)S] \rightsquigarrow \ldots \rightsquigarrow [S, S] \rightsquigarrow \varepsilon$.

Now, we can build a modal structure in which a formula can capture the notion that the initial word $[S_0, S_0]$ can reach an empty path, i.e. that a path with an empty valuation is accessible. We take a variation of the basic product model, $\mathcal{M} = (W, \rightarrow, \Rightarrow, I)$, where $W = \{(S_0, S_0)\} \cup (\Sigma \cup \{S, \lambda\}) \times (\Sigma \cup \{S, \lambda\})$, $\rightarrow = W \times W$,[5] $\Rightarrow = \{(\pi, \pi') \mid I(\pi) \Rightarrow_G I(\pi')\}$, and I is the identity on W barring (λ, λ), that is, $I = \{((\lambda, \lambda), \emptyset)\} \cup \{(s, s) \mid s \in W \setminus \{(\lambda, \lambda)\}\}$.[6]

We show that (Σ, U, V) is a valid instance of the PCP if and only if the following holds.

$$\mathcal{M}, [S_0, S_0] \vDash \Diamond \bigwedge_{s \in W \setminus \{(\lambda,\lambda)\}} \neg s \quad (=: \vartheta)$$

[Alternatively, we can introduce (λ, λ) as variables which can be evaluated in the model, and enquire whether $\mathcal{M}, [S_0\lambda, S_0\lambda] \vDash \Diamond(\lambda, \lambda)$].

It suffices to check that $L(G) \neq \emptyset$ is equivalent to the condition above, but this is just Proposition 2.6: $\varepsilon \in L(G)$ happens in the case that there exist $w_1, \ldots, w_n$ such that $[S_0, S_0] \Rightarrow_G w_1 \Rightarrow_G \ldots \Rightarrow_G w_n = \varepsilon$. This is equivalent to stating the existence of a collection of paths $w_1, \ldots, w_n$ in $\mathcal{M}$ witnessing $[S_0, S_0] \Rightarrow^* \varepsilon$ – that is, knowing that $I((\lambda, \lambda)) = \emptyset$, $\mathcal{M}, [S_0, S_0] \vDash \vartheta$. □

The proof can be adjusted to let "empty paths" refer to those of length zero, or those which only consist of the state (λ, λ). In contrast with the aforementioned harmonic grammars, rules in this context do substitute *sequences* of characters. The main difference between this construction and the traditional PCP is the additional capability to delete coinciding pairs: it is essential for the reduction to work in $\mathrm{MC}(\mathcal{L}_{\mathsf{X}})$, as we can not express the matching condition a priori without knowing an (actually uncomputable) upper bound of the length required to solve each particular instance. This circumstance will play a critical role in the decidability of model checking in other logics and reducts.

Observation 3.6. *The introduced rules of the form $[wS, w'S] \rightsquigarrow [wu_iS, w'v_iS]$ may involve arbitrarily large words: for instance, if $(u_i, v_i) = (00, 0)$ is a pair of the PCP instance, any $[1S, 10^nS]$ can have $[1(0^{2n}), 1(0^{2n})]$ as a derivation, hence ε. This representation of the reduction is technically not adjusted to our requirement that rule sets must be finite.*

It is possible to refine the grammar to bypass this aspect, even though another sort of "until"-complexity will still be involved (the deletion rule $[xw, xw'] \rightsquigarrow$

[5] For a more faithful representation, it can be assumed that no $s \in W$ satisfies $(\lambda, \lambda) \rightarrow s$.

[6] That is, we identify states and valuations. Note that s to a pair of characters. Here, we take (λ, λ) not as a possible combination of characters, but as a placeholder for a cell that has no characters, so that paths can be defined indefinitely: i.e., we can interpret $[w, w']$ as $[w\lambda^n, \lambda^n]$, for any $n \geq 0$. This representation will change for $\mathcal{L}_{\mathsf{U}}$.

$[w, w']$). We will introduce a *Tetris*-like rule, so that we can stack pairs of words represented wlog as $(u_i^1 \dots u_i^{n_i} \lambda \dots \lambda, v_i^1 \dots v_i^{n_i} \dots v_i^{m_i})$ and slide non-empty characters to the left-hand side. In other words, we always introduce new pairs of words at the same position and push λ characters to the end. For example:

$$S_0 \Rightarrow \begin{smallmatrix} 1 & \lambda & \lambda \\ 1 & 0 & 0 \end{smallmatrix} S \Rightarrow \begin{smallmatrix} 1 & \lambda & \lambda & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & \lambda & \lambda \end{smallmatrix} S \Rightarrow \begin{smallmatrix} 1 & \lambda & 0 & \lambda & 0 & 0 \\ 1 & 0 & 0 & 0 & \lambda & \lambda \end{smallmatrix} S \Rightarrow \begin{smallmatrix} 1 & \lambda & 0 & 0 & \lambda & 0 \\ 1 & 0 & 0 & 0 & \lambda & \lambda \end{smallmatrix} S \Rightarrow \begin{smallmatrix} 1 & \lambda & 0 & 0 & 0 & \lambda \\ 1 & 0 & 0 & 0 & \lambda & \lambda \end{smallmatrix} S \Rightarrow \dots$$

$$\Rightarrow \begin{smallmatrix} 1 & 0 & 0 & 0 & \lambda & \lambda \\ 1 & 0 & 0 & 0 & \lambda & \lambda \end{smallmatrix} S \Rightarrow \begin{smallmatrix} 0 & 0 & 0 & \lambda & \lambda \\ 0 & 0 & 0 & \lambda & \lambda \end{smallmatrix} S \Rightarrow \begin{smallmatrix} 0 & 0 & \lambda & \lambda \\ 0 & 0 & \lambda & \lambda \end{smallmatrix} S \Rightarrow \begin{smallmatrix} 0 & \lambda & \lambda \\ 0 & \lambda & \lambda \end{smallmatrix} S \Rightarrow \begin{smallmatrix} \lambda & \lambda \\ \lambda & \lambda \end{smallmatrix} S \Rightarrow \begin{smallmatrix} \lambda \\ \lambda \end{smallmatrix} S \Rightarrow S \Rightarrow \lambda_\epsilon$$

We have repurposed variables S_0 and S to appear individually, since word heads are always at the same position. Also, λ_ϵ is added to distinguish an end of string. So, the new set of variables will be $\{S_0, S, \lambda_\epsilon\} \cup (\Sigma \cup \{\lambda\})^2$. Then, for a similar model $\mathcal{M}$, the condition to check is $\mathcal{M}, S_0 \vDash \Diamond \lambda_\varepsilon$ and the new rule set is:

- $S_0 \mid S \rightsquigarrow \begin{pmatrix} u_i^1 \dots \overset{u_i^{n_i}}{v_i^{n_i}} & \overset{\lambda}{v_i^{n_i+1}} \dots \overset{\lambda}{v_i^{m_i}} \end{pmatrix} S$, for $1 \leq i \leq m$ (and similarly for $n_i \geq m_i$).
- $\left(\begin{smallmatrix} \lambda & c \\ a & b \end{smallmatrix}\right) \rightsquigarrow \left(\begin{smallmatrix} c & \lambda \\ a & b \end{smallmatrix}\right)$, $\left(\begin{smallmatrix} a & b \\ \lambda & c \end{smallmatrix}\right) \rightsquigarrow \left(\begin{smallmatrix} a & b \\ c & \lambda \end{smallmatrix}\right)$, for $c \in \Sigma$ and $a, b \in \Sigma \cup \{\lambda\}$.
- $\binom{a}{a} x \rightsquigarrow x$, for $a \in \Sigma \cup \{\lambda\}$ and $x \in (\Sigma \cup \{\lambda\})^2 \cup \{S, \lambda_\epsilon\}$.
- $S \rightsquigarrow \lambda_\epsilon$.

Corollary 3.7. *Model checking is undecidable for $\mathcal{L}_{\mathsf{X}}$.*

The current result also entails that the fragment $\{\Diamond\psi \mid \psi \in \mathrm{LTL} \restriction \mathsf{X}\}$ is not decidable. In particular, so is the fragment obtained by considering the closure of sets of formulas which admit the rule $\Diamond \mathsf{X}\psi \leftrightarrow \mathsf{X}\Diamond\psi$, since any given formula can thus be viewed as some $\Diamond\psi'$.

Decidability in fragments

Considering that the core of a proof of decidability does not reside only on finitude, but rather on mechanizing certain computations, we shall inspect which fragments of our logic are prone to find an attainable constraint to deal with the $\Diamond$ operator. Due to Example 3.3, even if we make no distinction between states with the same valuation, it is desirable that the length of candidate paths is somehow determined by the input formula. Among the commonly attributed traits of harmonic grammars, a consequence of the fact that rules are only applied to single variables is that only the initial segment of a sequence is needed to deduce any of its transformations:

Definition 3.8. *Let R be the rule set from a grammar G. We say that R is* prefix-continuous[7] *if there exists some $f : \mathbb{N} \to \mathbb{N}$ such that, for every natural n, whenever $u \Rightarrow_G^* v$ and $u \upharpoonright [0, f(n)) = u' \upharpoonright [0, f(n))$, there exists some v' such that $u' \Rightarrow_G^* v'$ and $v \upharpoonright [0, n) = v' \upharpoonright [0, n)$ holds.*[8] *That is, if $u \upharpoonright [0, f(n)) = u' \upharpoonright [0, f(n))$, then $\{v \upharpoonright [0, n) \mid u \Rightarrow_G^* v\} = \{v' \upharpoonright [0, n) \mid u' \Rightarrow_G^* v'\}$.*

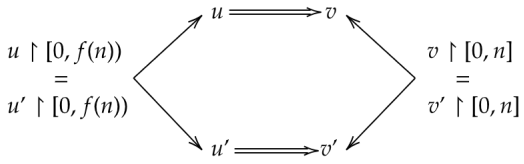

Figure 3.1: Condition of prefix continuity seen through a commutative diagram.

Equivalently, for u to access a path whose foremost n positions are different from those of any path accessible from u', u and u' must differ in some of their first $f(n)$ values. Given that, even with lineal functions of the form $f(n) = kn$ ($k > 1$), imposing such a restraint to Algorithm 1 still leads to exponential space, let us assume that our rule sets are prefix-continuous for some $f(n) = n + k$. This constitutes a proper fragment of $\mathcal{L}_{\mathsf{X}}$. The following is a lemma which will secure that our model checking procedures remain in PSPACE.

Lemma 3.9. *Let $(\mathcal{M}, \pi, \varphi)$ be the input for a* MC$(\mathcal{L}_{\mathsf{X}})$ *problem, where $\mathcal{M}$ is associated with a rule set prefix-continuous for some $f(n) = n + k$. Then, for every $n \geq d_k(\varphi) := d_{\mathsf{X}}(\varphi) + k(d_{\Diamond}(\varphi) + d_{\Diamond_1}(\varphi))$, $\mathcal{M}, \pi \vDash \varphi$ if and only if $\mathcal{M}, \pi[0, n) \vDash \varphi$.*

Proof. By induction on the construction of φ (taking the proof for every n each time). The cases of propositional variables and negation are clear.

- For $\varphi = \mathsf{X}\psi$, pick $n \geq d_k(\varphi) = d_k(\psi) + 1$. So $\mathcal{M}, \pi \vDash \mathsf{X}\psi$ iff $\mathcal{M}, \pi[1, +\infty) \vDash \psi$. Given that $n - 1 \geq d_k(\psi)$, by induction hypothesis, this occurs exactly when $\mathcal{M}, \pi[1, n - 1) \vDash \psi$, hence iff $\mathcal{M}, \pi[0, n) \vDash \mathsf{X}\psi$.
- For $\Diamond\psi$ (and, analogously, $\Diamond_1\psi$), take $n \geq d_k(\varphi) = d_k(\psi) + k$, so $\mathcal{M}, \pi \vDash \Diamond\psi$ iff there exists some $\pi \Rightarrow^* \pi'$ such that $\mathcal{M}, \pi' \vDash \psi$. By IH, this is if and only if $\mathcal{M}, \pi'[0, n - k) \vDash \psi$ and, by our assumption that rules are prefix-continuous, $\mathcal{M}, \pi[0, n) \vDash \Diamond\psi$.

[7] Since this notion can be traced back to k-local properties, we opt for a variation of its name.

[8] Note the transitive closure instead of single rules, which always holds under our assumption that rule sets are finite. Analogously, the accessibility relation $\Rightarrow \subseteq W^* \times W^*$ associated with a grammar is prefix-continuous if, for every $(\pi, \pi') \in \Rightarrow^*$, any ρ with $\pi[0, f(n)) = \rho[0, f(n))$ can access some ρ' that satisfies $\pi'[0, n) = \rho'[0, n)$.

– For $\varphi = \psi \wedge \chi$, let $n \geq d_k(\varphi) = \max\{d_k(\psi), d_k(\chi)\} \overset{\text{wlog}}{=} d_k(\psi) \geq d_k(\chi)$, so $\mathcal{M}, \pi \vDash \psi \wedge \chi$ iff, by IH, $\mathcal{M}, \pi[0,n) \vDash \psi$ and $\mathcal{M}, \pi[0,n) \vDash \chi$, that is, $\mathcal{M}, \pi[0,n) \vDash \varphi$.

□

This result applies to the grammar we considered in Section 1.2, since every rule prepends a chord to an existing one, or simply transforms one, so it is enough to know the first n chords of a sequence to reason about the initial segment of length n of every accessible sequence. In terms of complexity, this allows us to reassess our strategies for delimiting the amount of visited sequences when checking if $\Diamond\psi$ holds in some path π: revisiting Example 3.3, the key of the procedure may be visualized as attempting to set a polynomial bound to the height of a grid which displays the initial $d_k(\psi)$ positions of the possible transformations of π.

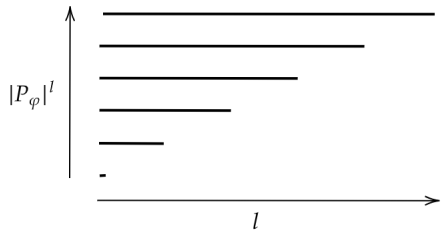

Figure 3.2: The decision of accessibility of a path is a bounded procedure for prefix-continuous rules.

It would also suffice to reuse Algorithm 1, but there exist more optimal strategies. We will recover some of the intuitions behind Savitch's Theorem (PSPACE = NSPACE) in [Ho01] to propose a proof for the central result of this subsection.

Theorem 3.10. *Under the forenamed prefix-continuous rule sets, model checking for* $\mathcal{L}_{\mathsf{X}}$ *is* PSPACE.

Proof. Fix a grammar G with rule set $R = \{r_0, \ldots, r_s\}$ and compute the $k \in \mathbb{N}$ for which it is prefix-continuous. Following [DGL16] and [Le14], we construct an algorithm which decides if a formula holds in a given path of a model.[9]

So, given $\mathcal{M} = \langle W, \rightarrow, \Rightarrow, V \rangle$, π and ψ, our goal for the $\Diamond\psi$ case is to examine if there exists a sequence of paths from π to some $\pi \Rightarrow^* \pi'$ such that $\mathcal{M}, \pi' \vDash \psi$.

[9] Technically, as seen in the proof of undecidability, the given grammar also works as an input of the problem, but –for the sake of convenience– we remove from the algorithm the calculation of k because it is mechanizable.

By the previous lemma, we can focus on their initial $d := d_k(\psi) \leq k|\psi|$ states: if there exists such a sequence of paths, we can assume it is optimal in the sense that it does not contain repeated copies, so we can assume it to be of length less than $|W|^d \cdot \frac{|W|+1}{|W|}$. The factor $\frac{|W|+1}{|W|}$ accounts for possible shorter starting paths, but we can omit it without loss of generality.

Our subprocedure must 1) find a path π' satisfying ψ, and 2) check whether such π' is reachable from π. So its general structure will be a for-loop along the possible goal paths (by means of an enumeration $0 \leq i < |W|$ of states[10] which will allow us to avoid non-determinism using little space), each of which will be run in an instance of $\mathrm{MC}'(\mathcal{L}_{\mathsf{X}})$ and, if successful, checked for accessibility from π. It is by compartmentalizing the inner paths and minimizing the number of open procedures that the problem will remain in PSPACE, in contrast to the expected EXPTIME in Algorithm 1 caused by cycling over the possible combinations of $m(\mathcal{M}, \pi, \psi)$-many rules. So, to some extent, the algorithm for 2) will act independently from 1).

Algorithm 2 Model checking in the fragment of $\mathcal{L}_{\mathsf{X}}$ with prefix-continuous rules

given $\mathcal{M}, \pi, \varphi$
run $\mathrm{MC}'(\mathcal{L}_{\mathsf{X}})(\mathcal{M}, \pi, \varphi)$ ▷ And output its result[a]
procedure $\mathrm{MC}'(\mathcal{L}_{\mathsf{X}})(\mathcal{M}, \pi, \phi)$
    switch ϕ
    case $p \in \mathrm{Prop}$: **return** $p \in V(\pi(0))$
    case $\neg\psi$: **return** (`not` $\mathrm{MC}(\mathcal{L}_{\mathsf{X}})(\mathcal{M}, \pi, \psi)$)
    case $\psi \wedge \chi$: **return** ($\mathrm{MC}(\mathcal{L}_{\mathsf{X}})(\mathcal{M}, \pi, \psi)$ `and` $\mathrm{MC}(\mathcal{L}_{\mathsf{X}})(\mathcal{M}, \pi, \chi)$)
    case $\mathsf{X}\psi$: **return** $\mathrm{MC}(\mathcal{L}_{\mathsf{X}})(\mathcal{M}, \pi[1, +\infty), \psi)$
    case $\Diamond_1\psi$:
    for $i = 0$; $i \leq s$; i++ **do**
        if $R_i = (\pi, \pi')$ for some π' **then**
            if $\mathrm{MC}(\mathcal{L}_{\mathsf{X}})(\mathcal{M}, \pi', \psi)$ **then**
                return `true`
            end if
        end if
    end for
    return `false`

[a] For the sake of convenience, let us assume that any algorithm ending with a **run** command accepts if it the procedure returns true, and rejects otherwise.

[10] With no gain in terms of complexity, it is also possible to consider just their valuations instead by picking an equivalent model.

Algorithm 2 (continued)

case $\Diamond\psi$:
for $\pi' = (0,\overset{(d)}{\ldots},0)$; $\pi' \leq (|W|-1,\overset{(d)}{\ldots},|W|-1)$; π'++$_{\text{lexicographically}}$ **do**
if $\pi' \in \Pi_{\mathcal{M}}$ and $\mathrm{MC}'(\mathcal{L}_{\mathsf{X}})(\mathcal{M},\pi',\psi)$ **then**
if $\mathrm{Split}(\pi[0,d),\pi'[0,d),d)$ **then** ▷ As $|W|^d + 1 \geq |W|^d$
return `true`
end if
end if
end for
return `false`
end switch
end procedure
procedure $\mathrm{Split}(\rho,\rho',i)$
if $i = 0$ **then**
if $\rho = \rho'$ or $\rho \Rightarrow \rho'$ **then**
return `true`
end if
end if
if $i > 0$ **then**
for $\overline{\sigma} := \begin{pmatrix} \sigma_1 \\ \vdots \\ \sigma_{|W|-1} \end{pmatrix} = \begin{pmatrix} 0 & \ldots & 0 \\ \vdots & & \vdots \\ 0 & \ldots & 0 \end{pmatrix}$; $\overline{\sigma} \leq \begin{pmatrix} |W|-1 & \ldots & |W|-1 \\ \vdots & & \vdots \\ |W|-1 & \ldots & |W|-1 \end{pmatrix}$; $\overline{\sigma}$++$_{\text{lexicographically}}$ **do**
let $b := 1$
let $b' := (\sigma_1 \in \Pi_{\mathcal{M}}) \wedge (\sigma_2 \in \Pi_{\mathcal{M}})$ ▷ If $|W| - 1 \geq 2$
for $j = 1$; $j < |W| - 1$ and $b \wedge b'$; j++ **do**
$b \leftarrow b \wedge \mathrm{Split}(\sigma_j,\sigma_{j+1},i-1)$
$b' \leftarrow b' \wedge (\sigma_{j+2} \in \Pi_{\mathcal{M}})$ ▷ For $j < |W| - 2$
end for
if $b \wedge \mathrm{Split}(\rho,\sigma_1,i-1) \wedge \mathrm{Split}(\sigma_{|W|-1},\rho',i-1)$ **then**
return `true`
end if
end for
end if
return `false`
end procedure

So Algorithm 2 accepts if and only if the given formula holds in the path: cases p, $\neg$, $\wedge$ and X are trivial. For $\Diamond_1\psi$, it checks if any of the rules can be applied to the path and returns true exactly when there is one which accesses a path where ψ holds. In the case $\Diamond\psi$, tests every goal path π' and attempts to bridge the gap

between π and π' by completing the optimal sequence of paths. Procedure Split applied to (ρ,ρ',i) assumes that the sequence between ρ and ρ' has length less than $|W|^i+1$ and divides it into $|W|$-many intervals. It guesses (or rather, tests every possible candidate) the paths which delimit these sectors and runs another instance for every interval, now taking into consideration that their sequence will not be longer than $|W|^{i-1}$.

Finally, Split is validated either when, at $i=0$, the paths are equal or accessible by $\Rightarrow$, or when $i>0$ and every subinstance has been resolved positively. Hence, Split(π,π',d) returns true if there is some sequence of paths between π and π' (including that where $\pi=\pi_1=\ldots=\pi'$), and false otherwise. Notice that the witnessing sequences can be shorter than $|W|^d+1$ as long as there exists an instance of Split which accepts with some $\sigma_j=\sigma_{j+1}$.

Our algorithm is deterministic and always ends, since each call to Split is done by reducing the index i in one unit. It runs in polynomial space: at any given moment, it is only storing position indices and $|W|$ paths from every active instance, thus a polynomial maximum of $d|W|+1$ paths; moreover, every step can be carried out in PSPACE. □

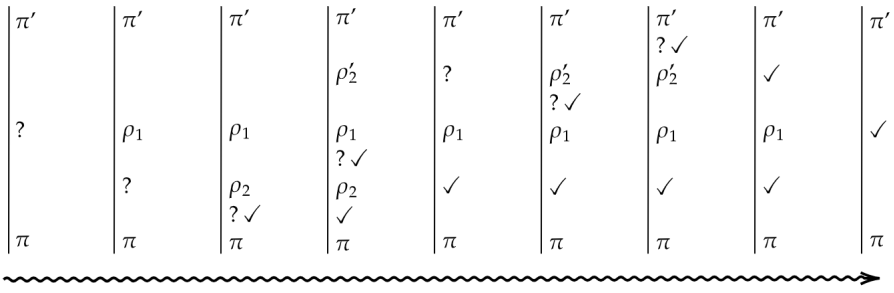

Figure 3.3: Procedure Split for $|W|=2$.

Observe that the grammars discussed in the proof of undecidability of MC$(\mathcal{L}_{\mathsf{X}})$ are not prefix-continuous due to the deletion rule: letting there only be $[S,S]\rightsquigarrow[0S',0S']$, $[wS',wS']\rightsquigarrow[w0S',w0S']$, $[xw,xw']\rightsquigarrow[w,w']$ and $[wS',w'S']\rightsquigarrow[w,w']$, no matter how broad $f(n)$ may be, $[0^{f(1)}1,0^{f(1)}1]\Rightarrow_{\mathsf{G}}^{*}[1,1]$ but $[0^{f(1)}0,0^{f(1)}0]\not\Rightarrow_{\mathsf{G}}^{*}[1,1]$. This, however, will vary in the case of $\mathcal{L}_{\mathsf{U}}$. But, thus far, we have already invoked an implicit 'until' operator to some extent:

Observation 3.11. *Let G be a grammar with rule set R. For models $\mathcal{M}=\langle W,\rightarrow,\Rightarrow,V\rangle$ where the graph of $\langle W,\rightarrow\rangle$ contains some loop having a subpath whose sequence belongs to the domain of $\Rightarrow$, the relation $\Rightarrow$ is not finite. We can either present it as:*

- *a list of pairs of subsequences* $R_0 = ((w_0, w_0'), \ldots, (w_s, w_s'))$ *which can eventually appear and be substituted in a considered path (i.e. finite rule schemata in the sense of Observation 3.2).*
- *an LTL-formula* φ_R *with two path variables that is satisfied, in our current model, by paths which differ exactly in some of such* (w_i, w_i')*. If it exists, we can also denote by* φ_R^* *some formula which defines the transitive closure of* $\Rightarrow$*.*

Both presentations refer to some string that may appear at a given point in a path: in particular, to reflect these occurrences, it is enough to use the eventuality operator $\mathsf{F}\varphi := \top \mathsf{U} \varphi$*. In the case of prefix-continuous rule sets for* $f(n) = n + k$*, the appearence of such sequences can be checked just with* X^k*, both for* $\Rightarrow$ *and its transitive closure.*

It is well known that model checking and satisfiability for LTL are PSPACE-complete [Ve01]. Since the size of our presentations of $\Rightarrow$ is independent of the models and formulas that our logic addresses, they signify no gain in complexity – still, unfolding the presentation into a practicable relation already entails some computation in PSPACE. Thus, we will take this complexity class as a reference for lower bounds. This allows us to summarize the results of the section as:

Proposition 3.12. *Model checking is decidable in* PSPACE *for any fragment of* $\mathcal{L}_\mathsf{X}$ *whose models are presented by some* $\varphi_R^* \in \mathrm{LTL} \restriction \mathsf{X}$*. It is undecidable otherwise, that is, whenever the fragment contains a model whose accessibility relations can only be presented by* φ_R^* *featuring* F *or* U*.*

In the next chapter, we will introduce a simplified procedure which relies in non-determinism and still works for the satisfiability problem, albeit restricted to a particular fragment of the logic.

3.2 Model checking of $\mathcal{L}_\mathsf{U}$

We remain in the same decision problem, but take as a domain the logic $\mathcal{L}_\mathsf{U}$, which involves the 'until' operator. Having proposed a semantic definition for our logics, since we still consider the same class of structures but expand the set of formulas to be discussed, anything that could be said in $\mathcal{L}_\mathsf{X}$ is expressible too in $\mathcal{L}_\mathsf{U}$. Thinking of $\mathcal{L}_\mathsf{X}$ as a fragment of $\mathcal{L}_\mathsf{U}$, it becomes clear that the undecidable problems in the latter are at least the ones in the former. Hence,

Corollary 3.13. *Model checking is undecidable for* $\mathcal{L}_\mathsf{U}$*.*

For decidable fragments, introducing an additional temporal operator that may evaluate an underlying formula at any point of a given path, possibly requires more demanding lower bounds for its complexity. In particular, results such as Lemma 3.9 may not provide restrictions as efficient as in $\mathcal{L}_{\mathsf{X}}$, if any at all. By itself, nesting of U along with X and $\Diamond_1$ does not blow up the length of paths to be checked. However, it is the combination of $\Diamond$ and U that gives rise to situations like the following:

Example 3.14. Let G the grammar over 12 major keys and 7 degrees, with rule set $R = \{X \rightsquigarrow \Delta/X\ X\}$. Consider $\psi(\phi) := \Diamond(\mathrm{F} \wedge (\phi \mathsf{U} \mathrm{C}))$, $\psi^1 := \psi(\top)$ and $\psi^{n+1} := \psi(\psi^n)$. For the basic product model $\mathcal{M}_0$ (from Definition 2.5) and the path $V(\pi) = (\mathrm{C})$,[11] any path that witnesses $\mathcal{M}_0, \pi \vDash \psi^n$ has length greater or equal than 7^n.

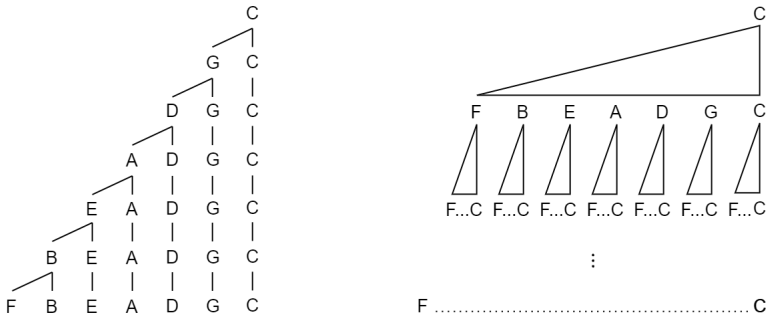

Figure 3.4: Derivation tree for the witness path for ψ^n.

While it is possible to study this case in PSPACE by opening and closing instances of a model checking procedure for $\Diamond$, it appears unclear that we can translate Lemma 2.9 to this context without handling memory allocation to avoid non-polynomial space. One might attempt to adopt a related approach and start by denoting, for any natural l, $\Pi^l_{\mathcal{M}}(\varphi) \subseteq \Pi_{\mathcal{M}}$ as the set of paths of length l which satisfy φ in $\mathcal{M}$, and

$$\Pi^{\leq l_0}_{\mathcal{M}}(\varphi) = \bigcup_{l \leq l_0} \Pi^l_{\mathcal{M}}(\varphi).$$

Based on the possible combinations of strings of a certain length, it is possible to establish an upper bound l for the sequences which are enough to check in order to assess the satisfaction of a formula $\psi \mathsf{U} \chi$. From this, given $\mathcal{M}, \pi, \Diamond\psi$, one can demonstrate the existence of an upper bound of $m(\mathcal{M}, \pi, \Diamond\psi)$ such that any path which witnesses $\mathcal{M}, \pi \vDash \Diamond\psi$ is $\pi \Rightarrow \pi_1 \rightarrow \ldots \rightarrow \pi_m$, for $m \leq m(\mathcal{M}, \pi, \Diamond\psi)$. However, such a bound is not computable in general, not even with prefix-continuous rules.

[11] For the sake of clarity, given that R cannot derive chords from different keys, let us express the state (d,C) as the chord that d takes in the key of C (e.g. (V,C) as G).

Thus, it is not possible to determine a value of l which would have sufficed for any $\psi \mathsf{U} \chi$ within the input formula, and hence to suggest an algorithm to construct the $\Pi^{\leq l_0}_{\mathcal{M}}(\psi)$ from the bottom up.

Reductions

We have already established that the PCP shows the undecidability of $\mathrm{MC}(\mathcal{L}_{\mathsf{U}})$. Although the result for $\mathcal{L}_{\mathsf{X}}$ only held when considering the full language, having access to the until operator allows us to narrow down the hypotheses to prefix-continuous rules.

Given an instance of the PCP, it will no longer be necessary to apply the deletion rule to see if an associated grammar can transform the initial variable into a string of identity pairs and, then, into an empty string. Instead, we can impose that the sequence found has both components of each of its characters coincide among the involved variables. More precisely, imposing that the sequence is not $[S_0, S_0]$ (and thus, that some rule has been applied),[12]

$$(\mathcal{M}, (S_0) \vDash) \Diamond \left(\neg (S_0 \vee S) \wedge \left(\bigvee_{a \in \Sigma \cup \{\lambda\}} (a,a) \right) \mathsf{U} \lambda_\varepsilon \right).$$

That is one unrestricted rule handled. The rules which switch a character with an empty space, $\binom{\lambda\ c}{a\ b} \rightsquigarrow \binom{c\ \lambda}{a\ b}$, appear to be prefix-continuous in that, given initial x cells, the possible reachable combinations are given by the length of the rules and the current blank characters. But it easy to see otherwise in the general setting, for instance, with $S_0 \rightsquigarrow S_0[0, \lambda]$ applied to a word $[\lambda, 0^n]$.

The main issue is that, given a sequence $[u, u']$ some of whose first n positions contain the λ character, it is possible that some $[v, v'] \Rightarrow [u, u']$ features a character $v(m) = a \neq \lambda$ or $v'(m) = a \neq \lambda$ being present in some $u(m')$ or $u'(m')$, for $m < n \leq m'$. This can be circumvented by reversing the representation of the PCP-words in grammar from the reduction: in that case, only characters equal to λ will be pushed to the beginning to the word. For example:

$$S_0 \Rightarrow S \begin{matrix} \lambda\ \lambda\ 1 \\ 0\ 0\ 1 \end{matrix} \lambda_\varepsilon \Rightarrow S \begin{matrix} 0\ 0\ 0\ \lambda\ \lambda\ 1 \\ \lambda\ \lambda\ 0\ 0\ 0\ 1 \end{matrix} \lambda_\varepsilon \Rightarrow \begin{matrix} 0\ 0\ 0\ \lambda\ \lambda\ 1 \\ \lambda\ \lambda\ 0\ 0\ 0\ 1 \end{matrix} \lambda_\varepsilon \Rightarrow \ldots \Rightarrow \begin{matrix} \lambda\ \lambda\ 0\ 0\ 0\ 1 \\ \lambda\ \lambda\ 0\ 0\ 0\ 1 \end{matrix} \lambda_\varepsilon$$

Nonetheless, notice that in the case of the top sub-word 000, it is necessary to discern how many λ are present in the remainder of the word in order to describe the possibly accessible words in the current position of 000. Seeing that the swap rule can shift said λ to any position, we can avoid the issue by adding an additional rule $\binom{a}{b} \rightsquigarrow \binom{a\ \lambda}{b\ \lambda}$, which just lets us append arbitrarily many λ. Therefore, any

[12] Otherwise, remove $\neg(S_0, S_0)$ and add $\Diamond_1$ before $\Diamond$.

combination arisen from adding λ into 000 is reachable. The current rule set becomes[13]:

- $S_0 \rightsquigarrow S \begin{pmatrix} \lambda & & \lambda & u_i^{n_i} & & u_i^1 \\ v_i^{m_i} & \dots & v_i^{n_i+1} & v_i^{n_i} & \dots & v_i^1 \end{pmatrix} \lambda_\varepsilon$, for $1 \leq i \leq m$ (and similarly for $n_i \geq m_i$).
- $S \rightsquigarrow S \begin{pmatrix} \lambda & & \lambda & u_i^{n_i} & & u_i^1 \\ v_i^{m_i} & \dots & v_i^{n_i+1} & v_i^{n_i} & \dots & v_i^1 \end{pmatrix}$, for $1 \leq i \leq m$ (likewise for $n_i \geq m_i$).
- $\begin{pmatrix} c & \lambda \\ a & b \end{pmatrix} \rightsquigarrow \begin{pmatrix} \lambda & c \\ a & b \end{pmatrix}$, $\begin{pmatrix} a & b \\ c & \lambda \end{pmatrix} \rightsquigarrow \begin{pmatrix} a & b \\ \lambda & c \end{pmatrix}$, for $c \in \Sigma$ and $a, b \in \Sigma \cup \{\lambda\}$.
- $x \rightsquigarrow x \begin{pmatrix} \lambda \\ \lambda \end{pmatrix}$, for $x \in (\Sigma \cup \{\lambda\})^2$.
- $S \rightsquigarrow \varepsilon$.

Any segment of a word accessed through the first two rules is determined by the same positions, a word of the same length; the same applies for the following two rules, since the accessed segments and the original ones only differ by some λ; finally, any $[w, w'] \restriction [0, n) \Leftarrow S[w, w'] \restriction [0, m)$ is derived from the first $m = n + 1$ positions. In other words, the rule set of the grammar is prefix-continuous for $f(x) = x + 1$. Therefore, we have ratified:

Proposition 3.15. *Let $\mathcal{L}'_{\mathsf{U}}$ be a fragment of $\mathcal{L}_{\mathsf{U}}$ which encompasses every grammar whose rule set is prefix-continuous for the function $f(x) = x + 1$. Then, model checking is undecidable for $\mathcal{L}'_{\mathsf{U}}$.*

In any case, one could challenge the notion of prefix-continuous rule sets as being representative of the simplicity of a grammar, given that such a property is not closed under subsets of grammars. On the other hand, it appears as a measure of some regularity[14] of a rule set, which proves useful for the design of some algorithms. Up to this point, either the usage of infinite or non-prefix-continuous rule sets had been used to reach undecidability. However, it could be argued that contrasting a left-branching grammar with a logic and a property both defined in a right-heading direction, is enough to grant our problems such computation complexities. Although for any further attempts it could be proposed to employ more general conditions such as k-local properties[15], this is an observation to keep in mind in order to study particular classes of grammars in the musical practise, such as the ones proposed by [Ro21].

[13] Technically, the models to which the PCP instance can be reduced must be constructed on a grammar whose alphabet contains the variables of the form $(a, a) \in (\Sigma \cup \{\lambda\}) \times (\Sigma \cup \{\lambda\})$. That way, the characterization PCP $\Leftrightarrow$ Grammar $\Leftrightarrow$ MC($\mathcal{L}_{\mathsf{U}}$) can be reproduced, with the language of our grammar not being the singleton of the empty word anymore, but some set of words of the form $[\lambda^n w, \lambda^n w']\lambda_\varepsilon$, with $w, w' \in \Sigma^*$.

[14] Not in the strict sense of regular grammars.

[15] See [Ma97].

Finally, being capable of constructing formulas with X, it could be proposed to characterize some models (or generated substructures of those) through the new operator. Even more, one could attempt to reduce the problem of model checking $\mathcal{L}_{\mathsf{U}}$ (or $\mathcal{L}_{\mathsf{X}}$) to $\mathrm{SAT}(\mathcal{L}_{\mathsf{U}})$, by means of including information of the model in the given formula. Still, even though this problem remains as future work, we could expect that it is not possible on the grounds of not having access to the alternative formalization of 'until' that can analyze the satisfaction on infinite paths. In particular,

$$\mathcal{M},\pi \vDash \psi \mathsf{X} \bot \quad \text{iff} \quad \mathcal{M},\pi[i,+\infty) \vDash \psi \text{ for every } i \text{ until } \bot \text{ holds}$$
$$\text{(i.e. } \mathcal{M},\pi[i,+\infty) \vDash \psi \text{ is always the case).}$$

Fragments

In general, in the situations that we can set a bound to the number of paths to be explored, we can think of (and implement) the problem of model checking of an 'until' formula as

$$\mathrm{MC}(\mathcal{M},\pi,\psi \mathsf{U} \chi) \equiv \bigvee_{j<|\pi|} \left(\mathrm{MC}(\mathcal{M},\pi[j,+\infty),\chi), \bigwedge_{i<j} \mathrm{MC}(\mathcal{M},\pi[i,+\infty),\psi) \right).$$

So one can substitute the corresponding clause for the case $\Diamond$ in Algorithm 2 in order to obtain a method to decide instances of model checking without the operator $\Diamond$. The complexity is preserved, because the longest the paths might become is $|\pi| + |\varphi| \cdot \max\{|v| - 1 \mid (u,v) \in G\}$, given an instance $\mathcal{M},\pi \vDash \varphi$ from a grammar G.

Proposition 3.16. *Model checking for the fragment of $\mathcal{L}_{\mathsf{U}}$ without $\Diamond$ is decidable in* PSPACE.

Another relevant type of construction is that of the formulas $\top \mathsf{U} \varphi$, which are satisfied if φ eventually happens in a path.[16] The problem of whether we can access a path where this formula holds is not decidable in general:

Let G be an unrestricted grammar and φ a formula whose satisfaction can be determined by finitely (and computably) many sequences up to final segments. For instance, $(p \vee q) \wedge \mathsf{X} r$ can hold for $(p,r)^\frown w$ and $(q,r)^\frown w$. Calling $u_0,\ldots,u_r$

[16] We mentioned before that the version of U that checked infinite paths was not at our disposal. This still holds for our case, as $\Diamond\neg(\top \mathsf{U} \varphi)$ assesses if any reached (finite) path satisfies φ nowhere. The computability of such a problem in grammars that are broad enough is beyond the scope of this thesis, but the complementary problem will be addressed in the following pages for a musically relevant rule set.

those sequences, we construct a grammar G' (and the corresponding model) which now includes deletion rules $wxu_iw' \rightsquigarrow wu_iw'$ and $wu_ixw' \rightsquigarrow wu_iw'$ for every $i \leq r$, $x \in \Sigma \cup V$ and $(\Sigma \cup V)^*$. So, essentially, G' contains some of the (finitely many) u_i if and only if G generates some string containing one of these words. The former is known as the *membership problem* for G' and it is known to be reducible to the halting problem ([Ho09]), which is undecidable.

Therefore, we need to narrow down our rule sets in order to possibly obtain decidability. A possible way to settle the decidability when dealing with context-sensitive grammars could result from making use of the capability of shifting substrings: recall that, given $x, y \in V$, the derivation $xy \Rightarrow_G yx$ can arise from adding some rules so that

$$xy \Rightarrow_G xz \Rightarrow_G z'z \Rightarrow_G z'x \Rightarrow_G yx.$$

is made possible. Then, by considering the words $u_0, \dots u_{r'}$ and similar exchange rules $xu_i^j \rightsquigarrow u_i^j x$, we can bring any u^i appearing in a derivation to the front.

Nevertheless, let us focus some context-free grammars that represent harmonic models such as [Ro20] or [St99]. For A, B variables and a a terminal character, let us consider any grammar $G = \langle V, \Sigma, P, S \rangle$ all whose rules are of the form:[17]

$$A \rightsquigarrow AB \qquad A \rightsquigarrow B \qquad A \rightsquigarrow a$$

We claim that it is possible to decide whether there exists a derivation of the initial variable S which contains a certain word $u = u^1 \dots u^n$ as a substring. Without loss of generality, given that the cited grammars usually allow for a "translation" of variables into their corresponding character via a terminal rule, let us assume that the u^j are variable symbols.

By an inductive argument on the length of u, let us first show how to find out if $u = u_1$ is reachable.[18] Consider the sets of words $W_0 = \{S\}$,

$$W_{i+1} = \{wxy \mid \exists z\, (wz \in W_i \text{ and } z \rightsquigarrow xy)\} \cup \{wx \mid \exists z\, (wz \in W_i \text{ and } z \rightsquigarrow x)\},$$

that is, each W_{i+1} is obtained from W_i by applying a rule to the last digit of any of the words that admits it. Since every variable thus far can appear as a derivation from the last symbol of a previous word, whenever no new variable is added for some i_1, no new one will appear for any $i \geq i_1$. This ensures that, after at most $|V|$ many iterations, the set $\{x \mid \exists w \exists i \leq i_1 \leq |V|\, (wx \in W_i)\}$ will stabilize.

In other words, it is decidable in time $\mathcal{O}(|P|^{|V|})$ to check if u_1 appears as a substring of some word in $L(G)$.

[17] Perhaps also featuring $S \rightsquigarrow \varepsilon$, for S the starting variable.

[18] See [Si12] for a derivation similar to right-regular grammars, which ours closely resembles.

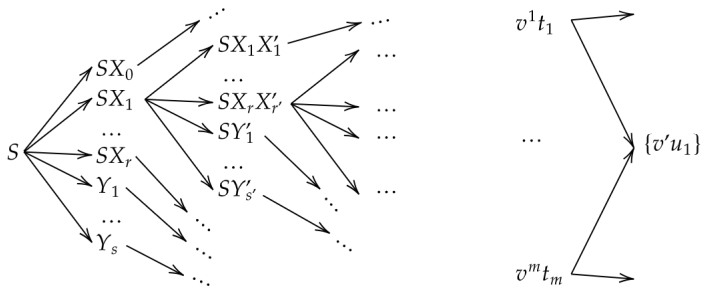

Figure 3.5: Determining the accessibility of u_1 and its predecessors.

Now, for $n \geq 2$, let us assume that we can decide whether some $wu_2 \dots u_n w'$ is derivable. Without loss of generality, since the rules are right-branching, it is enough to assume that for $wu_2 \dots u_n$ (and show it for some $w''u$). Since we developed an exhaustive proof (resume and adapt the notation $i_{n-1}, W_0, W_1, \dots$), consider the set

$$D = \{x \in V \mid \exists w \exists i \leq i_{n-1} \, (wxu_2(\dots u_n) \in W_i)\}.$$

Seeing that $ww'u \in L(G)$ if and only if there exists some $x \in D$ such that $x \Rightarrow_G^* w'u_1$, it is enough to determine if u_1 appears in any word derived from some $x \in D$ in order to decide if u appears as a substring of some element of $L(G)$.

Now, considering the same class of grammars, it is just a matter of evaluating ψ in a given path in order to decide model checking of $\Diamond(\psi \mathsf{U} \chi)$. In fact, this result holds for any $\psi, \chi \in \mathcal{L}_{\mathsf{X}}$, given that our grammar is prefix-continuous and we can establish the maximum needed length of the witnessing sequences $u_1, \dots, u_r$. The prefix continuity is provided by the fact that $wxw' \Rightarrow_G \alpha$ implies that $\alpha = wyw''$ for some y, w'', hence whenever v is derived from u which coincides in the first $n+1$ positions with some u', there will exist some v' derived from u' such that $v \restriction [0,n) = v' \restriction [0,n)$.

Proposition 3.17. *The decision problem of determining whether a given word appears as a substring of some element from the language of a fixed grammar G, is undecidable for unrestricted G and decidable for G with production rules among the following $\{A \rightsquigarrow AB, A \rightsquigarrow B, A \rightsquigarrow a\}$.*

Corollary 3.18. *Model checking on the fragment $\{\Diamond(\top \mathsf{U} \varphi) \mid \varphi \in \mathrm{LTL} \restriction \mathsf{X}\}$ is, in general, undecidable for the class of unrestricted grammars, and decidable for context-free grammars with rules among $\{A \rightsquigarrow AB, A \rightsquigarrow B, A \rightsquigarrow a\}$.*

Indeed, the decidable portion of the result applies to the models in [Ro21] and [Fi20], it is enough to reverse the representation of the rules ($X \rightsquigarrow V/X\ X$ becomes $X \rightsquigarrow X\ V/X$, etc.). In fact, the proof strategy works because we can generate sequences sequentially, in a way that we do not retrace visited parts of any intermediate word generated in the process, and due to the fact that no rule depends on two variables.

Of course, the fragment we considered is far from representative of all the musical notions we would hope to encapsulate with modal formulas. For example, take ψ describing some desired condition for a harmonic sequence, and χ_1,χ_2 of the form $p_0 \wedge \mathsf{X}p_1 \wedge \ldots \wedge \mathsf{X}^m p_m$. Then, the following

$$\psi \wedge I \wedge \neg\Diamond(\top \mathsf{U}(\chi_1 \wedge \neg(\top \mathsf{U}\chi_2)))$$

expresses that a path based on the first degree and satisfying ψ cannot access another path that eventually reaches χ_1 but not χ_2 anytime afterwards. Nonetheless, this is an expedient requirement to have when wanting a "question-answer" structure (in which an interrogative motif χ_1 is bound to be replied by some χ_2) or a reexpositive form (in which a χ_2 similar to an already presented χ_1 is expected after a trio or a development).

Chapter 4

Decision problems: satisfiability

At first glance, deciding whether a formula is satisfiable in some model at all seems like more of a complex problem than checking its truth value within a model, as some finite model property and others might have to be proven. Despite this, we can find more general decision problems which respond to the specifics of working without a fixed structure.

We also study the complexity of some fragments of $\mathcal{L}_X$ and $\mathcal{L}_U$, related to the restriction of certain formulas or operators, and to the selection of particular classes of grammars. Our results are largely based on the filtrations found in [Go87] and well-known methods featured in [DGL16].

4.1 Satisfiability without fixed grammars

In this section, we cannot make use of direct reductions from grammars or results to recursively set bounds to algorithms. Rather, we find more benefits in considering syntactical approaches.

Case $\mathcal{L}_X$: full language

Having proved the earlier properties of model checking for $\mathcal{L}_X$, determining if we can decide the satisfiability of some MC-decidable fragment $\mathcal{L}'_X$ of our logic becomes slightly more accessible. By some procedure analogous to filtration, if there exists an algorithm to generate every model where a given formula can hold, it will suffice to apply $MC(\mathcal{L}'_X)$ to each of them. However, given that there is no available algorithm for model checking in the full logic $\mathcal{L}_X$, we must resort to another approach for the general case.

The following does not presume a fixed grammar – however, showing that it is decidable to check if a formula is satisfiable in *some* grammar is still a broader

result than that of the fixed prefix-continuous case (which we will still detail). Given any formula φ, we will provide the definition of a structure $\mathcal{M}_\varphi$ consisting of sets of its subformulas consistent in some fashion, similar to how filtration with canonical models works. We will show it to possess the property that each formula holds in a state[1] if and only if it is one of its labels. Hence, φ will be satisfiable if it belongs to some such state.

The peculiarity of these models lies in its division into collections of states according to the modal (X) depth of its subformulas, so that model checking bypasses any looping which hinders decidability. Nonetheless, the consistency of formulas (let alone sets of) is unknown beforehand: despite it being enough for our purposes to prove some FMP-like condition, it is either through exhausting the combinations or through the description of our structures that we actually will construct them from the bottom up.

Consider an $\mathcal{L}_\mathsf{X}$-formula φ and P_φ the set of its propositional variables. Given that X commutes with negation and conjunction, i.e.

$$\mathsf{X}\neg\psi \equiv \neg\mathsf{X}\psi \quad \text{and} \quad \mathsf{X}(\psi \wedge \chi) \equiv \mathsf{X}\psi \wedge \mathsf{X}\chi,$$

we express φ as an equivalent formula in which X can only precede *atoms*, that is, some $p \in P$, $\Diamond_1$, $\Diamond$ or another X, done in linear time. In particular, $\varphi = \theta(\psi_1, \ldots, \psi_n)$ for some atoms ψ_i and a propositional $\theta(x_1, \ldots, x_n)$ on $\langle \{x_i\}_{i\leq n}, \neg, \wedge \rangle$.

Definition 4.1. *Let* Γ *be a set of* literals *(atoms and their negations) in* $\mathcal{L}_\mathsf{X}$ *or, more generally, a collection of formulas. A set* s *of* $\mathcal{L}_\mathsf{X}$*-formulas is* consistent *if* $s \nvdash_{\mathcal{L}_\mathsf{X}} \bot$*. We say that* s *is maximal consistent if it is consistent and, for every* $\psi \notin s$*,* $s \cup \{\psi\}$ *is inconsistent. We call* Γ*-maximal consistent sets (*$\mathrm{MCS}_\Gamma(s)$*) the equivalence classes of maximal consistent sets which coincide in* Γ*.*[2]

Without loss of generality, we can ignore the members of Γ-maximal consistent sets and drop "Γ" whenever there is not risk of confusion. Thus, we can view such classes as sets of formulas common to all their elements. Observe that this intersection is general not a subset of Γ, given that any set containing $\{\neg\Diamond p, \Diamond_1 q\}$ is always consistent with $\{\neg\Diamond_1 p, \Diamond\top\}$, but not always with some $\{r\}$. It is also possible to actually restrict the classes to Γ, although –in that case– some of the consistent formulas must be taken into consideration for filtrations.

In order to describe $\mathcal{M}_\varphi$, we work towards a collection of maximal consistent sets[3] of subformulas of φ marked with their relative X-depth. Every filtration will

[1] In the context of this chapter, meaning that it holds in some (the unique) path which starts at the current state.

[2] We can opt for the analogous maximal *satisfiable* sets, if the approach does not involve the an axiomatization and the truth values of formulas must be evaluated semantically.

[3] As elements of a structure, we also refer to them as *states*.

involve some set embeddable into Σ, the set of positive subformulas of φ. Even if it would have sufficed to maintain just the atoms (since every subformula ψ of φ arises as some boolean combination, and so does their consistency), we choose to contemplate such a Σ to achieve a more natural articulation of the truth lemma central to this section.

First define $d^{\varphi}_{\mathsf{X}}(\psi)$, the *depth of* $\psi \in \Sigma$ *in* φ, as the X-depth at which ψ appears in the derivation tree of φ, i.e.

$$d^{\varphi}_{\mathsf{X}}(\psi) = \min\{d_{\mathsf{X}}(\theta(x)) \mid \theta \in \mathcal{L}_{\mathsf{X}} \wedge \varphi = \theta[x/\psi]\}.$$

For each $i \leq d_{\mathsf{X}}(\varphi)$, let Σ'_i be the set of subformulas of φ which appear at depth i, along with their negations. More precisely, let

$$\Sigma'_i = \left\{\psi \mid (\psi \in \Sigma \vee \sim\psi \in \Sigma) \wedge d^{\varphi}_{\mathsf{X}}(\psi) = i\right\},$$

and define recursively

$$\Sigma_i = \begin{cases} \Sigma'_i, & i = d_{\mathsf{X}}(\varphi) \\ \Sigma'_i \cup \{\mathsf{X}\psi, \neg\mathsf{X}\psi \mid \psi \in \Sigma_{i+1} \wedge \psi \text{ positive}\}, & i < d_{\mathsf{X}}(\varphi). \end{cases}$$

These will differentiate several levels in our structure, within which the satisfaction of the path accessibility operators will be analyzed. Note that $\Sigma_{d_{\mathsf{X}}(\varphi)+1}$ should be empty, that $\Sigma_{d_{\mathsf{X}}(\varphi)}$ contains no X, that there exists a natural injective mapping $\Sigma_{i+1} \to \Sigma_i$, and that Σ_0 captures the depth in φ of each of its subformulas. It is possible to remove the hereditary aspect from the definition of the Σ_i and be left with a smaller model, but the current formulation ensures that the relation which corresponds to X is actually a function, that is, every state will already contain all the information about the *single* path it represents.

Letting $\Sigma^* = \bigcup_{i \leq d_{\mathsf{X}}(\varphi)} \Sigma_i$, the aforementioned states will be precisely maximal consistent sets from $\{s \subseteq \Sigma^* \mid \mathrm{MCS}(s)\}$. Our goal is to create a canonical model for $\mathcal{L}_{\Diamond_1\Diamond}$, where the relations for temporal operators are ignored, so that we can implement multiple filtrations to obtain structures representing the depth levels of φ. Hence, we will be able to evaluate φ in the reunion of such structures: this is opportune as, even if we ignore MCS's, the construction provides an upper bound for the size of the validating models, i.e. if φ is satisfiable for some (any) grammar, it will be so in a model below a maximum size. Conversely, if having checked all such structures up to that cardinality, none of them validates φ, then it is unsatisfiable.

Observation 4.2. *The logic $\mathcal{L}_{\Diamond_1\Diamond}$ is normal, since (1) it contains every propositional tautology and the distribution axiom*[4] $\Box_1(\phi \to \psi) \to (\Box_1\phi \to \Box_1\psi)$ *(ψ holds in every*

[4]We usually express axioms with the dual necessity operators.

accessed state which satisfies $\phi \to \psi$ and ϕ); and (2) the rules Modus Ponens and Necessitation hold (if ψ is satisfied in every state, it is clearly satisfied in every accessed state).

In some contexts, $\mathcal{L}_X$ is called a *fusion* of two logics[5]: the fragment LTL $\restriction$ X, containing no U, and the smallest extension of the logic K (with operator $\Diamond_1$) containing its transitive closure $\Diamond$. A fusion can be seen as a general combination of the logics where the interaction between different modalities is not expected to comply with any property. As a consequence, $\mathcal{L}_X$ is axiomatizable by the reunion of the axioms and rules from its components, that is, those of LTL $\restriction$ X, those of K, an axiom to ensure the transitive behavior of $\Diamond$,

$$\Box\phi \to \Box_1(\phi \wedge \Box\phi) \qquad (\Diamond_1(\phi \vee \Diamond\phi) \to \Diamond\phi),$$

the rule of induction,

$$\frac{\phi \to \Box_1\phi}{\phi \to \Box\phi} \qquad \left(\frac{\Diamond_1\phi \to \phi}{\Diamond\phi \to \phi}\right),$$

and some auxiliary axioms to define the semantically valid duality of our accessibility and necessity operators,

$$\Diamond_1\phi \leftrightarrow \neg\Box_1\neg\phi \quad \text{and} \quad \Diamond\phi \leftrightarrow \neg\Box\neg\phi.$$

These two facts are sufficient to construct the following model, which we will later base our filtrations on:

Definition 4.3. *The structure $\mathcal{M}_c = \langle W_c, R_c^1, R_c^\Diamond, V_c\rangle$[6] is called the* canonical model *for Σ^*, where $W_c = \{s \mid \mathrm{MCS}_{\Sigma^*}(s)\}$, $V_c(s) = p \cap s$ and, for given $s,t \in W_c$,*

$$sR_c^1t \;\; \textit{iff} \;\; \neg\Diamond_1\psi \in s \Rightarrow \neg\psi \in t,^{7} \quad \textit{and} \quad sR_c^\Diamond t \;\; \textit{iff} \;\; \neg\Diamond\psi \in s \Rightarrow \neg\psi, \neg\Diamond\psi \in t.$$

Having the transitive closure $\Diamond_1$ alters somewhat the usual definitions. Observe that, apparently, defining $R_c^\Diamond$ analogously to R_c^1 might lead to believe that $\Diamond_1\psi \in s \Leftrightarrow \Diamond\psi \in s$ holds in every $s \in W_c$: that is generally not the case, since the construction of W_c might have already ruled out the consistency of some formulas. For example,

$$\mathrm{MSS}_{\mathcal{L}_X}(s \supseteq \{\Diamond p, \neg\Diamond_1 p\}), \quad \text{while } \{\Diamond p, \neg\Diamond p\} \text{ is inconsistent.}$$

For now, as R_c^1 and $R_c^\Diamond$ refer to their same level, we can rest assured that interactions are well-behaved to some extent. The next example illustrates how these relations (let sRt) are better understood in terms of t respecting the forbidden formulas from s, and not t witnessing possible formulas in s.[8][9]

[5] See [ca07].

[6] We abridge $R_c^{\Diamond_1}$ as R_c^1.

[7] That is, $\{\neg\psi \mid \neg\Diamond_1\psi \in s\} \subseteq t$.

[8] In some diagrams, for the sake of readability, let the absence of ψ implicitly represent $\neg\psi$.

[9] If we assume that $\Diamond\top$ is a theorem of our logic (which we will soon argue to be the case), the

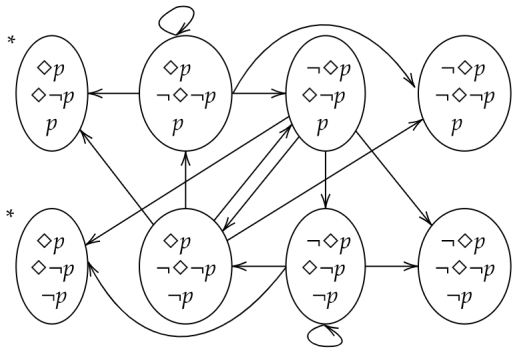

Figure 4.1: The structure $\mathcal{M}_c = \langle W_c, R_c^1 = \emptyset, R_c^{\Diamond}, V_c \rangle$ models $\varphi = (\Diamond p) \wedge (\Diamond \neg p)$, but no state s' with $sR_c^{\Diamond}s'$ witnesses both $s \vDash \Diamond p$ and $s \vDash \Diamond \neg p$. We have represented the $R_c^{\Diamond}$ with single-shafted arrows to favor readability – similarly, the * in some states denotes that they are connected with every world in the model.

We chose to exclude R^{X} for the moment because it does not affect the upcoming process, as it is the case for other equally effective definitions of $R_{\cdot}^1$. We will deal with it after proving that this sort of construction operates correctly. On the one hand, the current result is well-known (see, for example, [BB06]):

Proposition 4.4. *Assume, given $\mathcal{M}_c$, that $[\![\mathsf{X}\phi]\!] = \{s \in W_c \mid \mathsf{X}\phi \in s\}$ for every $\mathsf{X}\phi \in \Sigma^*$. Let $s \in W_c$ and $\psi \in \Sigma^*$. Then,*

$$\mathcal{M}_c, s \vDash \psi \Leftrightarrow \psi \in s.$$

The main lemma of this section will be of this form, hence the temporary assumption for X. Lindenbaum's Lemma was used for the proof, which also benefited from us regarding maximal consistent sets as closed under consequence: at some stage of the argument,[10] the validity of some formula in the calculus allows us to assert that some $(\Box_1\psi_1 \wedge \ldots \wedge \Box_1\psi_r) \rightarrow \Box_1\phi$ belongs to a given state s, but usually this is not a formula from Σ^*.

Having defined $R_c^{\Diamond}$, we ought to prove that it works identically to $(R_c^1)^*$, the (reflexive) transitive closure of R_c^1. This is sometimes referred to ([Fe24]) as an *ancestral lemma* (an ancestral of R, its "set of ancestors", being the relation R^*), and responds to the fact that $\Diamond$ is the transitive closure of $\Diamond_1$ in our calculus. Our devised proof differs from the one found in [Go87] in the type of relations involved, and that the domains for $(R_c^1)^*$ and $R_c^{\Diamond}$ coincide.

two rightmost states are inconsistent. For now, let us display them for the sake of illustrating the construction.

[10] See [Se08].

Lemma 4.5. *For every $s,t \in W_c$, the relation $R = (R_c^1)^*$ satisfies the following:*

(1) $sR_c^{\Diamond}t$ implies sRt.

(2) sRt implies that, for all $\neg\Diamond\psi \in s$, $\psi \notin t$.

Proof. For (1), for any $s \in W_c$, define the characteristic formula in Σ^* for s, $\chi(s) := \bigwedge(s \cap \Sigma^*) \wedge \neg\bigvee(\Sigma^* \setminus s)$. Following 9.7 in [Go87], $A_s := \bigvee\{\chi(s') \mid sRs'\}$ defines the set of states related with s through the closure of R_c^1, that is,

$$sRs' \Leftrightarrow A_s \in s'.$$

Given $s,t \in W_c$ such that $sR_c^{\Diamond}t$, suppose that sRt did not hold. First, by completeness of $\mathcal{M}_c$, notice that $\vdash_{\mathcal{L}_{\Diamond_1\Diamond}} A_s \to \Box_1 A_s$: any $u \in W_c$ with $A_s \in u$ satisfies sRu, so there exist $sR_c^1s_1R_c^1 \ldots R_c^1s_rR_c^1u$; then, if uR_c^1v for some $v \in W_c$, we have sRv and $A_s \in v$, whence $\Box_1 A_s \in u$.

By the induction rule, $\vdash_{\mathcal{L}_{\Diamond_1\Diamond}} A_s \to \Box A_s$. Knowing that $\chi(t)$ never concurs with A_s (since sRt is false), we can assert $\vdash_{\mathcal{L}_{\Diamond_1\Diamond}} A_s \to \neg\chi(t)$. Substituting A_s in the previous formula, we obtain $\vdash_{\mathcal{L}_{\Diamond_1\Diamond}} A_s \to \Box\neg\chi(t)$. Finally, $\vdash_{\mathcal{L}_{\Diamond_1\Diamond}} \chi(s) \to \Box\neg\chi(t)$ follows by MP, as $\vdash_{\mathcal{L}_{\Diamond_1\Diamond}} \chi(s) \to A_s$ holds by the reflexivity of R: but the fact that $sR_c^{\Diamond}t$, along with $\chi(s) \in s$, implies that $\chi(t) \notin t$, which never holds as $\chi(t)$ characterizes t. Therefore, sRt.

For (2), sRt implies the existence of some corresponding $s_0 = s, s_1, \ldots, s_r$. For every $j \leq r$, every $\neg\Diamond\psi \in s_j$ entails $\neg\Diamond_i\Diamond\psi \in s_j$, so $\neg\Diamond\psi \in s_{j+1}$ and we deduce they are in s_r too. Furthermore, this implies $\neg\Diamond_1\psi \in s_r$, thus $\neg\psi \in t$ and, ultimately, for every $\neg\Diamond\psi \in s$, $\neg\psi \in t$. □

The reasoning as it is relies on $\Diamond$ being associated to a reflexive relation, which is in consonance both with the rule $X \rightsquigarrow X$ being part in most musical systems and with the common use of reflexivity when defining derivations in unrestricted grammars. A similar result can also be developed when assuming that $R_c^{\Diamond}$ is not reflexive: then one has to enforce that some formulas of the sort $\neg\Diamond_1 \ldots \Diamond_1\Diamond\psi$ being in $sR_1^{\Diamond}t$ imply that $\neg\psi \in t$. As shown in the next example (Figure 4.2), this conveys that also "eventually" inaccessible formulas cannot be accessed through $R_c^{\Diamond}$. At any rate, we always have available the option of switching between assumptions: adding or restricting the possibility that a subformula ψ may hold for the same path where $\Diamond\psi$ is evaluated, we obtain the opposed version of $\Diamond$. In other words,

$$\Diamond_1^*\psi \leftrightarrow \psi \vee \Diamond_1^+\psi \qquad \text{and} \qquad \Diamond_1^+\psi \leftarrow \neg\psi \wedge \Diamond_1^*\psi.$$

Note, however, that this increases the length of the initial formulas quadratically, even if we do not specify the complexity of our algorithms beyond the classes PSPACE, EXPTIME, NEXPTIME, etc.

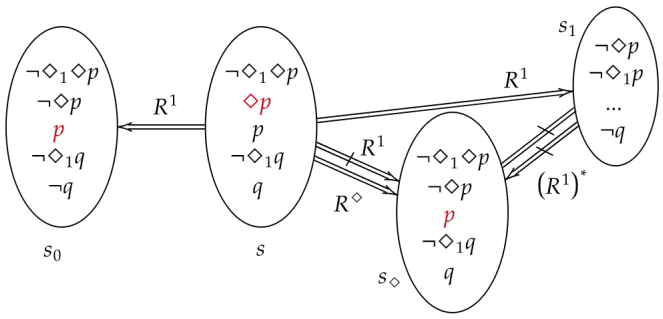

Figure 4.2: For a non-reflexive $\Diamond$, s_0 witnesses the satisfiability of s, but $sR^\Diamond s_\Diamond$ is not corresponded with $s(R^1)^* s_\Diamond$, as any accessible $sR^1 t$ is of the form $s_1 \subseteq t$, and s_1 cannot reach $s_\Diamond$ through the closure of this example R^1. However, with a reflexive $\Diamond$, the formulas in red are not consistent with the remaining ones in their state.

In line with the implication $\Box\psi \to \psi$ ($\psi \to \Diamond\psi$), the proof of the ancestral lemma relies on our maximal sets containing more (all) consequences outside of Σ^*. On the one hand, the fact that every harmonic sequence is in the domain of the grammar implies that $\Diamond_1\top$, i.e. $\neg\Diamond_1\psi$ and $\neg\Diamond_1\neg\psi$ cannot occur simultaneously. Likewise, the equivalence $\neg\Diamond_1\Diamond\psi \wedge \neg\Diamond_1\psi \leftrightarrow \neg\Diamond\psi$ (i.e. $\Diamond_1\Diamond\psi \vee \Diamond_1\psi \leftrightarrow \Diamond\psi$) encapsulates the transitive nature of $\Diamond$, as the following diagram depicts.

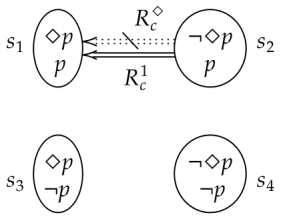

Figure 4.3: In case we only regarded formulas in $\bigcup_i \Sigma_i$, the canonical model of $\Diamond p$ would not satisfy the ancestral lemma, given that $\neg\Diamond p \in s_2$ would not imply $\neg\Diamond_1 p \in s_2$, hence $s_2 R_c^1 s_1$ but not $s_2 R_c^\Diamond s_1$. [For the sake of clarity, we only represent the relevant edges].

At this point, it appears to be the moment to mention that there exists another possible natural method to construct a model to analyze the satisfiability of a formula. Supposing that φ is satisfied in some $\mathcal{M}, w$, we may consider the set Σ of subformulas of φ and check that $\mathcal{M}/\Sigma$ has some state in which φ holds. Since $\mathcal{M}/\Sigma$ is also finite, we can conclude that the logic has the finite model property. Although the formalization is semantic, some of the previous proofs are simplified.

Case $\mathcal{L}_{\mathsf{X}}$: layering the model

We can now move on, so in order to address the evaluation of formulas with X, define, for any given Γ,

$$\Theta^{+}\Gamma = \{\psi \mid \mathsf{X}\phi \in \Gamma\} \qquad \text{and} \qquad \Theta^{-}\Gamma = \{\neg\psi \mid \neg\mathsf{X}\psi \in \Gamma\}.$$

They might be abbreviated as $\Theta\Gamma := \Theta^{+}\Gamma \cup \Theta^{-}\Gamma$, with hopes of relating some sort of states $\bar{s}$ and $\bar{t}$ whenever $\Theta\bar{s} \subseteq \bar{t}$.

Now, let us organize a new model in accordance to depth layers. For every $i \leq d_{\mathsf{X}}(\varphi)$, we introduce $\mathcal{M}_i = \mathcal{M}_c/\Sigma_i$, following a maximal filtration-like arrangement for each $i \leq n$. Start by defining an equivalence relation $\sim_i$ on W_c as $s \sim_i t$ if and only if $s \cap \Sigma_i = t \cap \Sigma_i$. Then, let $[s]_i = \{t \in W_c \mid s \sim_i t\}$, and put $W_i = \{[s]_i \mid s \in W_c\}$ as a new set of states. Again, in practice, we may regard $[s]_i$ as the set of formulas $\bigcap\{s' \mid s \sim_i s'\} \cap \Sigma_i = s \cap \Sigma_i$ and only discuss states in the definition of the relations.

The relation R_i^1 has $[s]_i R_i^1 [t]_i$ if and only if there exist $s' \sim_i s$ and $t' \sim_i t$ such that $sR_c^1 t$. Then, for every $\neg\Diamond_1\psi \in \Sigma_i$, whenever $\neg\Diamond_1\psi \in s$ we have $\neg\psi \in t$, given that any $s \sim_i s'$ coincide in their formulas at depth i (as well as $t \sim_i t'$). In particular, R_i^1 meets the criteria[11] to be a filtration relation of R_c^1 (modulo Σ_i), namely

$$sR_c^1 t \Rightarrow [s]_i R_i^1 [t]_i \quad \text{and} \quad [s]_i R_i^1 [t]_i \Rightarrow \{\psi \mid \neg\Diamond_1\psi \in s\} \cap t \cap \Sigma_i = \emptyset.$$

We interpret $R_i^{\Diamond}$ in an analogous way, which causes it to have the same properties as $R_c^{\Diamond}$, that is, $[s]_i R_i^{\Diamond} [t]_i$ if and only if $\neg\Diamond\psi \in [s]_i$ implies $\neg\psi, \neg\Diamond\psi \in [t]_i$.[12] Moreover, the ancestral lemma also holds for our new pairs of relations, so $(R_i^1)^*$ behaves as $R_i^{\Diamond}$.

In order to interpret $\mathcal{L}_{\mathsf{X}}$-formulas, we introduce a function f to handle the X operator. It assigns to each state in W_i the unique state in W_{i+1} which can be its immediate X-successor, i.e., it retrieves the formulas a level below (at depth $i+1$) and "forgets" the information from the position of the path it represents. Formally, for every $i < d_{\mathsf{X}}(\varphi)$ and $[s]_i \in W_i, [t]_{i+1} \in W_{i+1}$,

$$f([s]_i) = [t]_{i+1} \text{ iff } \Theta[s]_i \subseteq [t]_{i+1}.$$

The function is well-defined, because (1) if $[s]_i$ is consistent, so will be $\Theta[s]_i$, hence there exists some maximal set $t \supseteq \Theta[s]_i$ and $([s]_i, [t]_{i+1}) \in f$; and (2), whenever $f([s]_i) = [t]_{i+1} = [t']_{i+1}$, every $\psi \in \Sigma_{i+1} \cap t$ has $\mathsf{X}\psi \in \Sigma_i \cap s$ (by maximality of s with respect to Σ_i), hence $\psi \in t'$ and $t \sim_{i+1} t'$.

[11] See, for instance, [OLP21].

[12] In other words, for every $\neg\Diamond\psi \in \Sigma_i$, if $\neg\Diamond\psi \in s$ then $\neg\psi, \neg\Diamond\psi \in t$.

Thus far, we have constructed $\mathcal{M}_i = \langle W_i, R_i^1, R_i^{\Diamond}, V_i \rangle$ along f, where $V_i([s]_i) = s \cap \Sigma_i \cap P_\varphi = \{p \in P_\varphi \mid p \in s \cap \Sigma_i\}$. Observe how the resulting model is well-behaved in terms of complexity in that no level can be accessed from a deeper one by means of any of the relations we have presented (i.e. no looping due to arbitrarily long paths):

Definition 4.6. *The structure* $\mathcal{M}_\varphi = \langle W_\varphi, f, R_\varphi^1, R_\varphi^{\Diamond}, V_\varphi \rangle$ *is given by* $W_\varphi = \bigcup_{i \leq d_{\mathsf{X}}(\varphi)} W_i$, $R_\varphi^1 = \bigcup_{i \leq d_{\mathsf{X}}(\varphi)} R_i^1$, $R_\varphi^{\Diamond} = \bigcup_{i \leq d_{\mathsf{X}}(\varphi)} R_i^{\Diamond}$ *and* $V_\varphi = \bigcup_{i \leq d_{\mathsf{X}}(\varphi)} V_i$.

As mentioned previously, another means of constructing such models would have arisen from doing without the $\mathsf{X}^i\psi$ inherited from lower levels: we would have obtained a more concise structure, but its information would not have sufficed to make f a function, even if though it was always possible to choose some function within the resulting relation. For instance:

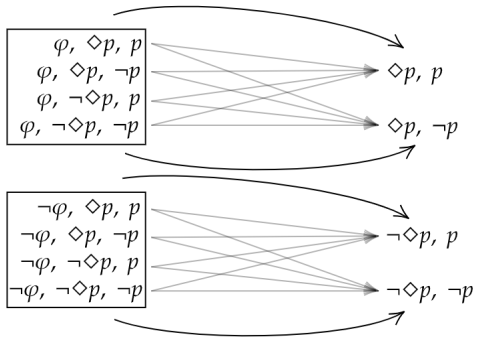

Figure 4.4: For $\varphi = \mathsf{X}\Diamond p$, suppose that our construction lets Σ_i not contain the formulas from $\mathsf{X}^j\psi$ with $\psi \in \Sigma_{i+j}$. Identify the states from W_i coinciding in the formulas from level Σ_i. Then, the relation f is not a function. For the sake of conciseness, let us neglect R_i^1 and $R_i^{\Diamond}$ in the drawing.

Returning to $\mathcal{M}_\varphi$, semantically, $\mathcal{L}_{\mathsf{X}}$-formulas in $\mathcal{M}_\varphi$ are evaluated as usual, with the exception of the now more general X,

$$\mathcal{M}_\varphi, [s]_i \vDash \mathsf{X}\psi \quad \text{iff} \quad i < d_{\mathsf{X}}(\varphi) \text{ and } \mathcal{M}_\varphi, f([s]_i) \vDash \psi,$$

so we have defined a structure whose relations are tied to the labelling of their states. In particular, our current goal consists in verifying the a truth lemma, that characterizes the formulas satisfied in a state through the formulas in its equivalence class.

Lemma 4.7. *For every* $i \leq d_{\mathsf{X}}(\varphi)$, $\psi \in \Sigma_i$, $[s]_i \in W_i$,

$$\mathcal{M}_\varphi, [s]_i \vDash \psi \quad \textit{if and only if} \quad \psi \in s\,(\cap \Sigma_i)\,.$$

Proof. By induction on $d_{\mathsf{X}}(\varphi) - i$, we show that, for every formula $\psi \in \Sigma_i$, we have $[\![\psi]\!] \cap W_i = \{[s]_i \in W_i \mid \psi \in s \cap \Sigma_i\}$. In the base case $i = d_{\mathsf{X}}(\varphi)$, there are no X operators, and we prove the claim by induction on the construction of ψ:

- $\psi = p \in P_\varphi$: $\psi \in s \cap \Sigma_i$ iff $p \in V_i([s]_i)$ iff $\mathcal{M}_\varphi, [s]_i \vDash p$.
- $\psi = \neg\chi$ follows by induction hypothesis (IH2), noting that $\neg\chi \in s \cap \Sigma_i \iff \chi \notin s \cap \Sigma_i$.
- $\psi = \chi \wedge \chi'$ may appear in $s \cap \Sigma_i$ not as a conjunction, but as both subformulas χ, χ' simultaneously. This is equivalent to $\mathcal{M}_\varphi, [s]_i \vDash \chi, \chi'$.
- $\psi = \Diamond_1\chi \in s \cap \Sigma_i$ iff there exists some $t \in W_c$ such that $\chi \in t \cap \Sigma_i$ and, for all $\neg\Diamond_1\chi' \in s$, we have $\neg\chi' \in t$. Such a t satisfies $[s]_i R_i^1 [t]_i$ and $\mathcal{M}_\varphi, [t]_i \vDash \chi$, i.e. $\mathcal{M}_\varphi, [s]_i \vDash \Diamond_1\chi$.
- $\psi = \Diamond\chi \in s \cap \Sigma_i$ iff there exists some $t \in W_c$ such that $\chi \in t \cap \Sigma_i$ and $sR^\Diamond t$. By Lemma 4.5, this is equivalent to state the existence of a $t \in W_c$ with $\mathcal{M}_\varphi, [t]_i \vDash \chi$ and $[s]_i (R_i^1)^* [t]_i$, so $\mathcal{M}_\varphi, [s]_i \vDash \Diamond\chi$.

The inductive step follows an analogous nested induction argument, but it now features the X operator. For $\psi = \mathsf{X}\chi$, $\mathsf{X}\chi \in s \cap \Sigma_i$ if and only if $\chi \in f([s]_i)$. By the first inductive hypothesis, $\mathcal{M}_\varphi, f([s]_i) \vDash \chi$, but that is equivalent to the desired $\mathcal{M}_\varphi, [s]_i \vDash \mathsf{X}\chi$. □

Observation 4.8. *Given that $\mathcal{M}_\varphi$ contains every Σ^*-maximal consistent set, any $\psi \in \Sigma_i$ is satisfiable iff it is contained in some Σ^*-maximal consistent set $s \in [s]_i \in W_i$.*

Proposition 4.9. *Satisfiability without a fixed grammar is decidable in* NEXPTIME.

Proof. Since every formula φ yields $|\varphi| =: n$ subformulas, and each $(\neg)\psi \in \Sigma^*$ refers either to such subformulas or some fixed variation, $\mathcal{M}_\varphi$ features at most $n2^n$ states, with finite relations. So it is enough to guess a model not larger than that and run a model-checking algorithm.

The fact that every path only has $d_{\mathsf{X}}(\varphi)$ relevant cells[13] ensures that model checking is decidable in PSPACE, due to 3.12. Observe that the relation $\Diamond_1$ we can deduce from $\Rightarrow$ indeed corresponds to a grammar, since every state $[s]_i$ where we have establish R_i^1 defines a single path as $([s]_i, f^{(1)}[s]_i, f^{(2)}[s]_i, \ldots)$. However, note that rules which we might try to describe as $u \rightsquigarrow v$ might be defined as $ux_f \rightsquigarrow vx_f$ instead, where x_f is a symbol that denotes the end of a sequence, so that the (context-sensitive) rules can only be applied to the words they are meant to.[14] □

[13] Read comment before Theorem 4.11.

[14] I.e. so that we cannot apply rules defined on smaller words to larger ones.

[The following is not necessary, given that we already have established our sought result and we can search models by brute force:] Nevertheless, we still do not know such MCS's, let alone $\mathcal{M}_\varphi$. But the same proof of Lemma 4.7 hints at how we can construct them and check which subformulas are satisfiable. Essentially, we check the satisfiability[15] of every maximal $s \subseteq \Sigma_i$, starting at $i = d_{\mathsf{X}}(\varphi)$ and working our way up: having completed each level, we can forget the previous one and only preserve the information we need – the satisfiability of a set of formulas some state is reaching to. The following result[16] to decide whether sets with $\Diamond_{(1)}$ are consistent will be convenient to complete the algorithm.

Theorem 4.10. *K modal logic with some operator $\Diamond$ and its transitive closure $\Diamond^*$ is* EXPTIME*-complete. Let* $\mathrm{SAT}_{\Diamond^*}(\Gamma)$ *be an algorithm of such a complexity which decides the satisfiability of any* Γ.

To be more precise, our algorithm[17] starts at the lowest level, taking every maximal subset of $\mathcal{P}(\Sigma_{d_{\mathsf{X}}(\varphi)})$ without complementary literals, and applying $\mathrm{SAT}_{\Diamond^*}$ in order to select the consistent ones. Then, it connects the states where the definitions of $R_i^{(1)}$ apply. For the upper levels W_i, it first chooses the maximal sets s whose Θs appears in W_{i+1} (otherwise, they are unsatisfiable) and it connects them via f. At this point, it can repeat the verification with $\mathrm{SAT}_{\Diamond^*}$ by disguising X-literals with fresh variables.

Observe that, at every W_i, every state $[s]_i$ contains the formulas supposed to be satisfied at that level, and not the $(\neg)\psi$ common to every $s' \in [s]_i$. For the sake of convenience, we will simply write s instead of their class notation $[s]_i$, since the original $s \in W_c$ are never used.

[15] And, thus, the consistency.

[16] Derived from [BHT13] and viewing the modal fragment of our logic as PDL.

[17] It is based in a previous method to derive a pseudo-canonical model on which the truth lemma also held.

Algorithm 3 Generation of a pseudo-model

given φ
compute $d = d_{\mathsf{X}}(\varphi)$
compute Σ and $\Sigma_d, \ldots, \Sigma_0$
let $\mathcal{M}_\varphi := (W_\varphi, f, R^1_\varphi, R^{\Diamond}_\varphi, V_\varphi)$, $W_d, \ldots, W_0$ be empty
let $C_d := \{s \in \mathcal{P}(\Sigma_d) \mid \forall \psi \in \Sigma_d \, (\psi \notin s \leftrightarrow {\sim}\psi \in s)\}$
for $s \in C_d$ **do**
    if $\mathrm{SAT}_{\Diamond^*}(s)$ **then**
        add $s \in W_d, W$ and let $V_\varphi(s) = s \cap P_\varphi$ ▷ Otherwise, do not copy
    end if
end for
for $s, s' \in W_d$ **do**
    add $(s, s') \in R^1_\varphi$ (resp. $R^{\Diamond}_\varphi$) if the definition of $sR^1_d s'$ ($sR^{\Diamond}_d s'$) holds
end for
for $i = d - 1$; $i \geq 0$; i-- **do**
    let $C_i := \{s \in \mathcal{P}(\Sigma_i) \mid \forall \psi \in \Sigma_i \, (\psi \notin s \leftrightarrow {\sim}\psi \in s)\}$
    for $s \in C_i$ **do**
        if $\Theta s \subseteq t$ for some $t \in W_{i+1}$ **then**
            for all $(\neg)\psi \in \Theta^{+/-} s$, replace every instance of $\mathsf{X}\psi$ by a fresh p_ψ
            if $\mathrm{SAT}_{\Diamond^*}(s)$ **then**
                reverse the change $[\mathsf{X}\psi / p_\psi]$, add $s \in W_i, W$ and let $V_\varphi(s) = s \cap P_\varphi$
            end if
        end if
    end for
    for $s, s' \in W_i$, $t \in W_{i+1}$ **do**
        add $(s, s') \in R^1_\varphi$ (resp. $R^{\Diamond}_\varphi$) and $(s, t) \in f$ if applicable
    end for
end for
if $\varphi \in s$ for some $s \in W_0$ **then**
    accept
end if
reject

Note that we can incorporate pairs of states into the relations at any point after their level has been constructed, and $f(s) = t$ can even be defined the moment s is attested to be satisfiable. Nevertheless, we opted to combine all of the additions in a single loop on grounds of better readability.

This algorithm provides models for every satisfiable subformula, and it does so in a complexity not lower than EXPTIME (because of $\mathrm{SAT}_{\Diamond^*}$ and the creation of the

Σ_i). As a matter of fact, though, it is necessary to append an empty level $W_{d_{\mathsf{X}}(\varphi)+1}$, since formulas are supposed to be objects of evaluation even if they contain more than $d_{\mathsf{X}}(\varphi)$-many X. So, create some $[s_\varepsilon]$ with no positive propositional variables such that, for every $[s]_{d_{\mathsf{X}}(\varphi)}$ in $W_{d_{\mathsf{X}}(\varphi)}$, $f([s]_{d_{\mathsf{X}}(\varphi)}) = [s_\varepsilon]$.

Theorem 4.11. *The problem of determining if the existence of a grammar for which a model satisfies a given $\mathcal{L}_{\mathsf{X}}$-formula φ is decidable in some complexity class $\mathcal{C}$ such that* EXPTIME $\subseteq \mathcal{C} \subseteq$ NEXPTIME.

The lower bound is obtained by considering the collection of formulas $\{\varphi \in \mathcal{L}_{\mathsf{X}} \mid d_{\mathsf{X}}(\varphi) = 0\}$, which reflects a fragment of $\mathcal{L}_{\mathsf{X}}$ that results EXPTIME-complete due to Theorem 4.10.

In the next section, we will make use of this construction to discuss the decidability of some fragments of $\mathcal{L}_{\mathsf{X}}$.

A note on $\mathcal{L}_{\mathsf{U}}$

The problem of deciding if any grammar can provide a model in which a given formula holds in the logic $\mathcal{L}_{\mathsf{U}}$ is probably, so to say, the most general question we have come across in this thesis. As such, we might find better results by reducing it to existing knowledge in the literature.

We have previously stated that $\mathcal{L}_{\mathsf{U}}$ does not possess the Church-Rosser property, hence [GKWZ03] affirms that it cannot be a product. However, analogously to $\mathcal{L}_{\mathsf{X}}$, it can be seen as a fusion of two logics: LTL and a version of K with an operator for the reflexive transitive closure of its modal operator. The latter can be reduced to propositional dynamic logic, PDL ([Tr07]). Let us present the following result from [KW91]:

Theorem 4.12. *Let $\mathcal{L}, \mathcal{L}'$ be two consistent, complete logics. Then, the fusion of $\mathcal{L}$ and $\mathcal{L}'$ is decidable if both $\mathcal{L}$ and $\mathcal{L}$ are decidable.*

By [CGP01] and [HKT00], LTL and PDL are decidable and complete with respect to their calculus. So $\mathcal{L}_{\mathsf{U}}$ can be reduced to their fusion and be decided as well. Additionally, it is axiomatized through the reunion of their axioms (and rules from the calculus).

4.2 Satisfiability in fragments and fixed grammars

In the first place, it is possible to find a reduction from the Post Correspondence Problem to $\mathrm{SAT}_G(\mathcal{L}_X)$, as in the case of $\mathrm{MC}(\mathcal{L}_X)$. Given (Σ, U, V) an instance of the PCP, consider the grammar G from Proposition 3.5 and $\varphi = S_0 \wedge \Diamond\left(\neg S_0 \wedge \bigwedge_{s\in(\Sigma\cup\{\lambda\})^2} \neg s\right)$.[18] On the one hand, whenever there exists a solution for (Σ, U, V), the basic model $\mathcal{M}$ described in Proposition 2.5 will satisfy φ in (S_0). But, if no such solution exists, by completeness of the basic product model, φ will be unsatisfiable for G: in other words, since $\mathcal{M}$ contains every possible path connected in accordance to the grammar's rule set, no path starting with S_0 (so, in particular, (S_0)) can reach the pertinent empty path in any model, because such path then would be reachable in $\mathcal{M}$.

At any rate, we are still able to provide positive results when considering specific conditions.

Fragments of $\mathcal{L}_X$

First of all, recall that our proof of general satisfiability allows us to provide prefix-continuous grammars (for $f(x) = x + 1$) for formulas which are satisfiable in some model, through the construction of some witness model. Given some formula φ and a grammar G, it is possible to run our previous algorithm to find every possible relevant "grammarless" model where φ is satisfied: while there is an option to check the admissibility of G in such models, it is not sure that we can rule out that such admissibility (or lack of thereof) is preserved in larger models.

Nevertheless, if G is a prefix-continuous grammar for some $f(x)$, note that for every path π where some subformula of φ may be interpreted, the valuation of its positions beyond $\pi[0, d_X(\varphi))$ is only relevant to assess the accessibility between paths, but not to check whether some propositional variable is satisfied. Thus, we can modify our construction of $\mathcal{M}_\varphi$ in order for it to represent the reach between paths as defined by G.

Start by finding $d_k(\varphi)$,[19] the bound on the number of cells needed to consider to perform model checking, as stated in Lemma 3.9. Then, compute the $l(\varphi)$ corresponding to the longest possible visited path during any such model checking. Consider m the maximum length of the right hand sides of the rules of G.

Now, for every $i \leq l(\varphi) + m$, define recursively the sets of propositional vari-

[18] Since the model with S_0 as a state is not available, we cannot quantify the conjunction over S_0, hence the added clause.

[19] Even though the notation finds its roots in prefix-continuous rule sets for $f(x) = x + k$, the procedure is analogous for any other kind of function.

ables Δ_i as

$$\Delta_{l(\varphi)+m} = \Delta := \{s \cup \{\neg p \mid p \in P_\varphi \setminus s\} \mid s \subseteq P_\varphi\},$$

the possible combinations of variables, and –for smaller i's–

$$\Delta_i = \{s_0 \cup \{\mathsf{X}\psi \mid \psi \in s_1 \text{ positive}\} \cup \{\neg\mathsf{X}\psi \mid \neg\psi \in s_1\} \mid s_0 \in \Delta \wedge s_1 \in \Delta_{i+1}\}.$$

This defines all possible paths of length $l(\varphi) + m$. Appending trivial empty states $s_{d_\mathsf{X}(\varphi)+1} \in W_{d_\mathsf{X}(\varphi)+1}, \ldots, s_{l(\varphi)+m} \in W_{l(\varphi)+m}$, consider the universe W obtained from the level-wise product of the W_i and the Δ_i, by only keeping the consistent sets. It can be assimilated if some (s, s') contain formulas in common.

Then, at each state s, include the formulas which encode the functioning of the grammar: on the one hand, given a rule $u := u_0 \ldots u_n \rightsquigarrow v_0 \ldots v_r =: v$, for every $i \leq l(\varphi)$, the accessibility condition

$$\mathsf{X}^i\chi_u \to \Diamond_1\mathsf{X}^i\chi_v,$$

where $\chi_u := \bigwedge_{j\leq n}\mathsf{X}^j u_j$ is the characteristic formula of the word u, and χ_v is similarly defined. On the other hand, we include $\neg\phi$ for every possible χ which describes a path inaccessible by a single rule application from the one which s defines through $(s \cap P_\varphi, \Theta^1 \cap P_\varphi, \Theta^2 \cap P_\varphi, \ldots)$.

Finally, define relations for $\Diamond_1$, $\Diamond$ and X in the same way as the construction of $\mathcal{M}_\varphi$.[20] The resulting structure not only preserves the truth lemma, but also incorporates the rule set of G in its relations.

Proposition 4.13. *Satisfiability for any fixed grammar of the fragment of $\mathcal{L}_\mathsf{X}$ with rule sets prefix-continuous for some $f(x) = x + k$ is decidable.*

Now we may discern particular cases of grammars, formulas and languages, starting with formulas without $\Diamond$.

[Sh03] showed the PSPACE-completeness of certain modal logics. Since the lowest complexity we could establish for $\mathrm{MC}(\mathcal{L}_\mathsf{X})$ is PSPACE, we may work under the assumption that $\mathrm{SAT}_\mathrm{G}(\mathcal{L}_\mathsf{X})$ will share this lower bound (otherwise, we could reduce model checking to this satisfiability). In the context of rule sets with specific attributes, this indicates that we may contemplate Algorithm 2 with no additional cost. But, for now, let us remain in rule sets with full generality and provide a result for a particular fragment.

Proposition 4.14. *Satisfiability (for fixed grammars) of $\mathcal{L}_\mathsf{X} \restriction \Diamond_1$, the fragment of $\mathcal{L}_\mathsf{X}$ which omits $\Diamond$, is decidable in* PSPACE.

[20] Another method of proceeding would have been to directly add such accessibility restrictions to the definition of the relations as such.

Proof. We start by showing the following:

Claim: Let φ for $\mathcal{L}_{\mathsf{X}} \restriction \Diamond_1$. Then, φ is satisfiable if and only if there exist $N \leq |\varphi|_{\Diamond_1}$ and some $\mathcal{M}, \pi \vDash \varphi$ such that, for some (possibly equal) finite $\pi_0, \ldots, \pi_N$, the set of paths of $\mathcal{M}$ is $\Pi_{\mathcal{M}} = \{\pi_i[j_i, j_i') \mid i \leq N, 0 \leq j_i < j_i' \leq |\pi_i|\}$. That is, φ holds in a model all whose paths are subsequences of some element among a fixed, finite collection of finite paths.

Proof: We show the direct implication by induction on the $\Diamond_1$-depth of φ. If $d_{\Diamond_1}(\varphi) = 0$, φ is and LTL$\restriction$X-formula, so any satisfying $\mathcal{M}, \pi \vDash \varphi$ only involves the sequence $\pi[0, d_{\mathsf{X}}(\varphi)]$. Hence, φ holds in the structure $\mathcal{M}' = \{\pi(0), \ldots, \pi(d_{\mathsf{X}}(\varphi))\}$ whose paths are subpaths of π.

For $d_{\Diamond_1}(\varphi) > 0$, express φ as $\theta(\Diamond_1\psi_1, \ldots, \Diamond_1\psi_k)$, for some $\theta(x_1, \ldots, x_k)$ with $d_{\Diamond_1}(\theta) = 0$ and ψ_i for lesser $\Diamond_1$-depth. Now, find some $\theta' \equiv \theta$ positive Boolean combination[21] of $\mathsf{X}^{x(a)}\theta_a$, where θ_a is either a literal, x_i or $\neg x_i$, so that $\varphi \equiv \theta'[x_i \leftarrow \Diamond_1\psi_i]$. Notice that $|\theta|_{x_i} = |\theta'|_{x_i}$, i.e., we are not adding any new instance of $\Diamond_1\psi_i$, as $\neg(\psi \wedge \chi) \equiv \neg\psi \vee \neg\chi$ and $\neg\mathsf{X}\psi \equiv \mathsf{X}\neg\psi$. Hence, if there exist some $\mathcal{M}, \pi_0 \vDash \varphi$, there is also some model $\mathcal{M}$ containing as paths only π_0 and the paths that $\mathsf{X}^{x(a)}\theta_a[x_i \leftarrow \Diamond_1\psi_i]$ reference.

Then, $\pi_0 \vDash \mathsf{X}^{x(a)}\theta_a[x_i \leftarrow \Diamond_1\psi_i]$ iff $\pi_0[x(a), +\infty) \vDash \theta_a[x_i \leftarrow \Diamond_1\psi_i]$ iff

- (for $\theta_a = p \in \mathrm{Prop}$) $p \in V(\pi(x(a)))$.
- (for $\theta_a = \neg p$, $p \in \mathrm{Prop}$) $p \notin V(\pi(x(a)))$.
- (for $\theta_a = x_i$) there is some $\pi \Rightarrow \pi'$ where ψ_i holds.
- (for $\theta_a = \neg x_i$) there is no $\pi \Rightarrow \pi'$ where ψ_i holds.

Given that only case 3 entails the existence of other satisfying paths, φ holds in π_0 for some $\mathcal{M}$ having π_0 and the paths which, by induction hypothesis, validate the positive $\Diamond_1\pi_i$. This concludes the inductive argument: for every $\Diamond_1$ in φ, we consider no more than a new path. ■

To check the satisfiability of φ, formulate an algorithm which allots $|\varphi|_{\Diamond_1} < |\varphi|$ slots for the possible maximal paths which define any possible model $\mathcal{M}$. As displayed in the first secondary induction step of Proposition 2.8, we can bound the length of these paths by $d_{\mathsf{X}}(\varphi)$ and the size of the left hand side of our derivation rules. This takes cubic space with respect to the length of the input formula, so applying $MC'(\mathcal{L}_{\mathsf{X}})$ to the resulting models is still PSPACE. □

Note that this argument does not find a direct translation for the general (continuous) case, as there would not only be computably many paths to check. If we turn to rules prefix-continuous for some $f(x) = x + k$, we can return to $\mathcal{L}_{\mathsf{X}}$ and lay

[21] This does not affect our complexity.

out an analogous argument to that of Theorem 3.10: checking the accessibility of a certain path can be simulated by generating all possible intermediate sequences.

Proposition 4.15. *Satisfiability (for fixed grammars) of the fragment* $\{\Diamond\varphi \mid d_{\Diamond_1}(\varphi) = d_{\Diamond}(\varphi) = 0\}$ *of* $\mathcal{L}_{\mathsf{X}}$*, with rules prefix-continuous for* $f(x) = x + k$*, is* PSPACE.

Proof. The current fragment only contains formulas of the type $\Diamond\varphi$, where φ is an LTL$\restriction$ X-formula. Given the space restrictions of rules and the fact that the truth value of such a φ is determined by the $d_{\mathsf{X}}(\varphi)$ first positions of the path it is evaluated in, it is enough to consider $l := d_{\mathsf{X}}(\varphi) + k$ positions to study $\Diamond\varphi$. This implies that any derivation sequence of paths between some surmised $\pi \vDash \Diamond\varphi$ and $\pi' \vDash \varphi$ which is longer than $|P_\varphi|^l$ contains some visited path and is redundant.

Therefore, our algorithm will accept $\Diamond\varphi$ exactly if there exists some verifier $|\pi| \leq l$ which can access some $\pi' \vDash$ in no more than $|P_\varphi|^l$ rule applications. It will do so by checking arbitrary derivations until a counter to $|P_\varphi|^l$ is exhausted:

Algorithm 4 Satisfiability of the positive fragment of $\mathcal{L}_{\mathsf{X}}$

1: given $(\Diamond)\varphi$
2: guess π of length l
3: let $c := 0$
4: **while** $c < |P_\varphi|^l$ and $\pi \nvDash \varphi$ **do**
5: guess $\pi \Rightarrow \pi'$
6: $\pi \leftarrow \pi'$
7: c++
8: **end while**

Since $\log(|P_\varphi|^l) \in \mathcal{O}(|\varphi|)$, our non-deterministic algorithm witnesses that the problem is NSPACE = PSPACE: it accepts a formula φ if and only if there exists a path satisfying it which is reachable from another. There is no need to discern whether any path should be in the model, as φ contains no instance of $\neg\Diamond$. □

Note that this proof cannot be reproduced for formulas beyond $\Diamond\varphi$, at least in NSPACE: let us inspect what would occur if, for a given $\Diamond\varphi \wedge \neg\Diamond\varphi'$, both $\Diamond\varphi$ and $\neg\Diamond\varphi'$ were satisfiable. Our algorithm would have to find a verifier the conjunction holds, and any witness for $\Diamond\varphi$ would certainly certify that the first clause holds. But, since the only way to assert that $\Diamond\varphi'$ cannot be satisfied is by letting out non-deterministic algorithm fail for every choice of a verifier. In other words, despite us being able to check in exponential time whether there is a path satisfying $\Diamond\varphi$ and no path satisfying $\Diamond\varphi'$, no NSPACE algorithm of this sort can address the problem because no sole witness can check $\neg\Diamond\varphi'$.

The procedure of the proof was also insubstantial for Theorem 3.10 to remain in PSPACE, because we would have expected it to work for negative formulas: for that, a deterministic version of the algorithm is needed – however, this implies storing every applied rule and requires exponential time (which was bypassed in Algorithm [2] by erasing data from closed instances). Conversely, it would not have been possible to use Savitch's trick for a proof of the full fragment of $\mathcal{L}_{\mathsf{X}}$, since we would still have had to consider exponentially-many potential models.

Now recall Theorem 4.17, which indicates a possible lower bound for the complexity of $\mathcal{L}_{\mathsf{X}}$, and denote by $\mathcal{L}_{\Diamond^*}$ the logic it alludes. Viewing $\mathcal{L}_{\mathsf{X}}$ as a logic whose main modal operator is $\Diamond_1$ (and $\Diamond$), this fact can be translated to $\mathcal{L}_{\mathsf{X}}$ in so far as X can be assimilated in some way. One possibility would arise from having our formulas possess the properties below.

$$\forall\varphi;\, \forall\mathcal{M}\forall\pi\ (\mathsf{X}\Diamond_1\varphi \leftrightarrow \Diamond_1\mathsf{X}\varphi)\ \wedge\ (\mathsf{X}\Diamond\varphi \leftrightarrow \Diamond\mathsf{X}\varphi)$$

In general, this is not the case for general rules, or even prefix-continuous ones:

Example 4.16. Let G be a grammar with $R = \{\mathrm{C} \rightsquigarrow \mathrm{GC}\}$. Consider the model $\mathcal{M}$ with universe $\{\mathrm{G},\mathrm{C},\mathrm{F}\}$,[22] $\rightarrow = \mathrm{Id} \cup \{(\mathrm{G},\mathrm{C}),(\mathrm{C},\mathrm{F})\}$, and the corresponding $\Rightarrow$ relation. For $\pi = (\mathrm{C},\mathrm{F})$, we have

$$\exists\pi \Rightarrow \pi' = (\mathrm{G},\mathrm{C},\mathrm{F})\ (V(\pi'(1)) = \mathrm{C}),\ \text{thus}\ \exists\pi \Rightarrow \pi'\ (\mathcal{M},\pi' \vDash \mathsf{X}\mathrm{C}),\ \text{i.e.}\ \mathcal{M},\pi \vDash \Diamond_1\mathsf{X}\mathrm{C},$$

but the sought equivalence is not fulfilled:

$$\nexists(\mathrm{F}) \Rightarrow \pi'\ (V(\pi'(0)) = \mathrm{C}),\ \text{so}\ \mathcal{M},(\mathrm{F}) \vDash \neg\Diamond_1\mathrm{C}\ \text{and}\ \mathcal{M},\pi \nvDash \mathsf{X}\Diamond_1\mathrm{C}.$$

Even for grammars whose rules are defined in a similar basis for every chord, it is possible that the property does not hold. This is the case of the fragment $\{X \rightsquigarrow X\ X,\ X \rightsquigarrow \Delta/X\ X,\ X \rightsquigarrow V/X\ X\}$ from [Ro20]'s full grammar rule set, which fails to satisfy $\mathrm{A}\flat\mathrm{C} \nvDash \Diamond_1\mathsf{X}\mathrm{A}\flat \leftrightarrow \mathsf{X}\Diamond_1\mathrm{A}\flat$. This happens despite such an interchange equivalence (commonly expressed as some commutativity principle in the context of product logics, [GKWZ03]) being coherent for models which are complete with respect to the set of derivable sequences of its associated grammar.

Theorem 4.17. *Satisfiability (with or without fixed grammars)*[23] *of* $\mathcal{L}_{\mathsf{X}}$*, under prefix-continuous rules, belongs to no proper subclass of* EXPTIME. *There exists some fragment with that precise complexity.*

[22] Identifying the notation for states and their valuations.

[23] The proof actually works without fixed grammars, but this problem can be reduced to fixed grammars.

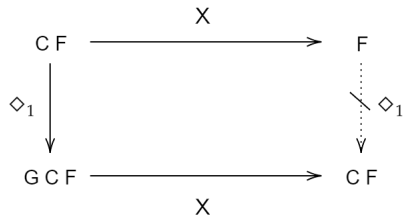

Figure 4.5: Non-commutativity of the operators $\Diamond_1$ and X.

Proof. Let $\mathcal{L}'_{\mathsf{X}}$ be the fragment where $\forall\varphi\forall\mathcal{M}\forall\pi\ (\mathsf{X}\star\varphi \leftrightarrow \star\mathsf{X}\varphi)$ holds for $\star \in \{\Diamond_1, \Diamond\}$. Every formula φ for $\mathcal{L}'_{\mathsf{X}}$ is equivalent to some $\phi = \phi'[x_i \leftarrow \mathsf{X}^{n_i}p_i]$, for some $p_i \in \mathrm{Prop}$ and ϕ' with null X-depth. Hence, taking $\mathrm{Prop}' := \{\mathsf{X}^n p \mid n \in \mathbb{N} \wedge p \in \mathrm{Prop}\}$, we can view formulas in the fragment as unimodal in $\Diamond_1$ and its transitive closure $\Diamond$.[24] Therefore, decidability of $\mathcal{L}'_{\mathsf{X}}$ reduces to $\mathrm{SAT}(\mathcal{L}_{\Diamond^*})$, which belongs to EXPTIME.

Now, supposing there was an algorithm to decide $\mathcal{L}'_{\mathsf{X}}$ more optimally than EXPTIME, given a modal relation R in some model $\mathcal{M} \vDash \mathcal{L}_{\Diamond^*}$ we can always interpret the formula in some model of $\mathcal{L}'_{\mathsf{X}}$. This is due to $\mathcal{L}'_{\mathsf{X}}$ being a normal modal logic, as it satisfies

$$\Box_1(\phi \to \psi) \to (\Box_1\phi \to \Box_1\psi) \equiv (\neg\Diamond_1\neg\phi \wedge \Diamond_1\neg\psi) \to \Diamond_1(\phi \wedge \neg\psi),$$

and the latter is valid, because the existence of an accessible path π where ψ does not hold –along with the fact that no path where ϕ fails is accessible– implies that π satisfies ϕ but not ψ. But that would imply that $\mathcal{L}_{\Diamond^*}$ reduces to $\mathcal{L}'_{\mathsf{X}}$ and is not EXPTIME-complete, a contradiction. □

Some relevant rule sets for which the aforementioned exchange property holds are those based on substitution principles, such as tritone or backdoor dominants, which are quite limited. However, the fragment $\{\Diamond\varphi \mid d_{\Diamond_1}(\varphi) = d_{\Diamond}(\varphi) = 0\}$ is already versatile enough to contemplate many musically motivated problems. Furthermore, the proof of Proposition 4.15 can be reproduced for formulas $\Diamond\varphi$ where φ may contain $\Diamond_1$ besides X, by just adding $d_{\Diamond_1}(\varphi) \cdot \max\{|u| - |v| \mid u \rightsquigarrow v\}$ to l.

At any rate, the discussion of such principles can often be overlooked due to the expressive power of the most widespread grammars. For instance, our previous case $\mathrm{A}\flat\mathrm{C} \nvDash \Diamond_1\mathrm{XA}\flat \leftrightarrow \mathrm{X}\Diamond_1\mathrm{A}\flat$ does hold when replacing $\Diamond_1$ with $\Diamond$: on the one hand, $\mathrm{A}\flat\mathrm{C}$ yields $\mathrm{A}\flat\mathrm{A}\flat\mathrm{C}$, which satisfies $\mathrm{XA}\flat$. On the other, $\mathrm{C} \vDash \Diamond\mathrm{A}\flat$ through $\mathrm{C} \Rightarrow_G \mathrm{GC} \Rightarrow_G \mathrm{DGC} \Rightarrow_G \mathrm{A}\flat\mathrm{GC}$. Even $\mathrm{A}\flat\mathrm{C}$ can be derived from C as

$$\mathrm{C} \Rightarrow_G \ldots \Rightarrow_G \mathrm{A}\flat\mathrm{GC} \Rightarrow_G \mathrm{A}\flat\mathrm{DGC} \Rightarrow_G \ldots \Rightarrow_G \mathrm{A}\flat\mathrm{CFBEADGC}.$$

[24] This effectively makes the model disregard states completely and treat (whole) paths as the only elements formulas refer to.

About $\mathcal{L}_{\mathsf{U}}$

Analogous to $\mathcal{L}_{\mathsf{X}}$, we can use the reduction for PCP to the model checking problem in $\mathcal{L}_{\mathsf{U}}$ in order to show that satisfiability for fixed grammars is undecidable. Employing the formula $S_0 \wedge \Diamond\left(\neg S_0 \wedge \left(\bigvee_{a\in\Sigma\cup\{\lambda\}}(a,a)\right)\mathsf{U}\lambda_\varepsilon\right)$ and the last version of the grammar, we conclude that fixed-grammar satisfiability is undecidable for the class of prefix-continuous rule sets for $f(x) = x + 1$.

In order to select certain fragments to study, one could inspect the interaction between operators and the musical motivation behind some formulas, in hopes of demarcating the most relevant options. In the first place, out of the combinations between U and $\neg$ or $\wedge$, only the first component of U is interchangeable, as

$$(\neg\psi)\mathsf{U}\chi \equiv \neg(\psi\mathsf{U}\chi) \qquad \text{and} \qquad (\psi\wedge\psi')\mathsf{U}\chi \equiv (\psi\mathsf{U}\chi)\wedge(\psi'\mathsf{U}\chi).$$

Generally, the second component does not possess such a behavior, as it is exemplified by,

$$(\mathrm{D},\mathrm{G},\mathrm{C}) \vDash (\top\mathsf{U}\mathrm{G})\wedge(\top\mathsf{U}\mathrm{C}), \quad \text{but} \quad (\mathrm{D},\mathrm{G},\mathrm{C}) \nvDash \top\mathsf{U}(\mathrm{G}\wedge\mathrm{C})$$

in the case of $\wedge$, and for $\neg$,

$$(\mathrm{D},\mathrm{A}) \vDash \mathrm{D}\mathsf{U}(\neg\mathrm{A}), \quad \text{but} \quad (\mathrm{D},\mathrm{A}) \nvDash \neg(\mathrm{D}\mathsf{U}\mathrm{A}).$$

The remaining modalities (X, $\Diamond_1$, $\Diamond$) do not commute with U in the most general settings, so we may turn to musical incentives to consider or not certain frameworks. For instance, we may demand a harmonic sequence representing the macro-scale of a piece that, until the B section, it is always possible to move towards a turnaround: letting χ entail a cadence and ψ represent a marker of the B section, we can write

$$(\Diamond\chi)\mathsf{U}\psi.$$

However, one could argue that it suffices to look for a sequence which can indeed access a timeline where turnarounds occur periodically until the B section takes place, i.e.

$$\Diamond((\chi\vee\mathsf{X}\chi\vee\ldots\vee\mathsf{X}^r\chi)\mathsf{U}\psi).$$

Otherwise, another condition that could imply the current request is the eventual existence of a ψ and always (never no) being ready to access a turnaround or in a B section:

$$(\top\mathsf{U}\psi)\wedge\neg(\top\mathsf{U}(\neg\Diamond\chi\wedge\neg\psi)).$$

Ultimately, these alternatives seem quite more powerful than their original intention, but give the impression that the second component of 'until' is more suitable for our purposes.

Another example of the flexibility of the operator $\top \mathsf{U} \cdot$ is related to the difference in structural levels in a harmonic skeleton of a piece. The formula

$$V \wedge \Diamond(\top \mathsf{U} V / \flat V)$$

expects a dominant sequence to be able to derive its tritone substitution, whereas the expression

$$V \wedge (\top \mathsf{U}(\Diamond V / \flat V))$$

holds whenever a dominant chord precedes some eventual dominant substitution. In the former, the bond between $V/\flat V$ and V is far stronger, as the substitute might be viewed as stemming of, being derived from the initial dominant. However, the second formula just states that, at some point, $V/\flat V$ will be accessible from some subsequence of the current progression, which (in most context) could more likely be interpreted as a reexposition or a callback, than a direct consequence.

In any case, we highlight two fragments coherent with these views, which incidentally can be addressed through the EXPTIME-completeness of PDL: in the first,

$$\{\theta(\psi_1, \ldots, \psi_r) \mid d_{\mathsf{X}}(\theta(x_1, \ldots, x_r)) + d_{\mathsf{U}}(\theta(x_1, \ldots, x_r)) = 0 \wedge \forall i \leq r(\psi_i \in \mathrm{LTL})\},$$

temporal formulas are pushed to the atoms of a $\Diamond_1, \Diamond$-formula, which is expected to be decidable with a complexity no smaller than EXPTIME. The second fragment,

$$\{\varphi \in \mathcal{L}_{\mathsf{U}} \mid (\chi \mathsf{U} \psi) \in Sub(\varphi) \Rightarrow \chi \in \mathrm{LTL}\},$$

assigns more importance to the second component of every occurrence of until, but it is undecidable in general (as seen with a reduction of the usual style).

Chapter 5

Overview

We review some of the notions and results obtained thus far, from the perspective of certain particular grammars. After that, we summarize and discuss the contents of our work.

5.1 Applications

Recall the model for jazz standards presented by Rohrmeier, as expressed in Table 1.1. Formally, it is a context-free grammar G, as every rule depends on a single variable and it is not compressive: for this reason, it is prefix-continuous for $f(x) = x$, hence its model checking and satisfiabilities are decidable. Furthermore, due to Proposition 3.17, the fragment of formulas $\Diamond(\top \mathsf{X}\varphi)$ is decidable. Such a complexity can be narrowed down to a class within exponential time by handling positions available for string replacement (possibly by introducing new variables) and allowing not more than $d(\varphi)$ many.

Otherwise, we can opt to work with a more concise grammar G' presented by [Fi20]. Chords (enharmonically) at a distance of a minor third are identified, and the only two generation rules are substitution (within the equivalence class) and the preparation by fifths. Thus, given the commutativity[1] of the rules in the fragment, it will suffice to keep track of where the preparation rules are applied.

In any case, as a corollary of the main results, parsing of sequences is PSPACE with any rule set (take $I \wedge \Diamond(c_1 \wedge X(c_2 \wedge X(...)))$ and apply SAT).

Observe the following:

Remark 5.1. The expressiveness of the current grammar G is considered to be enough to attain all possible harmonies in the studied corpus. In fact, all dia-

[1] In the sense that it is always possible to apply the substitution rules at the end, after the tree structure is deriven.

tonic chords can be derived from any starting degree: secondary dominant rule applications and major/minor tonicization derive all 24 keys, while diatonic rule instances allow us to cycle through the degrees of every fixed key. Notice that this also implies that any pair of chords can occur juxtaposed in a valid sequence (given any X, knowing that $X \Rightarrow_G^* Yw$ for every Y and some w, we can apply $X \Rightarrow XX \Rightarrow_G^* XYw$ to obtain XY). Hence, every harmonic sequence comprised of diatonic chords can be generated at some point of the string.

As in the case of G', it is natural to question if any subgrammar of G is equally as expressive, even if it is through more rule applications. Recall that we can also consider the fragment $\Diamond(\top \mathsf{U} \varphi)$, so that we can actually capture substrings of the elements from the language. For instance, the tritone substitution rule can be expressed as the concatenation of a modulation rule and a primary tritone modulation rule:

$$V/X_{key=Y} \rightsquigarrow V_{key=\flat V/X/Y} \quad \text{as} \quad V/X_{key=Y} \rightsquigarrow V_{key=X/Y} \rightsquigarrow V_{key=\flat V/X/Y}$$

In general, the same holds for other sorts of complex modulation transformations, such as the backdoor dominant rule:

$$V/X_{key=Y} \rightsquigarrow V_{key=\flat III/X/Y} \quad \text{as} \quad V/X_{key=Y} \rightsquigarrow V_{key=X/Y} \rightsquigarrow V_{key=\flat III/X/Y}$$

In any case, it is legitimate to ask if such simplifications (which do not affect the outcome in terms of expressiveness) work against the musical understanding of derivations. Despite the fact that every harmonic progression can be generated as a subsequence of an element of the language, part of the convenience of this kind of systems arises from the interpretation of certain disconnects or peculiarities of the derivation tree of a sequence.[2]

Based on the limitations of the grammars, we can propose two modifications to our current ones, so that either they are more representative or that their complexity is more optimal.

On the one hand, we may construct a grammar with distinctive (technical) features for each of the harmonic layers: having analyzed in a set of tunes what kind of transitions occur at a higher level of harmony (and which at a structural macro-scale), we could propose different rules for each of them. In the case that we only distinguish a surface level (L_1) and a structural level (L_0), it suffices to set an upper bound for the length of surface-level sequences which are considered to be a development of a single chord on the structural timeline. Then, the grammar for L_0 can be constructed as usual, and L_1 can be established through rules of the form $X \rightsquigarrow x_1 \dots x_r$, where X represents an L_0 chord and x_i surface ones. This

[2] Personal communication with C. Finkensiep.

system can be refined by introducing "terminal variables" for the structural level, that is, tagged chords which specify what kind of sequence can they be replaced for (I can stand for either a call or a response, modulating or not, etc.).

Additionally, with the cost of having longer sequences, one can transform unrestricted grammars[3] into context-sensitive ones: we can introduce a 'blank' variable such that it cannot be replaced, and that it completes the right hand side of every compressive rule. More precisely,

$$\text{every } u_1 \dots u_r \rightsquigarrow v_1 \dots v_s, \text{ for } s < r, \text{ is replaced by } u_1 \dots u_r \rightsquigarrow v_1 \dots v_s \lambda^{r-s}.$$

In order for the ends of rules substituted in to be operable as intended, we introduce the rule $x\lambda \rightsquigarrow \lambda x$, so that the language of the original grammar coincides with the set of final segments of the words from the new grammar.[4]

Lastly, a brief note regarding the interpretation of derivation trees for sequences in a harmonic grammar. It has been argued that some of the models in the literature have suboptimal implementations due to the fact that states might not preserve information about original tonalities from which the current passage modulated. Knowing a priori the maximal "depth" of embedded modulations, one could introduce additional propositional variables to reflect the route of keys that a chord has been derived from – granted, new rules will have to be considered (allowing for the coexistence of multiple variables in any position of a path) and the overall complexity will be higher.

5.2 Discussion

Throughout this thesis, we have presented a logic and its models, and worked on their relationship to harmonic grammars. In the theoretical aspect, we summarize the conditions for satisfiability (or unsatisfiability) of the most common decision problems:

- Model checking for $\mathcal{L}_X$: the problem is undecidable for any class of grammars all whose representations through formulas depend on some 'until', hence it is undecidable for context-sensitive[5] (and unrestricted) grammars in general. Decidability is achieved by assuming rule-sets are prefix-continuous, thus also context-free grammars. In terms of complexity, prefix-continuous rule sets for some $f(x) = x + k$ yield PSPACE, but lower bounds within PSPACE are achieved with regular grammars.

[3] They still only regard finite rule sets, in the sense of 3.2.

[4] As in the case of PCP-reductions, we can still introduce the rule $x \rightsquigarrow x\lambda$, so that the resulting grammar is prefix-continuous if certain conditions are met.

[5] Recall that swap rules can bring characters to the front.

- Model checking for $\mathcal{L}_{\mathsf{U}}$: again, it is undecidable in general, but for prefix-continuous grammars for $f(x) = x + 1$ too. To reach decidability, we can either remain in the fragment where $\Diamond$ is not available or consider formulas of the form $\Diamond(\top \mathsf{U} \varphi)$ ($\varphi \in \mathrm{LTL} \restriction \mathsf{X}$), with some sorts of context-free grammars.[6]

- Satisfiability (without fixed grammars): in general, the problem is decidable. For $\mathcal{L}_{\mathsf{X}}$, some filtration-like construction can be employed to prove the result; by considering a particular fragment, we show that the complexity is no less than EXPTIME.

- Satisfiability (fixing a grammar): it shares undecidability conditions with model checking. To remain in PSPACE, the formulas considered must often be restricted to the form $\Diamond\varphi$ or not having $\Diamond$.

Mainly, the first reason why undecidability is recurrent is the persistence of grammars which can be replicated in the models. Generally, there are no exchange properties for operators (commutativity, distributivity, etc.) which may reduce the problems to decidable ones: this stems from the fact that most grammars are not compatible with them, although other constructions where that occurs could be studied (like products, etc.).

Related, this particularity can be observed in the gap between model checking and fixed-model satisfiability. It is precisely the relationship between these problems what we initially expected, given that SAT with no given grammar turned out to be more attainable than MC. Indeed, the lack of the structure of a given grammar allows us to view $\Diamond$ in the most general way possible, so no condition beyond existence must be considered when relating two paths.

On the other hand, one of the principal difficulties when trying to reenact proofs from $\mathcal{L}_{\mathsf{X}}$ in $\mathcal{L}_{\mathsf{U}}$ is the appearance of the eventuality operator $\mathsf{F}\varphi := \top \mathsf{U} \varphi$. The analysis of fragments where a single instance of this operator is allowed suffices to illustrate how it is not possible anymore to enforce bounds for checking paths. Instead, more general methods (such as canonical constructions) could become more useful.

As much as reductions can be considered in order to "propagate" undecidability results to less extensive classes, the opposed can be said about some kinds of grammars. Even though some of the proofs have been established for concisely described grammars, we may actually reproduce some of the strategies to slightly more complex classes (by extending them through combinations of rules),

[6] Even though the obtained result only holds for rules in the form $A \rightsquigarrow AB|B|a$, we can extend this proof to more general grammars by decomposing more complex rules.

or simplify the formalization of apparently complex grammars from the musical literature.

Overall, the focus of our work has shifted towards theoretical endeavors. Even if the initial purpose was motivated by the possible computational and music-theoretic advantages of modal logic, the research experience has proven more substantial as an opportunity to learn about widespread methods in the field. It would be advisable to work towards the direction of concrete, less general results, for the musical reasoning to gain ground on general grammar properties.

As possible future work, we argue that some measure of similarity or well-behavior for rule sets could be proposed in order to refine some of the results and make them more musically coherent: recall that some of the reductions were given by seemingly arbitrary reductions, given by the fact that temporal operators and the definition of prefix continuity are oriented towards right-directedness. Another possible path to explore is the discussion on whether the representation of some sorts of models (or generated submodels) can be achieved through the use of the 'until' operator. This would effectively grant a proof for the undecidability of satisfiability with fixed grammars from some class, as the problem would be reduced of an undecidable model checking.

Bibliography

[AW04] Allan, M., and Williams, C. (2004), *Harmonising Chorales by Probabilistic Inference*. Advances in Neural Information Processing Systems 17.

[AIMT97] Asai, S., Inayaki, Y., Matsubara, S., Toyama, K. (1997), *Chart-Based Parsing and Transfer in Incremental Spoken Language Translation* (1997). The 4th Natural Language Processing Pacific Rim Symposium.

[BHT13] Balbiani, P., Herzig, A., and Troquard, N. (2013), *Dynamic Logic of Propositional Assignments: A Well-Behaved Variant of PDL*. 2013 28th Annual ACM/IEEE Symposium on Logic in Computer Science.

[Be19] Beach, D. (2019), *Schenkerian Analysis: Perspectives on Phrase Rhythm, Motive and Form*. Routledge.

[BB06] Blackburn, P., and van Benthem, J. (2006), *Modal Logic: A Semantic Perspective*. Institute for Logic, Language and Computation (ILLC), University of Amsterdam.

[BB10] Blackburn, P., and van Benthem, J. (2010), *Modal Logic for Open Minds*. Center for the Study of Language and Information.

[BS13] Broze, Y., and Shanahan, D. (2013), *Diachronic Changes in Jazz Harmony: A Cognitive Perspective*. Music Perception.

[BS19] Broze, Y., and Shanahan, D. (2019). *A Diachronic Analysis of Harmonic Schemata in Jazz*. International Conference on Music Perception and Cognition.

[CC89] Carey, N., and Clampitt, D. (1989), *Aspects of Well-Formed Scales*. Music Theory Spectrum.

[ca07] Carnielli, W. (2007), *Combining Logics*. Stanford Encyclopedia of Philosophy.

[Ch80] Chellas, B. F. (1980), *Modal Logic: An Introduction*. Cambridge University Press.

[CPP14] Chater, N., Perfors, A., and Piantadosi, S. T. (2014), *Language Processing and Language Learning*.

[CGP01] Clarke, E. M. Jr., Grumberg, O., and Peled, D. A. (2001), *Model Checking*. MIT Press.

[CN11] Clampitt, D., and Noll, T. (2011), *Modes, the Height-Width Duality, and Handschin's Tone Character*. Journal of Music Theory.

[CR08] Rohrmeier, M., and Cross, I. (2008), *Statistical Properties of Tonal Harmony in Bach's Chorales*.

[DN18] De Jong, K., and Noll, T. (2018), *Fundamental Bass and Real Bass in Dialogue: Tonal Aspects of the Structural Modes*. Music Theory Online.

[DN19] De Jong, K., and Noll, T. (2019), *Embedded Structural Modes: Unifying Scale Degrees and Harmonic Functions*. Mathematics and Computation in Music.

[DGL16] Demri, S., Goranko, V., and Lange, M. (2016), *Temporal Logics in Computer Science: Finite-State Systems*. Cambridge University Press.

[Fe24] Fernandez-Duque, D. (2024), *Non-Classical Temporal Logic in Topological Dynamics*. Topology, Algebra and Categories in Logic [Seminar lecture notes].

[FS66] Fine, K., and Schurz, G. (1996), *Transfer Theorems for Multimodal Logics*. Journal of Philosophical Logic.

[Fi20] Finkensiep, C. (2020), *Basic Music Theory*. [Seminar lecture notes].

[Ga06] Gabelaia, D. (2006), *Non-primitive Recursive Decidability of Products of Modal Logics with Expanding Domains*. Annals of Pure and Applied Logic.

[GKWZ03] Gabbay, D., Kurucz, A., Wolter, F., and Zakharyaschev, M., (2003), *Many-Dimensional Modal Logics: Theory and Applications*. Elsevier.

[Gj07] Gjerdingen, R. (2007), *Music in the Galant Style*. Oxford University Press.

[Go87] Goldblatt, R. (1987), *Logics of Time and Computation*. CSLI Publications.

[Gr14] Granroth-Wilding, M. (2014), *A Robust Parser-Interpreter for Jazz Chord Sequences*. Journal of New Music Research.

[Ha20] Harasim, D. (2020), *The Learnability of the Grammar of Jazz*. EPFL.

[HMNR19] Harasim, D., Moss, F. C., Neuwirth, M., and Rohrmeier, M. (2019), *Statistical characteristics of tonal harmony: A corpus study of Beethoven's string quartets*. PLoS One.

[HOR18] Harasim, D., O'Donnell, T. J., and Rohrmeier, M. (2018), *A Generalized Parsing Framework for Generative Models of Harmonic Syntax*. Proceedings of the 19th ISMIR Conference.

[Ha12] De Haas, W. (2012), *Music information retrieval based on tonal harmony*.

[HR09] De Haas, W., and Rohrmeier, M. (2009), *Modeling Harmonic Similarity Using a Generative Grammar of Tonal Harmony*. Proceedings of the 10th International Society for Music Information Retrieval Conference.

[HHT06] Hamanaka, M., Hirata, K., and Tojo, S. (2006), *Implementing "A Generative Theory of Tonal Music"*. Journal of New Music Research.

[HKT00] Harel, D., Kozen, D., and Tiuryn, J. (2000), *Dynamic Logic*. MIT Press.

[HR11] Hedges, T., and Rohrmeier, M. (2011), *Exploring Rameau and Beyond: A Corpus Study of Root Progression Theories*. Mathematics and Computation in Music.

[HT13] Herzig, A., and Troquard, N. (2013), *Dynamic Logic of Propositional Assignments*. 28th Annual IEEE/ACM Symposium on Logic in Computer Science.

[Ho01] Homer, S. (2001), *Computability and Complexity Theory*. Springer.

[Ho09] Hoogeboom, H. J. (2009), *Undecidable Problems for Context-free Grammars*. `https://liacs.leidenuniv.nl/~hoogeboomhj/second/` (last visited on 20/07/2024).

[Hu20] Hunter, T. (2020), *The Chomsky Hierarchy*. Blackwell Companion to Chomsky.

[JL83] Jackendoff, R., and Lerdahl, F. (1983), *A Generative Theory of Tonal Music*. MIT Press.

[KKWZ06] Konev, B., Kontchakov, R,, Wolter, F., and Zakharyaschev, M. (2006), *Dynamic topological logics over spaces with continuous functions*. Advances in Modal Logic.

[KOOTU20] Katsurada, K., Ogura, Y., Ohmura, H., Uehara, Y., Tojo, S. (2020), *Expectation-based parsing for Jazz Chord sequences*. Proceedings of the 17th Sound and Music Computing Conference.

[KW91] Kracht, M., and Wolter, F. (1991), *Properties of Independently Axiomatizable Bimodal Logics*. The Journal of Symbolic Logic.

[KM05] Kremer, P., and Mints, G. (2005), *Dynamic Topological Logic*. Annals of Pure and Applied Logic.

[La06] Lange, M. (2006), *Model Checking Propositional Dynamic Logic with All Extras*. Journal of Applied Logic.

[Le14] Lee, J. R. (2014), *Theory of Computation - Lecture 16: Savitch's Theorem*. `https://courses.cs.washington.edu/courses/cse431/14sp/scribes/lec16.pdf` (last visited on 20/07/2014).

[LP81] Lewis, H., and Papadimitriou, C. (1981), *Elements of the Theory of Computation*. Prentice Hall.

[Ma97] Magnaghi, A., and Tanaka, H. (1997), *A Procedure for K-Local Testability*. SIGAL Workshop Proceedings of the Special Interest Group on Algorithms.

[MP71] McNaughton, R., and Papert, S. (1971), *Counter-free Automata*. MIT Press.

[MR21] Rohrmeier, M., and Moss, F. (2021), "A Formal Model of Extended Tonal Harmony". Proc. of the 22nd Int. Society for Music Information Retrieval Conf.

[NR15] Neuwirth, M., and Rohrmeier, M. (2015), *Towards a Syntax of the Classical Cadence*. In "What is a cadence? Theoretical and analytical perspectives on cadences in the classical repertoire.".

[Ng05] Nguyen, L. A. (2005), *On the Complexity of Fragments of Modal Logics*. Advances in Modal Logic.

[OLP21] Open Logic Project, *Filtrations and Decidability*. Retrieved from: `https://builds.openlogicproject.org/content/normal-modal-logic/filtrations/filtrations.pdf` (last visited on 20/07/2024).

[Po46] Post, E. L. (1946), *A Variant of a Recursively Unsolvable Problem*. Bulletin of the American Mathematical Society.

[Ro11] Rohrmeier, M. (2011), *Towards a Generative Syntax of Tonal Harmony*.

[Ro20] Rohrmeier, M. (2020), *The Syntax of Jazz Harmony: Diatonic Tonality, Phrase Structure, and Form*. Music Theory and Analysis.

[Ro21] Rohrmeier, M. (2021), *A Formal Model of Extended Tonal Harmony*. Proceedings of the 22nd ISMIR Conference.

[Se08] Sergot, M. (2008), *Canonical Models for Normal Logics*.

[Sh03] Shapirovsky, I. (2003), *On PSPACE-decidability in Transitive Modal Logics*. Advances in modal logic.

[Si12] Sipser, M. (2012), *Introduction to the Theory of Computation*. Cengage Learning.

[St99] Steedman, M. (1999), *The Blues and the Abstract Truth: Music and Mental Models*. In "Mental models in cognitive science".

[Tr07] Troquard, N. (2007) *Propositional Dynamic Logic*. Stanford Encyclopedia of Philosophy.

[Ty12] Tymoczko, D. (2012), *The Generalized Tonnetz*. Journal of Music Theory.

[Ve01] Venema, Y. (2001), *Temporal Logic*. In "The Blackwell Guide to Philosophical Logic".