A Sequent Calculus for a Modal Logic on Finite Data Trees

David Baelde[1], Simon Lunel[2], and Sylvain Schmitz[3]

1 LSV, ENS Cachan & CNRS & Inria, Université Paris-Saclay, France
2 Inria Rennes, France
3 LSV, ENS Cachan & CNRS & Inria, Université Paris-Saclay, France

Abstract

We investigate the proof theory of a modal fragment of XPath equipped with data (in)equality tests over finite data trees, i.e. over finite unranked trees where nodes are labelled with both a symbol from a finite alphabet and a single data value from an infinite domain. We present a sound and complete sequent calculus for this logic, which yields the optimal PSPACE complexity bound for its validity problem.

1998 ACM Subject Classification F.2.2 [Nonnumerical Algorithms and Problems] Complexity of Proof Procedures, F.4.1 [Mathematical Logic] Modal Logic, H.2.3 [Languages] Query languages

Keywords and phrases XPath, proof systems, modal logic, complexity

Digital Object Identifier 10.4230/LIPIcs.CSL.2016.32

1 Introduction

Arguably the most widespread language for querying XML documents, XPath allows to select and extract elements and values from XML documents. It is embedded in the XSLT and XQuery languages and implemented through libraries in many general-purpose programming languages. The language in its successive revisions has evolved into a full-fledged programming language [22], but its distinguishing feature remains a navigational core (known as **CoreXPath** [5]) supplemented with the ability to perform data joins – this is captured in the fragment dubbed **CoreDataXPath** in [7].

Static analysis of XPath queries, typically inclusion or equivalence checks between queries, can be performed formally through the *validity problem* – or equivalently, for those XPath fragments with negation, through the satisfiability problem. In the data-oblivious case, satisfiability is decidable for **CoreXPath** even in the presence of DTDs [3]. The data-aware version **CoreDataXPath** however turns out to be undecidable [4], which has initiated a quest for decidable fragments and variants [7, 17, 15, 11, 12, 13] – often with prohibitively high complexities. This line of work relies on model-theoretic reasoning, and on the development of ad-hoc models of data automata tailored to capture the fragment at hand.

In this paper, we explore a different avenue, namely the usage of proof systems to analyse XPath queries. Proof systems provide indeed much more concrete *proof search* algorithms for query analysis than the typical 'find non-deterministically a model of size $f(n)$' algorithms that result from the model-theoretic and automata-theoretic approaches. They open the door to a wealth of formal results and practical heuristics and optimisations developed over the years for proof search techniques.

In the case of the data-oblivious **CoreXPath**, there are already several Hilbert-style axiomatisations of fragments [4, 24] and extensions with XPath 2.0 features [25]. By contrast, in this work we do not focus on the navigational aspects of XPath, but rather on understanding how to handle data tests through proof systems. Furthermore, while Hilbert-style axiomatisations provide purely syntactic rules to check the validity of formulæ, decidability and complexity results are rather derived from Gentzen-style sequent calculi or from tableaux systems, and we choose to work with the former.

More precisely, we present a sound and complete cut-free sequent calculus for a fragment of **CoreDataXpath**. For this first attempt at a proof system for a data-aware logic, we work in a somewhat simplified setting:

- our models are finite *data trees* rather than XML trees: these are ordered, unranked trees where each node carries exactly one datum from some infinite data domain $\mathbb{D}$ in addition to a label from some finite alphabet – whereas XML nodes might carry several data values –, and
- we strip **CoreDataXPath**'s navigational capabilities down to a simple *modal data logic* **DataGL** where the usual '$\Box$' modality is refined into two data-aware modalities: $\Box_=$ relates the current position to strict descendants labelled with the same data value, while $\Box_{\neq}$ relates it to strict descendants with a different data value (see Section 2).

  Our logic **DataGL** is a fragment of **CoreDataXPath**($\downarrow^+$), where navigation is restricted to the strict descendant axis $\downarrow^+$ (see [12]). As already noted by ten Cate, Fontaine, and Litak [23], the similarly defined data-oblivious **CoreXPath**($\downarrow^+$) corresponds naturally to the *provability logic* **GL** (named after Gödel and Löb). Although **GL** was originally intended to model provability in arithmetic, it is best understood for our purpose as the modal logic of finite trees: its set of axioms is sound and weakly complete for well-founded transitive frames (e.g. [6], Chapter 4, where the logic is called **KL**). By the same token, **DataGL** can be seen as the data-aware extension of **GL**.

Our calculus, defined and shown sound and complete with respect to finite data trees in Section 3, builds upon an existing sequent calculus for **GL** defined by Avron [2]. We found nonetheless that dealing with $\Box_{\neq}$ modalities brought significant new challenges – both when enforcing well-foundedness and when dealing with the non-transitivity of the associated 'descendant with different data' relation –, which we tackle by introducing so-called *histories* in the calculus.

Among the benefits of our calculus, we exhibit a complete proof search strategy in Section 4, and we show that this strategy works in PSPACE in Section 4.3. This is an improvement over the much more general upper bound shown by Figueira [12, Theorem 6.4] for the EXP-complete **CoreDataXPath**($\downarrow^+$), and matches the PSPACE-hardness of **GL** in the data-oblivious case (e.g. [9, Theorem 7]) – thus in **GL**, data can be added for free! Although there might be simpler ways to prove the PSPACE-completeness of **DataGL**, this shows that proof-theoretic methods do not necessarily come at the expense of algorithmic efficiency.

Due to space constrains, some material is omitted but can be found in the full paper at `https://hal.inria.fr/hal-01191172`.

2 Modal Logic on Finite Data Trees

We introduce in this section **DataGL**, a bimodal logic (see Section 2.2) defined over *finite data trees* (recalled first in Section 2.1). The logic **DataGL** has however a natural, equivalent formulation in terms of finite transitive irreflexive *data Kripke structures*, as shown in Section 2.2.2, allowing to reuse the model-theoretic tool set of modal logic – whereas similar results for **CoreDataXPath** fragments are rather more involved [14].

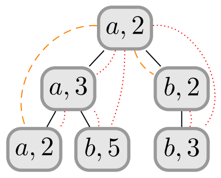


Figure 1 A finite data tree over $\Sigma \overset{\text{def}}{=} \{a, b\}$ and $\mathbb{D} \overset{\text{def}}{=} \mathbb{N}$. The $R_=$ relation is indicated through dashed orange arcs, and the $R_{\neq}$ relation through dotted red arcs.

2.1 Data Trees

A finite (ordered and unranked) *tree* $\mathfrak{t}$ over an alphabet $\mathbb{A}$ is a partial function from *positions* w in $\mathbb{N}^*$ (i.e. finite sequences of non-negative integers) to $\mathbb{A}$, with a finite non-empty domain $\operatorname{dom} \mathfrak{t}$, which is furthermore

prefix-closed: if $wv \in \operatorname{dom} \mathfrak{t}$ for some $w, v \in \mathbb{N}^*$, then $w \in \operatorname{dom} \mathfrak{t}$, and

predecessor-closed: if $w(i+1) \in \operatorname{dom} \mathfrak{t}$ for some $w \in \mathbb{N}^*$ and $i \in \mathbb{N}$, then $wi \in \operatorname{dom} \mathfrak{t}$.

Call the length $|w|$ the *height* of position w. The maximal such height $h(\mathfrak{t}) \overset{\text{def}}{=} \max_{w \in \operatorname{dom} \mathfrak{t}} |w|$ is called the *height* of $\mathfrak{t}$; this is well-defined since $\operatorname{dom} \mathfrak{t}$ is finite. The root of a tree $\mathfrak{t}$ is then denoted by the empty sequence ε, with height 0.

Let Σ be a finite set of tags and $\mathbb{D}$ an infinite countable set of data values. A finite *data tree* is a finite tree over the Cartesian product $\Sigma \times \mathbb{D}$; see Figure 1 for an example. For a position w in $\operatorname{dom} \mathfrak{t}$, we write $\ell(w)$ for its tag in Σ and $d(w)$ for its datum in $\mathbb{D}$; then $\mathfrak{t}(w) = (\ell(w), d(w))$. Given a data tree $\mathfrak{t}$, its *strict descendant* relation $R \overset{\text{def}}{=} \{(w, wv) \in \operatorname{dom} \mathfrak{t} \times \operatorname{dom} \mathfrak{t} \mid v \in \mathbb{N}^+\}$ between its positions can be partitioned into $R = R_= \uplus R_{\neq}$ by defining

$$R_= \overset{\text{def}}{=} \{(w, w') \in R \mid d(w) = d(w')\} , \qquad R_{\neq} \overset{\text{def}}{=} \{(w, w') \in R \mid d(w) \neq d(w')\} . \qquad (1)$$

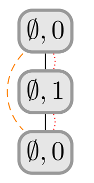
It is worth noting that $R_=$ is transitive, but $R_{\neq}$ is not – as seen for instance on the leftmost branch of the tree in Figure 1 –; however, $w \, R_{\neq} \, w' \, R_= \, w''$ or $w \, R_= \, w' \, R_{\neq} \, w''$ implies $w \, R_{\neq} \, w''$, a fact we dub *cross transitivity* – see for instance the rightmost branch of the tree in Figure 1.

2.2 Modal Data Logic

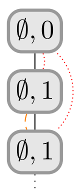
Our modal data logic **DataGL** is syntactically a modal logic with two modal operators, namely $\Box_=$ and $\Box_{\neq}$. Given a countable set A of atomic propositions, its set of formulæ is defined by the abstract syntax

$$\varphi ::= \bot \mid p \mid \varphi \supset \varphi \mid \Box_= \varphi \mid \Box_{\neq} \varphi$$

where p ranges over A. The usual Boolean connectives can be defined by $\neg\varphi \overset{\text{def}}{=} \varphi \supset \bot$, $\top \overset{\text{def}}{=} \neg\bot$, $\varphi \vee \psi \overset{\text{def}}{=} (\neg\varphi) \supset \psi$, $\varphi \wedge \psi \overset{\text{def}}{=} \neg(\neg\varphi \vee \neg\psi)$, and the diamonds by $\Diamond_= \varphi \overset{\text{def}}{=} \neg\Box_= \neg\varphi$ and $\Diamond_{\neq} \varphi \overset{\text{def}}{=} \neg\Box_{\neq}\neg\varphi$; finally the usual box and diamond are defined through $\Box\varphi \overset{\text{def}}{=} \Box_= \varphi \wedge \Box_{\neq} \varphi$ and $\Diamond\varphi \overset{\text{def}}{=} \neg\Box\neg\varphi$. Importantly, $\Diamond_=$ and $\Box_{\neq}$ are *not* dual, nor are $\Diamond_{\neq}$ and $\Box_=$.



(a) A counter-model to (2).



(b) An infinite counter-model to (3).

Figure 2 Counter-models to formulæ (2) and (3), where $\varphi \overset{\text{def}}{=} p$, $\Sigma \overset{\text{def}}{=} \{\emptyset, \{p\}\}$, and $\mathbb{D} \overset{\text{def}}{=} \mathbb{N}$.

2.2.1 Semantics

Given a finite data tree $\mathfrak{t}$ and a position $w \in \operatorname{dom} \mathfrak{t}$, we inductively define a **DataGL** formula φ to be *satisfied* in $\mathfrak{t}$ at w, denoted $\mathfrak{t}, w \models \varphi$, as usual:

$$\begin{aligned}
\mathfrak{t}, w &\models \bot && \text{never} ,\\
\mathfrak{t}, w &\models p && \text{iff } p \in \ell(w) ,\\
\mathfrak{t}, w &\models \varphi \supset \psi && \text{iff } \mathfrak{t}, w \models \varphi \text{ implies } \mathfrak{t}, w \models \psi ,\\
\mathfrak{t}, w &\models \Box_= \varphi && \text{iff } \forall w' \,.\, w \mathrel{R_=} w' \text{ implies } \mathfrak{t}, w' \models \varphi ,\\
\mathfrak{t}, w &\models \Box_{\neq} \varphi && \text{iff } \forall w' \,.\, w \mathrel{R_{\neq}} w' \text{ implies } \mathfrak{t}, w' \models \varphi .
\end{aligned}$$

A formula φ is *satisfiable* if there exists a finite data tree $\mathfrak{t}$ and a position w in $\operatorname{dom} \mathfrak{t}$ such that $\mathfrak{t}, w \models \varphi$. It is *valid* if for all finite data trees $\mathfrak{t}$ and positions w in $\operatorname{dom} \mathfrak{t}$, $\mathfrak{t}, w \models \varphi$; observe that φ is valid if and only if $\neg\varphi$ is not satisfiable, and that we can assume $w = \varepsilon$ without loss of generality by extracting the subtree of $\mathfrak{t}$ rooted by w. Note that for validity or satisfiability questions, we can assume A to be the finite set of atomic propositions appearing in φ, and work with the tag set $\Sigma \overset{\text{def}}{=} 2^A$.

▶ **Example 1** (Löb's Axiom). Consider the following formula, known as *Löb's axiom*:

$$\Box(\Box\varphi \supset \varphi) \supset \Box\varphi , \tag{L}$$

which can be viewed as an induction scheme over the depth of a node: to prove that φ holds at any node, it suffices to establish that it holds at every node assuming that it holds at deeper nodes. Since we are working on finite data trees, **L** is valid: for any finite data tree $\mathfrak{t}$ and any position w in $\operatorname{dom} \mathfrak{t}$, we can show that $\mathfrak{t}, w \models \mathbf{L}$. Indeed, if we assume $\mathfrak{t}, w \models \Box(\Box\varphi \supset \varphi)$, then by induction over $h(\mathfrak{t}) - |w'|$, if $w \mathrel{R} w'$ then $\mathfrak{t}, w' \models \varphi$: this holds for any leaf w', since then $\mathfrak{t}, w' \models \Box\varphi$ vacuously, and thus $\mathfrak{t}, w' \models \varphi$; and for an inner node w', by transitivity of R all its strict descendants are also strict descendants of w, thus by induction hypothesis they satisfy φ, hence $\mathfrak{t}, w' \models \Box\varphi$ and therefore $\mathfrak{t}, w' \models \varphi$ as desired.

▶ **Example 2.** Due to the non-transitivity of $R_{\neq}$, the following variant of **L** is not valid:

$$\Box_{\neq}(\Box_{\neq}\varphi \supset \varphi) \supset \Box_{\neq}\varphi . \tag{2}$$

A counter-model is depicted in Figure 2a in the case $\varphi \overset{\text{def}}{=} p$. Observe that $\mathfrak{t}, \varepsilon \not\models \Box_{\neq} p$ due to the middle node. Furthermore, $\mathfrak{t}, \varepsilon \models \Box_{\neq}(\Box_{\neq} p \supset p)$: the only node related to the root through $R_{\neq}$ is the middle node, and it satisfies $\Box_{\neq} p \supset p$ because it does not satisfy $\Box_{\neq} p$: indeed, the bottom node does not satisfy p.

▶ **Example 3.** The following, more involved formula is also valid over finite data trees, and further illustrates the interaction between $\Box_=$ and $\Box_{\neq}$:

$$\Box_{\neq}(\varphi \vee \Diamond_= \top) \supset \Diamond_{\neq}\varphi \vee \Box_{\neq}\bot \ . \tag{3}$$

Its validity relies on the finiteness assumption. Indeed, assume for the sake of contradiction that $\mathfrak{t}, w \models \Box_{\neq}(\varphi \vee \Diamond_= \top)$ but $\mathfrak{t}, w \not\models \Diamond_{\neq}\varphi$ and $\mathfrak{t}, w \not\models \Box_{\neq}\bot$, for some finite data tree $\mathfrak{t}$ and position w. Then there exists some w_0 in $\operatorname{dom}\mathfrak{t}$ with $w\, R_{\neq}\, w_0$, and $\mathfrak{t}, w_0 \not\models \varphi$. Necessarily, $\mathfrak{t}, w_0 \models \Diamond_= \top$: there exists w_1 in $\operatorname{dom}\mathfrak{t}$ such that $w_0\, R_=\, w_1$. By cross transitivity, $w\, R_{\neq}\, w_1$, and we can apply the same reasoning to w_1: there exists w_2 in $\operatorname{dom}\mathfrak{t}$ such that $w_1\, R_=\, w_2$, thus by transitivity of $R_=$ and cross transitivity, $w\, R_{\neq}\, w_2$, and so on and so forth. Hence there exists an infinite chain $w_0\, R_=\, w_1\, R_= \cdots$ of positions in $\mathfrak{t}$, which contradicts its finiteness.

Note that the previous argument describes a counter-model to (3) if we were to allow infinite data trees as models instead of only finite ones; see a counter-model in Figure 2b.

2.2.2 Data Kripke Structures

The data tree semantics of **DataGL** is actually a particular case of its semantics in terms of *data Kripke structures*. Such a structure is a tuple $\mathfrak{M} = (W, R, d, \ell)$ where W is a set of worlds, R is a binary relation over W, $d\colon W \to \mathbb{D}$ is a data labelling, and $\ell\colon W \to 2^A$ is a labelling using atomic propositions. Observe that a data tree $\mathfrak{t}$ defines such a structure with $W \overset{\text{def}}{=} \operatorname{dom}\mathfrak{t}$.

Given a data Kripke structure $\mathfrak{M}$, the relation R can be partitioned as $R = R_= \uplus R_{\neq}$ as in Equation (1), allowing to define the satisfaction relation $\mathfrak{M}, w \models \varphi$ exactly as in the case of data trees. Thus, a data Kripke structure can be seen as a Kripke structure with two relations $R_=$ and $R_{\neq}$. Working with this modal similarity type $\{\Diamond_=, \Diamond_{\neq}\}$ allows to readily apply the basic model-theoretic constructions of modal logic: satisfaction is invariant under e.g. *generated submodels* [6, Definition 2.5 and Proposition 2.6], *bounded morphisms* [6, Definition 2.12 and Proposition 2.14], and *bisimulations* [6, Definition 2.18 and Theorem 2.20].

We call a data Kripke structure a **DataGL** *model* if R is transitive irreflexive and W is finite. Given a root world w_0, the *height* of a world w is the length n of the maximal chain $w_0\, R\, w_1\, R \cdots R\, w_n = w$ from w_0 to w, and the *height* of $\mathfrak{M}$ the maximal height of its worlds. The semantics in terms of **DataGL** models is equivalent to that in terms of finite data trees: this is essentially due to the *tree-model property* of modal logic (see e.g. [6, Proposition 2.15], and proven via unfolding (see in the full paper for details and a reminder on bounded morphisms):

▶ **Proposition 4** (Tree-Model Property). *For any* **DataGL** *model $\mathfrak{M}$ and world w_0, there exists a finite data tree $\mathfrak{t}_{\mathfrak{M}}$ of the same height and a surjective bounded morphism f from $\mathfrak{t}_{\mathfrak{M}}$ to $\mathfrak{M}$ with $f(\varepsilon) = w_0$. Hence, for all* **DataGL** *formulæ φ and positions w in $\operatorname{dom}\mathfrak{t}_{\mathfrak{M}}$, $\mathfrak{t}_{\mathfrak{M}}, w \models \varphi$ if and only if $\mathfrak{M}, f(w) \models \varphi$.*

Since a finite data tree is a particular case of a **DataGL** model, Proposition 4 means that finite data trees and **DataGL** models can be used interchangeably.

2.2.3 Expressiveness

The logic **DataGL** is a strict fragment of **XPath**: when naming `@d` the unique data attribute of data trees, in concrete **XPath** syntax, $\Diamond_{\neq}\varphi$ could for instance be expressed as the node test `[./@d != ./descendant::*[`χ`]/@d]` assuming χ is the **XPath** translation of φ. More precisely, we only need the union-free fragment of **CoreDataXPath**$^{\varepsilon}(\downarrow^+)$ as defined by Figueira [12], whose satisfiability problem is EXP-complete. See in the full paper for a comparison between **DataGL** and **XPath**.

DataGL is also a strict fragment of $\mathbf{FO}^2(\sim, <)$ over data trees [7], which is the two-variables fragment of first-order logic with atomic predicates '$x \sim y$' to express that the positions at x and y have the same data values, and '$x < y$' to express that the position at y is a strict descendant of that at x. The standard translation of modal logics into first-order logics [6, Section 2.4] readily provides an equivalent $\mathbf{FO}^2(\sim, <)$ formula $\mathrm{ST}_x(\varphi)$ for any **DataGL** formula φ. To the best of our knowledge, whether satisfiability for $\mathbf{FO}^2(\sim, <)$ is decidable is open [7, 18].

3 Sequent Calculus

Developing sequent calculi for modal logics is often arduous. In order to obtain modular proof systems enjoying both cut elimination and a form of subformula property – which are desirable in order to show decidability –, advanced techniques are typically required: display logic [26], labelled calculi [20], nested sequents [8], or tree-hypersequents [21] to name a few. All these are motivated by the need to maintain additional information throughout proofs, using extra-logical means. As we are going to see, we also introduce a form of enriched sequents for **DataGL**, in the form of *histories* – which turn out to be similar to the ordered hypersequents recently employed by Indrzejczak [16] for linear-time modal logics. We define our calculus in Section 3.2, and prove it sound and complete in Sections 3.3 and 3.4. But first we present our main source of inspiration: Avron's sequent calculus for **GL** [2].

3.1 Avron's Sequent Calculus for GL

Our sequent calculus for **DataGL** is inspired by the work of Avron [2] who gave a sound and complete sequent calculus for **GL**, which does not require any extra-logical apparatus. In addition to the usual rules for Boolean connectives, Avron proposed the following sequent calculus rule ($\Box$) to deal with modalities in **GL** (the principal formula is coloured in orange):

$$\frac{\Box\Gamma, \Gamma, \Box\varphi \vdash \varphi}{\Box\Gamma \vdash \Box\varphi}\ \Box$$

As usual in classical sequent calculus, a sequent $\Gamma \vdash \Delta$ is made of two sets of formulæ: Γ (the *antecedents*) and Δ (the *consequents*) and should be interpreted as $\bigwedge_{\varphi\in\Gamma}\varphi \supset \bigvee_{\psi\in\Delta}\psi$. We simply use commas to denote set union. The notation $\Box\Gamma$ stands for the set of all $\Box\varphi$ formulæ for $\varphi \in \Gamma$, and later $\Box_=\Gamma$ will denote the set $\{\Box_=\varphi \mid \varphi \in \Gamma\}$, and similarly for $\Box_{\neq}$. A rule consists of a set of *premises* (the top sequents) along with a single *conclusion* (the bottom sequent). A sequent S is *derivable* from a set of sequents X if there exists a derivation with S as root and X as set of open leaves; a sequent is *provable* if it is derivable from the empty set, or equivalently if it has a *proof*, i.e. a derivation with no open leaves.

When Γ is empty, Avron's rule follows immediately from the **L** axiom: if we can prove $\Box\varphi \supset \varphi$, then we also have $\Box(\Box\varphi \supset \varphi)$ by necessitation, and $\Box\varphi$ follows by **L**. When Γ is not empty, the rule exploits the properties of $\Box$ in order to extract information from the antecedents in the conclusion sequent. Reading the rule bottom-up, the idea is that since we are assuming $\Box\Gamma$, then surely we can inductively assume Γ for any strict descendant where φ holds, but also $\Box\Gamma$ since our relation R is transitive.

▶ **Example 5** (Proof of **L** in Avron's Calculus)**.** The **L** axiom can be proven valid using ($\Box$) and the classical rules (see Figure 3 for the variants we will use later; we colour again principal formulæ in orange):

$$\dfrac{\dfrac{\dfrac{\dfrac{}{\Box(\Box\varphi\supset\varphi),\Box\varphi\vdash\Box\varphi,\varphi}\,\mathrm{ax}\qquad\dfrac{}{\Box(\Box\varphi\supset\varphi),\Box\varphi,\varphi\vdash\varphi}\,\mathrm{ax}}{\Box(\Box\varphi\supset\varphi),\Box\varphi,\Box\varphi\supset\varphi\vdash\varphi}\,\supset_L}{\Box(\Box\varphi\supset\varphi)\vdash\Box\varphi}\,\Box}{\vdash\Box(\Box\varphi\supset\varphi)\supset\Box\varphi}\,\supset_R$$

Note that the left branch of the proof could not be closed without the addition of $\Box\varphi$ among the antecedents of the premise of $(\Box)$.

The idea behind the rule $(\Box)$ applies immediately to our $\Box_=$ modality since $R_=$ is also transitive and well-founded. However, the $\Box_{\neq}$ modality cannot be handled in the same way because $R_{\neq}$ is not transitive. A sound rule for $\Box_{\neq}$ would be:

$$\dfrac{\Box_=\Gamma^{\neq},\Gamma^{\neq}\vdash\varphi}{\Box_=\Gamma^{=},\Box_{\neq}\Gamma^{\neq}\vdash\Box_{\neq}\varphi}$$

There is however no hope for this rule to yield a complete calculus. First, it fails to use in a significant way the assumptions $\Box_=\Gamma^=$. When we consider a strict descendant in the premise, we have forgotten about the position from which we came, and so we will never be able to tell when we reach a further descendant with the same datum, whether $\Gamma^=$ should hold. Second, it does not enforce tree finiteness. In that respect, note that adding an assumption $\Box_{\neq}\varphi$ to the antecedents of the top sequent (naively mimicking the $(\Box)$ rule) would yield an unsound rule. Both issues are solved in the following by enriching the structure of our sequents.

3.2 Sequent Calculus for DataGL

Our calculus employs sequents enriched with a *history*: those are sequences $\mathcal{H}=H_1;\ldots;H_n$ of sets of modal formulæ $H_i=\Box_=H_i^=,\Box_{\neq}H_i^{\neq}$ called its *cells*. The length of a history $\mathcal{H}$ is denoted by $|\mathcal{H}|$. Given a history $\mathcal{H}$ and $1\leq i\leq|\mathcal{H}|$, we always write H_i for its ith cell and $H_i^=$, $H_i^{\neq}$ for the sets such that $H_i=\Box_=H_i^=,\Box_{\neq}H_i^{\neq}$. If $\mathcal{H}$ is a history and H_i one of its cells, we write $\mathcal{H}\setminus H_i$ for the history obtained by removing H_i, i.e. for $H_1;\ldots;H_{i-1};H_{i+1};\ldots;H_{|\mathcal{H}|}$. The sequent calculus for **DataGL** deals with *sequents* of the form $\mathcal{H};\Gamma\vdash\Delta$ where Γ and Δ are sets of formulæ and $\mathcal{H}$ is a history. Its rules are given in Figures 3 and 4. The calculus enjoys several desirable properties, namely that it does not contain a 'cut' rule, and that it has the *subformula property*: in any rule, the formulæ found in the premises are subformulæ of those found in the conclusion.

3.2.1 Boolean Formulæ

The rules of Figure 3 for the Boolean connectives are not surprising: the history plays no role in these rules, and when we ignore it we recover the usual sequent calculus rules for propositional classical logic. These rules are formulated in a way that avoids explicitly considering the structural rules of contraction and weakening. Note that in rules $(\supset_L)$ and $(\supset_R)$ we have chosen to keep the principal formula $\varphi\supset\psi$ in the premises of the rule (in grey in Figure 3) – this is an inessential choice that simplifies the completeness argument later, in particular Lemma 11. In fact, weakening is admissible in the calculus (meaning that adding the weakening rules does not change the set of provable sequents; see in the full paper), which means that the rules without the greyed formulæ are also admissible:

▶ **Lemma 6** (Admissibility of Weakening). *The following rules are admissible:*

$$\dfrac{\mathcal{H};\Gamma\vdash\Delta}{\mathcal{H};\Gamma,\varphi\vdash\Delta}\,W_L\qquad\qquad\dfrac{\mathcal{H};\Gamma\vdash\Delta}{\mathcal{H};\Gamma\vdash\Delta,\varphi}\,W_R$$

$$\frac{}{\mathcal{H};\Gamma,\bot\vdash\Delta}\ \bot_L \qquad\qquad \frac{}{\mathcal{H};\Gamma,\varphi\vdash\varphi,\Delta}\ \mathrm{ax}$$

$$\frac{\mathcal{H};\Gamma,\varphi\supset\psi\vdash\varphi,\Delta \quad \mathcal{H};\Gamma,\varphi\supset\psi,\psi\vdash\Delta}{\mathcal{H};\Gamma,\varphi\supset\psi\vdash\Delta}\ \supset_L \qquad \frac{\mathcal{H};\Gamma,\varphi\vdash\varphi\supset\psi,\psi,\Delta}{\mathcal{H};\Gamma\vdash\varphi\supset\psi,\Delta}\ \supset_R$$

Figure 3 Sequent calculus for **DataGL**: classical sequent calculus rules. The principal formulæ are coloured in orange in the conclusions of the rules. The greyed formulæ $\varphi\supset\psi$ in the premises of $(\supset_L)$ and $(\supset_R)$ can be omitted; we do not use them in examples to reduce clutter.

$$\frac{\mathcal{H};\Gamma^{=},\Box_{=}\Gamma^{=},\Box_{\neq}\Gamma^{\neq},\{H_i^{\neq}\}_{1\leq i\leq|\mathcal{H}|},\Box_{=}\varphi\vdash\varphi}{\mathcal{H};\Gamma,\Box_{=}\Gamma^{=},\Box_{\neq}\Gamma^{\neq}\vdash\Box_{=}\varphi,\Delta}\ \Box_{=}$$

$$\frac{\begin{array}{c}\mathcal{H};(\Box_{=}\Gamma^{=},\Box_{\neq}\Gamma^{\neq},\Box_{\neq}\varphi);(\Gamma^{\neq},\{H_i^{\neq}\}_{1\leq i\leq|\mathcal{H}|})\vdash\varphi\\ \{\ \mathcal{H}\setminus H_j;(\Box_{=}\Gamma^{=},\Box_{\neq}\Gamma^{\neq},\Box_{\neq}\varphi);(\Gamma^{\neq},\{H_i^{\neq}\}_{1\leq i\leq|\mathcal{H}|,i\neq j},H_j^{=},H_j)\vdash\varphi\ \}_{1\leq j\leq|\mathcal{H}|}\end{array}}{\mathcal{H};\Gamma,\Box_{=}\Gamma^{=},\Box_{\neq}\Gamma^{\neq}\vdash\Box_{\neq}\varphi,\Delta}\ \Box_{\neq}$$

Figure 4 Sequent calculus for **DataGL**: modal rules. Rule $(\Box_{\neq})$ has $|\mathcal{H}|+1$ premises. The principal formulæ are coloured in orange in the conclusions of the rules. In both rules, Γ cannot contain any formula that is modal, i.e. of the form $\Box_\star\psi$ for $\star\in\{=,\neq\}$

3.2.2 Modal Formulæ

Before defining formally the semantics of our sequents, let us first provide an intuition for the complex rules of Figure 4 by presenting them informally from a proof search viewpoint.

Applying a modal rule bottom-up amounts to proving a sequent by considering all the possible descendants of a current (hypothetical) position in a data tree. In Avron's calculus all the important information about the current position (i.e. the modal antecedents of the conclusion sequent) could be transferred to the descendant as modal antecedents of the premise. In our case, this transfer is performed through the history. Intuitively, the history keeps track of which previous (hypothetical) positions have been visited, and of which modal formulæ were known to hold at these positions. For the same reason that Avron did not need a history, we do not need to remember past positions labelled with the same data as the current position. In fact, we only need to consider past positions labelled with mutually distinct data values: one value for the current position and one for each history cell – these values do not actually show up in the calculus, because its purpose is to establish the validity of a sequent, i.e. it simultaneously considers all possible data assignments.

The $(\Box_{=})$ rule is similar to Avron's rule, but also extracts information from the history: when we move to a strict descendant position with the same data value, it remains different from the data values of the past positions associated to history cells, and thus we know that all the $H_i^{\neq}$ formulæ hold at the new position. (Also note that this rule allows to weaken propositional formulæ, namely the parts Γ and Δ of the conclusion sequent; this is, again, to avoid considering explicit structural rules, and to facilitate proof search in our calculus.)

The $(\Box_{\neq})$ rule is the most complex one, as it not only extracts information from the history but also updates it. When moving to a strict descendant with a different data, the new data may or may not be different from the data of the previously visited positions in the history, leading to $|\mathcal{H}|+1$ premises:

- The first premise of the $(\Box_{\neq})$ rule covers the case of a totally fresh data value. In that case, we know that all the $H_i^{\neq}$ formulæ hold at the new position. We also update the history with a new cell corresponding to the position that we just left. Unsurprisingly, this cell contains the modal formulæ that were assumed about that position. It also contains $\Box_{\neq}\varphi$ as a way of mimicking well-founded inductive reasoning.
- Each of the remaining premises corresponds to the case where the new position has the same data as the position corresponding to history cell H_j. In such a case, the formulæ in $H_j^{\neq}$ are not known to hold at the new position. Instead, $H_j^{=}$ holds, as well as H_j itself, and thus there is no point in keeping a history cell for the past occurrence of the new data. As in the previous case, the history is updated with a new cell corresponding to the position we just left.

▶ **Example 7.** Let us consider again the invalid formula (2) from Example 2. Since our calculus is sound (see Theorem 9), proof search ought to fail for this formula. The first steps up to the first application of the $(\Box_{\neq})$ rule are:

$$\dfrac{\dfrac{\Box_{\neq}(\Box_{\neq}\varphi \supset \varphi), \Box_{\neq}\varphi; \Box_{\neq}\varphi \supset \varphi \vdash \varphi}{;\Box_{\neq}(\Box_{\neq}\varphi \supset \varphi) \vdash \Box_{\neq}\varphi}\,\Box_{\neq}}{;\vdash \Box_{\neq}(\Box_{\neq}\varphi \supset \varphi) \supset \Box_{\neq}\varphi}\,{\supset_R}$$

This creates a first history cell $H_1 \overset{\text{def}}{=} \Box_{\neq}(\Box_{\neq}\varphi \supset \varphi), \Box_{\neq}\varphi$ combining the modal formulæ of the antecedent and the principal formula. Soon after, proof search fails:

$$\dfrac{\dfrac{\dfrac{}{H_1; \Box_{\neq}\varphi; \Box_{\neq}\varphi \supset \varphi, \varphi \vdash \varphi}\,\mathrm{ax} \qquad \begin{array}{c}\text{no applicable rule}\\ \Box_{\neq}\varphi; \Box_{\neq}(\Box_{\neq}\varphi \supset \varphi), \Box_{\neq}\varphi \vdash \varphi\end{array}}{H_1; \vdash \Box_{\neq}\varphi, \varphi}\,\Box_{\neq} \qquad \dfrac{}{H_1; \varphi \vdash \varphi}\,\mathrm{ax}}{H_1; \Box_{\neq}\varphi \supset \varphi \vdash \varphi}\,{\supset_L}$$

This second application of $(\Box_{\neq})$ to $H_1; \vdash \Box_{\neq}\varphi, \varphi$, creates a new history cell $H_2 \overset{\text{def}}{=} \Box_{\neq}\varphi$, and has two premises: the first, which assumes a fresh data, copies the formulæ under $\Box_{\neq}$ from H_1 into the antecedent; the second assumes we have encountered the same data value as in the position remembered through H_1, thus copies the formulæ under $\Box_{=}$ in H_1 (there are none) into the antecedent, but also extracts H_1 itself from the history and puts it in the antecedent. Note that this search was deterministic: there were never any alternative applicable rule, hence formula (2) is unprovable.

3.3 Soundness

We now formally define the semantics of our sequents, and establish that the calculus is sound. An *annotated sequent* $(\mathcal{H}; \Gamma \vdash \Delta)^d$ is a sequent $\mathcal{H}; \Gamma \vdash \Delta$ together with a *data assignment* d, which is an injective function from $\{0, \ldots, |\mathcal{H}|\}$ to $\mathbb{D}$. We write $d_0, \ldots, d_{|\mathcal{H}|}$ for those distinct data values: the data value d_0 is understood as being associated to the bare sequent $\Gamma \vdash \Delta$, while each d_i for $0 < i \leq |\mathcal{H}|$ is associated to the cell H_i.

▶ **Definition 8** (Sequent Satisfiability and Validity). A finite data tree $\mathfrak{t}$ *satisfies* an annotated sequent $(\mathcal{H}; \Gamma \vdash \Delta)^d$ at a position w in $\operatorname{dom}\mathfrak{t}$ if and only if the following conditions together imply $\mathfrak{t}, w \models \varphi$ for some $\varphi \in \Delta$:

(a) $\mathfrak{t}, w \models \varphi$ for all $\varphi \in \Gamma$,
(b) $d(w) = d_0$,
(c) $\mathfrak{t}, w' \models \varphi$ for all $1 \leq i \leq |\mathcal{H}|$, all $\Box_{=}\varphi \in H_i^{=}$, and all w' with $w \mathrel{R} w'$ and $d(w') = d_i$, and
(d) $\mathfrak{t}, w' \models \varphi$ for all $1 \leq i \leq |\mathcal{H}|$, all $\Box_{\neq}\varphi \in H_i^{\neq}$, and all w' with $w \mathrel{R} w'$ and $d(w') \neq d_i$.

An annotated sequent is *valid* if and only if it is satisfied by all finite data trees at all positions. A finite data tree $\mathfrak{t}$ *satisfies* a sequent S at a position w, written $\mathfrak{t}, w \models S$, if it satisfies some annotation of S at w. A sequent is *valid* if all its annotations are valid.

Note that the validity of one annotation of a sequent is equivalent to the validity of all of its annotations, because any two annotations are related by a bijective renaming of data values, and satisfaction of a formula is invariant under such renamings. We prove soundness in in the full paper by checking that the rules preserve validity:

▶ **Theorem 9** (Soundness). *The sequent calculus for* **DataGL** *is sound, i.e. all provable sequents are valid.*

3.4 Completeness

We now turn to proving that our sequent calculus is complete: φ is valid in **DataGL** if and only if $\vdash \varphi$ is provable. We show more generally that our calculus is complete with respect to sequent validity:

▶ **Theorem 10** (Completeness). *The sequent calculus for* **DataGL** *is complete, i.e. all valid sequents are provable.*

This result is obtained by constructing for any invalid sequent an appropriate *canonical* (counter-)model, which is a **DataGL** model. More specifically, we follow Avron [2] in building a model based on unprovable saturated sequents, rather than a model based on saturated sets of formulæ as in the Hilbert-style approach [6]. This is not a minor difference: it allows us to obtain a canonical model that enjoys irreflexivity and well-foundedness, two properties that are not directly obtained in the Hilbert-style approach.

Let us call a sequent $\mathcal{H}; \Gamma \vdash \Delta$ *saturated* if $\varphi \supset \psi \in \Gamma$ implies $\varphi \in \Delta$ or $\psi \in \Gamma$, and $\varphi \supset \psi \in \Delta$ implies $\varphi \in \Gamma$ and $\psi \in \Delta$. We can restrict ourselves to work with saturated sequents without loss of generality:

▶ **Lemma 11** (Saturation Lemma). *For any unprovable sequent $S = \mathcal{H}; \Gamma \vdash \Delta$ there exists an unprovable sequent $S' = \mathcal{H}; \Gamma' \vdash \Delta'$ using only subformulæ of S, which is saturated and such that $\Gamma \subseteq \Gamma'$, $\Delta \subseteq \Delta'$. Furthermore if $\mathfrak{t}, w \models S$ then $\mathfrak{t}, w \models S'$.*

Proof sketch. The total size of all formulæ for which the saturation condition fails can be decreased by repeatedly applying the rules of Figure 3 bottom-up. This process yields a set of saturated sequents X, and a simple inspection of the rules shows that for all the sequents S' in X the formulæ initially present in Γ (resp. Δ) are still present, that S' only contains subformulæ of S, and that $\mathfrak{t}, w \models S$ implies $\mathfrak{t}, w \models S'$. Since S was unprovable, X contains at least one unprovable sequent. ◀

We shall build our canonical model based on annotated saturated sequents. In order to obtain a finite model, we restrict our sequents to only contain (sub)formulæ among a finite set, and we forbid duplicates in histories. This, in turn, allows us to bound the number of possible data assignments. In the remainder of this section, let us fix a finite set of formulæ $\mathcal{F}$ that is closed under taking subformulæ. Since we want to exhibit a counter-model, we can indeed work here with $\mathbb{D} \overset{\text{def}}{=} \mathbb{N}$ without loss of generality.

▶ **Definition 12** (Canonical Sequents). A *canonical sequent* $(\mathcal{H}; \Gamma \vdash \Delta)^d$ over $\mathcal{F}$ is unprovable saturated annotated sequent such that its formulæ belong to $\mathcal{F}$, $H_i \neq H_j$ for $1 \leq i \neq j \leq |\mathcal{H}|$, and $0 \leq d_i \leq 2^{|\mathcal{F}|}$ for every $0 \leq i \leq |\mathcal{H}|$.

Given a sequent $S = \mathcal{H}; \Gamma \vdash \Delta$, we write $S_i \overset{\text{def}}{=} H_i$ for $1 \leq i \leq |\mathcal{H}|$ for its history cells and $S_0 \overset{\text{def}}{=} \{\Box_\star\varphi \mid \Box_\star\varphi \in \Gamma, \star \in \{=, \neq\}\}$ for the set of modal formulæ found in Γ. We shall refer to the sets $(S_i)_{i\geq 0}$ as the *cells* of S.

▶ **Definition 13** (Canonical Relation). Given two sequents $S = \mathcal{H}; \Gamma \vdash \Delta$ and $S' = \mathcal{H}'; \Gamma' \vdash \Delta'$, S *embeds modally* into S', written $S \sqsubseteq S'$, if there exists an injective function $f\colon \{0, \ldots, |\mathcal{H}|\} \to \{0, \ldots, |\mathcal{H}'|\}$ such that $S_i \subseteq S'_{f(i)}$ for all $0 \leq i \leq |\mathcal{H}|$.

Given annotations d and d' for S and S', we define $S\ R^{\mathfrak{c}}\ S'$ to hold whenever $S \sqsubseteq S'$ and

(i) for all $0 \leq i \leq |\mathcal{H}|$, $d_i = d'_{f(i)}$,
(ii) $\varphi \in \Gamma'$ whenever $\Box_=\varphi \in S_i$ with $f(i) = 0$,
(iii) $\psi \in \Gamma'$ whenever $\Box_{\neq}\psi \in S_i$ with $f(i) \neq 0$, and
(iv) there exists some formula $\Box_\star\varphi \in (\Delta \setminus \Gamma) \cap S'_{f(0)}$ for $\star \in \{=, \neq\}$; that formula is called the *witness* associated to $S\ R^{\mathfrak{c}}\ S'$.

▶ **Lemma 14** (Transitivity and Irreflexivity). *The relation $R^{\mathfrak{c}}$ is transitive and irreflexive.*

Proof. Transitivity is obvious for $\sqsubseteq$ and conditions (i)–(iii). For condition (iv), it comes from the fact that witnesses are propagated by $\sqsubseteq$. Indeed, if $S\ R^{\mathfrak{c}}\ S'\ R^{\mathfrak{c}}\ S''$, then the witness $\Box_\star\varphi$ for $S\ R^{\mathfrak{c}}\ S'$ belongs to $\Delta \setminus \Gamma$ and to $S'_{f(i)}$ such that $d'_i = d_0$. Thus by condition (i) it also belongs to S''_j for $d''_j = d'_i = d_0$. Regarding irreflexivity, if $S\ R^{\mathfrak{c}}\ S$ then the witness would have to belong to $(\Delta \setminus \Gamma) \cap S'_0$ since $d_0 = d'_0$ (and all other data values are distinct from d_0), but that set is empty since $S'_0 \subseteq \Gamma$. ◀

Given a finite $\mathcal{F}$, observe that the set $C(\mathcal{F})$ of canonical sequents over $\mathcal{F}$ is also finite.

▶ **Definition 15** (Canonical Structure). The *canonical structure* over $\mathcal{F}$ is the data Kripke structure $\mathfrak{C}(\mathcal{F}) \overset{\text{def}}{=} (C(\mathcal{F}), R^{\mathfrak{c}}, d^{\mathfrak{c}}, \ell^{\mathfrak{c}})$ over canonical sequents. The data label of a sequent $S = (\mathcal{H}; \Gamma \vdash \Delta)^d$ is $d^{\mathfrak{c}}(S) \overset{\text{def}}{=} d_0$ and its propositional label is the set of atomic propositions found in Γ, i.e. $\ell^{\mathfrak{c}}(S) \overset{\text{def}}{=} \{p \in A \mid p \in \Gamma\}$.

By Lemma 14 and the finiteness of $\mathfrak{C}(\mathcal{F})$, it is a **DataGL** model. Hence Proposition 4 can be invoked to show the existence of a finite data tree satisfying the same sequents. The following lemma shows that $\mathfrak{C}(\mathcal{F})$ provides counter-models to validity in the sense of Definition 8 (see in the full paper for a proof):

▶ **Lemma 16** (Falsification Lemma). *For any canonical sequent $S = (\mathcal{H}; \Gamma \vdash \Delta)^d$ we have:*

- $\mathfrak{C}(\mathcal{F}), S \models \varphi$ *for all* $\varphi \in \Gamma$*;*
- $\mathfrak{C}(\mathcal{F}), S' \models \varphi$ *for all* $1 \leq i \leq |\mathcal{H}|$*, all* $\Box_=\varphi \in H_i$*, and all* S' *with* $S\ R^{\mathfrak{c}}\ S'$ *and* $d^{\mathfrak{c}}(S') = d_i$*;*
- $\mathfrak{C}(\mathcal{F}), S' \models \varphi$ *for all* $1 \leq i \leq |\mathcal{H}|$*, all* $\Box_{\neq}\varphi \in H_i$*, and all* S' *with* $S\ R^{\mathfrak{c}}\ S'$ *and* $d^{\mathfrak{c}}(S') \neq d_i$*;*
- $\mathfrak{C}(\mathcal{F}), S \not\models \varphi$ *for all* $\varphi \in \Delta$*.*

We can finally establish our result:

Proof of Theorem 10. Consider a sequent $\mathcal{H}; \Gamma \vdash \Delta$ that is unprovable. We can assume without loss of generality that it has no duplicate cell, as we can always add dummy formulæ to differentiate between identical cells, without making the sequent provable. Define $\mathcal{F}$ as its set of subformulæ. By Lemma 11 we obtain an unprovable saturated sequent $\mathcal{H}; \Gamma, \Gamma' \vdash \Delta, \Delta'$. This sequent can be annotated to obtain a canonical sequent, and by Lemma 16 we have a counter-model of that sequent, which is also a counter-model of $\mathcal{H}; \Gamma \vdash \Delta$. ◀

4 Proof Search

In this section we analyse further the structure of our proof system, deriving properties that are useful for proof search. We first discuss a straightforward complete proof search strategy in Section 4.1. In spite of its simplicity, it yields a polynomial bound on the depth of proofs (see Section 4.2) and therefore a PSPACE upper bound on **DataGL** validity, which is optimal (see Section 4.3).

4.1 Proof Search Strategy

Proof search can be understood intuitively as a game between two players Prover and Spoiler with sequents as positions. Given a sequent S, Prover first chooses an applicable rule, i.e. a rule whose conclusion matches S. Spoiler then chooses the new current sequent among the premises of the rule application. Any player with no possible move loses: Prover if no rule is applicable, and Spoiler if there are no premises, as with rules (ax) and ($\bot_L$); furthermore Spoiler wins if the play is infinite. A sequent is valid if and only if Prover has a winning strategy in this game.

When several rules are applicable, which one should Prover select?

- Clearly, if either (ax) or ($\bot_L$) is applicable, then she should pick it since she wins immediately.
- Furthermore, the rules ($\supset_L$) and ($\supset_R$) are *invertible*, i.e. their premises are provable if and only if their conclusion is provable: one direction is immediate since the premises allow to derive the conclusion, and conversely we can appeal to weakening (recall Lemma 6) to show the provability of the premises from that of the conclusion. An obvious complete proof search strategy is then to apply ($\supset_L$) and ($\supset_R$) eagerly, decomposing any Boolean formula that is not yet decomposed.
- After this phase, the only hope to prove a sequent is to use a modal rule: Prover has to select one principal modal formula on the right of the sequent and apply the corresponding ($\Box_=$) or ($\Box_{\neq}$) rule.

Although the obtained strategy only makes essential choices, it may *a priori* diverge: each application of a modal rule imports new Boolean formulas into the antecedent, which may yield new modal rules, and so on and so forth. It turns out, however, that the 'GL component' of our modal rules allows us to derive a small bound on the length of branches that should be considered in proof search attempts.

4.2 Small Proof Property

The *depth* of a proof Π is the maximal number of rule applications in any of its branches, in other words the maximal length of a play in the proof search game. The *size* of a proof $|\Pi|$ is the total number of rule applications in the proof tree. A proof is *minimal* if no other proof of its conclusion sequent is of strictly smaller size. Note that a minimal proof must necessarily apply the (ax) and ($\bot_L$) rules as soon as possible; it also cannot decompose twice the same Boolean formula between two applications of a modal rule ($\Box_=$) or ($\Box_{\neq}$).

Inspecting the rules of our calculus, it appears that a cell in the history – including S_0, the modal part of the current antecedent – may be displaced, perhaps moved to the antecedent of the sequent, be enriched with new modal formulæ, but can never be lost:

▶ **Lemma 17.** *Let Π be a derivation of a sequent S. We have $S \sqsubseteq S'$ for any sequent S' occurring in Π.*

▶ **Proposition 18** (Bounded ($\Box_{\neq}$) Applications). *Let Π be a minimal proof. For any formula $\Box_{\neq}\psi$, each branch of Π contains at most three applications of the ($\Box_{\neq}$) rule with $\Box_{\neq}\psi$ as principal formula.*

Proof. Let Π be a minimal proof of $\mathcal{H}^0; \Gamma^0 \vdash \Delta^0$. Consider a branch of Π that contains three applications of ($\Box_{\neq}$) to the same formula $\Box_{\neq}\psi$. Let $S^k = (\mathcal{H}^k; \Gamma^k \vdash \Delta^k)_{0\leq k\leq B}$ be the (unannotated) sequents along that branch, of length B, and let $p < q < r$ be the indices of the conclusion sequents of the three successive applications of ($\Box_{\neq}$) with $\Box_{\neq}\psi$ as principal formula. We shall establish that the axiom rule applies after the third rule application, thus a fourth application is impossible in a minimal proof.

The first application of ($\Box_{\neq}$) with $\Box_{\neq}\psi$ as principal formula introduces (in its premise sequent) a new history cell containing $\Box_{\neq}\psi$. By Lemma 17, for all $k > p$, there exists $0 \leq i_k \leq |\mathcal{H}^k|$ such that $\Box_{\neq}\psi \in S^k_{i_k}$.

By minimality of Π we know that the axiom rule does not apply when the second ($\Box_{\neq}$) rule with $\Box_{\neq}\psi$ as principal formula is performed, thus $\Box_{\neq}\psi \notin \Gamma^q$ and $i_q > 0$. For the same reason we also have $\psi \notin \Gamma^{q+1}$. Thus the premise of this second rule application cannot be the first premise of the ($\Box_{\neq}$) rule, for otherwise ψ would belong to Γ^{q+1}. For the same reason, it cannot correspond to one of the other premises with $j \neq i_q$. Hence, it must be the premise with $j = i_q$, and S^{q+1} contains two distinct cells containing $\Box_{\neq}\psi$, namely the antecedent and the freshly created cell. By Lemma 17 again, it follows that for all $k > q$, $\Box_{\neq}\psi \in S^k_{j_k}$ for some cell index $0 \leq j_k \neq i_k \leq |\mathcal{H}^k|$.

For the third rule application, the same argument applies, but this time all the premises of the rule contain ψ in their antecedent Γ^{r+1}, i.e. the axiom rule is applicable immediately after the third ($\Box_{\neq}$) application. ◀

Note that we can create a new history cell only when we apply a modal rule ($\Box_{\neq}$), thus Proposition 18 indirectly provides a bound on the length of the history in minimal proofs.

▶ **Proposition 19** (Bounded ($\Box_{=}$) Applications). *Let Π be a minimal proof in which history lengths are bounded by h. For any formula $\Box_{=}\varphi$, all branches of Π contain at most $h + 1$ applications of the ($\Box_{=}$) rule with $\Box_{=}\varphi$ as principal formula.*

Proof sketch. Each application of ($\Box_{=}$) can only occur with conclusion sequents where $\Box_{=}\varphi$ does not appear in the antecedent Γ, or the axiom rule could be applied instead. Thus its premise sequent has one more cell containing $\Box_{=}\varphi$ than its conclusion sequent, namely in the antecedent. By Lemma 17, we can only repeat this operation $h + 1$ times before being forced to see $\Box_{=}\varphi$ in the antecedent Γ. ◀

From the previous two results we easily obtain the following statement, which also takes Boolean rules into account to obtain a polynomial bound on the depth of minimal proofs.

▶ **Theorem 20** (Polynomial Proof Depth). *Let $S = \mathcal{H}; \Gamma \vdash \Delta$ be a sequent, $m^{\neq}$ (resp. $m^{=}$) be the number of distinct $\Box_{\neq}$ subformulæ (resp. $\Box_{=}$ subformulæ) occurring in S, and p be the number of other subformulæ. If S is provable, then it has a proof of depth at most $(3m^{\neq} + m^{=}(3m^{\neq} + 1) + 1)(p + 1) + 1$.*

Proof. By Proposition 18 and the subformula property, the number of ($\Box_{\neq}$) rule applications along any branch is bounded by $3m^{\neq}$, which is also a bound on the length of histories in minimal proofs. By Proposition 19 and the subformula property, the total number of modal rule applications is thus bounded by $3m^{\neq} + m^{=}(3m^{\neq} + 1)$. In between any two modal rules we can apply at most p Boolean rules without introducing the same formula twice, and possibly conclude by an axiom. ◀

4.3 Computational Complexity

The polynomial bound of Theorem 20 on the length of branches in minimal proofs yields a proof search algorithm working in alternating polynomial time. The algorithm implements the proof search game with Prover as existential player and Spoiler as universal player. Actually, Prover can follow the strategy of Section 4.1 without loss of generality nor of efficiency. Since our calculus is sound and complete for **DataGL**, and AP = PSPACE [10], it yields a PSPACE upper bound for the validity problem for **DataGL**. Because **DataGL** is closed under negation and PSPACE under complement, the same bound holds for the satisfiability problem. We show that this is actually a tight bound.

▶ **Theorem 21.** *The validity and satisfiability problems for* **DataGL** *are* PSPACE*-complete.*

The lower bound is obtained by reducing the satisfiability problem for **GL** to its **DataGL** counterpart. PSPACE-hardness follows since **GL** validity is known to be PSPACE-hard [9, Theorem 7]. We can indeed think of a finite tree without data as a finite data tree where all the nodes share the same datum, and in such a tree $\Box_=$ behaves exactly as the **GL** modality $\Box$. Given a **GL** formula φ we define the **DataGL** formula $\lfloor\varphi\rfloor$ by substituting $\Box_=$ for $\Box$:

$$\lfloor p\rfloor \overset{\text{def}}{=} p\,, \qquad \lfloor\bot\rfloor \overset{\text{def}}{=} \bot\,, \qquad \lfloor\varphi\supset\psi\rfloor \overset{\text{def}}{=} \lfloor\varphi\rfloor\supset\lfloor\psi\rfloor\,, \qquad \lfloor\Box\varphi\rfloor \overset{\text{def}}{=} \Box_=\lfloor\varphi\rfloor\,.$$

▶ **Claim 22.** *For any* **GL** *formula φ, φ is satisfiable in* **GL** *if and only if $\lfloor\varphi\rfloor\wedge\Box_{\neq}\bot$ is satisfiable in* **DataGL***.*

Proof sketch. The formula $\Box_{\neq}\bot$ ensures that a single data value is used throughout its models. Given such a finite single-data data tree $\mathfrak{t}$, a straightforward structural induction over φ establishes that $\mathfrak{t}, w\models\lfloor\varphi\rfloor$ if and only if $\lfloor\mathfrak{t}\rfloor, w\models\varphi$, where $\lfloor\mathfrak{t}\rfloor$ is the tree obtained by erasing all data information from $\mathfrak{t}$. ◀

A final observation about proof search is that the polynomial bound of Theorem 20 on the depth also applies to *failed* searches that follow the strategy of Section 4.1. This is exploited by Lunel [19] to extract small counter-models from failed proof attempts when focusing on saturated sequents (the completeness proof in Section 3.4 is indeed essentially based on proof search): if a sequent is unprovable, then it has a counter-model of polynomial height. Hence **DataGL** has a *strong finite model property*. It yields a different proof of the PSPACE upper bound of Theorem 21 when combined with Proposition 6.7 of [12], though with a rather less concrete algorithm than proof search.

5 Concluding Remarks

The sequent calculus for **DataGL** is to the best of our knowledge the first instance of a proof system for a data-aware logic on finite data trees. It provides an optimal proof search algorithm to establish the validity of **DataGL** formulæ, with a PSPACE complexity.

The logic **DataGL** is still a rather small syntactic fragment of **CoreDataXPath**. There are two natural directions for extending our proof system towards full **CoreDataXPath**:

- one is allowing path expressions as in **CoreDataXPath**$(\downarrow^+)$, which is EXP-complete [12],
- the other is to add other navigational axes, starting with the ancestor axis $\uparrow^+$, as in the non primitive recursive **CoreDataXPath**$(\uparrow^+,\downarrow^+)$ [15].

On both accounts, the use of sequents enriched with histories is a promising starting point.

From a proof theory perspective, two lines of inquiry seem interesting. The first would be to develop a cut elimination procedure for our sequent calculus; by completeness of the calculus, the (cut) rule is admissible, but this is a semantic proof rather than a syntactic one. The second is to consider an extension of **DataGL** where histories are integrated as logical connectives in the syntax instead of being a mere extra-logical mechanism; this might help in designing Hilbert-style axiomatisations for data logics, along the same lines as the recent work of Abriola et al. [1] on XPath equipped with the immediate child relation.

Acknowledgements. This work benefited greatly from discussions with Diego Figueira and Luc Segoufin. We thank the anonymous reviewers for their helpful comments.

References

1 Sergio Abriola, María Emilia Descotte, Raul Fervari, and Santiago Figueira. Axiomatizations for downward XPath on data trees. Preprint, 2016. URL: `http://arxiv.org/abs/1605.04271`.

2 Arnon Avron. On modal systems having arithmetical interpretations. *J. Symb. Log.*, 49(3):935–942, 1984. `doi:10.2307/2274147`.

3 Michael Benedikt, Wenfei Fan, and Floris Geerts. XPath satisfiability in the presence of DTDs. *Journal of the ACM*, 55(2):1–79, 2008. `doi:10.1145/1346330.1346333`.

4 Michael Benedikt, Wenfei Fan, and Gabriel Kuper. Structural properties of XPath fragments. *Theor. Comput. Sci.*, 336(1):3–31, 2005. `doi:10.1016/j.tcs.2004.10.030`.

5 Michael Benedikt and Christoph Koch. XPath leashed. *ACM Computing Surveys*, 41(1):3:1–3:54, 2009. `doi:10.1145/1456650.1456653`.

6 Patrick Blackburn, Marteen de Rijke, and Yde Venema. *Modal Logic*, volume 53 of *Cambridge Tracts in Theoretical Computer Science*. Cambridge University Press, 2001.

7 Mikołaj Bojańczyk, Anca Muscholl, Thomas Schwentick, and Luc Segoufin. Two-variable logic on data trees and XML reasoning. *Journal of the ACM*, 56(3):1–48, 2009. `doi:10.1145/1516512.1516515`.

8 Kai Brünnler and Lutz Straßburger. Modular sequent systems for modal logic. In *Tableaux 2009*, volume 5607 of *LNCS*, pages 152–166. Springer, 2009. `doi:10.1007/978-3-642-02716-1_12`.

9 A. V. Chagrov and M. N. Rybakov. How many variables does one need to prove PSPACE-hardness of modal logics? In Philippe Balbiani, Nobu-Yuki Suzuki, Frank Wolter, and Michael Zakharyaschev, editors, *AiML 2002*, pages 71–82. King's College Publications, 2003.

10 Ashok K. Chandra, Dexter C. Kozen, and Larry J. Stockmeyer. Alternation. *Journal of the ACM*, 28(1):114–133, 1981. `doi:10.1145/322234.322243`.

11 Diego Figueira. Alternating register automata on finite words and trees. *Logic. Meth. in Comput. Sci.*, 8(1):1–43, 2012. `doi:10.2168/LMCS-8(1:22)2012`.

12 Diego Figueira. Decidability of downward XPath. *ACM Trans. Comput. Logic*, 13(4):1–40, 2012. `doi:10.1145/2362355.2362362`.

13 Diego Figueira. On XPath with transitive axes and data tests. In *PODS 2013*, pages 249–260. ACM, 2013. `doi:10.1145/2463664.2463675`.

14 Diego Figueira, Santiago Figueira, and Carlos Areces. Model theory of XPath on data trees. Part I: bisimulation and characterization. *J. Artif. Intell. Res.*, 53(1):271–314, 2015. `doi:10.1613/jair.4658`.

15 Diego Figueira and Luc Segoufin. Bottom-up automata on data trees and vertical XPath. In *STACS 2011*, volume 9 of *LIPIcs*, pages 93–104. LZI, 2011. `doi:10.4230/LIPIcs.STACS.2011.93`.

16 Andrzej Indrzejczak. Linear time in hypersequent framework. *Bull. Symb. Logic*, 22(1):121–144, 2016. `doi:10.1017/bsl.2016.2`.

17 M. Jurdziński and R. Lazić. Alternating automata on data trees and XPath satisfiability. *ACM Trans. Comput. Logic*, 12(3):1–21, 2011. doi:10.1145/1929954.1929956.

18 Emmanuel Kieroński and Lidia Tendera. On finite satisfiability of two-variable first-order logic with equivalence relations. In *LICS 2009*, pages 123–132, 2009. doi:10.1109/LICS.2009.39.

19 Simon Lunel. Systèmes de preuves pour logiques modales à données. Mémoire de Master, LMFI, Université Paris-Diderot, September 2015.

20 Sara Negri. Proof analysis in modal logic. *J. Phil. Log.*, 34(5–6):507–544, 2005. doi:10.1007/s10992-005-2267-3.

21 Francesca Poggiolesi. The method of tree-hypersequents for modal propositional logic. In David Makinson, Jacek Malinowski, and Heinrich Wansing, editors, *Towards Mathematical Philosophy*, volume 28 of *Trends in Logic*, pages 31–51. Springer, 2009. doi:10.1007/978-1-4020-9084-4_3.

22 Jonathan Robie, Don Chamberlin, Michael Dyck, and John Snelson. XML Path Language (XPath) 3.0. W3C Recommendation, World Wide Web Consortium (W3C), 2014. URL: http://www.w3.org/TR/xpath-30/.

23 Balder ten Cate, Gaëlle Fontaine, and Tadeusz Litak. Some modal aspects of XPath. *J. Appl. Non-Classical Log.*, 20(3):139–171, 2010. doi:10.3166/jancl.20.139-171.

24 Balder ten Cate, Tadeusz Litak, and Maarten Marx. Complete axiomatizations for XPath fragments. *J. Appl. Logic*, 8(2):153–172, 2010. doi:10.1016/j.jal.2009.09.002.

25 Balder ten Cate and Maarten Marx. Axiomatizing the logical core of XPath 2.0. *Theor. Comput. Sys.*, 44(4):561–589, 2009. doi:10.1007/s00224-008-9151-9.

26 Heinrich Wansing. Sequent calculi for normal modal propositional logics. *J. Logic Comput.*, 4(2):125–142, 1994. doi:10.1093/logcom/4.2.125.