Data Graphs with Incomplete Information (and a Way to Complete Them)

Carlos Areces[1,2], Valentin Cassano[1,2,3], Danae Dutto[1,2,3], and Raul Fervari[1,2,4]

[1] Consejo Nacional de Investigaciones Científicas y Técnicas (CONICET), Argentina
[2] Universidad Nacional de Córdoba (UNC), Argentina
[3] Universidad Nacional de Río Cuarto (UNRC), Argentina
[4] Guangdong Technion - Israel Institute of Technology (GTIIT), China

Abstract. We introduce a modal language for reasoning about data graphs with incomplete information. Such data graphs are formally represented as models in which data value functions are partial — to capture what is unknown. In this setting, we also allow for unknown data values to be learned. Our main result is a sound and strongly complete axiomatization for the logic.

Keywords: Data Graphs · Incomplete Data · Modal/Intuitionistic Logic.

1 Modal Logic & Semistructured-Data Query Languages

Nowadays, there is a well established connection between modal logics [15,13] and semistructured-data query languages such as XPath and some of its relatives [20,25]. The main reason is that semistructured data is usually represented in the form of relational structures or graphs (e.g., an XML document), and modal logics are well suited for describing and reasoning over this kind of structures. This perspective enables us to use modal logic tools to reason with (and about) XPath-like languages (see, e.g., [9]), thereby helping us to develop methods to check the consistency of a database and to optimize queries, among other tasks. Some of these ideas have been explored, e.g., in [18,19], and also, in the presence of data comparisons, in, e.g., [11,3,1,8,6].

The connection mentioned above is illustrated in [16]. There, a version of XPath with (in)equality data tests between attributes in an XML document is named Core-Data-XPath, here called $\text{XPath}_{=}$. Models of $\text{XPath}_{=}$ are usually data trees which can be seen as XML documents. A data tree is a tree whose nodes contain a label from a finite alphabet and a data value from an infinite domain. From a modal logic perspective, these data trees are a particular class of *relational models*. Naturally, this view can be extended to more general relational structures, i.e., to arbitrary *data graphs*. Since data graphs are the underlying mathematical structure in *graph databases* (see, e.g., [28,30,4]), studying the meta-logical properties of languages to query this particular kind of models is important (see, e.g., [27,1]).

Let us provide an example to guide our discussion. Fig. 1 depicts a graph database modelling a piece of a library catalog. There, we can see attributes

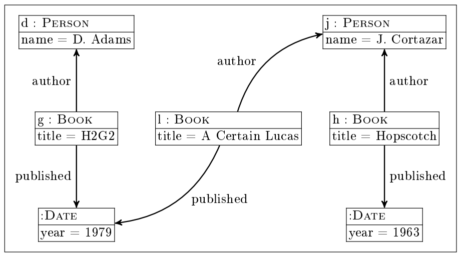

Fig. 1: Graph Database.

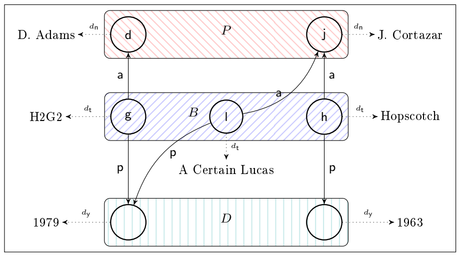

Fig. 2: Concrete Data Model.

(e.g., "author") as labels on edges; while nodes store data values represented as "tag = value" pairs (e.g., "title = H2G2"). Attributes are usually drawn from a finite set, while data values are assumed to be drawn from an infinite set. Additionally, some nodes may contain keys that uniquely identify a node (e.g., the ISBN number associated to a certain book, or the passport number of a person, indicated, e.g., "d:"). On this kind of graph database, we can state queries such as: "The book with the id l has the same author as the one with the id h". In this way, we can express both properties about the structure of the data graph, and about equality or inequality of the data values contained in it.

To explore graph databases and the possibility of querying them from a logical perspective we need an adequate logical language. We build on [6] and consider a hybrid modal language. In this setting, modalities encode attributes, proposition symbols encode types, and nominals from hybrid logic encode unique identifiers. Fig. 2 shows what the graph database in Fig. 1 looks like as a model of the hybrid modal language. In this figure, a and p are accessibility relations associated to their corresponding modalities (for the attributes "author" and "published"); P, B, and D, are proposition symbols (for the types "Person", "Book", and "Date"); and d, j, g, l, and h are nominals (for the unique identifiers appearing in the graph database, representing the passport number of a person or the ISBN of a book). Finally, data values in Fig. 2 are encoded as functions d_{n}, d_{t}, and d_{y} (standing for "name", "title", and "year" values).

To be noted, the model Fig. 2, usually called a *concrete model*, tries to remain as close as possible to the actual graph-database (the similarities between Fig. 1 and Fig. 2 should be obvious). However, depending on the expressive power of our logical language, a simpler, more abstract class of models may be better suited. In particular, many data query languages, like XPath, "abstract away" from the *actual* data values in the graph database, and only care about the result of performing *comparisons* between them (see, e.g., [22,23,3,1,6]). In other words, if the language includes only tests for (in)equality, for example, we can forget about actual data values and define an equivalent model (from the logic standpoint) with data equality relations instead of data value functions. Precisely, two nodes will be related by data equality if and only if they have the same data value. In [6] this connection is made explicit, and the simpler class of *abstract models* is exploited to prove a soundness and completeness result for a Hilbert-style axiomatization of $\text{XPath}_=$ extended with nominals and the satisfaction operator. Henceforth, we will refer to this logic as $\text{HXPath}_=$.

To tackle the main theme of this article, let us continue with our running example. Suppose that the library catalog from Fig. 1 contains a record of a rare book like the *Voynich Manuscript* [31]. If we treat books uniformly, the catalog will have a book node with data values for the name(s) of its author(s) and its publication year. The problem with the Voynich Manuscript is that neither the author(s) name(s) nor the exact year of publication are known. Notwithstanding, the book certainly has one or more authors, and certainly it was completed at some point in time. It is only that this information is unknown to us at the moment (but nothing prevents us from discovering this information in the future). It seems natural to consider, in this case, that the data values for author and year are undefined for the corresponding node (or, equivalently, to consider that they are assigned a special 'null' value). We call these models "partial data models". Formally, in the above-mentioned set up of concrete models, this calls for the use of *partial functions* for data representation, and to consider *update operators* that would complete the assignment whenever the previously unknown information becomes available. But once this is done, we should reconsider the relation between concrete and abstract data models, as it is in principle unclear how partial data models should be represented in an abstract way, and how are they related by updates. Intuitively, we could think that now abstract partial models would contain "partial equivalence relation" (but we would have to give up reflexivity!). And "learning" the value of an attribute (e.g., the date of edition of a book) may be thought of as "extending" these relations to reflect the (in)equalities that now are obtained. As we will discuss in this article, allowing for this perspective has a huge impact on the logic, and surprisingly at first (and less so after all the work is properly done) it leads to an intuitionistic version of $\text{HXPath}_=$. Interestingly, this perspective also points to a fresh connection between modal intuitionistic logics and dynamic logics like those in, e.g., [29,7,10].

Our contribution. We explore a novel approach for treating undefined values on data graphs. We define a notion of update and present it as a partial ordering on a collection of partial data models. To our knowledge this is the first time that these notions have been studied for variants of $\text{XPath}_=$. Our work builds on ideas present in [6] about the data-aware language $\text{HXPath}_=$. Moreover, we use tools from intuitionistic hybrid logic [17] to model the possibly undefined data values and their possible future definition. The result of putting these pieces together is a new logic that we call $\text{IHXPath}_=$. While at the syntactic level, the language of $\text{IHXPath}_=$ seems identical to the one of $\text{HXPath}_=$, semantics is much more involved (as is the case with classical and intuitionistic propositional logic). *Partial data models*, and the exact correspondence between concrete and abstract models in this new setting is presented in Sec. 2. A variant of the language $\text{HXPath}_=$ studied in [6], interpreted over abstract partial data models is given in Sec. 3. Sec. 4 presents an axiom system for $\text{IHXPath}_=$, which is strongly complete for any extension with so-called pure axioms and saturation rules (Sec. 5). Sec. 6 discusses our results and describes future lines of research.

2 Background and Motivation

The logic HXPath$_=$ in [6] formalizes a fragment of XPath that captures both topological and data (in)equality queries using elements from hybrid modal logic [5]. In HXPath$_=$ data graphs become models of the logic defined in two alternative, yet equivalent, ways. We explain what these models look like and use this explanation to motivate our work. In what follows, we assume Prop, Nom, Mod, and Cmp, are pairwise disjoint fixed sets of symbols for propositions, nominals, modalities, and data comparisons, respectively. Moreover, we assume Mod and Cmp to be finite; and Prop and Nom to be countably infinite.

Definition 1 (The models of HXPath$_=$). *A* concrete data model *is a tuple*

$$\mathfrak{C} = \langle N, \{R_{\mathsf{a}}\}_{\mathsf{a}\in\mathsf{Mod}}, D, \{d_{\mathsf{c}}\}_{\mathsf{c}\in\mathsf{Cmp}}, g, V\rangle,$$

where N is a non-empty set of nodes; each R_{a} is a binary accessibility relation on nodes; D is a non-empty set of data values; $d_{\mathsf{c}} : N \to D$ is a (total) function that assigns data values to nodes; $g : \mathsf{Nom} \to N$ is a (total) function that assigns nominals to nodes; and $V : \mathsf{Prop} \to 2^N$ is a valuation function. In turn, an abstract data model *is a tuple*

$$\mathfrak{A} = \langle N, \{R_{\mathsf{a}}\}_{\mathsf{a}\in\mathsf{Mod}}, \{\approx_{\mathsf{c}}\}_{\mathsf{c}\in\mathsf{Cmp}}, g, V\rangle,$$

where N, R_{a}, g, and V are as before; and each $\approx_{\mathsf{c}}$ is an equivalence relation on nodes (representing nodes with the same data value for c).

Remark 1. Notice that concrete and abstract data models are in correspondence to each other. The first yields the second by defining $\approx_{\mathsf{c}} = \{\, (n, n') \mid d_{\mathsf{c}}(n) = d_{\mathsf{c}}(n') \,\}$, while the second yields the first by defining $d_{\mathsf{c}}(n) = [n]_{\mathsf{c}}$.

Fig. 2 depicts a concrete data model. More precisely, it depicts only some relevant features of a concrete data model in the context of an example. This is particularly true, e.g., of data values functions such as d_{t}. In other words, as a total function, d_{t} must assign a value to each node; yet, only some of these values are present. Though not a technical issue, considering total data values functions is inelegant from a knowledge representation perspective. After all, why must we assign titles to nodes whose values are meant to represent, e.g., dates? This observation takes us to Def. 2.

Definition 2. *A* concrete partial data model *is a tuple $\mathfrak{C}$ as in Def. 1; with the exception that each $d_{\mathsf{c}} : N \nrightarrow D$ is a partial function.*

What about corresponding abstract data models? On a first glimpse, we may think of them via relations $\approx_{\mathsf{c}} = \{\, (n, n') \mid d_{\mathsf{c}}(n) = d_{\mathsf{c}}(n') \,\}$, which would turn out to be equivalence relations. However, this fails to account for the cases when d_{c} is undefined. Namely, n and n' need defined data values, i.e., need to belong to the domain of d_{c}. This forces us to abandon reflexivity and view $\approx_{\mathsf{c}}$ as a partial equivalence relation. This observation takes us to Def. 3.

Definition 3. *An* abstract partial data model *is a tuple $\mathfrak{A}$ as in Def. 1, except that $\approx_{\mathsf{c}}$ is a partial equivalence relation, i.e., it is symmetric and transitive.*

As a result of these changes, it turns out that now data inequality in abstract partial data models is not the complement of data equality. Namely, two nodes n and n' in an abstract partial data model are taken to have different data if and only if it is not the case that $n \approx_{\mathsf{c}} n'$ and in addition $n \approx_{\mathsf{c}} n$ and $n' \approx_{\mathsf{c}} n'$. This observation will have an important impact in the way we need to define our axiomatization in Sec. 4.

Remark 2. We can still build concrete partial data models from abstract partial data models by setting $d_{\mathsf{c}}(n) = \{\, n' \mid n \approx_{\mathsf{c}} n' \,\}$ if $\{\, n' \mid n \approx_{\mathsf{c}} n' \,\} \neq \emptyset$, and $d_{\mathsf{c}}(n)$ undefined otherwise. Similar to the case in Rem. 1, the correspondence between abstract and concrete partial data models is clear.

Now for the last piece of the puzzle. We have established a natural generalization of concrete and abstract data models capable of handling partial information. Let us consider again the case of the Voynich manuscript. It is clear that partial functions are all that we need to represent this book in our models. But suppose that at some point, we do learn the date of edition. This can be formalized as a relation on partial data models reflecting the new 'things' we have learned. One of the possibly different ways in which we can capture this idea is provided in Def. 4.

Definition 4. *Let $\mathfrak{C}$ and $\mathfrak{C}'$ be two concrete partial data model. We write $\mathfrak{C} \preccurlyeq \mathfrak{C}'$, and call $\mathfrak{C}'$ a concrete data update on $\mathfrak{C}$, iff $\mathfrak{C}'$ replaces some partial function d_{c} in $\mathfrak{C}$ by the partial function $d'_{\mathsf{c}} = d_{\mathsf{c}} \cup \{n \mapsto v\}$ s.t. for all $n' \in \mathrm{dom}(d_{\mathsf{c}})$*

$$d'_{\mathsf{c}}(n') = \begin{cases} v & \textit{if } d_{\mathsf{c}}(n) = d_{\mathsf{c}}(n') \\ d_{\mathsf{c}}(n') & \textit{otherwise} \end{cases}$$

and is otherwise equal to $\mathfrak{C}$. Similarly, let $\mathfrak{A}$ and $\mathfrak{A}'$ be two abstract partial data models. We write $\mathfrak{A} \preccurlyeq \mathfrak{A}'$, and call $\mathfrak{A}'$ an abstract data update on $\mathfrak{A}$, iff $\mathfrak{A}'$ replaces some relation $\approx_{\mathsf{c}}$ in $\mathfrak{A}$ by the relation $\approx'_{\mathsf{c}} = (\approx_{\mathsf{c}} \cup \{(n, n'), (n', n)\})^+$, where $n = n'$ or $n' \in \mathrm{dom}(\approx_{\mathsf{c}})$, and is otherwise equal to $\mathfrak{A}$.

The notion of update in Def. 4 is best explained by making explicit the kinds of updates that are allowed. First, (1) we can add a data value v of type c to an undefined node n. In the concrete case, we have $n \notin \mathrm{dom}(d_{\mathsf{c}})$. This means that (1) yields a partial function d'_{c} which adds the pair $n \mapsto v$ to d_{c}. In the abstract case, we have $n \notin \mathrm{dom}(\approx_{\mathsf{c}})$. This means that (1) yields a partial equivalence relation $\approx'_{\mathsf{c}}$ which adds the pair (n, n) to $\approx_{\mathsf{c}}$. Second, (2) we can "update" the data value v of type c assigned to a node n. On a first take, we restrict our attention to updates that preserve "data equality". In the concrete case, (2) yields a partial function d'_{c} that replaces all pairs $n' \mapsto d_{\mathsf{c}}(n)$ in d_{c} by a corresponding pair $n' \mapsto v$. This kind of update resembles an aliasing situation, i.e., if a data value is accessed through different nodes, then, modifying the data value through one node implicitly modifies the data values associated with all aliased nodes. In the abstract case, (2) captures the idea of partial equivalence classes in $\approx_{\mathsf{c}}$ being "merged" in $\approx'_{\mathsf{c}}$. Interestingly, if $\mathfrak{C}$ and $\mathfrak{C}'$ are concrete partial data models such that $\mathfrak{C} \preccurlyeq \mathfrak{C}'$, then, their abstract counterparts $\mathfrak{A}$ and $\mathfrak{A}'$ are such that $\mathfrak{A} \preccurlyeq \mathfrak{A}$, and viceversa.

3 Reasoning with Incomplete Information

In this section we introduce a modal logic to reason about collections of abstract partial data models related by updates. We refer to this modal logic as Intuitionistic Hybrid XPath with Data (IHXPath$_=$ for short). We define its syntax in Def. 5 and its semantics in Def. 7.

Definition 5. *The language of IHXPath$_=$ has* path *expressions (α, β, ...) and* node *expressions (φ, ψ, ...), mutually defined by the grammar:*

$$\alpha, \beta := \mathsf{a} \mid @_i \mid [\varphi] \mid \alpha\beta$$
$$\varphi, \psi := p \mid i \mid \bot \mid \varphi \vee \psi \mid \varphi \wedge \psi \mid \varphi \to \psi \mid \langle\alpha\rangle\varphi \mid [\alpha]\varphi \mid \langle\alpha *_{\mathsf{c}} \beta\rangle \mid [\alpha *_{\mathsf{c}} \beta].$$

In this grammar, $p \in \mathsf{Prop}$, $i \in \mathsf{Nom}$, $\mathsf{a} \in \mathsf{Mod}$, and $*_{\mathsf{c}} \in \{=_{\mathsf{c}}, \neq_{\mathsf{c}}\}$, with $\mathsf{c} \in \mathsf{Cmp}$. PE is the set of all path expressions and NE is the set of all nodes expressions.

We abbreviate $\neg\varphi := \varphi \to \bot$, $\top := \neg\bot$, $\epsilon := [\top]$, $@_i\varphi := \langle @_i\rangle\varphi$. We write $*_{\mathsf{c}}$ if it is indistinct to use $=_{\mathsf{c}}$ or $\neq_{\mathsf{c}}$. We refer to path expressions of the form $[\varphi]$ as *tests*; and to node expressions of the form $\langle\alpha *_{\mathsf{c}} \beta\rangle$ or $[\alpha *_{\mathsf{c}} \beta]$ as *data comparisons*. Intuitively, a path expression $@_i\alpha$ indicates an α path which starts at a node named i. Moreover, we read $\langle\alpha *_{\mathsf{c}} \beta\rangle$ as *the endpoints of some α and some β paths have the same/different data value (of type* c*)*, and $\langle\alpha\rangle\varphi$ as *φ holds at the endpoint of some α path.* We make clear the intuitive role of "box" modalities after introducing what models look like in our logic.

Remark 3. It is worth noting that, on the surface, the language for IHXPath$_=$ is inherently that of HXPath$_=$ [6]. In more detail, however, it contains some important changes. In particular, box data comparisons $[\alpha *_{\mathsf{c}} \beta]$ are introduced as primitive, as are the formulas $\langle\alpha\rangle\varphi$ and $[\alpha]\varphi$. As it will be made clear in what follows, $[\alpha *_{\mathsf{c}} \beta]$ and $[\alpha]\varphi$ are no longer definable in terms of $\langle\alpha *_{\mathsf{c}} \beta\rangle$ and $\langle\alpha\rangle\varphi$ due to the lack of duality between boxes and diamonds in an intuitionistic modal setting. The case of $\langle\alpha\rangle\varphi$ is particularly interesting as its definability in terms of $\langle\alpha =_{\mathsf{c}} \alpha\rangle$ in HXPath$_=$ hinged on the reflexivity of data comparisons, which we gave up to deal with partial data values.

Let us now turn our attention to the structures on which to interpret path and nodes expressions (Def. 6), and to the corresponding notion of satisfiability on these structures (Def. 7). Our definitions combine ideas found in [6] and [17].

Definition 6. *An abstract partial update structure (alias a model) is a tuple*

$$\mathcal{M} = \langle M, \preccurlyeq, \{\langle\mathfrak{A}_m, \cong_m\rangle\}_{m\in M}\rangle,$$

where $\langle M, \preccurlyeq\rangle$ is a poset, and for all m,

$$\mathfrak{A}_m = \langle N_m, \{R^{\mathsf{a}}_m\}_{\mathsf{a}\in\mathsf{Mod}}, \{\approx^{\mathsf{c}}_m\}_{\mathsf{c}\in\mathsf{Cmp}}, g_m, V_m\rangle$$

is an abstract partial data model (Def. 3) and $\cong_m$ is a congruence on $\mathfrak{A}_m$. In addition, for all $m \preccurlyeq m'$:

(i) $N_m \subseteq N_{m'}$, (ii) $\cong_m \subseteq \cong_{m'}$, (iii) $R^{\mathsf{a}}_m \subseteq R^{\mathsf{a}}_{m'}$, (iv) $\approx^{\mathsf{c}}_m \subseteq \approx^{\mathsf{c}}_{m'}$,
(v) for all $p \in \mathsf{Prop}$, $V_m(p) \subseteq V_{m'}(p)$, and (vi) for all $i \in \mathsf{Nom}$, $g_m(i) = g_{m'}(i)$.

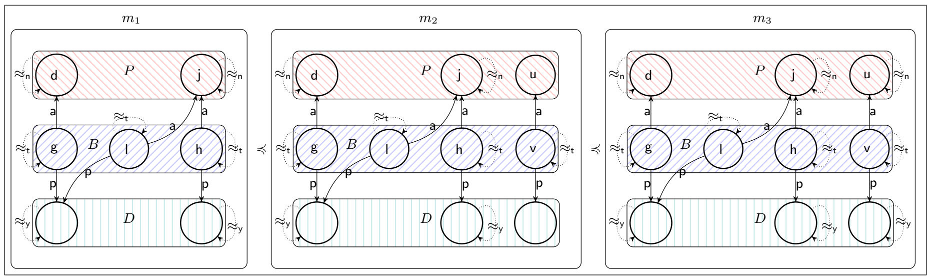

Fig. 3: Abstract Partial Data Update Structure.

Definition 7. *The relation $\Vdash$ of satisfiability is defined as:*

$\mathcal{M}, m, n, n' \Vdash \mathsf{a}$	*iff*	$nR^{\mathsf{a}}_m n'$
$\mathcal{M}, m, n, n' \Vdash @_i$	*iff*	$g_m(i) \cong_m n'$
$\mathcal{M}, m, n, n' \Vdash [\varphi]$	*iff*	$n \cong_m n'$ *and* $\mathcal{M}, m, n \Vdash \varphi$
$\mathcal{M}, m, n, n' \Vdash \alpha\beta$	*iff*	*exists* $n'' \in N_m$ *s.t.* $\mathcal{M}, m, n, n'' \Vdash \alpha$ *and* $\mathcal{M}, m, n'', n' \Vdash \beta$
$\mathcal{M}, m, n \Vdash \bot$		*never*
$\mathcal{M}, m, n \Vdash p$	*iff*	$n \in V_m(p)$
$\mathcal{M}, m, n \Vdash i$	*iff*	$n \cong_m g_m(i)$
$\mathcal{M}, m, n \Vdash \varphi \wedge \psi$	*iff*	$\mathcal{M}, m, n \Vdash \varphi$ *and* $\mathcal{M}, m, n \Vdash \psi$
$\mathcal{M}, m, n \Vdash \varphi \vee \psi$	*iff*	$\mathcal{M}, m, n \Vdash \varphi$ *or* $\mathcal{M}, m, n \Vdash \psi$
$\mathcal{M}, m, n \Vdash \varphi \to \psi$	*iff*	*for all* $m \preccurlyeq m'$, $\mathcal{M}, m', n \Vdash \varphi$ *implies* $\mathcal{M}, m', n \Vdash \psi$
$\mathcal{M}, m, n \Vdash \langle\alpha\rangle\varphi$	*iff*	*exists* $n' \in N_m$ *s.t.* $\mathcal{M}, m, n, n' \Vdash \alpha$ *and* $\mathcal{M}, m, n' \Vdash \varphi$
$\mathcal{M}, m, n \Vdash [\alpha]\varphi$	*iff*	*for all* $m \preccurlyeq m'$, $n' \in N_{m'}$ $\mathcal{M}, m', n, n' \Vdash \alpha$ *implies* $\mathcal{M}, m', n' \Vdash \varphi$
$\mathcal{M}, m, n \Vdash \langle\alpha =_{\mathsf{c}} \beta\rangle$	*iff*	*exists* $n', n'' \in N_m$ *s.t.* $\mathcal{M}, m, n, n' \Vdash \alpha$, $\mathcal{M}, m, n, n'' \Vdash \beta$, *and* $n' \approx^{\mathsf{c}}_m n''$
$\mathcal{M}, m, n \Vdash \langle\alpha \neq_{\mathsf{c}} \beta\rangle$	*iff*	*exists* $n', n'' \in N_m$ *s.t.* $\mathcal{M}, m, n, n' \Vdash \alpha$, $\mathcal{M}, m, n, n'' \Vdash \beta$, *and* $n' \approx^{\mathsf{c}}_m n'$, $n'' \approx^{\mathsf{c}}_m n''$, *and* $n' \not\approx^{\mathsf{c}}_m n''$
$\mathcal{M}, m, n \Vdash [\alpha =_{\mathsf{c}} \beta]$	*iff*	*for all* $m \preccurlyeq m'$, $n', n'' \in N_{m'}$ $\mathcal{M}, m', n, n' \Vdash \alpha$ *and* $\mathcal{M}, m', n, n'' \Vdash \beta$ *implies* $n' \approx^{\mathsf{c}}_v n''$
$\mathcal{M}, m, n \Vdash [\alpha \neq_{\mathsf{c}} \beta]$	*iff*	*for all* $m \preccurlyeq m'$, $n', n'' \in N_{m'}$ $\mathcal{M}, m', n, n' \Vdash \alpha$ *and* $\mathcal{M}, m', n, n'' \Vdash \beta$ *implies* $n' \approx^{\mathsf{c}}_m n'$, $n'' \approx^{\mathsf{c}}_m n''$, $n' \not\approx^{\mathsf{c}}_m n''$.

For $\Gamma \subseteq \mathsf{NE}$, we define $\mathcal{M}, m, n \Vdash \Gamma$ iff $\mathcal{M}, m, n \Vdash \gamma$ for all $\gamma \in \Gamma$. Moreover, for $\Gamma \cup \{\varphi\} \subseteq \mathsf{NE}$, we define $\Gamma \vDash \varphi$ iff $\mathcal{M}, m, n \Vdash \Gamma$ implies $\mathcal{M}, m, n \Vdash \varphi$.

Intuitively, models as in Def. 6 can be understood as collections of abstract partial data models related by abstract data updates (cf. Def. 4). These collections capture possible "histories" of updates. Interpreting "box" data comparisons and implications in an intuitionistic way permit us to reason about such "histories". The result in Prop. 1 is typical in an intuitionistic setting.

Proposition 1. *It follows that:*

(1) $\mathcal{M}, m, n \Vdash \varphi$ *and* $m \preccurlyeq m'$ *implies* $\mathcal{M}, m', n \Vdash \varphi$*;*
(2) $\mathcal{M}, m, n, n' \Vdash \alpha$ *and* $m \preccurlyeq m'$ *implies* $\mathcal{M}, m', n, n' \Vdash \alpha$*;*
(3) $\mathcal{M}, m, n \Vdash \varphi$ *and* $n \cong_m n'$ *implies* $\mathcal{M}, m, n' \Vdash \varphi$*;*
(4) $\mathcal{M}, m, n, n' \Vdash \alpha$ *and* $n \cong_m n''$ *and* $n' \cong_m n'''$ *implies* $\mathcal{M}, m, n'', n''' \Vdash \alpha$*.*

Example 1. We conclude this section with an example illustrating models and node and path expressions in use. For instance, the model in Fig. 3 may be understood as the history of what would occur if we add the Voynich manuscript (m_2) to our original library catalog (m_1), and later on we learn its author(s) and publication date (m_3). This history can be queried using node expressions such as $\langle @_{\mathsf{l}}\mathsf{a} =_{\mathsf{n}} @_{\mathsf{h}}\mathsf{a}\rangle$, stating that the author's name of *"A certain Lucas"* (l) and of *"Hopscotch"* (h) is the same, and it holds in m_1, m_2, and m_3. We can also write $@_{\mathsf{v}}B$, stating that *"The Voynich Manuscript"* (v) is a book on the catalog. This new node expression does not hold at m_1, but it holds at m_2 and m_3 –once the book has been added into the catalog. To be noted, adding a new named node is modeled by the addition of a new node together with the association, via $\cong$, of this new node to an already existing named node. Moreover, we can check that $@_{\mathsf{v}}\langle \mathsf{p}\rangle\top$, stating that v has a publication date, does not hold at m_1, but it holds at m_2 and m_3. Finally, we can check that $@_{\mathsf{v}}\langle \mathsf{p} =_{\mathsf{y}} \mathsf{p}\rangle$, stating that the publication date (p) of v is known, holds only at m_3.

4 Axiomatization and Completeness

In this section we present a strongly complete axiom system for IHXPath$_=$. This axiom system takes inspiration from [6] and [17], and consists of the axiom schemata in Tab. 1 and the inference rules in Tab. 2. The axioms under the heading 'Comparisons' in Tab. 1 deserve a short explanation. These axioms differ slightly from those in [6], since dealing with partial data values forced us to give up reflexivity for data equality. This implies that equality/inequality tests must ensure that they contain actual data. The rules of inference for 'Paths' generalize those from [17] and handle data comparisons. The axiom system for IHXPath$_=$ gives rise to a Hilbert-style notion of deduction of a node expression φ from a set of node expressions Γ, written $\Gamma \vdash \varphi$, defined inductively as usual. Prop. 2 is useful in our proof of completeness.

Proposition 2 (Agree). $\vdash @_i @_j \varphi \leftrightarrow @_j \varphi$*.*

Soundness and Completeness. The rest of this section covers the adequacy of the axiom system. Soundness is obtained by induction. We prove completeness by showing that every consistent set of node expressions is satisfiable. We use a Henkin-style construction akin to that for Hybrid Logic (see [24,14]). We take $\mathsf{NE}(\mathsf{Nom}')$, alias NE', for the set of node expressions with nominals in a set Nom'.

Table 1: Axioms for $\text{IHXPath}_=$

Basic	
(IPL)	Theorems of Intuitionistic Prop. Logic
Satisfaction	
$(\text{Distr}^{@}_{\wedge})$	$@_i(\varphi \wedge \psi) \leftrightarrow (@_i\varphi \wedge @_i\psi)$
$(\text{Distr}^{@}_{\vee})$	$@_i(\varphi \vee \psi) \leftrightarrow (@_i\varphi \vee @_i\psi)$
$(\text{Distr}^{@}_{\rightarrow})$	$@_i(\varphi \rightarrow \psi) \leftrightarrow (@_i\varphi \rightarrow @_i\psi)$
(Falsum)	$@_i\bot \rightarrow \bot$
(Refl@)	$@_i i$
Comparisons	
$(\text{T}\langle * \rangle)$	$\langle \alpha *_{\mathsf{c}} \beta \rangle \rightarrow \langle \alpha =_{\mathsf{c}} \alpha \rangle$
$(\text{B}\langle * \rangle)$	$\langle \alpha *_{\mathsf{c}} \beta \rangle \leftrightarrow \langle \beta *_{\mathsf{c}} \alpha \rangle$
$(4\langle = \rangle)$	$(\langle \alpha =_{\mathsf{c}} @_i \rangle \wedge \langle @_i =_{\mathsf{c}} \beta \rangle) \rightarrow \langle \alpha =_{\mathsf{c}} \beta \rangle$
(Irref)	$\neg\langle \alpha \neq_{\mathsf{c}} \alpha \rangle$
(CTran)	$(\langle \alpha \neq_{\mathsf{c}} \beta \rangle \wedge \langle @_i =_{\mathsf{c}} @_i \rangle) \rightarrow (\langle \alpha \neq_{\mathsf{c}} @_i \rangle \vee \langle @_i \neq_{\mathsf{c}} \beta \rangle)$
(Comp)	$(\langle \alpha =_{\mathsf{c}} @_i \rangle \wedge \langle @_i \neq_{\mathsf{c}} \beta \rangle) \rightarrow \langle \alpha \neq_{\mathsf{c}} \beta \rangle$
$(\text{EM}^{\langle = \rangle}_{\langle \neq \rangle})$	$(\langle \alpha =_{\mathsf{c}} \alpha \rangle \wedge \langle \beta =_{\mathsf{c}} \beta \rangle) \rightarrow (\langle \alpha =_{\mathsf{c}} \beta \rangle \vee \langle \alpha \neq_{\mathsf{c}} \beta \rangle)$

Paths	
(Cat)	$\langle \alpha \rangle \langle \beta \rangle \varphi \leftrightarrow \langle \alpha\beta \rangle \varphi$
(Id_ϵ)	$\langle \epsilon \rangle \varphi \leftrightarrow \varphi$
(Dist1)	$@_i\langle \alpha\beta *_{\mathsf{c}} \gamma \rangle \leftrightarrow @_i(\langle \alpha \rangle \langle \beta *_{\mathsf{c}} @_i\gamma \rangle)$
(Dist2)	$\langle \alpha \rangle \langle \beta *_{\mathsf{c}} \gamma \rangle \rightarrow \langle \alpha\beta *_{\mathsf{c}} \alpha\gamma \rangle$
(Dist3)	$\langle @_i\alpha *_{\mathsf{c}} @_i\beta \rangle \rightarrow @_i\langle \alpha *_{\mathsf{c}} \beta \rangle$
(Test)	$\langle [\psi]\alpha \rangle \varphi \leftrightarrow \psi \wedge \langle \alpha \rangle \varphi$
(Scope)	$\langle @_j\alpha *_{\mathsf{c}} \beta \rangle \rightarrow \langle @_i @_j \alpha *_{\mathsf{c}} \beta \rangle$
(Back)	$\langle \alpha @_i \beta *_{\mathsf{c}} \gamma \rangle \rightarrow \langle @_i\beta *_{\mathsf{c}} \gamma \rangle$
$(\langle \alpha \rangle \text{I})$	$(\langle \alpha \rangle i \wedge @_i\varphi) \rightarrow \langle \alpha \rangle \varphi$
$(\langle * \rangle \text{I})$	$(\langle \alpha \rangle i \wedge \langle @_i *_{\mathsf{c}} \beta \rangle) \rightarrow \langle \alpha *_{\mathsf{c}} \beta \rangle$
$([\alpha]\text{E})$	$(\langle \alpha \rangle j \wedge [\alpha]\varphi) \rightarrow @_j\varphi$
$([*]\text{E})$	$(\langle \alpha \rangle i \wedge \langle \beta \rangle j \wedge [\alpha *_{\mathsf{c}} \beta]) \rightarrow \langle @_i *_{\mathsf{c}} @_j \rangle$

Table 2: Rules of Inference for $\text{IHXPath}_=$

Basic
$\dfrac{\varphi \quad \varphi \rightarrow \psi}{\psi}\,(\text{MP})$
Satisfaction
$\dfrac{\varphi}{@_i\varphi}\,(@\text{I'})$
$\dfrac{@_i j \quad @_i\varphi}{@_j\varphi}\,(\text{Nom})$
$\dfrac{@_i\varphi}{\varphi}\,(@\text{E})^\dagger$
† i does not occur in φ.

Paths
$\dfrac{\varphi \rightarrow @_i\langle \alpha\beta *_{\mathsf{c}} \gamma \rangle \wedge (\varphi \wedge @_i\langle \alpha \rangle j \wedge @_i\langle @_j\beta *_{\mathsf{c}} \gamma \rangle) \rightarrow \psi}{\varphi \rightarrow \psi}\,(\langle * \rangle \text{E})^\dagger$
$\dfrac{(\varphi \rightarrow @_i\langle \alpha \rangle \chi) \wedge ((\varphi \wedge @_j\chi \wedge @_i\langle \alpha \rangle j) \rightarrow \psi)}{\varphi \rightarrow \psi}\,(\langle \alpha \rangle \text{E})^\dagger$
$\dfrac{(\varphi \wedge @_i(\langle \alpha \rangle j \wedge \langle \beta \rangle k)) \rightarrow \langle @_j *_{\mathsf{c}} @_k \rangle}{\varphi \rightarrow @_i[\alpha *_{\mathsf{c}} \beta]}\,([*]\text{I})^\dagger$
$\dfrac{(\varphi \wedge @_i\langle \alpha \rangle j) \rightarrow @_j\psi}{\varphi \rightarrow @_i[\alpha]\psi}\,([\alpha]\text{I})^\dagger$
† j and k do not occur in α, β, γ, χ nor ψ.

Definition 8 (Saturated). *Let* $\mathsf{Nom}' \subset \mathsf{Nom}''$; $\Gamma'' \subseteq \mathsf{NE}(\mathsf{Nom}'')$ *is* saturated *iff:*

1. $\Gamma'' = \{ \varphi \mid \Gamma'' \vdash \varphi \} \subset \mathsf{NE}(\mathsf{Nom}'')$;
2. $@_i(\varphi \vee \psi) \in \Gamma''$ *implies* $@_i\varphi \in \Gamma''$ *or* $@_i\psi \in \Gamma''$;
3. *exists* $i \in \mathsf{Nom}''$ *s.t.* $i \in \Gamma''$;
4. $@_i\langle \mathsf{a} \rangle \varphi \in \Gamma''$ *implies exists* $j \in \mathsf{Nom}''$ *s.t.* $\{@_j\varphi, @_i\langle \mathsf{a} \rangle j\} \subseteq \Gamma''$
5. $@_i\langle @_j \mathsf{a}\alpha *_{\mathsf{c}} \beta \rangle \in \Gamma''$ *implies*
   exists $k \in (\mathsf{Nom}'' \setminus \mathsf{Nom}')$ *s.t.* $\{@_j\langle \mathsf{a} \rangle k, @_i\langle @_k\alpha *_{\mathsf{c}} \beta \rangle\} \subseteq \Gamma''$.

The conditions above have the following names: 1. $\vdash$*-closed; 2. the disjunction property; 3. named; 4.* $\langle \alpha \rangle$*-pasted; and 5.* $\langle * \rangle$*-pasted.*

Now we are in position to establish Lem. 1, a.k.a., the *Lindenbaum Lemma*. This lemma states a crucial result: consistent sets can be extended to *saturated* sets (enriching the language with new symbols for nominals).

Lemma 1 (Saturation Lemma). *Let $\mathsf{Nom}' \subset \mathsf{Nom}''$, and $\Gamma' \cup \{\psi\} \subseteq \mathsf{NE}'$ be s.t. $\Gamma' \nvdash \psi$. There is $\Gamma'' \subseteq \mathsf{NE}''$ s.t. (1) $\Gamma' \subseteq \Gamma''$, (2) Γ'' is saturated, and (3) $\Gamma'' \nvdash \psi$.*

Proof. Enumerate all node expressions in NE'' and let $k \in (\mathsf{Nom}'' \setminus \mathsf{Nom}')$ be the first nominal in this enumeration. Define $\Sigma_0 = \Gamma' \cup \{k\}$. Now, suppose that we have defined Σ_n, for $n \geq 0$. Let $\varphi_{(n+1)}$ be the $(n+1)^{\text{th}}$ node expression in the enumeration. If $\Sigma_n \cup \{\varphi_{(n+1)}\} \vdash \psi$, then, define $\Sigma_{(n+1)} = \Sigma_n$. Otherwise, i.e., if $\Sigma_n \cup \{\varphi_{(n+1)}\} \nvdash \psi$, then, define $\Sigma_{(n+1)} = \Sigma_n \cup \{\varphi_{(n+1)}\} \cup \Sigma'$ where:

$$\Sigma' = \begin{cases} \emptyset & \text{if } \varphi_{(n+1)} \notin \{@_i(\theta \vee \chi), @_i\langle \mathsf{a}\rangle\varphi, @_i\langle @_j \mathsf{a}\alpha *_{\mathsf{c}} \beta\rangle\} \\ \{@_i\theta\} & \text{if } \varphi_{(n+1)} \text{ is } @_i(\theta \vee \chi) \text{ and } \Sigma_n \cup \{\varphi_{(n+1)}, @_i\theta\} \nvdash \psi \\ \{@_i\chi\} & \text{if } \varphi_{(n+1)} \text{ is } @_i(\theta \vee \chi) \text{ and } \Sigma_n \cup \{\varphi_{(n+1)}, @_i\theta\} \vdash \psi \\ \{@_i\langle \mathsf{a}\rangle j, @_j\varphi\} & \text{if } \varphi_{(n+1)} \text{ is } @_i\langle \mathsf{a}\rangle\varphi \\ & \quad \text{and } j \in \mathsf{Nom}'' \text{ does not appear in } \Sigma_n \cup \{\varphi_{(n+1)}\}. \\ \{@_j\langle \mathsf{a}\rangle k, @_i\langle @_k\alpha *_{\mathsf{c}} \beta\rangle\} & \text{if } \varphi_{(n+1)} \text{ is } @_i\langle @_j \mathsf{a}\alpha *_{\mathsf{c}} \beta\rangle \\ & \quad \text{and } k \in \mathsf{Nom}'' \text{ does not appear in } \Sigma_n \cup \{\varphi_{(n+1)}\}. \end{cases}$$

Define $\Sigma = \bigcup_{n\geq 0} \Sigma_n$. It is possible to prove by induction that $\Sigma \nvdash \psi$. The proof finishes if Σ is saturated. We prove only the cases $\langle\alpha\rangle$-pasted and $\langle *\rangle$-pasted.

($\langle\alpha\rangle$-pasted) Let $@_i\langle \mathsf{a}\rangle\varphi \in \Sigma$ and, w.l.o.g., $\varphi_{(n+1)} = @_i\langle \mathsf{a}\rangle\varphi$. It follows that, $\{@_i\langle \mathsf{a}\rangle j, @_j\varphi\} \subseteq \Sigma_{(n+1)} \subseteq \Sigma$ for j a nominal in $\mathsf{Nom}'' \setminus \mathsf{Nom}'$.

($\langle *\rangle$-pasted) Let $@_i\langle @_j \mathsf{a}\alpha *_{\mathsf{c}} \beta\rangle \in \Sigma$ and, w.l.o.g., $\varphi_{(n+1)} = @_i\langle @_j \mathsf{a}\alpha *_{\mathsf{c}} \beta\rangle$. It follows that, $\{@_i\langle \mathsf{a}\rangle j, @_i\langle @_k\alpha *_{\mathsf{c}} \beta\rangle\} \subseteq \Sigma_{(n+1)} \subseteq \Sigma$ for $j \in \mathsf{Nom}''$.

Lem. 1 enables us to build the model needed for proving completeness (Def. 9).

Definition 9 (Extracted Model). *Let $\{\mathsf{Nom}'_i\}_{i\in\mathbb{N}}$ be a family of pairwise disjoint denumerable sets of nominals. Moreover, let $\mathsf{Nom}^*_n = \bigcup_{i=1}^{n} \mathsf{Nom}'_i$; and $\mathsf{NE}^*_n = \mathsf{NE}(\mathsf{Nom} \cup \mathsf{Nom}^*_n)$. For every consistent set $\Gamma \subseteq \mathsf{NE}$; define*

$$\mathcal{M}_\Gamma = \langle M, \subseteq, \{\langle \mathfrak{A}_{\Gamma'}, \cong_{\Gamma'}\rangle\}_{\Gamma' \in M}\rangle$$

where: $\mathfrak{A}_{\Gamma'} = \langle N_{\Gamma'}, \{R^{\mathsf{a}}_{\Gamma'}\}_{\mathsf{a}\in\mathsf{Mod}}, \{\approx^{\mathsf{c}}_{\Gamma'}\}_{\mathsf{c}\in\mathsf{Cmp}}, g_{\Gamma'}, V_{\Gamma'}\rangle$ and

1. *$M = \{\, \Gamma' \subseteq \mathsf{NE}^*_n \mid n \in \mathbb{N}$ and $\Gamma \subseteq \Gamma'$ and Γ' is saturated$\}$;*
2. *for all $N_{\Gamma'} = \{\, i \mid i$ is a nominal appearing in $\Gamma' \,\}$;*
3. *for all $\cong_{\Gamma'} = \{\, (i, j) \mid @_i j \in \Gamma' \,\}$;*
4. *for all $R^{\mathsf{a}}_{\Gamma'} = \{\, (i, j) \mid @_i\langle \mathsf{a}\rangle j \in \Gamma' \,\}$;*
5. *for all $\approx^{\mathsf{c}}_{\Gamma'} = \{\, (i, j) \mid \langle @_i =_{\mathsf{c}} @_j\rangle \in \Gamma' \,\}$;*
6. *for all $V_{\Gamma'} : \mathsf{Prop} \to 2^{N_{\Gamma'}}$, it follows that $V_{\Gamma'}(p) = \{\, i \mid @_i p \in \Gamma' \,\}$; and*
7. *for all $g_{\Gamma'} : \mathsf{Nom} \to N_{\Gamma'}$, it follows that $g_{\Gamma'}(i) = i$.*

It can be checked that the structure in Def. 9 is a model in the sense of Def. 6. On this basis, we state the Truth Lemma 2 and the Completeness Theorem 1.

Lemma 2 (Truth Lemma). *Let $\mathcal{M}_\Gamma$ be as in Def. 9, it follows that:*

(1) $\mathcal{M}_\Gamma, \Gamma', i, j \Vdash \alpha$ *iff* $@_i\langle\alpha\rangle j \in \Gamma'$ (2) $\mathcal{M}_\Gamma, \Gamma', i \Vdash \varphi$ *iff* $@_i\varphi \in \Gamma'$.

Proof. The proof is by mutual induction on path and node expressions. The inductive hypotheses are

$$(\mathrm{IH}_1)\ \mathcal{M}_\Gamma, \Gamma', i, j \Vdash \alpha \text{ iff } @_i\langle\alpha\rangle j \in \Gamma'; \quad (\mathrm{IH}_2)\ \mathcal{M}_\Gamma, \Gamma', i \Vdash \varphi \text{ iff } @_i\varphi \in \Gamma'.$$

We prove the inductive case for $[\alpha =_\mathsf{c} \beta]$ as one of the most interesting. For this case, we need to prove $\mathcal{M}_\Gamma, \Gamma', i \Vdash [\alpha =_\mathsf{c} \beta]$ iff $@_i[\alpha =_\mathsf{c} \beta] \in \Gamma'$.

($\Rightarrow$) The proof proceeds by contradiction. Suppose: (a) $\mathcal{M}_\Gamma, \Gamma', i \Vdash [\alpha =_\mathsf{c} \beta]$ and (b) $@_i[\alpha =_\mathsf{c} \beta] \notin \Gamma'$. We prove (c) $\Gamma' \cup \{@_i\langle\alpha\rangle j, @_i\langle\beta\rangle k\} \nvdash @_i\langle @_j =_\mathsf{c} @_k\rangle$ for j, k arbitrary in $N_{\Gamma'}$. From not (c), i.e., $\Gamma' \cup \{@_i\langle\alpha\rangle j, @_i\langle\beta\rangle k\} \vdash @_i\langle @_j =_\mathsf{c} @_k\rangle$, we get $\Gamma' \vdash @_i((\langle\alpha\rangle j \wedge \langle\beta\rangle k) \rightarrow \langle @_j =_\mathsf{c} @_k\rangle)$; and using ([$*$]I) we obtain $\Gamma' \vdash @_i[\alpha =_\mathsf{c} \beta]$; this contradicts (b). Then, from Lem. 1,

$$\text{(d) exists } \Gamma'' \supseteq \Gamma' \cup \{@_i\langle\alpha\rangle j, @_i\langle\beta\rangle k\} \text{ s.t. } @_i\langle @_j =_\mathsf{c} @_k\rangle \notin \Gamma''.$$

The claim is: (d) contradicts (a). Suppose that exists such a Γ'', using (IH_1), we get that exists $\Gamma'' \supseteq \Gamma'$ and $\{j, k\} \subseteq N_{\Gamma''}$ s.t. $\mathcal{M}_\Gamma, \Gamma'', i, j \Vdash \alpha$, $\mathcal{M}_\Gamma, \Gamma'', i, k \Vdash \beta$, and $j \not\approx^\mathsf{c}_{\Gamma''} k$. This means that $\mathcal{M}_\Gamma, \Gamma', i \nVdash [\alpha =_\mathsf{c} \beta]$; which is a contradiction. Therefore, $\mathcal{M}_\Gamma, \Gamma', i \Vdash [\alpha =_\mathsf{c} \beta]$ implies $@_i[\alpha =_\mathsf{c} \beta] \in \Gamma'$.

($\Leftarrow$) Let $@_i[\alpha =_\mathsf{c} \beta] \in \Gamma'$. Proving $\mathcal{M}_\Gamma, \Gamma', i \Vdash [\alpha =_\mathsf{c} \beta]$ is equiv. to proving that for all $\Gamma'' \supseteq \Gamma'$ and all $\{j, k\} \subseteq N_{\Gamma''}$, if $\mathcal{M}_\Gamma, \Gamma'', i, j \Vdash \alpha$ and $\mathcal{M}_\Gamma, \Gamma'', i, k \Vdash \beta$, then, $j \approx^\mathsf{c}_{\Gamma''} k$. Let $\Gamma'' \supseteq \Gamma'$ and $\{j, k\} \subseteq N_{\Gamma''}$ be s.t. $\mathcal{M}_\Gamma, \Gamma'', i, j \Vdash \alpha$ and $\mathcal{M}_\Gamma, \Gamma'', i, k \Vdash \beta$. The proof is concluded if $j \approx^\mathsf{c}_{\Gamma''} k$, i.e., $\langle @_j =_\mathsf{c} @_k\rangle \in \Gamma''$. From (A) $\{@_i\langle\alpha\rangle j, @_i\langle\beta\rangle k\} \subseteq \Gamma''$. Since $\Gamma' \subseteq \Gamma''$, $@_i[\alpha =_\mathsf{c} \beta] \in \Gamma''$. Using ([$*$]E), we get $@_i\langle @_j =_\mathsf{c} @_k\rangle \in \Gamma''$. Thus, $j \approx^\mathsf{c}_{\Gamma''} k$.

Theorem 1 (Completeness). *$\Gamma \vDash \varphi$ implies $\Gamma \vdash \varphi$.*

Proof. We prove $\Gamma \nvdash \varphi$ implies $\Gamma \nvDash \varphi$. Let $\mathcal{M}_\Gamma$ be as in Def. 9. From Lem. 1 we know that exists $\Gamma' \supseteq \Gamma$ s.t. $\Gamma' \in \mathcal{M}_\Gamma$ and $\varphi \notin \Gamma'$. From Lem. 2, it is clear that for some nominal $i \in N_{\Gamma'}$, $\mathcal{M}_\Gamma, \Gamma', i \Vdash \Gamma$ and $\mathcal{M}_\Gamma, \Gamma', i \nVdash \varphi$. This proves $\Gamma \nvDash \varphi$.

5 Extended Axiomatic Systems

In this section we briefly cover extensions of IHXPath$_=$ with *pure axioms* and *existential saturation rules*, a family of axioms and rules that enables us to extend our strong completeness result for a wider family of logics. These ideas make use of the ability of hybrid logics to fully internalize the first-order conditions that are needed to characterize many interesting frame classes. Similarly to what is done in [14,6], if we add pure axioms and existential saturation rules into the axiom system for IHXPath$_=$, the completeness proof in the previous section automatically yields strong completeness for the extended axiom systems, with respect to their respective classes of models.

Standard Translation. We will define a *standard translation* of the language of IHXPath$_=$ into the language of Intuitionistic FOL with equality (IFOL). This translation is needed to characterize the frame conditions that the new axioms and rules define. The semantics of IFOL, can be found, e.g., in [21].

Table 3: Standard Translation into IFOL

Propositional	Paths
$\mathsf{ST}'_x(\bot) = \bot$	$\mathsf{ST}'_x(\langle\alpha\rangle\varphi) = \exists y(\mathsf{ST}'_{x,y}(\alpha) \wedge \mathsf{ST}'_y(\varphi))$
$\mathsf{ST}'_x(p) = p(x)$	$\mathsf{ST}'_x([\alpha]\varphi) = \forall y(\mathsf{ST}'_{x,y}(\alpha) \rightarrow \mathsf{ST}'_y(\varphi))$
$\mathsf{ST}'_x(i) = x = x_i$	$\mathsf{ST}'_{x,y}(\mathsf{a}) = a(x,y)$
$\mathsf{ST}'_x(\varphi \vee \psi) = \mathsf{ST}'_x(\varphi) \vee \mathsf{ST}'_x(\psi)$	$\mathsf{ST}'_{x,y}(@_i) = y = x_i$
$\mathsf{ST}'_x(\varphi \wedge \psi) = \mathsf{ST}'_x(\varphi) \wedge \mathsf{ST}'_x(\psi)$	$\mathsf{ST}'_{x,y}([\varphi]) = (x = y) \wedge \mathsf{ST}'_y(\varphi)$
$\mathsf{ST}'_x(\varphi \rightarrow \psi) = \mathsf{ST}'_x(\varphi) \rightarrow \mathsf{ST}'_x(\psi)$	$\mathsf{ST}'_{x,y}(\alpha\beta) = \exists z(\mathsf{ST}'_{x,z}(\alpha) \wedge \mathsf{ST}'_{z,y}(\beta))$
Comparisons	
$\mathsf{ST}'_x(\langle\alpha =_{\mathsf{c}} \beta\rangle) = \exists y \exists z(\mathsf{ST}'_{x,y}(\alpha) \wedge \mathsf{ST}'_{x,z}(\beta) \wedge c(y,z))$	
$\mathsf{ST}'_x(\langle\alpha \neq_{\mathsf{c}} \beta\rangle) = \exists y \exists z(\mathsf{ST}'_{x,y}(\alpha) \wedge \mathsf{ST}'_{x,z}(\beta) \wedge c(y,y) \wedge c(z,z) \wedge \neg c(y,z))$	
$\mathsf{ST}'_x([\alpha =_{\mathsf{c}} \beta]) = \forall y \forall z(\mathsf{ST}'_{x,y}(\alpha) \wedge \mathsf{ST}'_{x,z}(\beta) \rightarrow c(y,z))$	
$\mathsf{ST}'_x([\alpha \neq_{\mathsf{c}} \beta]) = \forall y \forall z(\mathsf{ST}'_{x,y}(\alpha) \wedge \mathsf{ST}'_{x,z}(\beta) \rightarrow (c(y,y) \wedge c(z,z) \wedge \neg c(y,z)))$	
Partial Equality	
$\sigma_{\mathsf{c}} = \forall x \forall y(c(x,y) \rightarrow c(x,y))$	$\tau_{\mathsf{c}} = \forall x \forall y \forall z(c(x,y) \wedge c(y,z) \rightarrow c(x,z))$

Definition 10. *Fix an alphabet of denumerable sets of: unary predicate symbols (*P*), binary relation symbols (*R*), constant symbols (*Con*),and variable symbols (*Var*). The language of IFOL is defined by the grammar:*

$$\varphi, \psi := p(t) \mid r(t,t') \mid \bot \mid t = t' \mid \varphi \vee \psi \mid \varphi \wedge \psi \mid \varphi \rightarrow \psi \mid \exists x\varphi \mid \forall x\varphi,$$

where t, t' are terms –i.e., symbols for constants or variables, $p \in \mathsf{P}$, and $r \in \mathsf{R}$.

Def. 11 establishes a correspondence from the language of IHXPath$_=$ to that of IFOL. Notice that symbols in Prop and P are in a one to one correspondence; that we map each $\mathsf{a} \in \mathsf{Mod}$ to a unique $a \in \mathsf{Rels}$; and that $x, y, z, \ldots$ are symbols in Var. Prop. 3 indicates that such a correspondence preserves satisfiability.

Definition 11. *The* standard translation *of a node expression φ is defined as:* $\mathsf{ST}_x(\varphi) = \left(\bigwedge_{\mathsf{c} \in \mathsf{Cmp}}(\sigma_{\mathsf{c}} \wedge \tau_{\mathsf{c}})\right) \wedge \mathsf{ST}'_x(\varphi)$*; where σ_{c}, τ_{c}, and ST'_x are as in Tab. 3.*

Proposition 3. $\mathcal{M}, m, n \Vdash \varphi$ *iff* $\mathcal{M}, m \Vdash \mathsf{ST}_x(\varphi)[x \mapsto w, x_i \mapsto g_m(i)]$.

Pure Axioms and Existential Saturation Rules. We introduce pure axioms and existential saturation rules, and their associated frame conditions. Briefly, pure axioms are formulas that use no proposition symbols; and existential saturation rules are instantiations of first-order formulas with a $\forall\exists$ quantification pattern. These axioms and rules are incorporated to the axiom system from Sec. 4, which contains *unorthodox inference rules*, i.e., rules with side conditions. As discussed in [14,6], unorthodox rules are crucial to obtain a general completeness result.

Notation. We use $\boldsymbol{i}_n$ to indicate a sequence $i_1 \ldots i_n$ of nominals; and $\varphi(\boldsymbol{i}_n)$ to indicate a node expression with no proposition symbols and with nominals in $\boldsymbol{i}_n$. Lastly, we use $Q\boldsymbol{i}_n\varphi$ to indicate $Qx_{i_1} \ldots Qx_{i_n}\varphi$ for $Q \in \{\forall, \exists\}$.

Definition 12. *A node expression is called* pure *iff it has no proposition symbols. A rule of the form*

$$\frac{\varphi(\boldsymbol{i}_n\boldsymbol{j}_m) \to \psi}{\psi}\ (\rho)$$

is called an existential saturation rule *iff $\boldsymbol{i}_n$ and $\boldsymbol{j}_m$ are disjoint sequences of nominals, and the nominals in ψ are not in $\boldsymbol{j}_m$. We use $\mathrm{hd}(\rho)$ to indicate $\varphi(\boldsymbol{i}_n\boldsymbol{j}_m)$. Sets Π and P of pure expressions and existential saturation rules, respectively, define a* frame condition $\mathsf{FC}(\Pi \cup \mathrm{P})$ *defined as:*

$$\bigwedge\{\, \forall x \forall \boldsymbol{i}_n (\mathsf{ST}_x(\varphi(\boldsymbol{i}_n))) \mid \varphi(\boldsymbol{i}_n) \in \Pi \,\} \wedge \bigwedge\{\, \forall x \forall \boldsymbol{i}_n \exists \boldsymbol{j}_m (\mathsf{ST}_x(\mathrm{hd}(\rho))) \mid \rho \in \mathrm{P} \,\}.$$

Extending the axiom system of IHXPath$_=$ with pure axioms and existential saturation forces us to revise the definition of saturation and the saturation lemma in Def. 8 and Lem. 1; necessary for completeness. This is done in Def. 13 and Lem. 3, respectively. Thm. 2 states the generalized completeness theorem.

Definition 13 (P-saturated). *Let $\mathsf{Nom}' \subset \mathsf{Nom}''$ and P a set of existential saturation rules; $\Gamma'' \subseteq \mathsf{NE}''$ is* P-saturated *iff Γ'' is* saturated *and*

for all $\rho \in \mathrm{P}$ and all nominals $\boldsymbol{k}_n \in \Gamma''$,
exists $\boldsymbol{l}_m \subseteq (\mathsf{Nom}'' \setminus \mathsf{Nom}')$ s.t. $\varphi[\boldsymbol{i}_n\boldsymbol{j}_m/\boldsymbol{k}_n\boldsymbol{l}_m] \in \Gamma''$ –where $\varphi(\boldsymbol{i}_n\boldsymbol{j}_m) = \mathrm{hd}(\rho)$.

Lemma 3 (P-saturation Lemma). *Let $\mathsf{Nom}' \subset \mathsf{Nom}''$; Π be a set of pure axioms and P a set of existential saturation rules. Moreover, let $\Gamma' \subseteq \mathsf{NE}'$ be s.t. $\Gamma' \nvdash \psi$. Exists $\Gamma'' \subseteq \mathsf{NE}''$ s.t.: (1) $\Gamma' \subseteq \Gamma''$; (2) Γ'' is P-saturated; and (3) $\Gamma'' \nvdash \psi$.*

Theorem 2. *Let Π and P be sets of pure axioms and existential saturation rules, respectively. The axiomatic system obtained by extending that from Tabs. 1 and 2 with Π and P as additional axioms and rules, is strongly complete w.r.t. the class of models whose frames satisfy* $\mathsf{FC}(\Pi \cup \mathrm{P})$.

6 Final Remarks

We presented a logic –IHXPath$_=$– that provides an intuitionistic reading of XPath with (in)equality checks for attribute values in the presence of partial data value functions and updates. For this logic, we first identified suitable notions of concrete and abstract models, and characterized a certain class of update functions. This lead us to the definition of abstract partial update structures, which became the models of our logic. To our knowledge, this is the first approach for dealing with incomplete information based on partial orders between data graphs. Other attempts to address similar problems use completely different ideas, such as those presented in [12,26,2]. Moreover, by defining a suitable notion of updates (defining certain cases in which new information can be added to a graph database) we discovered a novel link between dynamic data updates and intuitionistic logic.

We provided an axiomatization and a strong completeness result. Moreover, we showed that our system preserves strong completeness when extended with pure axioms and existential saturation rules (with respect to the corresponding class of models). These extensions allow to characterize several interesting classes of models (cf, e.g., [14,6]).

Much remains to be done. For instance, besides (in)equalities, we would like to explore comparison operators such as less than, greater than, etc. Moreover, we would like to study this logic from a purely dynamic logic perspective, *à la* [29,10,7]. It would also be interesting to characterize particular classes of models (e.g., the case were the accessibility relations define trees). Other model theoretic questions should also be addressed, e.g., defining a proper notion of bisimulation that captures the expressive power of the logic. Finally, decidability and complexity of different reasoning tasks (e.g., model-checking, satisfiability) should be established. We conjecture, e.g., that the notions of filtration defined in [6] for $\text{HXPath}_=$, might be extended and adapted to $\text{IHXPath}_=$ to prove decidability of the satisfiability problem, together with upper-bounds to its complexity. These are some initial thoughts that deserve further exploration.

Acknowledgments. We thank the reviewers for their valuable comments. Our work is supported by the Laboratoire International Associé SINFIN, the EU Grant Agreement 101008233 (MISSION), the ANPCyT projects PICT-2020-3780, PICT-2021-00400, PICT-2021-00675, and PICTO-2022-CBA-00088, and the CONICET projects PIBAA-28720210100428CO, PIBAA-28720210100165CO, and PIP-11220200100812CO.

References

1. Abriola, S., Barceló, P., Figueira, D., Figueira, S.: Bisimulations on data graphs. Journal of Artificial Intelligence Research **61**, 171–213 (2018)
2. Abriola, S., Cifuentes, S., Martinez, M., Pardal, N., Pin, E.: An epistemic approach to model uncertainty in data-graphs. International Journal of Approximate Reasoning **160**, 108948 (2023)
3. Abriola, S., Descotte, M.E., Fervari, R., Figueira, S.: Axiomatizations for downward XPath on data trees. Journal of Computer and System Sciences **89**, 209–245 (2017)
4. Angles, R., Gutiérrez, C.: Survey of graph database models. ACM Computing Surveys **40**(1), 1:1–1:39 (2008)
5. Areces, C., ten Cate, B.: Hybrid logics. In: Handbook of Modal Logic, pp. 821–868. Elsevier (2006)
6. Areces, C., Fervari, R.: Axiomatizing hybrid XPath with data. Logical Methods in Computer Science **17**(3) (2021)
7. Areces, C., Fervari, R., Hoffmann, G.: Relation-changing modal operators. Logic Journal of the IGPL **23**(4), 601–627 (2015)
8. Areces, C., Fervari, R., Seiler, N.: Tableaux for hybrid XPath with data. In: 18th EPIA Conference on Artificial Intelligence (EPIA 2017). LNCS, vol. 10423, pp. 611–623. Springer (2017)
9. Arenas, M., Fan, W., Libkin, L.: On verifying consistency of XML specifications. In: 21st ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems (PODS'02). pp. 259–270. ACM (2002)

10. Aucher, G., van Benthem, J., Grossi, D.: Modal logics of sabotage revisited. Journal of Logic and Computation **28**(2), 269–303 (2018)
11. Baelde, D., Lunel, S., Schmitz, S.: A sequent calculus for a modal logic on finite data trees. In: 25th EACSL Annual Conference on Computer Science Logic (CSL 2016). LIPIcs, vol. 62, pp. 32:1–32:16. Schloss Dagstuhl (2016)
12. Barceló, P., Libkin, L., Reutter, J.L.: Querying regular graph patterns. Journal of the ACM **61**(1), 8:1–8:54 (2014)
13. Blackburn, P., van Benthem, J., Wolter, F.: Handbook of Modal Logic. Elsevier (2006)
14. Blackburn, P., ten Cate, B.: Pure extensions, proof rules, and hybrid axiomatics. Studia Logica **84**(2), 277–322 (2006)
15. Blackburn, P., de Rijke, M., Venema, Y.: Modal Logic, Cambridge Tracts in Theoretical Computer Science, vol. 53. Cambridge University Press (2001)
16. Bojańczyk, M., Muscholl, A., Schwentick, T., Segoufin, L.: Two-variable logic on data trees and XML reasoning. Journal of the ACM **56**(3) (2009)
17. Braüner, T.: Hybrid Logics and its Proof-Theory, Applied Logics Series, vol. 37. Springer (2011)
18. ten Cate, B., Fontaine, G., Litak, T.: Some modal aspects of XPath. Journal of Applied Non-Classical Logics **20**(3), 139–171 (2010)
19. ten Cate, B., Litak, T., Marx, M.: Complete axiomatizations for XPath fragments. Journal of Applied Logic **8**(2), 153–172 (2010)
20. Clark, J., DeRose, S.: XML path language (XPath). Website (1999), W3C Recommendation, url = http://www.w3.org/TR/xpath
21. van Dalen, D.: Logic and structure. Universitext, Springer, 5 edn. (2013)
22. Figueira, D.: Reasoning on Words and Trees with Data. PhD thesis, Laboratoire Spécification et Vérification, ENS Cachan, France (2010)
23. Figueira, D.: Decidability of downward XPath. ACM Transactions on Computational Logic **13**(4), 34 (2012)
24. Goldblatt, R.: An abstract setting for Henkin proofs. Topoi **3**(1), 37–41 (1984)
25. Gottlob, G., Koch, C., Pichler, R.: Efficient algorithms for processing XPath queries. ACM Transactions on Database Systems **30**(2), 444–491 (2005)
26. Grabon, M., Michaliszyn, J., Otop, J., Wieczorek, P.: Querying data graphs with arithmetical regular expressions. In: 25th International Joint Conference on Artificial Intelligence (IJCAI 2016). pp. 1088–1094. IJCAI/AAAI Press (2016)
27. Libkin, L., Martens, W., Vrgoč, D.: Querying graphs with data. Journal of the ACM **63**(2), 14:1–14:53 (2016)
28. Libkin, L., Vrgoč, D.: Regular path queries on graphs with data. In: International Conference on Database Theory (ICDT'12). pp. 74–85. ACM (2012)
29. Plaza, J.: Logics of public communications. Synthese **158**(2), 165–179 (2007)
30. Robinson, I., Webber, J., Eifrem, E.: Graph Databases. O'Reilly Media, Inc. (2013)
31. Schinner, A.: The Voynich manuscript: Evidence of the hoax hypothesis. Cryptologia **31**(2), 95–107 (2007)