Tableaux for Reasoning About Atomic Updates

Christian G. Fermüller Georg Moser Richard Zach

Technische Universität Wien, Austria
{chrisf,moser,zach}@logic.at

Abstract. A simple model of dynamic databases is studied from a modal logic perspecitve. A state α of a database is an atomic update of a state β if at most one atomic statement is evaluated differently in α compared to β. The corresponding restriction on Kripke-like structures yields so-called update logics. These logics are studied also in a many-valued context. Adequate tableau calculi are given.

1 Introduction

Various approaches employing modal logics for the representation of knowledge and for (mechanized) reasoning about data have been investigated. See, e.g., [3, 15, 6, 10, 9] for some recent work of relevance to database theory.

Here we investigate a particularly simple model of (dynamic) databases. The states of a database are identified with assignments of truth values to basic propositions. Some states are considered as results of *updating* other states of the database. In other words, a binary *update relation* is defined over the set of possible states. This amounts to defining usual Kripke interpretations. Standard normal modal logics arise if we augment classical propositional logic (over the signature of basic propositions of the database) with the modalities $\Box$ and $\Diamond$, interpreted as "in all updated states" and "in some updated state", respectively. However, as we shall see below, interesting deviations from standard modal logics are needed to model atomic, i.e. stepwise, updates instead of arbitrary ones.

Literature on the so-called "update problem" usually aims at formalizing changes in databases triggered by *arbitrary complex* changes in the environment to which the database refers. Here however, we want to model only *atomic* or *"single-step" updates*[1]. More exactly, each update operation is assumed to change the truth value of at most one basic proposition at a time. In general, atomic updates reflect adaption to a changing environment (or improved knowledge) only via *sequences* of such atomic update operations. However, we think that considering *atomic* updates leads to a more realistic model of the actual computational behavior of dynamic databases. At a fundamental level the evolution of any database proceeds in basic steps, each of which corresponds to some well defined atomic action that can be performed on a database entry. We aim at a conceptually clear as well as technically simple logical model of this aspect of dynamic databases.

[1] What we call "atomic update" here was called "single-step updates" in a previous — unpublished — version of this paper by the first author. This preliminary version of the paper is accessible at http://www.cin.ufpe.br/~wollic/wollic2000/proceedings/.

Nieuwenhuis, Robert and Andrei Voronkov (eds.), *Logic for Programming, Artificial Intelligence and Reasoning. 8th International Conference, LPAR 2001.* Proceedings. LNAI 2250. Springer, Berlin, 2001. pp. 635–649

The subtle constraint on the update relation seems to have dramatic effects for the corresponding modal logics: the set of formulas valid in all corresponding Kripke interpretations is not closed under substitution. In response to this fact we propose to use a *two-sorted* propositional language that allows us to distinguish between "atoms" (basic propositions of the database) and genuine propositional variables; and consequently between "concrete" and "schematic" statements about data. We define a corresponding semantics and provide complete and sound *tableau calculi* for the resulting logics.

We generalize this model of atomic dynamic databases to scenarios allowing for incomplete and inconsistent information. Replacing classical logic by Belnap's well-known four-valued logic [4] opens the space for new types of modal operators over corresponding update models. Some examples of such *distribution modalities* expressing properties of updates will be investigated. We claim that in general the concept of distribution modalities is a versatile tool to model a broad range of updates in (dynamic) databases. A variant of tableaux for finite-valued logics with distribution modalities introduced in [7] turns out to be adequate for formalizing reasoning in corresponding logics.

We emphasize that the concepts and results presented here should be considered only as a first step in exploring the scope and limits of many-valued Kripke structures and distribution modalities in the context of reasoning about dynamic databases. Accordingly, we conclude with a list of future topics of research.

2 Atomic databases and Kripke interpretations

Our first object of investigation is arguably the simplest logical model of a database. It refers to a fixed set of atomic units of information (propositions, called atoms) and presumes that the only information explicitly contained in the database is which of those atomic propositions hold and which do not hold.

More formally, by a *(classical) state (of a database)* we mean a total function of type $\mathsf{atoms} \mapsto \{\mathbf{t},\mathbf{f}\}$, where atoms is a non-empty, countable set of propositional atoms. Obviously we can evaluate classical propositional formulas over the signature atoms (and standard connectives) with respect to a state α as usual:

- $v_\alpha(p) = \alpha(p)$, for $p \in \mathsf{atoms}$
- $v_\alpha(\top) = \mathbf{t}$ and $v_\alpha(\bot) = \mathbf{f}$
- $v_\alpha(\neg A) = \mathbf{t}$ iff $v_\alpha(A) = \mathbf{f}$
- $v_\alpha(A \circ B) = \widetilde{\circ}(v_\alpha(A), v_\alpha(B))$

where $\circ \in \{\wedge, \vee, \supset, \equiv\}$ and $\widetilde{\circ}$ is the classical boolean function associated with the binary connective $\circ$. In other words, a *query* is an arbitrary propositional formula A over atoms, which receives the answer $v_\alpha(A)$ if the database is in state α.

We are interested in the dynamic structure of a database; i.e., the possible transitions from states to states triggered by update operations. As explained above, we focus on the—arguably—most elementary type of an update operation: A single application of such an update operation changes the truth status of at most one atomic unit of information. Correspondingly, state α' is called an *atomic update* of a state α if the following condition is satisfied:

(au) $\alpha(p) \neq \alpha'(p)$ for *at most one* $p \in \mathsf{atoms}$.

Throughout the paper we will consider *atomic* updates *only*, and therefore often drop the adjective "atomic."

Definition 1. *An* (atomic-)update model *is a pair* $\mathcal{D} = (\Sigma, U)$ *where*

- Σ *is a set of states of a database over (a fixed set)* atoms*, i.e., a set of functions of type* $\mathsf{atoms} \mapsto \{\mathbf{t}, \mathbf{f}\}$*, and*
- U *is a binary relation over* Σ*, subject to the restriction that* $\forall \alpha, \alpha' \in \Sigma\colon \alpha U \alpha'$ *implies that* α' *is an atomic update of* α*.*

We extend the expressibility of the query language by adding to it the modal operators $\Box$ and $\Diamond$, with the intended meaning "in all (reachable) updates" and "in some (reachable) update", respectively. More exactly—referring to states α of a atomic-update model $\mathcal{D} = (\Sigma, U)$—we extend the definition of v_α as follows:

- $v^{\mathcal{D}}_\alpha(\Box A) = \mathbf{t}$ iff $\forall \beta \in \Sigma$: if $\alpha U \beta$, then $v^{\mathcal{D}}_\beta(A) = \mathbf{t}$, and
- $v^{\mathcal{D}}_\alpha(\Diamond A) = \mathbf{t}$ iff $\exists \beta \in \Sigma\colon \alpha U \beta$ and $v^{\mathcal{D}}_\beta(A) = \mathbf{t}$.

This simple logical machinery allows for the expression of statements that refer not only to the current state of a database but also to possible updates of a states.

Example 2. The formula $A \supset \Box A$ may be paraphrased as "If the statement A is currently validated by the database, then all possible (atomic) updates will still validate A". Similarly $\Diamond A \wedge \Diamond \neg A$ expresses that A is contingent, i.e., a statement that will be evaluated differently in different possible updates of the current state of the database. Likewise we can express the fact that there is no possible update of the current state by "$\neg\Diamond\top$". The statement that for every possible update (of the current state) a further update is possible is expressed by "$\Box\Diamond\top$."

There is a close connection between states of a database and worlds of a Kripke structure where a world β is accessible from a world α iff β is an atomic update of α.

Definition 3. *A* (Kripke) interpretation *is a triple* $\mathcal{M} = (W, R, V)$ *where*

- W *is a non-empty* set of worlds,
- R *is a binary* accessibility relation *on* W,
- $V : \mathsf{PV} \times W \mapsto \{\mathbf{t}, \mathbf{f}\}$ *is a* truth value assignment *to the infinite set* PV *of* propositional variables.

The corresponding evaluation function $v^{\mathcal{M}}$ *that assigns a truth value to each formula* A *in each world* $w \in W$ *is defined as usual.* $\mathcal{M}$ *is a* (counter-)interpretation *for a formula* A *if* $v^{\mathcal{M}}(A, w) = \mathbf{t}$ ($\mathbf{f}$) *for some* $w \in W$. A *is* valid *in* $\mathcal{M}$ *if* $\mathcal{M}$ *is not a counter-interpretation for* A*; i.e., if* $v^{\mathcal{M}}(A, w) = \mathbf{t}$ *for all* $w \in W$.

Definition 4. *The* skeleton $T(\mathcal{M})$ *of an interpretation* $\mathcal{M} = (W, R, V)$ *is the undirected graph with* W *as set of nodes and an edge between* $v, w \in W$ *iff* $v \neq w$ *and either* vRw *or* wRv*. We call an interpretation* $\mathcal{M}$ tree-like *if its skeleton* $T(\mathcal{M})$ *is a tree (i.e., a connected acyclic graph).*

Clearly, condition (**au**) corresponds to condition

(au′) *wRv* implies $V(p,v) \neq V(p,w)$ for *at most one* $p \in \mathsf{PV}$.

for Kripke interpretations. We say that an interpretation $\mathcal{M} = (W,R,V)$ fulfills condition (**au′**) *on a subset* $P \subseteq \mathsf{PV}$ if for all $v,w \in W$: if *wRv* then $V(p,v) = V(p,w)$ for all except at most one $p \in P$.

If a Kripke interpretation $\mathcal{M} = (W,R,V)$ satisfies (**au′**) for all its worlds then it corresponds to a (unique) update model $\mathcal{D}_{\mathcal{M}} = (\Sigma,U)$, where atoms is identified with PV, $\Sigma = \{\lambda p[V(p,w)] \mid w \in W\}$, and $\alpha U \beta \Longleftrightarrow vRw$ where $\alpha = \lambda p[V(p,v)]$ and $\beta = \lambda p[V(p,w)]$. Conversely, every update model $\mathcal{D}$ corresponds to Kripke interpretation $\mathcal{M}_{\mathcal{D}}$ that satisfies (**au′**).

By requiring the update relation in a model to fulfill simple properties we can adapt our model to databases which obey certain dynamic constraints. For instance, requiring the update relation U to be *symmetric* corresponds to modeling databases for which every update is reversible. Similarly in many applications it will be useful to require U to be *reflexive* (corresponding to: the "empty" update operation is always applicable) or *serial* (corresponding to: every state can be updated). Observe however that, e.g., *transitivity* does not in general make sense for atomic updates: the atomic update of an atomic update is not expected to be atomic itself.

Definition 5. *The class of all update models is called* update K-models. *An update model is called an* update KB-, D-, T-, *or* TB-model *if its update relation is symmetric, serial, reflexive, or symmetric and reflexive, respectively.*

3 Concrete versus schematic statements

It might seem as if—so far—we have only described just another view of normal modal logic. However, by insisting that the truth value of at most *one* atom can be changed in one update operation we ensured that, e.g., the formula

$$F = (p \wedge q) \supset \Box(p \vee q)$$

is evaluated true in all states of all models, if p and q are different atoms. By contrast, substituting (in F) p for q results in a formula which is false in all states in which p is **t** and where there is an atomic update in which p is **f**. In other words, the set of formulas true in all states of an update model is in general *not closed under substitution*.

There is a simple way of recovering closure under substitution:

Definition 6. *A formula A is* schematically valid *in an (atomic-)update model* $\mathcal{D} = (\Sigma,U)$ *if* $v^{\mathcal{D}}_{\alpha}(A') = \mathbf{t}$ *for all substitution instances* A' *of A and all* $\alpha \in \Sigma$.

The set of formulas that are schematically valid in all update Λ*-models is called* update-Λ *(for* $\Lambda \in \{\mathsf{K}, \mathsf{KB}, \mathsf{D}, \mathsf{T}, \mathsf{TB}\}$*).*

By definition, each update-Λ is closed under substitution. It is easy to see that they are also closed under modus ponens and the necessity rule. Therefore they can be considered as ordinary modal logics and can be directly compared to the corresponding

standard logics (which we identify with the sets of formulas valid in all corresponding Kripke interpretations).

Our first main result is the following

Theorem 7. *For* $\Lambda \in \{\mathsf{K}, \mathsf{KB}, \mathsf{D}, \mathsf{T}, \mathsf{TB}\}$, *update-*$\Lambda$ *and (ordinary)* Λ *coincide.*

Proof. If a formula is valid in all Λ-interpretations then, in particular, it is valid in all Λ-interpretations satisfying condition (**au**′) for all worlds. Since such interpretations correspond to update models it follows that $\Lambda \subseteq$ update-Λ.

For the converse we prove the following:

Claim. Every tree-like Λ-interpretation $\mathcal{M} = (W, R, V)$ can be transformed into a Λ-interpretation $\mathcal{M}' = (W, R, V')$ such that condition (**au**′) is satisfied and for all $w \in W$: $v^{\mathcal{M}}(A, w) = v^{\mathcal{M}'}(A\theta, w)$ for all formulas A and some substitution θ.

The claim implies that if $\mathcal{M}$ is a counter-interpretation for A then $\mathcal{D}_{\mathcal{M}'}$ is a counter-update-model for $A\theta$. It is a consequence of the usual tableau-based completeness proofs for $\Lambda \in \{\mathsf{K}, \mathsf{B}, \mathsf{D}, \mathsf{T}, \mathsf{TB}\}$, that without loss of generality a (counter)-Λ-interpretation for any formula A may be assumed to be tree-like (Recall Definition 4 and see, e.g., [8], but also Theorem 14 below.) Therefore update-$\Lambda \subseteq \Lambda$ follows from the claim.

To establish the claim, consider the skeleton $T(\mathcal{M})$ of $\mathcal{M}$ and define

$$\mathrm{diff}_V^P(v, w) = |\{p \in P \mid V(p, v) \neq V(p, w)\}|$$

for each edge (v, w) in $T(\mathcal{M})$. Obviously, if $\mathrm{diff}_V^{\mathsf{PV}}(v, w) \leq 1$ for all edges (v, w) then $\mathcal{M}$ satisfies the atomic update condition and nothing is left to prove. Since only finitely many propositional variables can occur in a single formula A we restrict our attention to the assignments in $\mathcal{M}$ of a finite subset P of PV; more exactly we assume that—at the beginning of our construction—$V(p, w) = \mathbf{t}$ for all $w \in W$ and all $p \in \mathsf{PV} - P$.

Let $\mathrm{diff}_V^P(v, w) > 1$; then there are two different propositional variables $p, q \in P$ such that $V(p, v) \neq V(p, w)$ and $V(q, v) \neq V(p, w)$. We set

$$\theta = \{p \leftarrow (e \equiv f), q \leftarrow (f \equiv g)\}$$

for pairwise different variables $e, f, g \notin P$. We now update the truth value assignment V of $\mathcal{M}$ to an assignment V' such that the following three conditions are satisfied:

1. $\mathrm{diff}_V^{P'}(v, w) < \mathrm{diff}_{V'}^{P'}(v, w)$, where $P' = P \cup \{e, f, g\}$,
2. $\mathrm{diff}_V^{\mathsf{PV}}(u, u') \leq \mathrm{diff}_{V'}^{\mathsf{PV}}(u, u')$ for all edges (u, u') in $T(\mathcal{M})$,
3. $v_{\mathcal{M}'}(A\theta, u) = v_{\mathcal{M}}(A, u)$ for all $u \in W$ and all formulas A built up from variables in P, where $\mathcal{M}' = (W, R, V')$.

We start by assigning appropriate truth values to e, f, g in v and w. Without loss of generality, we may assume that either

(a) $V(p, v) = V(q, v) = \mathbf{t}$ and $V(p, w) = V(q, w) = \mathbf{f}$, or
(b) $V(p, v) = V(q, w) = \mathbf{t}$ and $V(p, w) = V(q, v) = \mathbf{f}$.

In case (a) we set $V'(e,v) = V'(f,v) = V'(g,v) = V'(e,w) = V'(g,w) = \mathbf{t}$ and $V'(f,w) = \mathbf{f}$. In case (b) we set $V'(e,v) = V'(f,v) = V('e,w) = \mathbf{t}$ and $V'(g,v) = V'(f,w) = V'(g,w) = \mathbf{f}$. In both cases condition 1 is satisfied and $v_{\mathcal{M}'}(A\theta,u) = v_{\mathcal{M}}(A,u)$ for $u \in \{v,w\}$.

Observe that p and q are not relevant for evaluating $A\theta$; we may thus set $V'(p,u) = V'(q,u) = \mathbf{t}$ in all worlds $u \in W$.

The assignment of truth values to e, f, g in worlds u distinct from v and w is defined by induction on the distance $d(u)$ to the world v in $T(\mathcal{M})$; where $d(u)$ is defined as the minimal number of edges in a sequence $u, u_1, \ldots, u_k, v$ of adjacent nodes. The induction hypothesis is:

(IH) Conditions 2 and 3, above, are satisfied if we only consider the worlds $u \in W$ for which $d(u) \leq n$.

(IH) trivially holds for $n = 1$.

Let u' be a world with $d(u') = n+1$. Since $T(\mathcal{M})$ is a tree there is a unique u with (u',u) in $T(\mathcal{M})$ and $d(u) = n$. By induction hypothesis, we have already defined an appropriate assignment to e, f, g in u. To find the appropriate truth values for e, f, g in u' we distinguish the following cases.

(1) $V(p,u) = V(q,u) = \mathbf{t}$. (IH) leaves two possibilities for V' with respect to e, f, g in u:
  (1.1) $V'(e,u) = V'(f,u) = V'(g,u) = \mathbf{t}$. We set $V'(e,u') = V(p,u')$ and $V'(f,u') = \mathbf{t}$ and $V'(g,u') = V(q,u')$.
  (1.2) $V'(e,u) = V'(f,u) = V'(g,u) = \mathbf{f}$. We set $V'(e,u') = \widetilde{\neg} V(p,u')$ and $V'(f,u') = \mathbf{f}$ and $V'(g,u') = \widetilde{\neg} V(q,u')$.
(2) $V(p,u) = \mathbf{t}$ and $V(q,u) = \mathbf{f}$. Again, (IH) leaves two possibilities:
  (2.1) $V'(e,u) = V'(f,u) = \mathbf{t}$ and $V'(g,u) = \mathbf{f}$. V' is like in case (1.1).
  (2.2) $V'(e,u) = V'(f,u) = \mathbf{f}$ and $V'(g,u) = \mathbf{t}$. V' is like in case (1.2).
(3) $V(p,u) = \mathbf{f}$ and $V(q,u) = \mathbf{t}$. Like case (2), except for swapping $\mathbf{t}$ and $\mathbf{f}$ in the assignments to f in u'.
(4) $V(p,u) = \mathbf{f}$ and $V(q,u) = \mathbf{f}$. Like case (1), except for swapping $\mathbf{t}$ and $\mathbf{f}$ in the assignments to f in u'.

In all cases it easy to check that (IH) holds for $n+1$ after the described adjustments. Therefore the construction eliminates the particular counter-example to (**au**$'$) without introducing a new one. The whole construction is repeated for each pair (x,y) of adjacent worlds where $\mathrm{diff}^{\mathsf{PV}}_{V^*}(x,y) > 1$ until (**au**$'$) is satisfied. (V^* is the respective valuation from the previous step.) □

Remark 8. As is to be expected from the intended semantics of *atomic* updates update-Λ=Λ *does not hold* in general if Λ is a logic for which the accessibility relation is *transitive*. E.g., one can check that the formula

$$F = (p \wedge q) \supset [\Diamond\Box\bot \vee \Box(\neg p \vee q) \vee \Diamond\Diamond(p \vee q)]$$

is schematically valid in all atomic-update models with transitive update relation. However, it is easy to construct a (Kripke) counter-interpretation with transitive accessibility

relation for F. (Modulo obvious augmentations of Definitions 5 and 6) this fact can be expressed as update-$\mathsf{K4} \neq \mathsf{K4}$. Similarly, the "update counterparts" of $\mathsf{K5}$, $\mathsf{S4}$, $\mathsf{S5}$, etc. do *not* coincide with the respective standard logics.

Remark 9. Independently of any considerations on update models, Theorem 7 can be viewed as a *strengthening of the completeness theorem* for the *standard* normal logics K, KB, D, T, and TB. It states that for every non-valid formula F there is a counter-interpretation of an instance of F that obeys the atomic-update restriction (**au**′). Indeed, the proof of the theorem consists in an explicit construction of such a substitution instance and its corresponding update counter-model.

We are interested in reasoning about dynamic databases both at the level of "schematic" statements and by evaluating statements referring to concrete atoms of a database. Theorem 7 tells us that we remain within standard normal modal logics as long as only schematic validity is considered. In order to be able to refer to the schematic as well as the concrete level simultaneously we define the language $\mathcal{UL}$ over a *two-sorted* propositional signature:

- An *atomic formula* of $\mathcal{UL}$ is either an element $p \in \mathsf{atoms}$ or a schematic variable or $\top$ or $\bot$. (The set of propositional variables and atoms are disjoint.)
- *Complex formulas* of $\mathcal{UL}$ are built up as usual from the atomic formulas using the connectives $\neg, \wedge, \vee, \supset$ and the modalities $\Box$, $\Diamond$.

A formula of our extended language is called *concrete* if it does not contain propositional variables. Otherwise, it is called *schematic*. For concrete formulas F, $v_\alpha^{\mathcal{M}}(F)$ is defined as in Section 2. An arbitrary (possibly schematic) formula F is called *valid* in $\mathcal{M}$ if for all concrete formulas F' that arise by substituting the propositional variables of F with concrete formulas we have $v_\alpha^{\mathcal{M}}(F') = \mathbf{t}$ for all states α of $\mathcal{M}$.

Notation. We use lower case letters for atoms. Different letters always denote different atoms. Propositional variables are denoted by upper case letters from the end of the alphabet.

Example 10. The concrete formula $(a \wedge \neg b) \supset \Box(b \supset a)$ is valid in all update models. However the schematic formula $(X \wedge \neg b) \supset \Box(b \supset X)$ is not valid in most update models.

The concrete formula $\phi = \Diamond(a \wedge b) \wedge \Diamond(a \wedge \neg b) \wedge \Diamond(\neg a \wedge b) \wedge \Diamond(\neg a \wedge \neg b)$ can never evaluate to **t**, since this would mean that in at least one of the accessible updates both atoms, a and b, are evaluated differently than in the current state. In other words $\neg\phi$ is valid in all update models. In contrast, it is easy to find counter models for $\Box(X \vee Y) \vee \Box(X \vee \neg Y) \vee \Box(\neg X \vee Y) \vee \Box(\neg X \vee \neg Y)$.

4 Prefixed tableaux adapted to update models

We have defined adequate syntax and semantics of a language that allows to express various statements with respect to changing databases (of a particularly simple type). To substantiate the claim that this formalism provides a basis for *reasoning* we have to

define sound and complete calculi, suitable for automated proof search. Fortunately, Fitting's analytic *prefixed tableaux* [8] for standard normal logics turn out to be adaptable to our scenario. (See also [12] for an overview and history of related methods.)

We assume familiarity with tableaux but review the relevant terminology.

A *prefix* is a finite sequence of natural numbers (separated by dots). A prefix τ is a *simple extension* of a prefix σ if $\tau = \sigma.n$ for some $n \in \mathbb{N}$. A *prefixed formula* is a pair consisting of a prefix σ and formula F written as $\sigma :: F$. (Kripke-) interpretations are extended to prefixed formulas by referring to an *assignment* ϕ of worlds to prefixes. More formally, we define $v^{\mathcal{M}}_{\phi}(\sigma :: F) = v^{\mathcal{M}}(F, \phi(\sigma))$. If S is a set of prefixed formulas, then Pre(S) is the set of prefixes in S..

Prefixed tableaux are downward rooted trees of prefixed formulas, generated by appending new prefixed formula to a branch according to three types of rules.

Non-modal rules:

The rules for negation and disjunction are as follows:

$$(\neg\neg)\ \frac{\sigma :: \neg\neg F}{\sigma :: F} \qquad (\vee)\ \frac{\sigma :: F \vee G}{\sigma :: F \mid \sigma :: G} \qquad (\neg\vee)\ \frac{\sigma :: \neg(F \vee G)}{\begin{array}{c}\sigma :: \neg F \\ \sigma :: \neg G\end{array}}$$

We refer to $\sigma :: F$ and $\sigma :: G$ in $(\vee)$ as the two *sides* of the conclusion. The rules for conjunction and implication are similar.

Modal rules:

The rules for analyzing the modality $\Box$ in the basic modal logic K are

$$(\mathrm{K})\ \frac{\sigma :: \Box F}{\sigma.n :: F} \qquad (\pi)\ \frac{\sigma :: \neg\Box F}{\sigma.n :: \neg F}$$

where for (π) n is such that the prefix $\sigma.n$ is new to the current branch and for (K) $\sigma.n$ has been already used in the current branch. $\Diamond$ is treated as $\neg\Box\neg$. For serial, reflexive, and symmetric models we have to add the following rules, respectively:

$$(\mathrm{D})\ \frac{\sigma :: \Box F}{\sigma :: \Diamond F} \qquad (\mathrm{T})\ \frac{\sigma :: \Box F}{\sigma :: F} \qquad (\mathrm{KB})\ \frac{\sigma.n :: \Box F}{\sigma :: F}$$

Closure rules:

The closure rules for standard modal logics are

$$\frac{\begin{array}{c}\sigma :: \neg F \\ \sigma :: F\end{array}}{\texttt{closed}} \qquad \frac{\sigma :: \neg\top}{\texttt{closed}}$$

To accommodate the difference between atoms and schematic variables in the language $\mathcal{UL}$ as well as for the atomic update condition in update models it suffices to extend the standard tableau calculi by additional closure rules.

(Atomic) update closure rules:

$$\frac{\begin{array}{c}\sigma :: a \\ \sigma.n :: \neg a \\ \sigma :: b \\ \sigma.n :: \neg b\end{array}}{\texttt{closed}} \qquad \frac{\begin{array}{c}\sigma :: \neg a \\ \sigma.n :: a \\ \sigma :: b \\ \sigma.n :: \neg b\end{array}}{\texttt{closed}} \qquad \frac{\begin{array}{c}\sigma :: a \\ \sigma.n :: \neg a \\ \sigma :: \neg b \\ \sigma.n :: b\end{array}}{\texttt{closed}} \qquad \frac{\begin{array}{c}\sigma :: \neg a \\ \sigma.n :: a \\ \sigma :: \neg b \\ \sigma.n :: b\end{array}}{\texttt{closed}}$$

where a and b are different atoms.

If Λ is one of the logics K, KB, D, T, or TB, then a tableau constructed according to the above rules and the corresponding modal rules is called an *update* Λ*-tableau*.

A branch B of a tableau is *closed* if one of the above closure rules is applicable; otherwise B is called *open*. Let B be an open branch; the result B' of applying a rule ρ to one of the prefixed formulas in B and adding the prefixed formula(s) of (one side of) the conclusion of ρ to B is called an *extension of* B, as usual.

If all branches in a tableau $\mathbf{T}$ are *closed*, then $\mathbf{T}$ is called *closed*.

A closed update Λ-tableau with root $1 :: \neg F$ is a *tableau proof* of F. We will establish soundness and completeness of the presented tableau calculi, following essentially the proofs for standard normal logics as presented, e.g., in [8, 14, 12].

Let Π be a set of prefixes. Let $\sigma \rhd \tau$, $(\sigma, \tau \in \Pi)$ denote that τ is Λ-*accessible* from σ. The definition of $\rhd$ is given in the following table. (We call a prefix σ a Λ-*deadend* if Λ is non-serial and if there is no τ accessible from σ. In the case of the serial counterpart of Λ we demand that any Λ-deadend is made reflexive.)

Λ	$\sigma \rhd \tau$ iff
K	$\tau = \sigma.n$ for some $n \geq 1$
KB	$\tau = \sigma.n$ or $\sigma = \tau.m$
D	K-condition or (σ is a K-deadend and $\sigma = \tau$)
T	$\tau = \sigma.n$ or $\tau = \sigma$
TB	$\tau = \sigma$ or $\tau = \sigma.n$ or $\sigma = \tau.m$

This definition implies that $\langle \Pi, \rhd \rangle$ is a Λ-frame for $\Lambda \in \{\mathsf{K}, \mathsf{KB}, \mathsf{D}, \mathsf{T}, \mathsf{TB}\}$. In the following, we identify the set of propositional atoms atoms with a subset of PV (this subset is again denoted as atoms), thus treating our two-sorted (prefixed) $\mathcal{UL}$-formulas as a ordinary (prefixed) formula of modal logic.

A branch B of an update Λ-tableau is *satisfied* by a Λ-interpretation $\mathcal{M} = (W, R, V)$ if there is an assignment ϕ such that $v_\phi^{\mathcal{M}}(\sigma :: F) = \mathbf{t}$ for all $\sigma :: F$ in B.

Observe that open branches of update tableaux are, by definition, also branches of ordinary (modal) tableaux. Hence the following lemma is standard.

Lemma 11. *Let B be an open branch in an update Λ-tableau. Assume that B is satisfied by an Λ-interpretation $\mathcal{M}$ such that for all $w \in W$, condition $(\mathbf{au}')$ is fulfilled. Then every extension B' of B is also satisfied by $\mathcal{M}$.*

Theorem 12 (Soundness). *Let $\Lambda \in \{\mathsf{K}, \mathsf{KB}, \mathsf{D}, \mathsf{T}, \mathsf{TB}\}$. If F has an update Λ-tableau proof then F is valid in all update Λ-models.*

Proof. (Indirectly.) Suppose that F is not valid in all update Λ-models. Then some instance F' of F has a counter-Λ-interpretation $\mathcal{M} = (W, R, V)$, which fulfills condition $(\mathbf{au}')$ for all $w \in W$.

Now assume that there exists a tableau proof $\mathbf{T}$ of F. We can instantiate $\mathbf{T}$ to obtain a closed tableau $\mathbf{T}'$ with root $1 :: \neg F'$. By the first assumption $v_\phi^{\mathcal{M}}(\neg F') = \mathbf{t}$. Using Lemma 11 inductively, if follows that there exists a branch in $\mathbf{T}$' satisfied by $\mathcal{M}$. This contradicts the assumption that $\mathbf{T}$ and hence also $\mathbf{T}$' is closed. □

A set S of prefix $\mathcal{UL}$-formulas is *atomically closed* if

1. There is a formula A such that both $\sigma :: A$ and $\sigma :: \neg A$ occur in S, or
2. $\sigma :: \neg\top$ occurs in S, or
3. one of the following cases holds, where a, b are different atoms:

$$\{\, \sigma :: a,\ \ \sigma.n :: \neg a,\ \ \sigma :: b,\ \ \sigma.n :: \neg b \,\} \subseteq S$$
$$\{\, \sigma :: \neg a,\ \ \sigma.n :: a,\ \ \sigma :: b,\ \ \sigma.n :: \neg b \,\} \subseteq S$$
$$\{\, \sigma :: a,\ \ \sigma.n :: \neg a,\ \ \sigma :: \neg b,\ \ \sigma.n :: b \,\} \subseteq S$$
$$\{\, \sigma :: \neg a,\ \ \sigma.n :: a,\ \ \sigma :: \neg b,\ \ \sigma.n :: b \,\} \subseteq S$$

A set S of prefix $\mathcal{UL}$-formulas is Λ-*downward saturated* if it is *not* atomically closed and the usual conditions for downward saturatedness are satisfied by composite formulas F. (See, e.g., [12, 8].) We recall only the case where $F = \sigma :: \Box A$: Let $\Pi = \mathrm{Pre}(S)$, if $\sigma :: \Box A$ occurs in S, then $\tau :: A \in S$ for every $\tau \in \Pi$ such that $\sigma \rhd \tau$.

We use the following corollary, extracted from the proof of Theorem 7.

Corollary 13. *Let $\mathcal{M} = (W, R, V)$ be a tree-like Λ-interpretation that, for all $w \in W$ fulfills the atomic update condition (**au**′) on some subset P of* PV. *Then there exists an Λ-interpretation $\mathcal{M}' = (W, R, V)$ such that for all $w \in W$: w fulfills the atomic update condition (**au**′) on all* PV *and $v^{\mathcal{M}}(A, w) = v^{\mathcal{M}'}(A\theta, w)$ for all formulas A and some substitution θ with domain* $\mathsf{PV} - P$.

Theorem 14 (Completeness). *For $\Lambda \in \{$*K, KB, D, T, TB*$\}$, if an $\mathcal{UL}$-formula F is valid in all update Λ-models, then there exists a tableau proof of F.*

Proof. (Indirectly.) Suppose that all tableaux with root $1 :: \neg F$ have an open branch. Then a systematic tableau construction, as described in [12] or [8], yields an open branch B that is downward saturated. As in the standard completeness proofs, one can show that B is satisfied by a tree-like Λ-interpretation $\mathcal{M} = (W, R, V)$. In particular, we have $v_\phi^{\mathcal{M}}(1 :: F) = \mathbf{f}$ for some assignment ϕ. Moreover, since B is Λ-downward saturated, (**au**′) is fulfilled on (the subset of PV called) atoms (because of clause 3 in the definition of atomical closure, above.) By Corollary 13 we obtain a counter-Λ-interpretation $\mathcal{M}' = (W, R, V')$ for F' that fulfills (**au**′) on all variables, where in F' only variables that are not in atoms have been instantiated. But this implies that F cannot be valid in all Λ-update models. □

Remark on integrity constraints

In reasoning about changing states of a database, *integrity constraints* are of central importance. By an integrity constraint we simply mean a condition, referring to specific atoms and/or schematic variables, that has to be fulfilled in all states of a given database. Assuming that, in reference to single state, those conditions are expressible in $\mathcal{UL}$, the framework of prefixed tableaux allows the inclusion of integrity constraints in reasoning about databases by simply treating the corresponding formulas of $\mathcal{UL}$ as *global axioms* (in the sense of [8, 14]).

5 Incomplete and inconsistent data

Our model of the dynamic behavior of a database is yet too simple to capture phenomena like possibly incomplete and inconsistent data. However, we claim that the basic formalism of atomic update models and corresponding tableaux is easily adapted to such scenarios.

Belnap's four-valued logic [4] has been suggested repeatedly as a tool for reasoning about (possibly) inconsistent and incomplete information. The main intuition in this context is that a database may not only contain information implying that a statement is *false* or *true*, but such information may also be absent or inconsistent. The four possible states of knowledge are represented by the four truth values **f** (*false*), **u** (*undetermined*), $\bot$(*inconsistent*), and **t** (*true*), respectively. This intended interpretation induces the following truth functions for the connectives $\neg$, $\wedge$, and $\vee$:

	$\neg$
f	**t**
u	**u**
$\bot$	$\bot$
t	**f**

$\wedge$	**f**	**u**	$\bot$	**t**
f	**f**	**f**	**f**	**f**
u	**f**	**u**	**f**	**u**
$\bot$	**f**	**f**	$\bot$	$\bot$
t	**f**	**u**	$\bot$	**t**

$\vee$	**f**	**u**	$\bot$	**t**
f	**f**	**u**	$\bot$	**t**
u	**u**	**u**	**t**	**t**
$\bot$	$\bot$	**t**	$\bot$	**t**
t	**t**	**t**	**t**	**t**

For the definition of other connectives (in particular forms of implication) and the choice of designated truth values we refer to the extensive investigations of Avron and Arieli (see, e.g., [2, 1]).

The many-valued context allows to extend the classical universal and existential modalities to the more general concept of *distribution modalities*, introduced in [7]. Let $\mathcal{V}$ be the set of truth values; and, correspondingly, let a state of a database be an assignment $\alpha : \mathsf{atoms} \mapsto \mathcal{V}$. Then any function $\widetilde{\mu}$ of type $2^{\mathcal{V}} \mapsto \mathcal{V}$ induces a truth function of a distribution modality μ by:

$$v_\alpha^{\mathcal{D}}(\mu F) = \widetilde{\mu}(\{v_\beta^{\mathcal{D}}(F) \mid \beta \in \Sigma\colon \alpha U \beta\}).$$

Here Σ are the states of the update model $\mathcal{D}$ and U is its accessibility relation. $\{v_\beta^{\mathcal{D}}(F) \mid \beta \in \Sigma\colon \alpha U \beta\}$ is called the *distribution* of F in $\mathcal{D}$ at α. Again, a (many-valued) update model corresponds to a (many-valued) Kripke interpretation. In particular, we call the update models in which the states of the database consist in assignments of type $\mathsf{atoms} \mapsto \{\mathbf{t}, \mathbf{u}, \mathbf{f}, \bot\}$ *Belnap update structures*. This context allows us to define modalities like

- $det(F)$ with the intended meaning: "No update renders information on F incomplete or inconsistent", and
- $unif(F)$ with the intended meaning: "F is evaluated uniformly in all updates".

Since $det(F)$ is intended to express a meta-linguistic (and therefore classical) property of F within the object language itself, it always evaluates to **t** or **f**. More exactly, its semantics is fixed by:

$$\widetilde{det}(W) = \begin{cases} \mathbf{t} & \text{if } W = \emptyset, \{\mathbf{f}\}, \{\mathbf{t}\}, \text{or } \{\mathbf{t}, \mathbf{f}\} \\ \mathbf{f} & \text{otherwise} \end{cases}$$

On the other hand "uniform evaluation" admittedly is an ambiguous concept. Certainly, we want $\mathit{unif}(F)$ to be *true* if either F evaluates to **t** in all updates, or to **f** in all updates. Likewise, it is clear that $\mathit{unif}(F)$ is *false* if the distribution contains **t** *and* **f** (i.e., if there is an update evaluating the formula to **t**, but also another update that evaluates it to **f**.) But we want $\mathit{unif}(F)$ to be *undetermined* if the distribution of F contains **u**. One way to round off and formalize these intuitions is to define the truth function for *unif* as follows:

$W \subseteq \mathcal{V}$	$\widetilde{\mathit{unif}}(W)$
$\emptyset, \{\mathbf{f}\}, \{\mathbf{t}\}$	**t**
$\{\mathbf{u}\}, \{\mathbf{u},\mathbf{f}\}, \{\mathbf{u},\mathbf{t}\}$	**u**
$\{\mathbf{f},\mathbf{t}\}, \{\mathbf{f},\mathbf{u},\mathbf{t}\}, \{\mathbf{f},\bot,\mathbf{t}\}, \{\mathbf{f},\mathbf{u},\bot,\mathbf{t}\}$	**f**
$\{\bot\}, \{\mathbf{f},\bot\}, \{\bot,\mathbf{t}\}, \{\mathbf{u},\bot\}, \{\mathbf{f},\mathbf{u},\bot\}, \{\mathbf{u},\bot\,\mathbf{t}\}$	$\bot$

Of course, *det* and *unif* are just two simple examples. Observe that there are 4^{2^4} possible distribution modalities definable over Belnap update structures. All of them refer to properties of the "truth status" of statements with respect to the class of possible stepwise evolutions of the database. We also remind the reader that Belnap update models come in different variants according to different constraining properties of the update relation.

To define a particular *Belnap update logic* with respect to a class of Belnap update models we therefore have to fix three independent parameters (in addition to the set of atoms and propositional variables):

(1) a set of designated truth values $\mathcal{V}_D \subseteq \mathcal{V}$; usually $\{\mathbf{t}\}$ or $\{\mathbf{t},\bot\}$
(2) a set of $\{\mu_1,\ldots,\mu_n\}$ of distribution modalities (with associated truth functions $\widetilde{\mu_1},\ldots,\widetilde{\mu_n}$ and four-valued connectives (specified by their truth tables)
(3) properties like symmetry or reflexivity, which we want the update relation to observe.

We call a concrete formula F *valid* in such a logic if $v^{\mathcal{D}}_{\alpha}(F) \in \mathcal{V}_D$ for all states α of all corresponding atomic update models $\mathcal{D}$. This is extended to schematic formulas in the obvious way. (See Section 3.)

6 Prefixed signed tableaux for Belnap update models

It is well known that appropriate analytic calculi for all (truth functional) finite valued logics can be defined using *signed* versions of tableaux (see, e.g., [13]). These can be extended to finite valued modal logics by combining prefixes (denoting worlds) and signs (denoting truth values) as was shown in [7]. We describe a simplified example of the latter calculi, adapted for update structures, for the special case of Belnap update models with serial (but otherwise general) update relation and (only) modality *det*.[2]

A *prefixed signed formula* is a triple consisting of a finite sequence of natural numbers σ (prefix), a truth value v, and a formula F, written as $\sigma{:}[v]{:}F$.

[2] Other properties of the update relation result in simple technical variations of some modal rules. The corresponding calculi are omitted here for space reasons.

Remark 15. In classical logic the prefixed signed formulas $\sigma\colon[\mathbf{t}]\colon F$ and $\sigma\colon[\mathbf{f}]\colon F$ are just notational variants of the prefixed formulas $\sigma :: F$ and $\sigma :: \neg F$, respectively. For many-valued logics truth value signs are not only an elegant way to make semantic information explicit but are, in general, *needed* to obtain complete tableau calculi.

Again, a *(prefixed signed)* tableau is a downward rooted tree of prefixed signed formulas, constructed using the following rules.

Non-modal rules: can directly be read off from the truth tables of connectives. We refer to [13, 16] for general methods and results about constructing optimal rules.

Closure rules: The standard closure rule is

$$\frac{\begin{array}{c}\sigma\colon[v]\colon F\\ \sigma\colon[w]\colon F\end{array}}{\texttt{closed}}$$

where v and w are different truth values. (F need not be atomic.)

A modal operator μ induces an additional closure rules if a formula μF never evaluates to a particular truth value. For instance, the modality *det* triggers the following two closure rules:

$$\frac{\sigma\colon[\mathbf{u}]\colon det(F)}{\texttt{closed}} \qquad \frac{\sigma\colon[\bot]\colon det(F)}{\texttt{closed}}$$

Modal rules:

A general method for constructing modal rules from associated truth functions is described in [7]. We present greatly simplified[3] versions for the remaining cases of *det*-modalized formulae:

$$\frac{\sigma\colon[\mathbf{t}]\colon det(F)}{\sigma.n\colon[\mathbf{t}]\colon F \mid \sigma.n\colon[\mathbf{f}]\colon F} \qquad \frac{\sigma\colon[\mathbf{f}]\colon det(F)}{\sigma.n\colon[\mathbf{u}]\colon F \mid \sigma.n\colon[\bot]\colon F}$$

where $\sigma.n$ already occurs on the branch.

(Atomic) update closure rule:

$$\frac{\begin{array}{c}\sigma\colon[u_1]\colon a\\ \sigma.n\colon[v_1]\colon a\\ \sigma\colon[u_2]\colon b\\ \sigma.n\colon[v_2]\colon b\end{array}}{\texttt{closed}}$$

where a and b are different atoms and $v_i \neq u_i$ for $i = 1$ and $i = 2$.

The results of [7] and Theorems 12 and 14 can be combined straightforwardly to obtain

Theorem 16. *A formula F is valid in a Belnap update logic if and only if for all non-designated truth values v there exists a corresponding update tableaux with root $1\colon[v]\colon F$ that is closed.*

Remark 17. For all mentioned variants of update logics, systematic and terminating tableau construction procedures can be defined as usual. This, in particular, implies the *decidability* of these logics.

[3] The simplification makes essential use of the fact that *det* is the only modal operator and that the update relation is serial, but otherwise unrestricted.

7 Open ends

Other types of update operations. Obviously atomic updates as defined by condition (**au**) are only a special case.[4] One might, e.g., study multiple update relations that are indexed by the "new information" that triggers the update. This information is often represented by a boolean combination of atoms and thus naturally induces a corresponding algebra of update relations (similar to the algebra of programs in dynamic logic).

Different underlying many-valued logics. Update models can be defined over all kinds of truth functional logics as mechanism for "local" evaluation. As an interesting example we mention the bi-lattice based logics suggested by M.L. Ginsberg [11] for modeling default reasoning. Also dynamic *fuzzy* databases can be modeled by building on an appropriate fuzzy logic (e.g., some finite-valued Łukasiewicz logic).

Other useful distribution modalities. As explained above, every function of type $2^{\mathcal{V}} \mapsto \mathcal{V}$ induces a distribution modality. A systematic investigation of expressibility, complexity of corresponding rules and functional dependency between different sets of modalities is still lacking.

Modeling global update constraints. As a simple example consider the condition—for Belnap update models—that updates can only *increase knowledge* about data. Technically this corresponds to requiring $\alpha(a) \leq_k \beta(a)$ if $\alpha U \beta$, where $\leq_k$ is the partial "knowledge order" defined by $\mathbf{u} \leq_k \mathbf{t} \leq_k \bot$ and $\mathbf{u} \leq_k \mathbf{f} \leq_k \bot$.

First-order reasoning. Both, update models and corresponding tableau, are readily generalized to the first-order level. This move, of course, vastly improves the expressibility and complexity of the corresponding logics. Their strength and limits should also be explored.

References

[1] Ofer Arieli and Arnon Avron. The logical role of the four-valued bilattice. In *13th Annual IEEE Symposium on Logic in Computer Science (LICS)*, pages 118–126, Indianapolis (USA), 1998. IEEE Computer Society.

[2] Ofer Arieli and Arnon Avron. The value of the four values. *Artificial Intelligence*, 102(1):97–141, 1998.

[3] Philippe Balbiani. A modal logic for data analysis. In Wojciech Penczek and Andrzej Szalas, editors, *Proc. Mathematical Foundations of Computer Science, 21st MFCS*, volume 1115 of *Lecture Notes in Computer Science*, pages 167–179, Cracow, Poland, 1996. Springer-Verlag.

[4] Nuel D. Belnap. A useful four-valued logic. In J.M. Dunn and G. Epstein, editors, *Modern Uses of Multiple-Valued Logic*, pages 8–37. D. Reidel Publishing Co., 1977.

[5] Thomas Eiter and Georg Gottlob. On the complexity of propositional knowledge base revision, updates, and counterfactuals. *Artificial Intelligence*, 57:227–270, 1992.

[6] Dieter Fensel, Rix Groenboom, and G.R.Renardel de Lavalette. Modal change logic (MCL): Specifying the reasoning of knowledge-based systems. *Data Knowl. Eng.*, 25(1-2):243–269, 1998.

[4] For a thorough computational analysis of different related concepts see [5].

[7] Christian G. Fermüller and Herbert Langsteiner. Tableaux for finite-valued logics with arbitrary distribution modalities. In Harrie de Swart, editor, *Proc. Automated Reasoning with Analytic Tableaux and Related Methods (TABLEAUX'98)*, volume 1397 of *Lecture Notes in Computer Science*, pages 156–171, Oisterwijk (Netherlands), 1998. Springer-Verlag.

[8] Melvin C. Fitting. *Proof Methods for Modal and Intuitionistic Logics*. D. Reidel Publishing Co., Dordrecht, 1983.

[9] Melvin C. Fitting. Modality and databases. In Roy Dyckhoff, editor, *Proc. Automated Reasoning with Analytic Tableaux and Related Methods (TABLEAUX 2000)*, pages 19–39, Lecture Notes in Computer Science 1847, St. Andrews, Scotland, 2000. Springer-Verlag.

[10] Nir Friedman and Joseph Y. Halpern. Modeling belief in dynamic systems. II: Revision and update. *J. Artif. Intell. Res. (JAIR)*, 10:17–167, 1999.

[11] Matthew L. Ginsberg. Multi-valued logics. *Computational Intelligence*, 4(3), 1988.

[12] Rajeev Goré. Tableau methods for modal and temporal logics. In M. D'Agostino, D. Gabbay, R. Hähnle, and J. Posegga, editors, *Handbook of Tableau Methods*, pages 297–396. Kluwer Academic Publishers, 1999.

[13] Reiner Hähnle. *Automated Deduction in Multiple-valued Logics*. Clarendon Press, Oxford, 1993.

[14] Fabio Massacci. Single Step Tableaux for Modal Logics. *J. Automated Reasoning*, 24(3):319–364, 2000.

[15] Mark Ryan and Pierre-Yves Schobbens. Counterfactuals and updates as inverse modalities. *J. Logic Lang. Inf.*, 6(2):123–146, 1997.

[16] Gernot Salzer. Optimal axiomatizations for multiple-valued operators and quantifiers based on semilattices. In Michael McRobbie and John Slaney, editors, *Proc. 13th Conference on Automated Deduction, New Brunswick/NJ, USA*, volume 1104 of *Lecture Notes in Computer Science*, pages 688–702. Springer-Verlag, 1996.