Intuitionistic S4 is decidable

Marianna Girlando, Roman Kuznets, Sonia Marin, Marianela Morales, Lutz Strassburger

Intuitionistic S4 is decidable

Marianna Girlando
University of Amsterdam
Amsterdam, Netherlands

Roman Kuznets
TU Wien
Vienna, Austria

Sonia Marin
University of Birmingham
Birmingham, UK

Marianela Morales
Inria Saclay
Palaiseau, France

Lutz Straßburger
Inria Saclay
Palaiseau, France

***Abstract*—In this paper we demonstrate decidability for the intuitionistic modal logic S4 first formulated by Fischer Servi. This solves a problem that has been open for almost thirty years since it had been posed in Simpson's PhD thesis in 1994. We obtain this result by performing proof search in a labelled deductive system that, instead of using only one binary relation on the labels, employs two: one corresponding to the accessibility relation of modal logic and the other corresponding to the order relation of intuitionistic Kripke frames. Our search algorithm outputs either a proof or a finite counter-model, thus, additionally establishing the finite model property for intuitionistic S4, which has been another long-standing open problem in the area.**

1. Introduction

For a logic to be decidable, there must be a recursive procedure that determines, for each formula, whether or not it is a theorem of the logic, ideally terminating with either a proof in a deductive system or a suitable countermodel.

Decidability of intuitionistic propositional logic. Gentzen was the first to show decidability of intuitionistic propositional logic in 1935, using a sequent calculus he had designed [11]. His approach was to bound the number of consequences inferred from some given initial sequents. It was later observed (independently by Ono [23], Ketonen [14], and Kleene [15]) that, if structural rules were built in into the notation, it allowed for a root-first proof-search approach. Namely, search for a sequent calculus proof until it either terminates with a proof or else reaches a sequent that has already occurred along the branch leading to it, in which case it is possible to reconstruct a Kripke countermodel from that branch [28]. For a detailed historical account, see Dyckhoff's survey [5].

Decidability for classical modal logics. Decidability for modal logics (on the classical propositional base) has been investigated early on. Ladner provided decision procedures for some common modal logics, including S4, based purely on Kripke semantics, with no mention of sequent calculi [17]. Otherwise, the procedure described above can similarly be applied to modal logic S4, as it is also the logic of reflexive transitive Kripke frames. Generally, the approach via proof search in a sequent calculus is available for those modal logics that have a cut-free sequent system [25] (or, in certain cases, a nested sequent system using a similar approach [3]).

Decision procedures using labelled sequents. Labelled sequents internalize certain elements of Kripke semantics into the syntax of sequents, which turns out to have several interesting consequences for decision procedures for intuitionistic propositional logic [6], as well as modal logics [22].

This is the author version of the LICS'23 paper with detailed appendices.

Firstly, in the setting of intuitionistic propositional logic, it is not necessary to restrict oneself to single-conclusion sequents (i.e., sequents with at most one formula in the succedent). This means that all the rules can be made invertible, thus, removing the need for backtracking and making the proof search procedure deterministic. Secondly, in both settings, the loop-check for ensuring termination becomes local to the topmost sequent of a branch of a proof-search tree rather than applied between two sequents on this branch: it is sufficient to check whether two labels of the topmost sequent carry the same formulas. Finally, in the case when a loop is found, a countermodel can be built directly from the topmost sequent. Indeed, the syntax of labelled sequents makes it easy to read off a countermodel from a sequent by simply substituting labels or adding back edges, both constructions being available directly in the labelled syntax.

Intuitionistic modal logic. Intuitionistic modal logic IS4 is a way to combine intuitionistic propositional logic and modal logic S4. It is an extension of intuitionistic modal logic IK, which was first studied in [8], [24] and investigated in detail in [26], where decidability was shown for most logics in the intuitionistic modal S5-cube. Decidability of IS5 (aka MIPQ) had been shown earlier by Mints [21, §11], but for IK4 and IS4 decidability remained open. Indeed, the question of IS4's decidability hides a lot more complexity than either of its progenitors. As IS4 can be interpreted on Kripke frames that combine the preorder relation of intuitionistic Kripke frames with the accessibility relation of modal Kripke frames, the well-known problems of looping in decision procedures based on respective sequent calculi are exacerbated by the interactions between modalities and intuitionistic implication.

Contribution. In this paper, we show that IS4 is decidable and conjecture that the same method can also be applied to show decidability of IK4. We provide a constructive decision procedure, that, given a formula, produces either a proof showing the formula to be valid or a finite countermodel falsifying the formula. The procedure is based on a *fully labelled proof system* for IS4 presented in [20]. This system inherits the advantages of labelled systems for intuitionistic propositional logic and for classical modal logics, in particular, all inference rules are invertible and there is a direct correspondence between sequents and models. This choice was crucial to the emergence of the solution, which also required an intricate organization of proof search and loop-checking to reach a full decision procedure.

Related work. Several strategies have been explored for

decidability of IS4 and related logics. The $\Box$-only fragments of a wide range of intuitionistic multi-modal logics was shown to be decidable via a proof-search based procedure in [10].

Moreover, for $\Box$-only fragments of intuitionistic modal logics, decidability results were obtained through embedding into classical bimodal logics, defined as *fusions* of normal monomodal logics [29]. Logic IS4 is closely related to a classical bimodal logic defined as the *product* S4 $\times$ S4 [8] in that both are sound and complete with respect to Kripke models with two preorder relations forming a "grid." Interestingly, logic S4 $\times$ S4 was shown to be undecidable by Gabelaia *et al.* [9]. This result shows how easily undecidability arises in modal logic; however, the proof strategy from [9] does not scale to IS4, whose models need to satisfy monotonicity over one of the preorder relations. Similarly, embeddings of intuitionistic modal logics into fragments of monadic second-order logic, which is decidable, were used to prove decidability [1], but, as explained by the authors, the method could not be applied to IS4. Another common method of demonstrating decidability applied to some intuitionistic modal logics in [12] is semantic filtration, but again the specificities of the logics from the intuitionistic S5-cube have so far prevented its application to them. On the other hand, constructive S4 (a different intuitionistic variant of modal logic S4) was shown to be decidable in [13], and the finite model property for it was proved in [2].

Organization. In Sect. 2, we recall the syntax and semantics of IS4. In Sect. 3, we explain the difficulties of devising a decision procedure for it. In Sect. 4, we present the first ingredient of our solution: proof system labIS4$_{\leq}$ for IS4 based on fully labelled sequents. In Sect. 5, we describe how to read off a model from such a sequent. In Sect. 6, we establish useful properties of sequents occurring during a proof search in labIS4$_{\leq}$. In Sect. 7, we develop concepts necessary to present our decision procedure based on a modified proof-search algorithm. In Sect. 8, we demonstrate how to retrieve a countermodel from a failed proof search. In Sect. 9, we show how to reconstruct a real proof when the algorithm succeeds. In Sect. 10, we prove that the algorithm always terminates. Finally, Sect. 11, contains our conclusions. All the proofs and some detailed examples can be found in the Appendix.

2. Preliminaries: Logic IS4

In this paper, formulas are denoted by capital letters A, B, C, ... and are constructed from a countable set $\mathcal{A}$ of *atomic propositions* (denoted by lowercase a, b, c, ...) as

$$A ::= \bot \mid a \mid (A \wedge A) \mid (A \vee A) \mid (A \supset A) \mid \Box A \mid \Diamond A$$

Intuitionistic modal logic K, or IK for short, is obtained from intuitionistic propositional logic by extending the syntax with two modalities $\Box$ and $\Diamond$, standing most generally for *necessity* and *possibility* respectively, and by adding k-*axioms*

$$\begin{array}{ll} \mathsf{k}_1\colon \Box(A\supset B)\supset(\Box A\supset\Box B) & \mathsf{k}_3\colon \Diamond(A\vee B)\supset(\Diamond A\vee\Diamond B) \\ \mathsf{k}_2\colon \Box(A\supset B)\supset(\Diamond A\supset\Diamond B) & \mathsf{k}_4\colon (\Diamond A\supset\Box B)\supset\Box(A\supset B) \\ & \mathsf{k}_5\colon \Diamond\bot\supset\bot \end{array} \quad (1)$$

A formula is a theorem of IK iff it is derivable from this set via the rules of *necessitation* and *modus ponens*:

$$\mathsf{nec}\ \frac{A}{\Box A} \qquad \text{and} \qquad \mathsf{mp}\ \frac{A \qquad A\supset B}{B}$$

Intuitionistic modal logic S4, or IS4 for short, is obtained from IK by adding axioms

$$\begin{array}{ll} \mathsf{4}\colon & (\Diamond\Diamond A\supset\Diamond A)\wedge(\Box A\supset\Box\Box A) \\ \mathsf{t}\colon & (A\supset\Diamond A)\wedge(\Box A\supset A) \end{array} \quad (2)$$

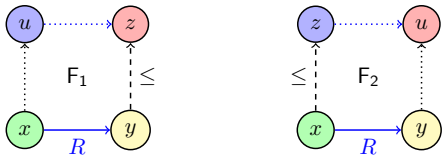
Note that in the classical case axioms k_2–k_5 in (1) would follow from k_1, but due to the lack of De Morgan duality, this is not the case in intuitionistic logic. Similarly, in (2) both conjuncts are needed because they do not follow from each other as in the classical case.

Let us now recall the *birelational models* [7], [24] for intuitionistic modal logics, which combine the Kripke semantics for intuitionistic propositional logic and classical modal logics.

Definition 2.1. A ***birelational frame*** $\mathcal{F}$ is a triple $\langle W, R, \leq\rangle$ of a nonempty set W of ***worlds*** equipped with an ***accessibility relation*** R and a preorder $\leq$ (i.e., a reflexive and transitive relation) satisfying:

(F_1) For all $x, y, z \in W$, if xRy and $y{\leq}z$, there exists $u \in W$ such that $x{\leq}u$ and uRz (see figure below left);

(F_2) For all $x, y, z \in W$, if $x{\leq}z$ and xRy, there exists $u \in W$ such that zRu and $y{\leq}u$ (see figure below right).



Definition 2.2. A ***birelational model*** $\mathcal{M}$ is a quadruple $\langle W, R, \leq, V\rangle$ with $\langle W, R, \leq\rangle$ a birelational frame and $V\colon W \to 2^{\mathcal{A}}$ a ***valuation function***, that is, a function mapping each world w to the subset of propositional atoms that are true at w, additionally subject to the ***monotonicity condition***: if $w{\leq}w'$, then $V(w) \subseteq V(w')$.

We write $\mathcal{M}, w \Vdash a$ iff $a \in V(w)$ and recursively extend relation $\Vdash$ to all formulas following the rules for both intuitionistic and modal Kripke models: $\mathcal{M}, w \nVdash \bot$;

$$\begin{array}{rl} \mathcal{M}, w \Vdash A\wedge B & \text{iff } \mathcal{M}, w \Vdash A \text{ and } \mathcal{M}, w \Vdash B; \\ \mathcal{M}, w \Vdash A\vee B & \text{iff } \mathcal{M}, w \Vdash A \text{ or } \mathcal{M}, w \Vdash B; \\ \mathcal{M}, w \Vdash A\supset B & \text{iff for all } w' \text{ with } w{\leq}w', \\ & \quad \text{if } \mathcal{M}, w' \Vdash A, \text{ then } \mathcal{M}, w' \Vdash B; \\ \mathcal{M}, w \Vdash \Box A & \text{iff for all } w' \text{ and } u \text{ with } w{\leq}w' \\ & \quad \text{and } w'Ru, \text{ we have } \mathcal{M}, u \Vdash A; \\ \mathcal{M}, w \Vdash \Diamond A & \text{iff there exists } u \text{ such that} \\ & \quad wRu \text{ and } \mathcal{M}, u \Vdash A. \end{array}$$

From the monotonicity of valuation function V, we get the monotonicity property for relation $\Vdash$:

Proposition 2.3. *(Monotonicity) For any formula A and for any $w, w' \in W$, if $w{\leq}w'$ and $\mathcal{M}, w \Vdash A$, then $\mathcal{M}, w' \Vdash A$.*

Definition 2.4 (Validity)**.** A formula A is ***valid in a model*** $\mathcal{M} = \langle W, R, \leq, V\rangle$ iff $\mathcal{M}, w \Vdash A$ for all $w \in W$. A formula A is ***valid in a frame*** $\mathcal{F} = \langle W, R, \leq\rangle$ iff it is valid in $\langle W, R, \leq, V\rangle$ for all valuations V.

The correspondence between syntax and semantics for IS4 can be stated as follows:

Theorem 2.5 (Completeness [8], [24])**.** *A formula A is a theorem of* IS4 *if and only if A is valid in every birelational frame $\langle W, R, \leq\rangle$ where R is reflexive and transitive.*

3. Why Is IS4's Decidability Hard?

In this section we highlight the main difficulties that we encountered in tackling the decidability problem for IS4 and give a hint at the key ingredients of our method in the process.

One way to prove decidability for a logic is to perform proof search in a sound and complete deductive system with the intention of either finding a proof or constructing a countermodel from a failed proof search.

For IS4 several such deductive systems exist, the first being Simpson's labelled systems [26]. Moreover, there are various kinds of nested sequents systems: single-conclusion [27], multiple conclusion [16], and also focused variants [4]. A natural question to ask is why none of these systems has been used to prove decidability of IS4.

The need for more labelling. The aforementioned systems rely on what we could call a mixed approach: they internalize the modal accessibility relation R within the sequent syntax, using either labels and relational atoms or nesting, but they rely on a traditional structural approach for the intuitionistic aspect of the logic, e.g., single-conclusion sequents (at least in certain rules). One might think that combining the traditional loop-check for the intuitionistic part with the label-based loop-check for the modal part would be a way to a decision procedure. But the situation is more complicated because

1) the classical S4-loop test, which is looking for a repetition along the R-relation, cannot be applied to the right-hand-side of a sequent, as the conclusion formula can sometimes be replaced by a new one;
2) the structural approach to the intuitionistic system also means the rules are not all invertible and the procedure requires backtracking, so the modality loop-check also needs to be combined with the necessary backtracking.

Both of these problems can be overcome by using a fully labelled proof system that incorporates both relations R and $\leq$ [19], [20]. This has the same advantages as moving from a structural to a labelled approach for intuitionistic propositional logic, mentioned in the introduction. Not only does this system re-establish the close relationship between a sequent and a model, as is known from classical modal logic, it also enables us to make all rules in the system invertible. Moreover, explicit relational atoms in the sequent syntax make it easy to implement the loop-checks and to represent the back edges explicitly when constructing a countermodel.

The backtracking/termination trade-off. Naive proof search is not terminating, with two possible sources of non-termination: the first inherited from the classical modal logic S4, and the second from intuitionistic propositional logic. In both those cases independently, the problem would be solved by a simple loop-check. However, it is not straightforward to combine the two, as the following example shows:

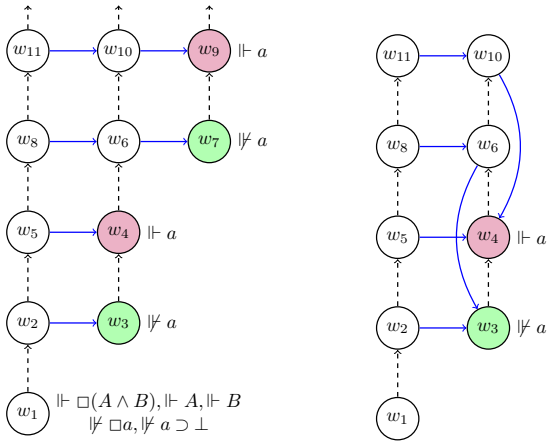


Fig. 1. Left: Illustration of the potential non-termination issue. Right: Illustration of a break of condition F_1 when identifying nodes unrestrictedly.

Example 3.1. Consider the following formula, which is not provable in IS4:

$$\Box\Big((\Box a \supset \bot) \wedge \big((a \supset \bot) \supset \bot\big)\Big) \supset \bot. \tag{3}$$

Let $A = \Box a \supset \bot$ and $B = (a \supset \bot) \supset \bot$. In order to construct a countermodel $\mathcal{M}$, we need a world w_1 that forces $\Box(A \wedge B)$ and, therefore, also A and B. Consequently, every world w' such that $w_1 \leq w$ and wRw' for some w should force these formulas. Then all these worlds must force neither $\Box a$ nor $a \supset \bot$. The latter means that for each such world w', there must be a world v with $w' \leq v$ and $\mathcal{M}, v \Vdash a$. Of course, in turn, v must not force $\Box a$, so there must be worlds u and u' with $v \leq u'$, $u'Ru$, and $\mathcal{M}, u \nVdash a$. But this u must not force $a \supset \bot$ so there must be a world v_1 with $u \leq v_1$ and $\mathcal{M}, v_1 \Vdash a$, and so on. Thus, a naive implementation of a countermodel construction via proof search will keep adding worlds *ad infinitum* because neither of the two loop-checks will detect the repetition (see Fig. 1, Left). In other words,

3) to account for the interaction of modalities and intuitionistic implications, the two loop-checks must get along well with each other.

The interleaving of proof search and loops. For solving this third problem, we have to implement a more sophisticated loop-check involving both relations. This directly leads us to the fourth problem.

Example 3.2. Assume, for the sake of example, that we designed a suitable loop-check such that we could stop proof search at the stage of the structure presented on Fig. 1 (Left) and that we identified w_7 to w_3 and w_9 to w_4. This would create "backlinks" between w_{10} and w_4 as well as

$$\mathsf{id}\ \frac{}{\mathcal{R}, x{\leq}y, \Gamma, x{:}a \Longrightarrow \Delta, y{:}a} \qquad \bot^{\bullet}\ \frac{}{\mathcal{R}, \Gamma, x{:}\bot \Longrightarrow \Delta}$$

$$\wedge^{\bullet}\ \frac{\mathcal{R}, \Gamma, x{:}A, x{:}B \Longrightarrow \Delta}{\mathcal{R}, \Gamma, x{:}A \wedge B \Longrightarrow \Delta} \qquad \wedge^{\circ}\ \frac{\mathcal{R}, \Gamma \Longrightarrow \Delta, x{:}A \quad \mathcal{R}, \Gamma \Longrightarrow \Delta, x{:}B}{\mathcal{R}, \Gamma \Longrightarrow \Delta, x{:}A \wedge B}$$

$$\vee^{\bullet}\ \frac{\mathcal{R}, \Gamma, x{:}A \Longrightarrow \Delta \quad \mathcal{R}, \Gamma, x{:}B \Longrightarrow \Delta}{\mathcal{R}, \Gamma, x{:}A \vee B \Longrightarrow \Delta} \qquad \vee^{\circ}\ \frac{\mathcal{R}, \Gamma \Longrightarrow \Delta, x{:}A, x{:}B}{\mathcal{R}, \Gamma \Longrightarrow \Delta, x{:}A \vee B}$$

$$\supset^{\bullet}\ \frac{\mathcal{R}, x{\leq}y, x{:}A \supset B, \Gamma \Longrightarrow \Delta, y{:}A \quad \mathcal{R}, x{\leq}y, \Gamma, y{:}B \Longrightarrow \Delta}{\mathcal{R}, x{\leq}y, \Gamma, x{:}A \supset B \Longrightarrow \Delta} \qquad \supset^{\circ}\ \frac{\mathcal{R}, x{\leq}z, \Gamma, z{:}A \Longrightarrow \Delta, z{:}B}{\mathcal{R}, \Gamma \Longrightarrow \Delta, x{:}A \supset B}\ z \text{ fresh}$$

$$\Box^{\bullet}\ \frac{\mathcal{R}, x{\leq}y, yRz, \Gamma, x{:}\Box A, z{:}A \Longrightarrow \Delta}{\mathcal{R}, x{\leq}y, yRz, \Gamma, x{:}\Box A \Longrightarrow \Delta} \qquad \Box^{\circ}\ \frac{\mathcal{R}, x{\leq}u, uRz, \Gamma \Longrightarrow \Delta, z{:}A}{\mathcal{R}, \Gamma \Longrightarrow \Delta, x{:}\Box A}\ u, z \text{ fresh}$$

$$\Diamond^{\bullet}\ \frac{\mathcal{R}, xRy, \Gamma, y{:}A \Longrightarrow \Delta}{\mathcal{R}, \Gamma, x{:}\Diamond A \Longrightarrow \Delta}\ y \text{ fresh} \qquad \Diamond^{\circ}\ \frac{\mathcal{R}, xRy, \Gamma \Longrightarrow \Delta, x{:}\Diamond A, y{:}A}{\mathcal{R}, xRy, \Gamma \Longrightarrow \Delta, x{:}\Diamond A}$$

$$\leq\!\mathsf{rf}\ \frac{\mathcal{R}, x{\leq}x, \Gamma \Longrightarrow \Delta}{\mathcal{R}, \Gamma \Longrightarrow \Delta} \qquad \leq\!\mathsf{tr}\ \frac{\mathcal{R}, x{\leq}y, y{\leq}z, x{\leq}z, \Gamma \Longrightarrow \Delta}{\mathcal{R}, x{\leq}y, y{\leq}z, \Gamma \Longrightarrow \Delta}$$

$$R\mathsf{rf}\ \frac{\mathcal{R}, xRx, \Gamma \Longrightarrow \Delta}{\mathcal{R}, \Gamma \Longrightarrow \Delta} \qquad R\mathsf{tr}\ \frac{\mathcal{R}, xRy, yRz, xRz, \Gamma \Longrightarrow \Delta}{\mathcal{R}, xRy, yRz, \Gamma \Longrightarrow \Delta}$$

$$\mathsf{F}_1\ \frac{\mathcal{R}, xRy, y{\leq}z, x{\leq}u, uRz, \Gamma \Longrightarrow \Delta}{\mathcal{R}, xRy, y{\leq}z, \Gamma \Longrightarrow \Delta}\ u \text{ fresh} \qquad \mathsf{F}_2\ \frac{\mathcal{R}, xRy, x{\leq}z, y{\leq}u, zRu, \Gamma \Longrightarrow \Delta}{\mathcal{R}, xRy, x{\leq}z, \Gamma \Longrightarrow \Delta}\ u \text{ fresh}$$

Fig. 2. System $\mathsf{labIS4}_{\leq}$

$$\mathsf{mon}^{\bullet}\ \frac{\mathcal{R}, x{\leq}y, \Gamma, x{:}A, y{:}A \Longrightarrow \Delta}{\mathcal{R}, x{\leq}y, \Gamma, x{:}A \Longrightarrow \Delta} \qquad 4^{\bullet}\ \frac{\mathcal{R}, xRy, x{:}\Box A, y{:}\Box A, \Gamma \Longrightarrow \Delta}{\mathcal{R}, xRy, x{:}\Box A, \Gamma \Longrightarrow \Delta} \qquad \mathsf{cont}^{\bullet}\ \frac{\mathcal{R}, \Gamma, x{:}A, x{:}A \Longrightarrow \Delta}{\mathcal{R}, \Gamma, x{:}A \Longrightarrow \Delta}$$

$$\mathsf{weak}\ \frac{\mathcal{R}, \Gamma \Longrightarrow \Delta}{\mathcal{R}, \mathcal{R}', \Gamma, \Gamma' \Longrightarrow \Delta, \Delta'} \qquad 4^{\circ}\ \frac{\mathcal{R}, xRy, \Gamma \Longrightarrow \Delta, x{:}\Diamond A, y{:}\Diamond A}{\mathcal{R}, xRy, \Gamma \Longrightarrow \Delta, x{:}\Diamond A} \qquad \mathsf{cont}^{\circ}\ \frac{\mathcal{R}, \Gamma \Longrightarrow \Delta, x{:}A, x{:}A}{\mathcal{R}, \Gamma \Longrightarrow \Delta, x{:}A}$$

Fig. 3. Admissible rules

between w_6 and w_3. However, this would lead to a violation of F_1 as now $w_{10}Rw_4{\leq}w_6$ but there is no w' such that $w_{10}{\leq}w'Rw_6$. (see Fig. 1, Right). This means that

4) the standard method of constructing a (finite) countermodel from a failed proof search via identifying labels/worlds that create a loop fails in the setting of birelational models. We break the $\mathsf{F}_1/\mathsf{F}_2$ properties, which in turn would force us to add new worlds, which would mean we have to continue proof search.

We solve this problem by identifying (substituting) labels not only after the completion of but also during proof search. This preserves unprovability, but could, *a priori*, be unsound. This means that, when terminating a branch on a non-axiomatic sequent, it is still possible to extract a countermodel from it. However, when reaching only axiomatic leaves, it remains to be shown that a sound proof can be obtained from the proof attempt (potentially containing identification of labels). So, instead of doing naive proof search and then constructing a countermodel from a failed proof search by "folding" the failed sequent, we perform the folding already during the proof search, which is now a countermodel search, and then construct a proper proof from a failed countermodel search by "unfolding" the search tree. The loop-check ensuring termination has to be subtly calibrated for this final step of unfolding the proof attempt into a real proof.

4. Preliminaries: Fully Labelled Sequents

In this section we present the fully labelled sequent proof system $\mathsf{labIS4}_{\leq}$ [20] for intuitionistic modal logics we are using in order to prove decidability of IS4. The starting point is the notion of a *labelled formula* which is a pair $x{:}A$ of a label x and a formula A. A *relational atom* is either an expression xRy or $x{\leq}y$ where x and y are labels. A *(labelled) sequent* $\mathfrak{G}$ is a triple $\mathcal{R}, \Gamma \Longrightarrow \Delta$ where $\mathcal{R}$ is a set of relational atoms and Γ and Δ are multisets of labelled formulas, all written

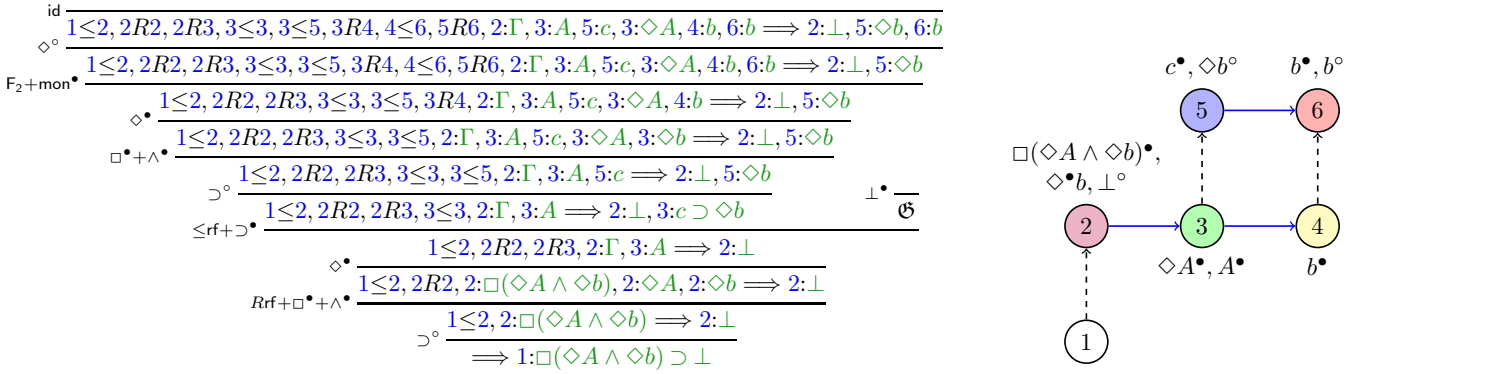


Fig. 4. Left: Proof in labIS4$_\leq$ of $\Box(\Diamond A \wedge \Diamond b) \supset \bot$ with $A = (c \supset \Diamond b) \supset \bot$, $\Gamma = \{\Box(\Diamond A \wedge \Diamond b), \Diamond b\}$ and $\mathfrak{G} = 1{\leq}2, 2R2, 2R3, 3{\leq}3, 2{:}\Gamma, 3{:}\bot \Longrightarrow 2{:}\bot$. Right: Diagrammatic representation of the top left sequent in the derivation.

as comma-separated lists. Figure 2 shows the inference rules of labIS4$_\leq$.[1] Figure 3 shows some additional rules, and we define labIS4$'_\leq$:= labIS4$_\leq \cup \{\mathsf{mon}^\bullet, \mathsf{weak}, 4^\bullet, 4^\circ, \mathsf{cont}^\bullet, \mathsf{cont}^\circ\}$.

A ***derivation*** is a finite tree where each edge is a sequent and each internal node is a rule. A derivation is a ***proof*** of the sequent at the root iff each leaf is a rule with no premises, i.e., $\bot^\bullet$ or id. As shown in [20], labIS4$_\leq$ is sound and complete for IS4.

Theorem 4.1 ([20]). *A formula A is a theorem of* IS4 *iff for every x, the sequent $\Longrightarrow x{:}A$ is derivable in* labIS4$_\leq$.

Example 4.2. On the left of Fig. 4 we have a proof in labIS4$_\leq$ for the valid formula $F = \Box(\Diamond A \wedge \Diamond b) \supset \bot$, with $A = (c \supset \Diamond b) \supset \bot$, where, for reasons of space, several rules are sometimes condensed into a single step.

Corollary 4.3. *Rules* $\mathsf{mon}^\bullet$, weak, $4^\bullet$, 4°, $\mathsf{cont}^\bullet$, *and* cont° *are admissible for* labIS4$_\leq$. *Therefore, a formula A is derivable in* labIS4$'_\leq$ *iff it is derivable in* labIS4$_\leq$.

5. From Sequents to Models

In this section we establish the promised correspondence between sequents and models.

Notation 5.1. Let $\mathfrak{G}$ be a sequent $\mathcal{R}, \Gamma \Longrightarrow \Delta$. We write

- $x \leq_\mathfrak{G} y$ iff $x{\leq}y$ occurs in $\mathcal{R}$;
- $x R_\mathfrak{G} y$ iff xRy occurs in $\mathcal{R}$;
- $\mathfrak{G}, x{:}A^\bullet$ iff $x{:}A$ occurs in Γ (in this case we also say that $A^\bullet$ ***occurs*** at x in $\mathfrak{G}$);
- $\mathfrak{G}, x{:}A^\circ$ iff $x{:}A$ occurs in Δ (in this case we also say that A° ***occurs*** at x in $\mathfrak{G}$);
- $\ell(\mathfrak{G})$ for the set of labels occurring in $\mathfrak{G}$.

This notation is used extensively throughout the paper.[2]

Example 5.2. This notation also enables us to represent sequents graphically, as shown on the right of Fig. 4, which depicts the top-left sequent of the proof showcased on the left of the figure. The sequent is represented by means of a directed graph whose nodes are the labels of the sequent (for convenience we shall use natural numbers), dashed edges are the $\leq$-relations, and solid edges are the R-relations. We can now simply write a formula $A^\bullet$ (A°) next to a label w, to indicate that the labelled formula $w{:}A$ occurs on the left-hand side (right-hand side) of $\Longrightarrow$.

Definition 5.3 (Happy labelled formula)**.** A formula $A^\bullet$ (A°) is ***happy*** at a label x in a sequent $\mathfrak{G}$, or $\mathfrak{G}, x{:}A^\bullet$ ($\mathfrak{G}, x{:}A^\circ$) is ***happy*** for short, iff the following conditions hold:

- $\mathfrak{G}, x{:}a^\bullet$ is always happy;
- $\mathfrak{G}, x{:}a^\circ$ is happy iff we do not have $\mathfrak{G}, x{:}a^\bullet$;
- $\mathfrak{G}, x{:}\bot^\bullet$ is never happy;
- $\mathfrak{G}, x{:}\bot^\circ$ is always happy;
- $\mathfrak{G}, x{:}A \wedge B^\bullet$ is happy iff $\mathfrak{G}, x{:}A^\bullet$ and $\mathfrak{G}, x{:}B^\bullet$;
- $\mathfrak{G}, x{:}A \wedge B^\circ$ is happy iff $\mathfrak{G}, x{:}A^\circ$ or $\mathfrak{G}, x{:}B^\circ$;
- $\mathfrak{G}, x{:}A \vee B^\bullet$ is happy iff $\mathfrak{G}, x{:}A^\bullet$ or $\mathfrak{G}, x{:}B^\bullet$;
- $\mathfrak{G}, x{:}A \vee B^\circ$ is happy iff $\mathfrak{G}, x{:}A^\circ$ and $\mathfrak{G}, x{:}B^\circ$;
- $\mathfrak{G}, x{:}A \supset B^\bullet$ is happy iff $\mathfrak{G}, x{:}A^\circ$ or $\mathfrak{G}, x{:}B^\bullet$;
- $\mathfrak{G}, x{:}A \supset B^\circ$ is happy iff $\mathfrak{G}, y{:}A^\bullet$ and $\mathfrak{G}, y{:}B^\circ$ for some y s.t. $x \leq_\mathfrak{G} y$;
- $\mathfrak{G}, x{:}\Box A^\bullet$ is happy iff $\mathfrak{G}, z{:}A^\bullet$ and $\mathfrak{G}, z{:}\Box A^\bullet$ for all z with $x R_\mathfrak{G} z$;
- $\mathfrak{G}, x{:}\Box A^\circ$ is happy iff $\mathfrak{G}, z{:}A^\circ$ for some y, z s.t. $x \leq_\mathfrak{G} y$ and $y R_\mathfrak{G} z$;
- $\mathfrak{G}, x{:}\Diamond A^\bullet$ is happy iff $\mathfrak{G}, y{:}A^\bullet$ for some y s.t. $x R_\mathfrak{G} y$;
- $\mathfrak{G}, x{:}\Diamond A^\circ$ is happy iff $\mathfrak{G}, y{:}A^\circ$ and $\mathfrak{G}, y{:}\Diamond A^\circ$ for all y s.t. $x R_\mathfrak{G} y$.

Otherwise, the formula is ***unhappy***.

Definition 5.4 (Happy label)**.** A label x occurring in a sequent $\mathfrak{G}$ is ***happy*** iff all formulas occurring at x in $\mathfrak{G}$ are happy. Label x is ***almost happy*** iff all formulas occurring at x are happy except, possibly, those of the shapes $\bot^\bullet$, a°, $A \supset B^\circ$, and $\Box A^\circ$. Label x is ***naively happy*** iff all formulas occurring at x are happy except, possibly, those of the shapes $\bot^\bullet$, $\Diamond A^\bullet$, a°, $A \supset B^\circ$, and $\Box A^\circ$.

Definition 5.5 (Structurally saturated sequent)**.** A sequent $\mathfrak{G}$ is ***structurally saturated*** iff the following holds:

[1] Note that labIS4$_\leq$ has no cut rule, meaning that cut is admissible. For a syntactic proof of cut elimination, see [20].

[2] The use of $\bullet$ and $\circ$ in this way goes back to Lamarche [18].

(mon$^\bullet$) if $x \leq_{\mathfrak{G}} y$ and $\mathfrak{G}, x{:}C^\bullet$, then $\mathfrak{G}, y{:}C^\bullet$;
(F_1) if $xR_{\mathfrak{G}}y$ and $y \leq_{\mathfrak{G}} z$, then there is u such that $x \leq_{\mathfrak{G}} u$ and $uR_{\mathfrak{G}}z$;
(F_2) if $xR_{\mathfrak{G}}y$ and $x \leq_{\mathfrak{G}} z$, then there is u such that $y \leq_{\mathfrak{G}} u$ and $zR_{\mathfrak{G}}u$;
($\leq$tr) if $x \leq_{\mathfrak{G}} y$ and $y \leq_{\mathfrak{G}} z$, then $x \leq_{\mathfrak{G}} z$;
($\leq$rf) $x \leq_{\mathfrak{G}} x$ for all x occurring in $\mathfrak{G}$;
(Rtr) if $xR_{\mathfrak{G}}y$ and $yR_{\mathfrak{G}}z$, then $xR_{\mathfrak{G}}z$;
(Rrf) $xR_{\mathfrak{G}}x$ for all x occurring in $\mathfrak{G}$.

Definition 5.6 (Happy sequent)**.** A sequent $\mathfrak{G}$ is ***happy*** iff it is structurally saturated and all labels in the sequent are happy.

Definition 5.7 (Model of a sequent)**.** Let $\mathfrak{G}$ be a sequent. We define the ***model*** $\mathcal{M}_{\mathfrak{G}}$ ***of*** $\mathfrak{G}$ to be the quadruple $\mathcal{M}_{\mathfrak{G}} = \langle \ell(\mathfrak{G}), \leq_{\mathfrak{G}}, R_{\mathfrak{G}}, V \rangle$ where $\ell(\mathfrak{G}), \leq_{\mathfrak{G}}, R_{\mathfrak{G}}$ are as defined in Notation 5.1 and V is the function $V\colon \ell(\mathfrak{G}) \to 2^{\mathcal{A}}$ defined such that for all atoms $a \in \mathcal{A}$ we have $a \in V(w)$ iff $\mathfrak{G}, w{:}a^\bullet$.

Theorem 5.8 (Completeness)**.** *For a happy sequent $\mathfrak{G}$, its model $\mathcal{M}_{\mathfrak{G}} = \langle \ell(\mathfrak{G}), \leq_{\mathfrak{G}}, R_{\mathfrak{G}}, V \rangle$ is a transitive and reflexive birelational model with the following two properties:*

- *if $\mathfrak{G}, x{:}A^\bullet$, then $\mathcal{M}_{\mathfrak{G}}, x \Vdash A$;*
- *if $\mathfrak{G}, x{:}A^\circ$, then $\mathcal{M}_{\mathfrak{G}}, x \nVdash A$.*

Definition 5.9 (Axiomatic sequent)**.** A sequent $\mathfrak{G}$ is ***axiomatic*** iff there is a label x such that either $\mathfrak{G}, x{:}a^\bullet$ and $\mathfrak{G}, x{:}a^\circ$ for some a, or $\mathfrak{G}, x{:}\bot^\bullet$. Otherwise, $\mathfrak{G}$ is called ***non-axiomatic***.

Remark 5.10. An axiomatic sequent $\mathfrak{G}$ is never happy, because either $\mathfrak{G}, x{:}a^\circ$ or $\mathfrak{G}, x{:}\bot^\bullet$ is unhappy.

6. Layers, Clusters, and Saturated Sequents

All derivations of $\Longrightarrow x{:}A$ obtained by a proof search in labIS4$_\leq$ have a particular structure: their labels are partitioned into *layers*, with each layer having a tree structure. This plays an important role in our proof search algorithm.

Definition 6.1 (Layer)**.** For a sequent $\mathfrak{G}$, we define the relation $R^{\leftrightarrow}_{\mathfrak{G}}$ to be the transitive and reflexive closure of $R_{\mathfrak{G}} \cup R^{-1}_{\mathfrak{G}}$. Since this is an equivalence relation, we can define a ***layer*** L in $\mathfrak{G}$ to be an equivalence class of $R^{\leftrightarrow}_{\mathfrak{G}}$.

Definition 6.2 (Layered sequent)**.** We say that a sequent $\mathfrak{G}$ is ***layered*** iff for any labels x, x', y, and y' occurring in $\mathfrak{G}$:

1) if $xR^{\leftrightarrow}_{\mathfrak{G}}y$ for $x \neq y$, then $x \not\leq_{\mathfrak{G}} y$ and $y \not\leq_{\mathfrak{G}} x$; and
2) if $xR^{\leftrightarrow}_{\mathfrak{G}}y$, $x'R^{\leftrightarrow}_{\mathfrak{G}}y'$, and $x \leq_{\mathfrak{G}} x'$ for $x \neq x'$, then $y' \not\leq_{\mathfrak{G}} y$.

For layers L_1 and L_2, we define $L_1 \leq L_2$ whenever there are labels $x \in L_1$ and $y \in L_2$ such that $x \leq_{\mathfrak{G}} y$. We write $L_1 < L_2$ iff $L_1 \leq L_2$ and $L_1 \neq L_2$.

Proposition 6.3. *For a layered structurally saturated sequent $\mathfrak{G}$, relation $\leq$ is an order relation on its layers.*

The second condition in Def. 6.2 is only needed to establish antisymmetry in Prop. 6.3. If $\leq_{\mathfrak{G}}$ on labels is already antisymmetric then the second condition follows from the first. In all sequents that we discuss here, the order relation $\leq$ on layers defines a tree structure.

Definition 6.4 (Tree-layered sequent)**.** A layered sequent is ***tree-layered*** iff (i) there is a layer L_0 such that $L_0 \leq L$ for all layers L, and (ii) for all layers L, L', and L'', whenever $L' \leq L$ and $L'' \leq L$, then either $L' \leq L''$ or $L'' \leq L'$.

Definition 6.5 (Topmost and inner layer)**.** A layer L in a layered sequent $\mathfrak{G}$ is called a ***topmost layer*** iff it is maximal with respect to $\leq$, i.e., whenever $L \leq L'$ for some layer L', then $L = L'$. Otherwise, the layer L is called an ***inner layer***.

Note that in a tree-layered structurally saturated sequent, topmost layers are the leaves of the tree structure w.r.t. $\leq$.

In a sequent $\mathfrak{G}$ constructed by labIS4$_\leq$, each layer also has a tree structure with respect to $R_{\mathfrak{G}}$. But, as said before, in order to search for a proof and a countermodel at the same time, we need to weaken this tree structure on the layers.

Definition 6.6 (Cluster)**.** If $\mathfrak{G}$ is structurally saturated, then $R_{\mathfrak{G}} \cap R^{-1}_{\mathfrak{G}}$ is an equivalence relation, and we can define a ***cluster*** C in $\mathfrak{G}$ to be an equivalence class $C = \{x_1, ..., x_n\}$ of $R_{\mathfrak{G}} \cap R^{-1}_{\mathfrak{G}}$. A cluster $C = \{x_1\}$ containing only one label is called ***singleton***. On clusters, we define two binary relations:

- $C_1 \leq_{\mathfrak{G}} C_2$ iff for all $y \in C_2$ there is $x \in C_1$ with $x \leq_{\mathfrak{G}} y$.
- $C_1 R_{\mathfrak{G}} C_2$ iff there are $x \in C_1$ and $y \in C_2$ with $xR_{\mathfrak{G}}y$.

We sometimes abuse the notation and replace one of the clusters by a label x even when $\{x\}$ is not a cluster. Nevertheless, the definitions are then applied verbatim to $\{x\}$.

Proposition 6.7. *For a structurally saturated sequent $\mathfrak{G}$, relation $R_{\mathfrak{G}}$ is an order and $\leq_{\mathfrak{G}}$ is a preorder on its clusters. If $\mathfrak{G}$ is layered, then $\leq_{\mathfrak{G}}$ is also an order.*

Definition 6.8 (Tree-clustered sequent)**.** A structurally saturated sequent $\mathfrak{G}$ is called ***tree-clustered*** iff (i) for any clusters C' and C'' there is a cluster C, such that $CR_{\mathfrak{G}}C'$ and $CR_{\mathfrak{G}}C''$, and (ii) for all clusters C, C', and C'', whenever $C'R_{\mathfrak{G}}C$ and $C''R_{\mathfrak{G}}C$, then either $C'R_{\mathfrak{G}}C''$ or $C''R_{\mathfrak{G}}C'$.

Definition 6.9 (Happy layer)**.** A layer L is ***happy/almost happy/naively happy*** iff all labels of L are happy/almost happy/naively happy respectively.

Remark 6.10. Observe that the definitions of happy labelled formulas for $A \supset B^\bullet$ and $\Box A^\bullet$ in Def. 5.3 are different from the forcing conditions of the corresponding connectives. This mismatch is intentional with the missing conditions on $\leq$ outsourced to (mon$^\bullet$)-structural saturation (Def. 5.5) instead. As a result, happiness becomes almost a local property for layers: if a sequent is modified (by adding labelled formulas or relational atoms) outside of a given happy layer, the only formulas whose happiness needs to be reinspected are $A \supset B^\circ$ and $\Box A^\circ$. We use this observation in the notion of stable sequents below, where inner layers are never modified and always remain happy.

Definition 6.11 (Stable and saturated sequent)**.** A sequent $\mathfrak{G}$ is ***stable*** iff it is tree-layered and tree-clustered and, additionally, all its inner layers are happy. A sequent $\mathfrak{G}$ is ***saturated*** iff it is stable and all its topmost layers are almost happy. It is ***semi-saturated*** iff it is stable and all its topmost

layers are naively happy. A set $\mathfrak{S}$ of sequents is called ***stable/saturated/semi-saturated*** iff it is finite and all elements of $\mathfrak{S}$ are stable/saturated/semi-saturated respectively.

Since tree-layered sequents are layered and tree-clustered sequents are structurally saturated (in fact, the definition of clusters does not make sense otherwise), all stable sequents are layered and structurally saturated.

7. Search Algorithm

The algorithm that we will present here works on stable sequents. Roughly speaking, we work on the topmost layers of such sequents until these layers are almost happy. This is achieved by first semi-saturating (which essentially means applying all inference rules to a layer that do not add new labels to the sequent), then picking an unhappy $x{:}\Diamond A^\bullet$ in the layer and making it happy (either by applying the $\Diamond^\bullet$-rule or, in case of loop detection, by creating a cluster). Then we semi-saturate again, and continue until all $x{:}\Diamond A^\bullet$ in the layer are happy and, therefore, the layer is almost happy.

After all layers are almost happy, the only unhappy formulas remaining in the resulting saturated sequent are in topmost layers and of the shape $x{:}\bot^\bullet$, $x{:}a^\circ$, $x{:}A \supset B^\circ$ or $x{:}\Box A^\circ$. For the first two shapes, the sequent is axiomatic, and we can stop working on it.

If all unhappy formulas are in topmost layers and of the shape $x{:}A \supset B^\circ$ or $x{:}\Box A^\circ$, we make them happy by creating new layers that become new topmost layers to be saturated while formerly topmost layers turn into happy inner layers. This process is repeated, until we see a repetition of layers. In order to achieve termination of this process, we implement two other loop detection mechanisms (different from the one used for $x{:}\Diamond A^\bullet$ formulas).

In the remainder of this section, we formally introduce the concepts needed to understand the details of this algorithm, which is then shown in Fig. 7.

A. Semi-saturation

We begin by making happy all formulas that are not of the form $x{:}\bot^\bullet$, $x{:}a^\circ$, $x{:}\Diamond A^\bullet$, $x{:}A \supset B^\circ$, or $x{:}\Box A^\circ$.

Definition 7.1 (Semi-saturation)**.** We define a rewrite relation $\rightsquigarrow_{\mathsf{s}}$ on sets $\mathfrak{S}$ of sequents: $\mathfrak{S} \rightsquigarrow_{\mathsf{s}} \mathfrak{S}'$ iff there is a sequent $\mathfrak{G} \in \mathfrak{S}$ such that some labelled formula C of one of the below shapes occurs at x in $\mathfrak{G}$ and is unhappy, and $\mathfrak{S}'$ is obtained by replacing sequent $\mathfrak{G}$ in set $\mathfrak{S}$ either with sequent $\mathfrak{G}'$ according to cases 1)–4) below, or with sequents $\mathfrak{G}'$ and $\mathfrak{G}''$ according to cases 5)–7) below, as follows:

1) for $C = A \wedge B^\bullet$, obtain $\mathfrak{G}'$ by adding to $\mathfrak{G}$ all of $x{:}A^\bullet$ and $x{:}B^\bullet$ that are missing;
2) for $C = A \vee B^\circ$, obtain $\mathfrak{G}'$ by adding to $\mathfrak{G}$ all of $x{:}A^\circ$ and $x{:}B^\circ$ that are missing;
3) for $C = \Box A^\bullet$, obtain $\mathfrak{G}'$ by adding to $\mathfrak{G}$ all $z{:}A^\bullet$ and $z{:}\Box A^\bullet$ that are missing whenever $xR_{\mathfrak{G}}z$;
4) for $C = \Diamond A^\circ$, obtain $\mathfrak{G}'$ by adding to $\mathfrak{G}$ all $y{:}A^\circ$ and $y{:}\Diamond A^\circ$ that are missing whenever $xR_{\mathfrak{G}}y$;
5) for $C = A \vee B^\bullet$, obtain $\mathfrak{G}'$ (respectively $\mathfrak{G}''$) by adding to $\mathfrak{G}$ the labelled formula $x{:}A^\bullet$ (respectively $x{:}B^\bullet$);
6) for $C = A \wedge B^\circ$, obtain $\mathfrak{G}'$ (respectively $\mathfrak{G}''$) by adding to $\mathfrak{G}$ the labelled formula $x{:}A^\circ$ (respectively $x{:}B^\circ$);
7) for $C = A \supset B^\bullet$, obtain $\mathfrak{G}'$ (respectively $\mathfrak{G}''$) by adding to $\mathfrak{G}$ the labelled formula $x{:}A^\circ$ (respectively $x{:}B^\bullet$).

We write $\rightsquigarrow_{\mathsf{s}}^*$ for the transitive and reflexive closure of $\rightsquigarrow_{\mathsf{s}}$. If $\mathfrak{S} \rightsquigarrow_{\mathsf{s}}^* \mathfrak{S}'$ and $\mathfrak{S}'$ is in normal form w.r.t. $\rightsquigarrow_{\mathsf{s}}$, i.e., if all labels occurring in all sequents from $\mathfrak{S}'$ are naively happy, then $\mathfrak{S}'$ is called the ***semi-saturation*** of $\mathfrak{S}$.

The use of this term is justified because, the semi-saturation of a stable set of sequents is semi-saturated.

Lemma 7.2. *All the following statements hold:*

a) If a set $\mathfrak{S}$ is stable and $\mathfrak{S} \rightsquigarrow_{\mathsf{s}} \mathfrak{S}'$, then $\mathfrak{S}'$ is stable.
b) On finite sets $\mathfrak{S}$, the rewrite relation $\rightsquigarrow_{\mathsf{s}}$ is terminating.
c) A set $\mathfrak{S}$ is semi-saturated iff it is stable and in normal form w.r.t. $\rightsquigarrow_{\mathsf{s}}$.

B. Saturation

In the next step we show how to deal with unhappy $x{:}\Diamond A^\bullet$. This is the first source of non-termination, and for this we employ the same method that is commonly used for classical S4. However, whereas in the case of classical S4, loop detection completes the proof search in the current branch, here we have to continue proof search. For this reason we realize loops by clusters.

Definition 7.3 (Equivalent labels/clusters)**.** Labels x and y occurring in sequents $\mathfrak{G}$ and $\mathfrak{H}$ respectively are ***equivalent***, in symbols ${}_{\mathfrak{G}}x \sim {}_{\mathfrak{H}}y$, iff $\mathfrak{G}, x{:}A^\bullet \iff \mathfrak{H}, y{:}A^\bullet$ and also $\mathfrak{G}, x{:}A^\circ \iff \mathfrak{H}, y{:}A^\circ$ for all formulas A. If $\mathfrak{G}$ and $\mathfrak{H}$ are clear from the context, we simply write $x \sim y$. For structurally saturated sequents $\mathfrak{G}$ and $\mathfrak{H}$, we can generalize this to clusters: ${}_{\mathfrak{G}}C_1 \sim {}_{\mathfrak{H}}C_2$ iff there is a bijection $f\colon C_1 \to C_2$, such that $fx \sim x$ for all $x \in C_1$.

Definition 7.4 (Having no past)**.** A label x in a sequent $\mathfrak{G}$ ***has no past*** iff $y \leq_{\mathfrak{G}} x$ implies $y = x$ for all y.

Definition 7.5 (Saturation)**.** We define a binary relation $\rightsquigarrow_\Diamond$ on sequents as follows: $\mathfrak{G} \rightsquigarrow_\Diamond \mathfrak{G}'$ iff there is a label y and a formula A, such that $\mathfrak{G}, y{:}\Diamond A^\bullet$ is unhappy, and all labels $u \neq y$ with $uR_{\mathfrak{G}}y$ but not $yR_{\mathfrak{G}}u$ are almost happy, and either

1) Option 1: there is a label $x \neq y$ such that $xR_{\mathfrak{G}}y$ and $x \sim y$, and $\mathfrak{G}, x{:}\Diamond A^\bullet$ is happy, and every label u with $xR_{\mathfrak{G}}u$ has no past in $\mathfrak{G}$. Then $\mathfrak{G}'$ is obtained from $\mathfrak{G}$ by first substituting x for each occurrence of y and then closing the resulting $R_{\mathfrak{G}}$ under transitivity; or
2) Option 2: There is no such label x. Then $\mathfrak{G}'$ is obtained from $\mathfrak{G}$ by first adding $z \leq z$, yRz, $z{:}A^\bullet$ for some fresh label z, and then closing the resulting $R_{\mathfrak{G}}$ under transitivity and reflexivity.

For sets $\mathfrak{S}$ and $\mathfrak{S}'$ of sequents, we write $\mathfrak{S} \rightsquigarrow_\Diamond \mathfrak{S}'$ iff $\mathfrak{S}$ is semi-saturated, $\mathfrak{G} \rightsquigarrow_\Diamond \mathfrak{G}'$ for some $\mathfrak{G} \in \mathfrak{S}$ and $\mathfrak{S}'$ is a semi-

saturation of $(\mathfrak{S}\backslash\{\mathfrak{G}\})\cup\{\mathfrak{G}'\}$.[3] We write $\rightsquigarrow^*_\Diamond$ for the transitive and reflexive closure of $\rightsquigarrow_\Diamond$. If $\mathfrak{S}\rightsquigarrow^*_\Diamond\mathfrak{S}'$ and $\mathfrak{S}'$ is in normal form w.r.t. $\rightsquigarrow_\Diamond$, then $\mathfrak{S}'$ is called a ***saturation*** of $\mathfrak{S}$.

As before, this term is justified because, a saturation of a semi-saturated set of sequents is saturated.

Lemma 7.6. *All the following statements hold:*

a) If a set $\mathfrak{S}$ is semi-saturated and $\mathfrak{S}\rightsquigarrow_\Diamond\mathfrak{S}'$, then $\mathfrak{S}'$ is semi-saturated.
b) The rewrite relation $\rightsquigarrow_\Diamond$ is terminating.
c) A set $\mathfrak{S}$ is saturated iff it is semi-saturated and in normal form w.r.t. $\rightsquigarrow_\Diamond$.

C. Lifting saturation

After having discussed how to saturate sets of sequents, we are now going to show how to deal with unhappy $A\supset B^\circ$ and $\Box A^\circ$. For this, we have to create new layers. The naive way would be to just copy the R-structure of the layer that contains the unhappy $x{:}A\supset B^\circ$ or $x{:}\Box A^\circ$. However, due to the presence of clusters, we sometimes need to create two copies of the cluster that contains the unhappy formula.

Definition 7.7 (Sequent sum)**.** Let $\mathfrak{G}$ and $\mathfrak{G}'$ be sequents $\mathcal{R},\Gamma\Longrightarrow\Delta$ and $\mathcal{R}',\Gamma'\Longrightarrow\Delta'$ respectively. We define their ***sum*** $\mathfrak{G}+\mathfrak{G}'$ to be the sequent $\mathcal{R},\mathcal{R}',\Gamma,\Gamma'\Longrightarrow\Delta,\Delta'$.

Construction 7.8 (Layer Lifting I)**.** Let $\mathfrak{G}$ be a saturated sequent, L be one of its topmost layers, and let $x\in L$. Then x belongs to a cluster $C_x=\{x_1,\ldots,x_h\}$ with $h\geq 1$ and $x=x_m$ for some $1\leq m\leq h$, which we often abbreviate as $m=1..h$. In particular, $x_iR_{\mathfrak{G}}x_{i'}$ for any $i,i'=1..h$. Let $\{y_1,\ldots,y_l\}=L\setminus C_x$ where $l\geq 0$. Then, if $h=1$, let $\hat{L}$ be the set of fresh labels $\{\hat{y}_1,\ldots,\hat{y}_l,\hat{x}\}$. Otherwise, if $h>1$, let $\hat{L}:=\{\hat{y}_1,\ldots,\hat{y}_l,\hat{x},\hat{x}'_1,\ldots,\hat{x}'_h,\hat{x}''_1,\ldots,\hat{x}''_h\}$ where again all labels are fresh. We define $\mathfrak{G}{\uparrow^x}$ to consist of:

1) relational atoms $v{\leq}v$ and vRv for all $v\in\hat{L}$;
2) for each $i=1..l$, each $j=1..h$, and each $w\in\ell(\mathfrak{G})$:
   a) relational atom $w{\leq}\hat{y}_i$ whenever $w{\leq_{\mathfrak{G}}}y_i$,
   b) relational atom $w{\leq}\hat{x}$ whenever $w{\leq_{\mathfrak{G}}}x$,
   c) only for $h>1$: relational atoms $w{\leq}\hat{x}'_j$ and $w{\leq}\hat{x}''_j$ whenever $w{\leq_{\mathfrak{G}}}x_j$;
3) for all $i,i'=1..l$ and $j,j'=1..h$,
   a) relational atom $\hat{y}_iR\hat{y}_{i'}$ whenever $y_iR_{\mathfrak{G}}y_{i'}$,
   b) only for $h>1$: relational atoms $\hat{y}_iR\hat{x}'_j$ and $\hat{y}_iR\hat{x}''_j$ whenever $y_iR_{\mathfrak{G}}x_j$,
   c) relational atom $\hat{y}_iR\hat{x}$ whenever $y_iR_{\mathfrak{G}}x$,
   d) only for $h>1$: relational atoms $\hat{x}'_jR\hat{y}_i$ and $\hat{x}''_jR\hat{y}_i$ whenever $x_jR_{\mathfrak{G}}y_i$,
   e) relational atom $\hat{x}R\hat{y}_i$ whenever $xR_{\mathfrak{G}}y_i$,
   f) only for $h>1$: relational atoms $\hat{x}'_jR\hat{x}'_{j'}$, $\hat{x}'_jR\hat{x}$, $\hat{x}R\hat{x}''_j$, and $\hat{x}''_jR\hat{x}''_{j'}$.
4) For each $i=1..l$, each $j=1..h$, and each C, add:
   a) labelled formulas $\hat{y}_i{:}C^\bullet$ whenever $\mathfrak{G},y_i{:}C^\bullet$,
   b) only for $h>1$: labelled formulas $\hat{x}'_j{:}C^\bullet$ and $\hat{x}''_j{:}C^\bullet$ whenever $\mathfrak{G},x_j{:}C^\bullet$,
   c) labelled formulas $\hat{x}{:}C^\bullet$ whenever $\mathfrak{G},x{:}C^\bullet$.

[3] Strictly speaking, in the first case $\mathfrak{G}'$ is already semi-saturated, but to simplify the presentation, we semi-saturate in both cases.

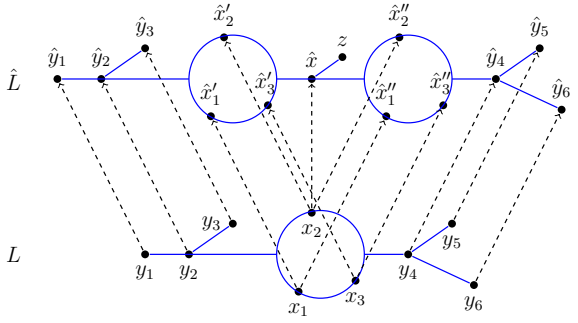


Fig. 5. Layer lifting as defined in Constructions 7.8 and 7.9

This construction lifts a layer L of $\mathfrak{G}$ w.r.t. a label $x\in L$. If x is a singleton cluster, this is a simple lifting as one might expect. Otherwise, if x is in a non-singleton cluster, this cluster is duplicated and the lifting of x is put in between the two copies, as indicated in Fig. 5 (ignore label z for the moment).

In Construction 7.8, Points 1)–2) ensure that in $\mathfrak{G}+\mathfrak{G}{\uparrow^x}$, the new layer is indeed above L as intended and ($\leq$tr), ($\leq$rf), and (Rrf) are satisfied; Point 3) ensures (Rtr), (F_1), and (F_2); Point 4) ensures ($\mathsf{mon}^\bullet$). Thus, $\mathfrak{G}+\mathfrak{G}{\uparrow^x}$ is tree-clustered, tree-layered, and structurally saturated. However, it may not be stable because L is now an inner layer. It is almost happy due to the saturation of $\mathfrak{G}$ but may contain unhappy $A\supset B^\circ$ and $\Box B^\circ$ formulas.

Construction 7.9 (Layer Lifting II)**.** Let $\mathfrak{G}$ be a saturated sequent with $\mathfrak{G},x{:}F^\circ$ being unhappy for some label x and some formula F of shape $A\supset B$ or $\Box B$. We reuse the notation from Construction 7.8 (note that L must be a topmost layer of $\mathfrak{G}$ due to it being saturated). We define $\mathfrak{G}{\uparrow^{x:F}}$ as follows:

- If $F=A\supset B$, then $\mathfrak{G}{\uparrow^{x:A\supset B}}$ is $\mathfrak{G}{\uparrow^x}$ to which we add labelled formulas $\hat{x}{:}A^\bullet$ and $\hat{x}{:}B^\circ$ and call $\hat{x}$ a ***suricata label of*** x
- If $A=\Box B$, then $\mathfrak{G}{\uparrow^{x:\Box B}}$ is $\mathfrak{G}{\uparrow^x}$ to which we add zRz, $z{\leq}z$, and $z{:}B^\circ$ for a fresh label z, and additionally add relational atoms vRz whenever $v\in\hat{L}$ and $vR\hat{x}$ is in $\mathfrak{G}{\uparrow^x}$. Here z is called a ***suricata label of*** x.

Informally speaking, sequent $\mathfrak{G}+\mathfrak{G}{\uparrow^{x:F}}$ contains precisely the relational atoms and labeled formulas that need to be added to $\mathfrak{G}+\mathfrak{G}{\uparrow^x}$ so that (i) the unhappy $\mathfrak{G},x{:}F^\circ$ becomes happy (the suricata label contains the white formula responsible for this happiness) and (ii) the result is still structurally saturated (and tree-layered and cluster-layered). Figure 5 shows the case of $x{:}\Box B^\circ$ with the additional fresh label z (that has no past). In order to also preserve the property of being stable, we need to add such a layer for all unhappy $\mathfrak{G},x{:}F^\circ$ in the same layer L.

Construction 7.10 (Layer Lifting III)**.** Let L be a topmost layer in a saturated non-axiomatic sequent $\mathfrak{G}$ where $x_1{:}F_1^\circ$, …, $x_m{:}F_m^\circ$ (of the form $A\supset B$ or $\Box B$) are all the unhappy

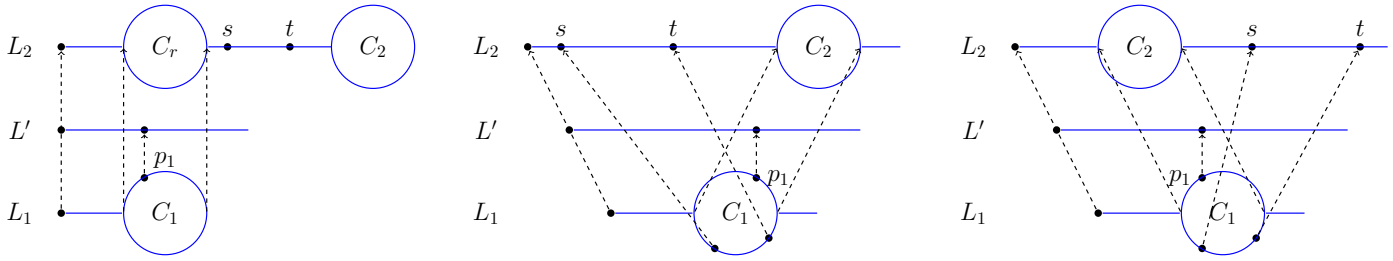


Fig. 6. Left: Structure of an unhappy R-triangle loop defined in Def. 7.16 Middle and Right: Structure of an unhappy U-triangle loop defined in Def. 7.17

formulas in L. Then $\mathfrak{G}{\uparrow^L} := \mathfrak{G}{\uparrow^{x_1:F_1}} + \cdots + \mathfrak{G}{\uparrow^{x_m:F_m}}$ where we assume the sets of fresh labels introduced in each of $\mathfrak{G}{\uparrow^{x_1:F_1}}, \ldots, \mathfrak{G}{\uparrow^{x_m:F_m}}$ to be pairwise disjoint.

If we do this addition of layers naively, we will not terminate. Therefore, we need a loop-check, which could be viewed as a generalized version of a loop-check in sequent calculi for intuitionistic logic.

Definition 7.11 (Simulation of layers)**.** Let L' and L be layers in a layered sequent $\mathfrak{G}$. A ***layer simulation*** between L' and L is a non-empty binary relation $\mathsf{S} \subseteq (L' \times L) \cap \sim$ such that for all $x' \in L'$, $x, y \in L$,

(S1) whenever $x' \mathsf{S} x$ and $x R_{\mathfrak{G}} y$, there exists $y' \in L'$ such that $x' R_{\mathfrak{G}} y'$ and $y' \mathsf{S} y$, and

(S2) whenever $x' \mathsf{S} x$ and $y R_{\mathfrak{G}} x$, there exists $y' \in L'$ such that $y' R_{\mathfrak{G}} x'$ and $y' \mathsf{S} y$.

We say L' ***simulates*** L iff there is a layer simulation between L' and L.

Proposition 7.12. *Let L' and L be layers in a layered sequent $\mathfrak{G}$. If S is a layer simulation between L' and L then, for all $x \in L$, there is a $x' \in L'$ such that $x' \mathsf{S} x$.*

Definition 7.13. Let L be a topmost layer in a layered sequent $\mathfrak{G}$. We say L ***is simulated*** iff there is a layer L' in $\mathfrak{G}$, such that $L' < L$ and L' simulates L.

Definition 7.14 (Lifting Saturation)**.** Let $\mathfrak{G}$ be a saturated non-axiomatic sequent and $L_1, \ldots, L_n$ be all topmost layers in $\mathfrak{G}$ that are not simulated. We define the ***lifting saturation*** of $\mathfrak{G}$ as $\mathfrak{G}{\uparrow} := \mathfrak{G} + \mathfrak{G}{\uparrow^{L_1}} + \cdots + \mathfrak{G}{\uparrow^{L_n}}$ where the sets of fresh labels introduced in each of $\mathfrak{G}{\uparrow^{L_1}}, \ldots, \mathfrak{G}{\uparrow^{L_n}}$ are pairwise disjoint.

Lemma 7.15. *If $\mathfrak{G}$ is saturated, then $\mathfrak{G}{\uparrow}$ is stable.*

D. Loop saturation

We have presented all the steps that make the sequent grow, by adding new labelled formulas or relational atoms. As a result, unhappy formulas become happy, or the sequent becomes structurally saturated. Definition 7.14 exhibits a condition under which we stop applying the growing steps: namely, when a topmost layer is simulated by a layer below.

However, observe that, in general, new layers are larger than previous ones and there are two sources for the growth. First, in the case of an unhappy $x{:}\Box B^{\circ}$, a fresh label z is added, and second, if label x is in a non-singleton cluster, this cluster is duplicated. Both effects can be seen in Fig. 5.

For this reason, we need to find a way to *shrink* a layer. This will be done by creating clusters similar to how Def. 7.5 does for $\Diamond^{\bullet}$. The difference is that this time the potential repetition will involve several layers rather than being local to a single layer. The difficulty is that some part of a layer L_1 will be repeated in a layer L_2 occurring above it, but in order to keep the sequent tree-layered, we cannot create clusters across several layers. The solution we implement here is to create a cluster inside layer L_2 for a part that would be repeated in a future layer, provided that we can repeat in layer L_2 what happened in layer L_1. We call such loops *triangle loops*. There are two kinds of such loops. The first kind occurs if the cluster to be created in L_2 is in a part of L_2 that has no past in L_1. This could be caused for example by repetitions of $\Box^{\circ}$. We call these loops *R-triange loops*. The second kind occurs when the repetition is caused by a repeated duplication of clusters in the layer lifting (see Fig. 5). We call these loops *U-triangle loops*.

Before we give the formal definitions, observe that all new layers that are created in our algorithm are of the shape $\mathfrak{G}{\uparrow^{x:F}}$, as defined in Construction 7.9, and each such layer contains exactly one suricata label. This can, therefore, be called *the suricata label of the layer*, and it is (immediately after the lifting) the only label that contains a white formula.[4]

Definition 7.16 (R-triangle loop)**.** Let $\mathfrak{G}$ be a saturated sequent with two layers $L_1 < L_2$. We say that clusters $C_1 \subseteq L_1$ and $C_2 \subseteq L_2$ form an ***R-triangle loop*** iff

1) $C_1 \sim C_2$;
2) there is a label $p_1 \in C_1$ such that there is a layer L' with $L_1 < L' \leq L_2$ that contains a suricata label of p_1;
3) there is a cluster C_r such that $C_1 \leq_{\mathfrak{G}} C_r$, and $C_r R_{\mathfrak{G}} C_2$, and no label $v \in L_2 \setminus C_r$ with $C_r \; R_{\mathfrak{G}} \; v \; R_{\mathfrak{G}} \; C_2$ has a past in L_1, i.e., $u \not\leq_{\mathfrak{G}} v$ for any $u \in L_1$.

This R-triangle loop is ***unhappy*** iff additionally

4) L_2 is a topmost layer;
5) there are labels $s, t \in L_2 \setminus C_2$ such that $s \sim t$, and $s \neq t$, and $C_r \; R_{\mathfrak{G}} \; s \; R_{\mathfrak{G}} \; t \; R_{\mathfrak{G}} \; C_2$;
6) there is no suricata label u with $C_r \; R_{\mathfrak{G}} \; u \; R_{\mathfrak{G}} \; t$;
7) we do *not* have $C_2 \; R_{\mathfrak{G}} \; t$.

(Note that s and t may be in a common cluster.)

Figure 6 (Left) illustrates this definition.

[4]On the other hand, observe that a label x can have several suricata labels in other layers if there is more than one $\supset^{\circ}$- or $\Box^{\circ}$-formula in x.

0) Given a formula F, define $\mathfrak{G}_0(F)$ to be the sequent $r{\leq}r, rRr \Longrightarrow r{:}F$ and let $\boldsymbol{\mathfrak{G}}'_0 := \{\mathfrak{G}_0(F)\}$.
1) For set $\boldsymbol{\mathfrak{G}}'_i$, calculate a full saturation $\boldsymbol{\mathfrak{G}}_i$.
2) If all sequents in $\boldsymbol{\mathfrak{G}}_i$ are axiomatic, then terminate.
   The formula F is provable and we can give a proof of $\Longrightarrow r{:}F$ in labIS4$_{\leq}$ (see Sect. 9).
3) Otherwise, pick a non-axiomatic sequent $\mathfrak{G}_i \in \boldsymbol{\mathfrak{G}}_i$ and compute its lifting saturation $\mathfrak{G}_i{\uparrow}$ (see Def. 7.14).
4) If $\mathfrak{G}_i{\uparrow} = \mathfrak{G}_i$, then terminate.
   The formula F is not provable, and sequent $\mathfrak{G}_i$ defines a countermodel (see Sect. 8).
5) Otherwise, set $\boldsymbol{\mathfrak{G}}'_{i+1} := \big(\boldsymbol{\mathfrak{G}}_i \setminus \{\mathfrak{G}_i\}\big) \cup \{\mathfrak{G}_i{\uparrow}\}$.
6) Go to Step 1).

Fig. 7. Proof search algorithm

Definition 7.17 (U-triangle loop)**.** Let $\mathfrak{G}$ be a saturated sequent with two layers $L_1 < L_2$. We say that clusters $C_1 \subseteq L_1$ and $C_2 \subseteq L_2$ form a ***U-triangle loop*** iff

1) $C_1 \sim C_2$;
2) there is a label $p_1 \in C_1$ such that there is a layer L' with $L_1 < L' \leq L_2$ that contains a suricata label of p_1;
3) $C_1 {\leq_{\mathfrak{G}}} C_2$.

The U-triangle loop is ***unhappy*** iff additionally

4) L_2 is a topmost layer;
5) there are labels $s, t \in L_2 \setminus C_2$ such that $C_1 {\leq_{\mathfrak{G}}} s$, and $C_1 {\leq_{\mathfrak{G}}} t$, and $s \sim t$, and $s \neq t$, and either $C_2\, R_{\mathfrak{G}}\, s\, R_{\mathfrak{G}}\, t$ or $s\, R_{\mathfrak{G}}\, t\, R_{\mathfrak{G}}\, C_2$;
6) there is no suricata label u with $s\, R_{\mathfrak{G}}\, u\, R_{\mathfrak{G}}\, t$;
7) we do *not* have $s\, R_{\mathfrak{G}}\, C_2\, R_{\mathfrak{G}}\, t$.

(Again, s and t may be in the same cluster.)

The two possibilities envisioned by this definition, depending on whether $C_2\, R_{\mathfrak{G}}\, s\, R_{\mathfrak{G}}\, t$ or $s\, R_{\mathfrak{G}}\, t\, R_{\mathfrak{G}}\, C_2$, are illustrated in Fig. 6 (Middle and Right).

Informally, we speak of a *triangle loop* when we can reproduce the steps that started with the creation of L' and led to L_2. The loop is unhappy if we can observe some repetition in the new part of L_2. If this is the case, we can collapse this repetition by creating a cluster, or shrinking an existing cluster, as follows:

Definition 7.18 (Loop saturation)**.** Let $\mathfrak{G}$ and $\mathfrak{G}'$ be saturated sequents. We write $\mathfrak{G} \rightsquigarrow_{\circlearrowleft} \mathfrak{G}'$ iff there is an unhappy R-triangle or U-triangle loop in $\mathfrak{G}$ where the labels s and t are as in Def. 7.16 or Def. 7.17 respectively and $\mathfrak{G}'$ is obtained from $\mathfrak{G}$ by substituting s for all occurrences of t and closing the resulting $R_{\mathfrak{G}}$ under transitivity. For sets $\boldsymbol{\mathfrak{G}}$ and $\boldsymbol{\mathfrak{G}}'$ of sequents, we write $\boldsymbol{\mathfrak{G}} \rightsquigarrow_{\circlearrowleft} \boldsymbol{\mathfrak{G}}'$ iff $\boldsymbol{\mathfrak{G}}$ is saturated, $\mathfrak{G} \rightsquigarrow_{\circlearrowleft} \mathfrak{G}'$ for some $\mathfrak{G} \in \boldsymbol{\mathfrak{G}}$, and $\boldsymbol{\mathfrak{G}}' = \big(\boldsymbol{\mathfrak{G}} \setminus \{\mathfrak{G}\}\big) \cup \{\mathfrak{G}'\}$. We write $\rightsquigarrow^*_{\circlearrowleft}$ for the transitive and reflexive closure of $\rightsquigarrow_{\circlearrowleft}$. If $\boldsymbol{\mathfrak{G}} \rightsquigarrow^*_{\circlearrowleft} \boldsymbol{\mathfrak{G}}'$ and $\boldsymbol{\mathfrak{G}}'$ is in normal form w.r.t. $\rightsquigarrow_{\circlearrowleft}$, then $\boldsymbol{\mathfrak{G}}'$ is a ***loop saturation*** of $\boldsymbol{\mathfrak{G}}$.

As before, this term is justified because a loop saturation of a saturated set of sequents is saturated.

Lemma 7.19. *It holds that:*

a) If $\boldsymbol{\mathfrak{G}}$ is saturated and $\boldsymbol{\mathfrak{G}} \rightsquigarrow_{\circlearrowleft} \boldsymbol{\mathfrak{G}}'$, then $\boldsymbol{\mathfrak{G}}'$ is saturated.
b) The rewrite relation $\rightsquigarrow_{\circlearrowleft}$ is terminating.

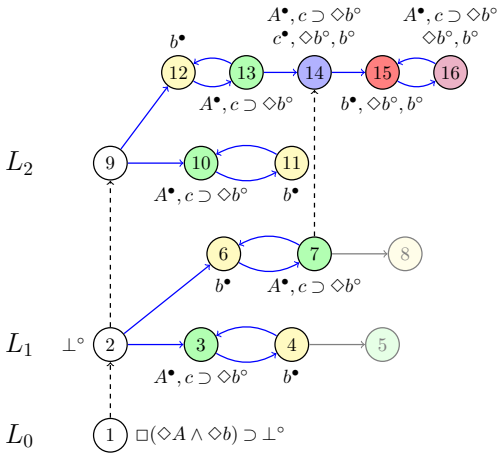


Fig. 8. Formula $\Box(\Diamond A \wedge \Diamond b) \supset \bot$, with $A = (c \supset \Diamond b) \supset \bot$. The set $\Gamma^\bullet = \{\Box(\Diamond A \wedge \Diamond b)^\bullet, \Diamond A \wedge \Diamond b^\bullet, \Diamond A^\bullet, \Diamond b^\bullet\}$ is in all nodes, except 1.

Definition 7.20 (Full saturation)**.** Let $\boldsymbol{\mathfrak{G}}$ be a set of stable sequents. Let $\boldsymbol{\mathfrak{G}}'$ be a semi-saturation of $\boldsymbol{\mathfrak{G}}$, and $\boldsymbol{\mathfrak{G}}''$ be a saturation of $\boldsymbol{\mathfrak{G}}'$, and $\boldsymbol{\mathfrak{G}}'''$ be a loop saturation of $\boldsymbol{\mathfrak{G}}''$. Then $\boldsymbol{\mathfrak{G}}'''$ is a ***full saturation*** of $\boldsymbol{\mathfrak{G}}$.

Corollary 7.21. *If a set $\boldsymbol{\mathfrak{G}}$ of sequents is stable and $\boldsymbol{\mathfrak{G}}'''$ is its full saturation, then $\boldsymbol{\mathfrak{G}}'''$ is saturated.*

E. Search algorithm

We have now all the ingredients for our proof/countermodel search algorithm, which is presented in Fig. 7. It produces a sequence $\boldsymbol{\mathfrak{G}}_0, \boldsymbol{\mathfrak{G}}_1, \boldsymbol{\mathfrak{G}}_2, \ldots$ of sets of saturated sequents using the transformations discussed in this section. We terminate at step i if either all sequents in $\boldsymbol{\mathfrak{G}}_i$ are axiomatic (in that case we can produce a proof in labIS4$_{\leq}$, see Sect. 9) or there is a sequent $\mathfrak{G}_i \in \boldsymbol{\mathfrak{G}}_i$ that does not grow anymore (in that case we can construct a countermodel, see Sect. 8).

Example 7.22. Figure 8 contains a (partial) diagrammatic representation of one sequent generated by the algorithm, when run on the valid IS4 formula from Example 4.2. At layer L_1 clusters $\{3, 4\}$ and $\{6, 7\}$ are generated by saturation (Option 1 of Def. 7.5). Layer L_2 is created by lifting saturation (Def. 7.14), where 14 is a suricata label and both clusters

$\{12, 13\}$ and $\{15, 16\}$ come from $\{6, 7\}$. The sequent is axiomatic because of 15.

8. Countermodel Construction

Assume we initiate the algorithm with a formula F. If we terminate at Step 4, we have found a saturated sequent $\mathfrak{G}_i$ such that $\mathfrak{G}_i, r{:}F^\circ$ and $\mathfrak{G}{\uparrow_i} = \mathfrak{G}_i$. This means that each topmost layer of $\mathfrak{G}_i$ is either happy or simulated. Thus, for each unhappy layer L there is some happy inner layer L' that simulates L via a simulation S_L. We now define $\mathfrak{G}_i^\star$ to be obtained from $\mathfrak{G}_i$ by adding a relational atom $x{\leq}x'$ whenever $x'\mathsf{S}_L x$ for some unhappy layer L, and by closing the result under transitivity of $\leq$. This makes all unhappy $A \supset B^\circ$ and $\Box A^\circ$ in the topmost layers happy and preserves structural saturation. Hence, $\mathfrak{G}_i^\star$ is a happy sequent, which allows to apply Theorem 5.8 to obtain a finite countermodel for F.

Theorem 8.1. *If the algorithm shown in Fig. 7 terminates in Step 4, then formula F is not a theorem of* IS4.

Example 8.2. Figure 9 represents (a $\leq$-branch of) a sequent, generated by the algorithm when run on formula (3) from Sect. 3, that is on $\Box(A \wedge B) \supset \bot$ with $A = \Box a \supset \bot$ and $B = (a \supset \bot) \supset \bot$. If we include label 12 (but remove atom $9 \leq 14$), the figure depicts the moment when the algorithm finds an unhappy R-triangle loop: employing the terminology from Def. 7.16, we take $C_1 = \{3\}$, $p_1 = 3$, $C_2 = \{13\}$, $s = 14$, $C_r = \{s\}$, and $t = 12$. Then 12 is replaced with 14, and after the loop saturation is no longer present in the sequent. Then the search on the depicted $\leq$-branch stops because the topmost layer L_6 can be simulated by layer L_4. By adding relational atoms $15{\leq}8$, $14{\leq}6$, and $13{\leq}7$ to the sequent (Theorem 8.1), we obtain (a part of) the countermodel for our formula.

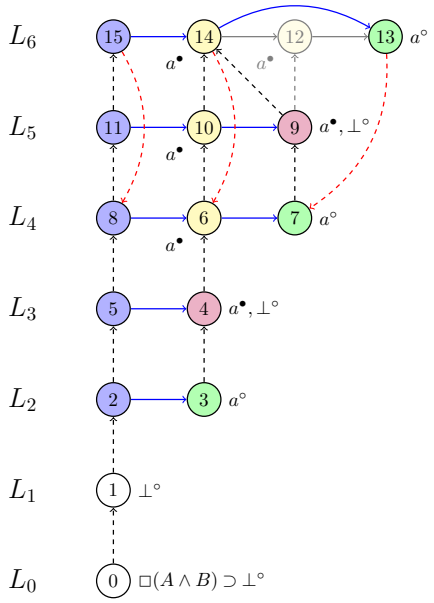


Fig. 9. Formula $\Box(A \wedge B) \supset \bot$ with $A = \Box a \supset \bot$ and $B = (a \supset \bot) \supset \bot$. $\Gamma^\bullet = \{\Box(A \wedge B)^\bullet, A \wedge B^\bullet, A^\bullet, B^\bullet, \Box a^\circ, a \supset \bot^\circ\}$ is in all nodes but 0.

9. Sequent Proof Unfolding

Let us now turn to the case when the algorithm terminates in Step 2. Then all sequents in $\boldsymbol{\mathfrak{G}}_i$ are axiomatic, and we want to construct a proof of $\Longrightarrow r{:}F$ in labIS4$_\leq$. For this we are going to simulate the steps of the algorithm by applying the inference rules of labIS4$'_\leq$, starting with rules Rrf and $\leq$rf to obtain the sequent $r{\leq}r, rRr \Longrightarrow r{:}F$ that is the input of the algorithm. On the one hand, this unfolding seems easy because our search algorithm was designed as organized proof search in labIS4$'_\leq$ and most steps can indeed be executed by applying the rules of labIS4$'_\leq$. However, the difficulties come from the fact that sequents produced by labIS4$'_\leq$ do not have non-singleton clusters. We call such sequents *proper*.

Definition 9.1 (Vertical sequent)**.** A layered sequent $\mathfrak{G}$ is ***vertical*** iff for all labels u, u', v, and v' in $\mathfrak{G}$, (a) if $u{\leq_\mathfrak{G}}v$, and $u{\leq_\mathfrak{G}}v'$, and $vR^{\leftrightarrow}_\mathfrak{G}v'$, then $v = v'$, and (b) if $u{\leq_\mathfrak{G}}v$, and $u'{\leq_\mathfrak{G}}v$, and $uR^{\leftrightarrow}_\mathfrak{G}u'$, then $u = u'$, i.e., for each label there is at most one future per layer and at most one past per layer.

Definition 9.2 (Proper layer/sequent)**.** A layer L in a tree-clustered sequent $\mathfrak{G}$ is called ***proper*** iff all clusters $C \subseteq L$ are singletons. A tree-clustered sequent $\mathfrak{G}$ is ***proper*** iff it is tree-layered and vertical and all its layers are proper.

The basic idea of constructing our proof is to mimic the algorithm with a derivation that works "layer by layer." The key observation is that whenever we create a cluster in the algorithm, we can *unfold* it, i.e., repeat this cluster arbitrarily often in an actual proof with only proper sequents. And this property is preserved when we lift the cluster to the next layer.

Definition 9.3 (Unfolding)**.** For $n \in \mathbb{N}$, a proper sequent $\hat{\mathfrak{G}}$ is called an ***n-unfolding*** of a stable sequent $\mathfrak{G}$ iff there is a binary ***n-unfolding relation*** $\mathsf{U} \subseteq \ell(\mathfrak{G}) \times \ell(\hat{\mathfrak{G}})$ that has the following properties, where we assume $x\mathsf{U}\hat{x}$ and $y\mathsf{U}\hat{y}$ and write L_u to denote the layer of u:

(U1) *formula invariance*: $x \sim \hat{x}$;
(U2) *R-invariance*: if x and y are not in the same cluster in $\mathfrak{G}$, then $xR_\mathfrak{G}y$ iff $\hat{x}R_{\hat{\mathfrak{G}}}\hat{y}$;
(U3) *layer invariance*: $L_x \leq_\mathfrak{G} L_y$ iff $L_{\hat{x}} \leq_{\hat{\mathfrak{G}}} L_{\hat{y}}$;
(U4) *"no-past" invariance*: x has no past in $\mathfrak{G}$ iff $\hat{x}$ has no past in $\hat{\mathfrak{G}}$;
(U5) *singleton-cluster invariance*: if $C = \{z\}$ is a singleton cluster in $\mathfrak{G}$, then there is a unique $\hat{z} \in \ell(\hat{\mathfrak{G}})$ with $z\mathsf{U}\hat{z}$;
(U6) *non-singleton-cluster unfolding*: if C is a cluster in $\mathfrak{G}$ with $|C| = k \geq 2$, then for all $i = 1..k$ and $j = 1..n$ there are $z_i \in \ell(\mathfrak{G})$ and $\hat{z}_{i,j} \in \ell(\hat{\mathfrak{G}})$ such that we have $z_i\mathsf{U}\hat{z}_{i,j}$ and $C = \{z_1, \ldots, z_k\}$ and additionally $\hat{z}_{1,1}R\cdots R\hat{z}_{k,1}R\hat{z}_{1,2}R\cdots R\hat{z}_{k,2}R\cdots R\hat{z}_{1,n}R\cdots R\hat{z}_{k,n}$ in $\hat{\mathfrak{G}}$;
(U7) *injectivity*: if $\hat{x} = \hat{y}$, then $x = y$.

Note that every $(n+1)$-unfolding U is also an n-unfolding and that any proper stable sequent $\mathfrak{G}$ is an n-unfolding of itself for any n, via the diagonal relation $\{(x, x) \mid x \in \ell(\mathfrak{G})\}$.

To ensure that unfolding only operates with proper sequents, we will now introduce labIS4$^\star_\leq$, a proof system whose rules

correspond to applications of several $\mathsf{labIS4}'_{\leq}$ steps, thus forcing the rules of $\mathsf{labIS4}'_{\leq}$ to be applied in a certain way.

Let $\mathfrak{G}$ be a sequent $\mathcal{R}, \Gamma \Longrightarrow \Delta$ with $x{:}F$ occurring in Δ. Then we write $(\mathcal{R}, \Gamma \Longrightarrow \Delta)\Uparrow^{x:F}$ for $\mathfrak{G} + \mathfrak{G}\uparrow^{x:F}$. This allows us to define the following inference rules:

$$\supset^{\circ}_{\star}\frac{(\mathcal{R}, \Gamma \Longrightarrow \Delta, x{:}A \supset B)\Uparrow^{x:A\supset B}}{\mathcal{R}, \Gamma \Longrightarrow \Delta, x{:}A \supset B} \qquad \Box^{\circ}_{\star}\frac{(\mathcal{R}, \Gamma \Longrightarrow \Delta, x{:}\Box A)\Uparrow^{x:\Box A}}{\mathcal{R}, \Gamma \Longrightarrow \Delta, x{:}\Box A}$$

For a label x in a sequent $\mathfrak{G}$, we define $\overleftarrow{x}$ to be the set of labels x' with $x' R_{\mathfrak{G}} x$. For a set X of labels, we write XRy for the set $\{xRy \mid x \in X\}$ of relational atoms. With this notation, we can define the rule:

$$\Diamond^{\bullet}_{\star}\frac{\mathcal{R}, \overleftarrow{x} R y, yRy, y{\leq}y, \Gamma, x{:}\Diamond A, y{:}A \Longrightarrow \Delta}{\mathcal{R}, \Gamma, x{:}\Diamond A \Longrightarrow \Delta}\ y \text{ fresh}$$

Now define ***system*** $\mathsf{labIS4}^{\star}_{\leq}$ to be

- inference rules id, $\bot^{\bullet}$ and $\Diamond^{\circ}$ as in Fig. 2;
- rule $\Box^{\bullet}_{\star}$ obtained by setting $x = y$ in $\Box^{\bullet}$ from Fig. 2;
- rules $\wedge^{\bullet}_{\star}$, $\wedge^{\circ}_{\star}$, $\vee^{\bullet}_{\star}$, $\vee^{\circ}_{\star}$, and $\supset^{\bullet}_{\star}$ that are variants of $\wedge^{\bullet}$, $\wedge^{\circ}$, $\vee^{\bullet}$, $\vee^{\circ}$, and $\supset^{\bullet}$ from Fig. 2 respectively, where the principal formula is not deleted in the premises, the premises are closed under monotonicity, and, additionally, $x = y$ for $\supset^{\bullet}_{\star}$;
- rules $4^{\bullet}$ and 4° from Fig. 3;
- rules $\supset^{\circ}_{\star}$, $\Box^{\circ}_{\star}$, and $\Diamond^{\bullet}_{\star}$ as shown above.

Note that there are no structural rules in $\mathsf{labIS4}^{\star}_{\leq}$.

Proposition 9.4. *Every rule* r *in* $\mathsf{labIS4}^{\star}_{\leq}$ *is derivable in* $\mathsf{labIS4}'_{\leq}$*, and if* $\mathfrak{G}$ *is the conclusion of an instance of* r*, then each premise has the form* $\mathfrak{G} + \mathfrak{H}$ *for some* $\mathfrak{H}$*.*

When writing a premise $\mathfrak{G}+\mathfrak{H}$, we assume that $\mathfrak{G}$ and $\mathfrak{H}$ are disjoint, i.e., all labelled formulas and relational atoms of $\mathfrak{H}$ are absent from the conclusion.

Definition 9.5 (Touch)**.** Let r be an instance of a rule in $\mathsf{labIS4}^{\star}_{\leq}$ with a sequent $\mathfrak{G}$ in its conclusion. We say that r ***touches*** a label $x \in \ell(\mathfrak{G})$ iff for some premise $\mathfrak{G} + \mathfrak{H}$ of r, either we have $u{\leq_{\mathfrak{H}}}x$ or $uR_{\mathfrak{H}}x$ for some label u or we have $\mathfrak{H}$,$x{:}B^{\bullet}$ or $\mathfrak{H}$,$x{:}B^{\circ}$ for some formula B. We say that r ***touches a set*** X ***of labels*** iff r touches some $x \in X$.

Definition 9.6 (Restricted sequent/forbidden label)**.** A sequent $\mathfrak{G}$ is called ***restricted*** iff it is equipped with a set $f(\mathfrak{G}) \subseteq \ell(\mathfrak{G})$ of ***forbidden*** labels satisfying two conditions:

(a) if $y \in f(\mathfrak{G})$ and $xR_{\mathfrak{G}}y$, then $x \in f(\mathfrak{G})$, i.e., all parents of a forbidden label are forbidden;

(b) if $L_x <_{\mathfrak{G}} L_y$ for some $y \in \ell(\mathfrak{G})$, then $x \in f(\mathfrak{G})$, i.e., all labels from the inner layers are forbidden.

All labels in $\ell(\mathfrak{G})$ that are not forbidden are ***allowed***.

In the following, we assume that all sequents are restricted, in particular, all sequents occurring in derivations.

Definition 9.7 (Tidy rule/derivation)**.** Let $\mathfrak{G}$ be the conclusion of an instance of a rule r in $\mathsf{labIS4}^{\star}_{\leq}$. We say that r is ***tidy*** iff (i) r does not touch any forbidden label in $\mathfrak{G}$, and (ii) the forbidden labels for each premise $\mathfrak{K}$ of r are

$$f(\mathfrak{K}) = \begin{cases} \ell(\mathfrak{G}) & \text{if } \mathsf{r} \in \{\supset^{\circ}_{\star}, \Box^{\circ}_{\star}, \Diamond^{\bullet}_{\star}\}, \\ f(\mathfrak{G}) & \text{otherwise.} \end{cases} \tag{4}$$

A derivation $\mathcal{D}$ is ***tidy*** iff it consists of tidy rule instances.

In other words, rules in tidy derivations do not touch forbidden labels and either keep all labels and forbidden labels unchanged or introduce new allowed labels turning all old labels into forbidden. This is sufficient, in particular, to ensure that all sequents in a tidy derivation are proper.

Proposition 9.8. *Let* $\mathcal{D}$ *be a tidy derivation. If the conclusion of* $\mathcal{D}$ *is proper, then every sequent in* $\mathcal{D}$ *is also proper.*

Definition 9.9 (Unfoldable)**.** A derivation $\mathcal{D}_n$ is called an n-***unfolding of a stable set*** $\mathfrak{S}$ iff each premise of $\mathcal{D}_n$ is an n-unfolding of some sequent from $\mathfrak{S}$. For a formula F, a stable set $\mathfrak{S}$ is F***-unfoldable*** iff for every $n \in \mathbb{N}$ there is a tidy derivation $\mathcal{D}_n$ of $\mathfrak{G}_0(F)$ (as defined in Step 0 of the algorithm in Fig. 7) such that $\mathcal{D}_n$ is an n-unfolding of $\mathfrak{S}$.

Clearly, $\mathfrak{S}'_0 = \{\mathfrak{G}_0(F)\}$ is F-unfoldable. It now has to be shown that all operations performed in the algorithm, in particular, the rewrite relations $\rightsquigarrow_{\mathsf{s}}$, $\rightsquigarrow_{\Diamond}$, and $\rightsquigarrow_{\circlearrowleft}$, preserve this propery. For $\rightsquigarrow_{\mathsf{s}}$, this is straightforward, as we simply apply the inference rules of $\mathsf{labIS4}^{\star}_{\leq}$ to match the semi-saturation steps. For $\rightsquigarrow_{\Diamond}$ and $\rightsquigarrow_{\circlearrowleft}$, we need to unfold the newly created clusters. The basic idea is to simply repeat all proof steps that lead to the first occurrence of the cluster n-times.

Lemma 9.10 (Unfolding Lemma)**.** *All sets* $\mathfrak{S}_i$ *of sequents generated by the algorithm from Fig. 7 are* F*-unfoldable.*

Hence, when we terminate in Step 2, we can construct a 1-unfolding of $\mathfrak{S}_i$, which constitutes a proof in $\mathsf{labIS4}^{\star}_{\leq}$ because an unfolding of an axiomatic sequent is always axiomatic.

Theorem 9.11. *If the algorithm shown in Fig. 7 terminates in Step 2, then the formula* F *is a theorem of* IS4*.*

10. Termination

We have already established in Sect. 7 that every step in our algorithm (shown in Fig. 7) terminates. It remains to show that we cannot run through the main loop forever, i.e., sequence $\mathfrak{S}_0, \mathfrak{S}_1, \ldots, \mathfrak{S}_i, \ldots$ eventually terminates either in Step 2 (we find a proof) or in Step 4 (we find a countermodel). The basic idea is to restrict the size of a layer in the sequents of $\mathfrak{S}_i$. Then the number of distinct possible layers is finite, hence, we will eventually find a simulation (Defs. 7.11, 7.13).

For the remainder of this section, we assume that a formula F in Step 0 has n subformula occurrences.

We define the ***size*** $|x|$ of a label x in a sequent $\mathfrak{G}$ to be the number of distinct formula occurrences $x{:}A^{\bullet}$ or $x{:}A^{\circ}$ in $\mathfrak{G}$.

Lemma 10.1. *The size of a label occurring in a sequent of some* $\mathfrak{S}_i$ *is at most* n*. And there are* 2^n *many equivalence classes of labels with respect to* $\sim$*.*

The ***size*** $|C|$ of a cluster C is the number of labels in C.

Lemma 10.2. *The size of a cluster occurring in a sequent of some* $\mathfrak{S}_i$ *is at most* 2^n*. And there are* $2^{2^n}-1$ *many equivalence classes of (non-empty) clusters with respect to* $\sim$*.*

The algorithm visits only stable sequents. A layer L in such a sequent is a tree of clusters. Let M be a branch in L, i.e., a sequence $C_1, C_2, \ldots, C_l$ of clusters from the root to a leaf. The ***length*** $|M|$ of M is the sum $|C_1| + |C_2| + \cdots + |C_l|$.

Lemma 10.3. *The length of a branch in a layer in a sequent in a set* $\mathfrak{S}_i$ *is bounded, and the bound is determined by* F*.*

Theorem 10.4 (Termination)**.** *The proof search algorithm given in Fig. 7 is terminating.*

11. Conclusion

In this paper we have solved a problem open for almost 30 years. Our solution has two key ingredients.

First, the use of the fully labelled system with relational atoms for both binary relations enabled us to give a proof system that has only invertible rules and also gives a closer correspondence between sequents and models.

Second, although the identification of labels during during proof search to realize loops is *a priori* unsound, however, under the right circumstances, we can preserve soundness if we organize the proof search in a certain systematic way.

We conjecture that the same method can also be applied to IK4, which is IS4 without the t-axiom and which is the other logic in the intuitionistic version of the S5-cube for which decidability is an open problem. In fact, the overall argument is the same, but in many definitions and proof arguments, there would be subtle differences due to the absence of reflexivity. For this reason a full treatment of IK4 would go beyond the scope of this paper.

The implementation of the algorithm is object of current work. In a future work we would like to investigate the complexity of provability in IS4.

Acknowledgements. Roman Kuznets is supported by the Austrian Science Fund (FWF) project ByzDEL (P 33600). Marianna Girlando is supported by the Horizon 2021 programme, under the Marie Skłodowska-Curie grant CYDER (101064105).

References

[1] N. Alechina and D. Shkatov. A general method for proving decidability of intuitionistic modal logics. *Journal of Applied Logic*, 4(3):219–230, 2006. doi:10.1016/j.jal.2005.06.007.

[2] P. Balbiani, M. Diéguez, and D. Fernández-Duque. Some constructive variants of S4 with the finite model property. In *LICS 2021*. IEEE, 2021. doi:10.1109/LICS52264.2021.9470643.

[3] K. Brünnler. Deep sequent systems for modal logic. *Archive for Mathematical Logic*, 48(6):551–577, 2009. doi:10.1007/s00153-009-0137-3.

[4] K. Chaudhuri, S. Marin, and L. Straßburger. Modular focused proof systems for intuitionistic modal logics. In *FSCD 2016*. LZI, 2016. doi:10.4230/LIPIcs.FSCD.2016.16.

[5] R. Dyckhoff. Intuitionistic decision procedures since Gentzen. In *Advances in Proof Theory*, pages 245–267. Springer, 2016. doi:10.1007/978-3-319-29198-7_6.

[6] R. Dyckhoff and S. Negri. Proof analysis in intermediate logics. *Archive for Mathematical Logic*, 51(1–2):71–92, 2012. doi:10.1007/s00153-011-0254-7.

[7] W. B. Ewald. Intuitionistic tense and modal logic. *Journal of Symbolic Logic*, 51(1):166–179, 1986. doi:10.2307/2273953.

[8] G. Fischer Servi. Axiomatizations for some intuitionistic modal logics. *Rendiconti del Seminario Matematico - PoliTO*, 42(3):179–194, 1984.

[9] D. Gabelaia, A. Kurucz, F. Wolter, and M. Zakharyaschev. Products of 'transitive' modal logics. *Journal of Symbolic Logic*, 70(3):993–1021, 2005. doi:10.2178/jsl/1122038925.

[10] D. Garg, V. Genovese, and S. Negri. Countermodels from sequent calculi in multi-modal logics. In *LICS 2012*, pages 315–324. IEEE, 2012. doi:10.1109/LICS.2012.42.

[11] G. Gentzen. Untersuchungen über das logische Schließen. I. *Mathematische Zeitschrift*, 39:176–210, 1935. doi:10.1007/BF01201353.

[12] Y. Hasimoto. Finite model property for some intuitionistic modal logics. *Bulletin of the Section of Logic*, 30(2):87–97, 2001.

[13] S. Heilala and B. Pientka. Bidirectional decision procedures for the intuitionistic propositional modal logic IS4. In *CADE 2007*, pages 116–131. Springer, 2007. doi:10.1007/978-3-540-73595-3_9.

[14] O. Ketonen. Untersuchungen zum Prädikatenkalkül. *Annales Academiae Scientiarum Fennicae, Series A, I*, 23, 1944.

[15] S. C. Kleene. *Introduction to Metamathematics*. North-Holland, 1952.

[16] R. Kuznets and L. Straßburger. Maehara-style modal nested calculi. *Archive for Mathematical Logic*, 58(3–4):359–385, 2019. doi:10.1007/s00153-018-0636-1.

[17] R. E. Ladner. The computational complexity of provability in systems of modal propositional logic. *SIAM Journal on Computing*, 6(3):467–480, 1977. doi:10.1137/0206033.

[18] F. Lamarche. On the algebra of structural contexts. Accepted at *Mathematical Structures in Computer Science*, 2001. URL: https://hal.inria.fr/inria-00099461.

[19] P. Maffezioli, A. Naibo, and S. Negri. The Church–Fitch knowability paradox in the light of structural proof theory. *Synthese*, 190(14):2677–2716, 2013. doi:10.1007/s11229-012-0061-7.

[20] S. Marin, M. Morales, and L. Straßburger. A fully labelled proof system for intuitionistic modal logics. *Journal of Logic and Computation*, 31(3):998–1022, 2021. doi:10.1093/logcom/exab020.

[21] G. E. Minc. On some calculi of modal logic. *Proceedings of the Steklov Institute of Mathematics*, 98(1968):97–124, 1971.

[22] S. Negri. Proof analysis in modal logic. *Journal of Philosophical Logic*, 34(5–6):507–544, 2005. doi:10.1007/s10992-005-2267-3.

[23] K. Ono. Logische Untersuchungen über die Grundlagen der Mathematik. *Journal of the Faculty of Science, Imperial University of Tokyo, section I*, III(7):329–389, 1938.

[24] G. Plotkin and C. Stirling. A framework for intuitionistic modal logics. In *TARK 1986*, pages 399–406, 1986. URL: http://www.tark.org/proceedings/tark_mar19_86/p399-plotkin.pdf.

[25] M. Sato. A study of Kripke-type models for some modal logics by Gentzen's sequential method. *Publications of the RIMS*, 13(2):381–468, 1977. doi:10.2977/PRIMS/1195189814.

[26] A. K. Simpson. *The Proof Theory and Semantics of Intuitionistic Modal Logic*. PhD thesis, University of Edinburgh, 1994. URL: http://hdl.handle.net/1842/407.

[27] L. Straßburger. Cut elimination in nested sequents for intuitionistic modal logics. In *FoSSaCS 2013*, pages 209–224. Springer, 2013. doi:10.1007/978-3-642-37075-5_14.

[28] A. S. Troelstra and H. Schwichtenberg. *Basic Proof Theory*. Cambridge University Press, 2000. doi:10.1017/CBO9781139168717.

[29] F. Wolter and M. Zakharyaschev. Intuitionistic modal logics as fragments of classical bimodal logics. In *Logic at Work*, pages 168–186. Physica, 1999.

APPENDIX

A. Examples

This section contains some examples illustrating how the search algorithm (Fig. 7) works. Refer to Example 5.2 for an explanation of the diagrammatic notation.

For reasons of space, we represent only one sequent of those generated by the algorithm, that is, either an axiomatic sequent or a sequent from which a countermodel for the formula at the root can be generated. Recall that even if an axiomatic sequent is found the algorithm might continue running: for it to stop (at step 2), all sequents in the set need to be axiomatic. Conversely, the algorithm stops (at step 4) if it finds a sequent containing enough information to build a countermodel.

Example A.1 (Valid formula)**.** Let us consider formula $\Box(\Diamond A \wedge \Diamond b) \supset \bot$, with $A = (c \supset \Diamond b) \supset \bot$, which is valid in IS4. Figure 10 represents one sequent generated by the algorithm. Let $\Gamma^\bullet$ be the set $\{\Box(\Diamond A \wedge \Diamond b)^\bullet, \Diamond A \wedge \Diamond b^\bullet, \Diamond A^\bullet, \Diamond b^\bullet\}$. To each label in the figure (except 1) we associate $\Gamma^\bullet$, plus the formulas explicitly displayed next to the node in the figure. The following $\leq$-relations are not displayed but are present in the sequent: $3{\leq}10$, $4{\leq}11$, $6{\leq}12$, $6{\leq}15$, $7{\leq}13$ and $7{\leq}16$.

At layer L_1 the search on both R-branches stops in virtue of option 1 of Def. 7.5 (saturation): there, 5 is replaced with 3, and 8 with 6. This generates two non-singleton clusters: $C_1 = \{3, 4\}$ and $C_2 = \{6, 7\}$. Then, since all labels in L_1 are almost happy, lifting saturation is applied: the rewrite "lifts" one copy of the layer for each $\supset^\circ$-formula present in the sequent. We only represent one of such layers, L_2, generated from formula $7{:}c \supset \Diamond b^\circ$. Since 7 belongs to the cluster C_2, the cluster is duplicated in L_2, following Defs. 7.8–7.9. Then, after a semi-saturation step, we have that both $15{:}b^\bullet$ and $15{:}b^\circ$. Thus we stop the search along this branch. If all the sequents manipulated by the algorithm eventually result in axiomatic sequents, then the algorithm stops and, according to Theorem 9.11, the formula at the root is a theorem of IS4. Indeed, its derivation in $\mathsf{labIS4}_{\leq}$ can be found in Fig. 4 (Left).

Example A.2 (Non-valid formula, R-triangle loop)**.** We now consider formula 3 from Sect. 3, that is, $\Box(A \wedge B) \supset \bot$ with $A = \Box a \supset \bot$ and $B = (a \supset \bot) \supset \bot$. On the left of Fig. 11 we represent one sequent generated by the algorithm. Let $\Gamma^\bullet = \{\Box(A \wedge B)^\bullet, A \wedge B^\bullet, A^\bullet, B^\bullet, \Box a^\circ, a \supset \bot^\circ\}$. To each label in the figure (except 0), we associate $\Gamma^\bullet$, plus the formulas explicitly displayed next to the node. To be precise, the figure displays only one $\leq$-branch of the sequent: formulas $\Box a^\circ$ and $a \supset \bot^\circ$ are in all the labels, and every time they are unhappy they are also lifted, generating a new layer (which is not displayed).

At layer L_6 an unhappy R-triangle loop is detected: employing the terminology from Def. 7.16, we take $C_1 = \{3\}$, $p_1 = 3$, $C_2 = \{13\}$, $s = 14$, $C_r = \{s\}$ and $t = 12$. On the right of Fig. 11 is represented the sequent resulting from the loop saturation, where 12 is replaced by 14. Observe that the cluster $\{14\}$ remains unchanged. After the loop saturation,

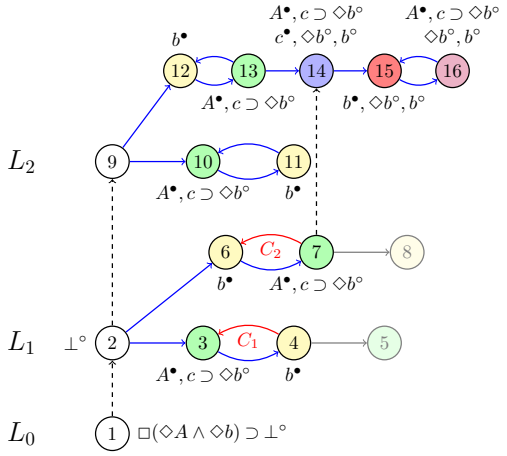


Fig. 10. Formula $\Box(\Diamond A \wedge \Diamond b) \supset \bot$, with $A = (c \supset \Diamond b) \supset \bot$.

proof search along the $\leq$-branch depicted stops: this is because layer $L_{6'}$ can be simulated by layer L_4. By adding relational atoms $15{\leq}8$, $14{\leq}6$ and $13{\leq}7$ to the sequent (Theorem 8.1), which are the dashed arrows pointing downwards in the figure, we obtain (a part of) the countermodel for our formula. To complete the countermodel, we need to take into account the layers generated from unhappy $\Box^\circ$ and $\supset^\circ$ formulas present in the sequent. Specifically, $8{:}a \supset \bot^\circ$ and $8{:}a \supset \bot^\circ$ are unhappy. These formulas give rise to more $\leq$-branches in the sequent (but all branches are finite, and there is only a finite number of them).

Example A.3 (Non-valid formula, U-triangle loop)**.** Let us consider formula $\Box(\Diamond D \wedge \Diamond c) \supset \bot$, with $D = (a \supset b) \supset \bot$. Figure 12 illustrates one sequent generated by the search algorithm. Let $\Gamma^\bullet = \{\Box(\Diamond D \wedge \Diamond c)^\bullet, \Diamond D \wedge \Diamond c^\bullet, \Diamond D^\bullet, \Diamond c^\bullet\}$. To each label we associate the following sets, color-coded in the figure:

1	$\Gamma^\bullet \cup \{\Box(\Diamond D \wedge \Diamond c) \supset \bot^\circ\}$
2	$\Gamma^\bullet \cup \{\bot^\circ\}$
$10 \sim 19 \sim 29$	$\Gamma^\bullet$
$3 \sim 8 \sim 6 \sim$ $18 \sim 15 \sim 12 \sim$ $28 \sim 23 \sim 21$	$\Gamma^\bullet \cup \{D^\bullet, a \supset b^\circ\}$
$4 \sim 9 \sim 7 \sim$ $17 \sim 14 \sim 13 \sim$ $27 \sim 24 \sim 22 \sim$	$\Gamma^\bullet \cup \{c^\bullet\}$
$16 \sim 26 \sim 25$	$\Gamma^\bullet \cup \{D^\bullet, a^\bullet, a \supset b^\circ\}$
$5 \sim 11 \sim 20$	$\Gamma^\bullet \cup \{D^\bullet, a^\bullet, b^\circ, a \supset b^\circ\}$

As in the previous example, only one $\leq$-branch is represented. Moreover, for reasons of space, we have not pictured a second R-branch originating from 2 by saturation, to make formula $2{:}\Diamond c^\bullet$ happy. The labels originated by lifting this branch are present in all layers.

As in Example A.1, in layer L_1, after saturation, a non-singleton loop is created, $\{3, 4\}$. Then, layer L_2 is generated by lifting saturation, which duplicates the cluster. The process is repeated to generate L_3. At L_3, an unhappy U-triangle

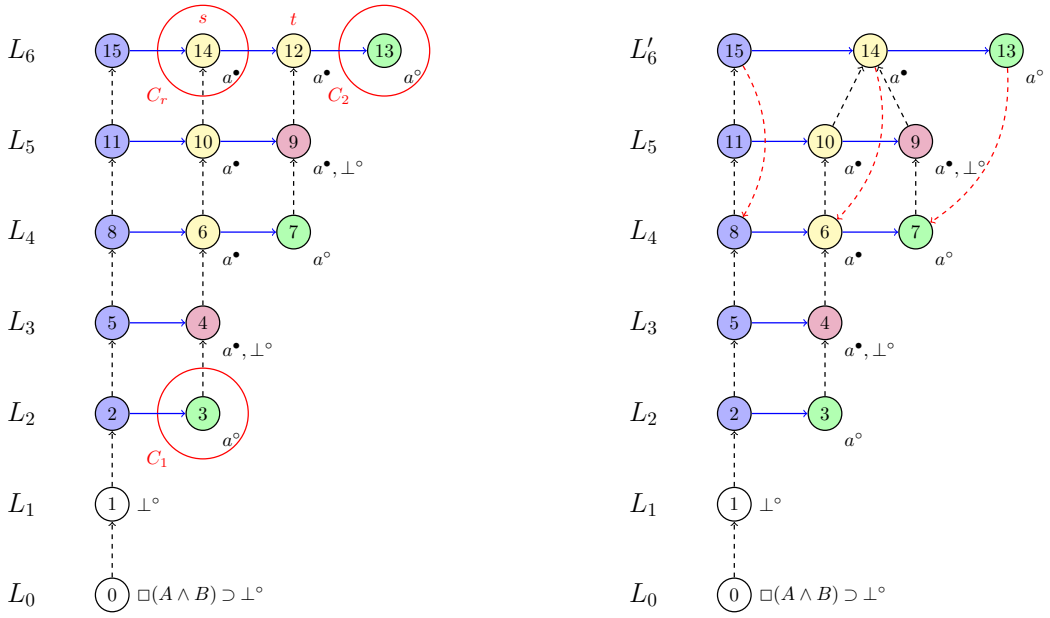


Fig. 11. Formula $\Box(A \wedge B) \supset \bot$ with $A = \Box a \supset \bot$ and $B = (a \supset \bot) \supset \bot$. A loop saturation step is applied to the leftmost sequent, and the result of the saturation is the rightmost sequent.

loop is found. Using the terminology from Def. 7.17, take $C_1 = \{3, 4\}$, $C_2 = \{12, 13\}$, $s = 18$ and $t = 15$. Thus, we substitute 15 with 18. Moreover, a second unhappy U-triangle loop is found, by taking $s' = 17$ and $t' = 14$ (refer to the topmost sequent of Fig. 12). Layer $L_{3'}$ is the result of the loop saturation: it contains a cluster $\{18, 17, 16\}$. The algorithm produces a sequent consisting of layers L_0 - L_2, $L_{3'}$, and continues (middle sequent in Fig. 12).

Then, after a lifting saturation, layer L_4 is generated. Here, three unhappy U-triangle loops are present: take $C_1 = \{3, 4\}$, $C_2 = \{21, 22\}$, and then $s = 28$ and $t = 23$, $s' = 27$ and $t' = 24$ and $s'' = 26$ and $t'' = 25$. After the loop saturation, we obtain layer $L_{4'}$, which can be simulated by layer $L_{3'}$ (lowermost sequent in Fig. 12). By adding relational atoms $29 \leq 19$, $28 \leq 18$, $27 \leq 17$, $26 \leq 16$, $20 \leq 11$, $21 \leq 12$ and $22 \leq 13$ we have a (partial) countermodel for the formula at the root. To complete the countermodel, we need to include the layers generated by lifting the layers of the sequent in correspondence to each remaining unhappy formula $(a \supset b)^\circ$.

B. Proofs from Section 5

Theorem 5.8 (Completeness)**.** *For a happy sequent $\mathfrak{G}$, its model $\mathcal{M}_\mathfrak{G} = \langle \ell(\mathfrak{G}), \leq_\mathfrak{G}, R_\mathfrak{G}, V \rangle$ is a transitive and reflexive birelational model with the following two properties:*

- *if $\mathfrak{G}$,x:$A^\bullet$, then $\mathcal{M}_\mathfrak{G}, x \Vdash A$;*
- *if $\mathfrak{G}$,x:A°, then $\mathcal{M}_\mathfrak{G}, x \nVdash A$.*

Proof. The worlds W of $\mathcal{M}_\mathfrak{G}$ are the labels of the sequent. Conditions F_1 and F_2, the transitivity and reflexivity of $\leq_\mathfrak{G}$ and $R_\mathfrak{G}$, and the monotonicity of V all follow by construction due to structural saturation. Thus, $\mathcal{M}_\mathfrak{G}$ is a transitive and reflexive birelational model. It only remains to show the two properties about forcing, that we prove by mutual induction on the size of A, proceeding by case analysis on the main connective of A:

- $\mathfrak{G}$,x:$\bot^\bullet$: it is not possible for a happy sequent.
- $\mathfrak{G}$,x:$\bot^\circ$: we have $\mathcal{M}_\mathfrak{G}, x \nVdash \bot$ by definition.
- $\mathfrak{G}$,x:$a^\bullet$: by Definition 5.7, $\mathcal{M}_\mathfrak{G}, x \Vdash a$.
- $\mathfrak{G}$,x:a°: it is not the case that $\mathfrak{G}$,x:$a^\bullet$ by happiness of x, hence, $\mathcal{M}_\mathfrak{G}, x \nVdash a$ by Definition 5.7.
- $\mathfrak{G}$,x:$B \wedge C^\bullet$: by happiness of x, both $\mathfrak{G}$,x:$B^\bullet$ and $\mathfrak{G}$,x:$C^\bullet$. Then $\mathcal{M}_\mathfrak{G}, x \Vdash B$ and $\mathcal{M}_\mathfrak{G}, x \Vdash C$ by IH. Therefore, $\mathcal{M}_\mathfrak{G}, x \Vdash B \wedge C$.
- $\mathfrak{G}$,x:$B \wedge C^\circ$: by happiness of x, either $\mathfrak{G}$,x:B° or $\mathfrak{G}$,x:C°. Then either $\mathcal{M}_\mathfrak{G}, x \nVdash B$ or $\mathcal{M}_\mathfrak{G}, x \nVdash C$ by IH. Therefore, $\mathcal{M}_\mathfrak{G}, x \nVdash B \wedge C$.
- Cases for $\mathfrak{G}$,x:$B \vee C^\bullet$ and $\mathfrak{G}$,x:$B \vee C^\circ$ are analogous.
- $\mathfrak{G}$,x:$B \supset C^\bullet$: consider any y with $x \leq_\mathfrak{G} y$. By (mon$^\bullet$)-structural saturation, $\mathfrak{G}$,y:$B \supset C^\bullet$. By happiness of y, either $\mathfrak{G}$,y:B° or $\mathfrak{G}$,y:$C^\bullet$. By IH, either $\mathcal{M}_\mathfrak{G}, y \nVdash B$ or $\mathcal{M}_\mathfrak{G}, y \Vdash C$. Thus, $\mathcal{M}_\mathfrak{G}, y \Vdash B$ implies $\mathcal{M}_\mathfrak{G}, y \Vdash C$ for all y with $x \leq_\mathfrak{G} y$. Therefore, $\mathcal{M}_\mathfrak{G}, x \Vdash B \supset C$.
- $\mathfrak{G}$,x:$B \supset C^\circ$: by happiness of x, there is a world y such that $x \leq_\mathfrak{G} y$, $\mathfrak{G}$,y:$B^\bullet$, and $\mathfrak{G}$,y:C°. By IH, $\mathcal{M}_\mathfrak{G}, y \Vdash B$ and $\mathcal{M}_\mathfrak{G}, y \nVdash C$. Therefore, $\mathcal{M}_\mathfrak{G}, x \nVdash B \supset C$.
- $\mathfrak{G}$,x:$\Diamond B^\bullet$: by happiness of x, there is a world y such that $x R_\mathfrak{G} y$ and $\mathfrak{G}$,y:$B^\bullet$. By IH, we have $\mathcal{M}_\mathfrak{G}, y \Vdash B$ and, therefore, $\mathcal{M}_\mathfrak{G}, x \Vdash \Diamond B$.
- $\mathfrak{G}$,x:$\Diamond B^\circ$: by happiness of x, we have $\mathfrak{G}$,y:B° for all worlds y such that $x R_\mathfrak{G} y$. Thus, by IH, $\mathcal{M}_\mathfrak{G}, y \nVdash B$ whenever $x R_\mathfrak{G} y$. Therefore, $\mathcal{M}_\mathfrak{G}, x \nVdash \Diamond B$.
- $\mathfrak{G}$,x:$\Box B^\bullet$: consider arbitrary y and z with $x \leq_\mathfrak{G} y$ and $y R_\mathfrak{G} z$. By (mon$^\bullet$)-structural saturation, $\mathfrak{G}$,y:$\Box B^\bullet$. By happiness of y, we have $\mathfrak{G}$,z:$B^\bullet$. Thus, by IH, $\mathcal{M}_\mathfrak{G}, z \Vdash B$ whenever $x \leq_\mathfrak{G} y$ and $y R_\mathfrak{G} z$. Therefore, $\mathcal{M}_\mathfrak{G}, x \Vdash \Box B$.

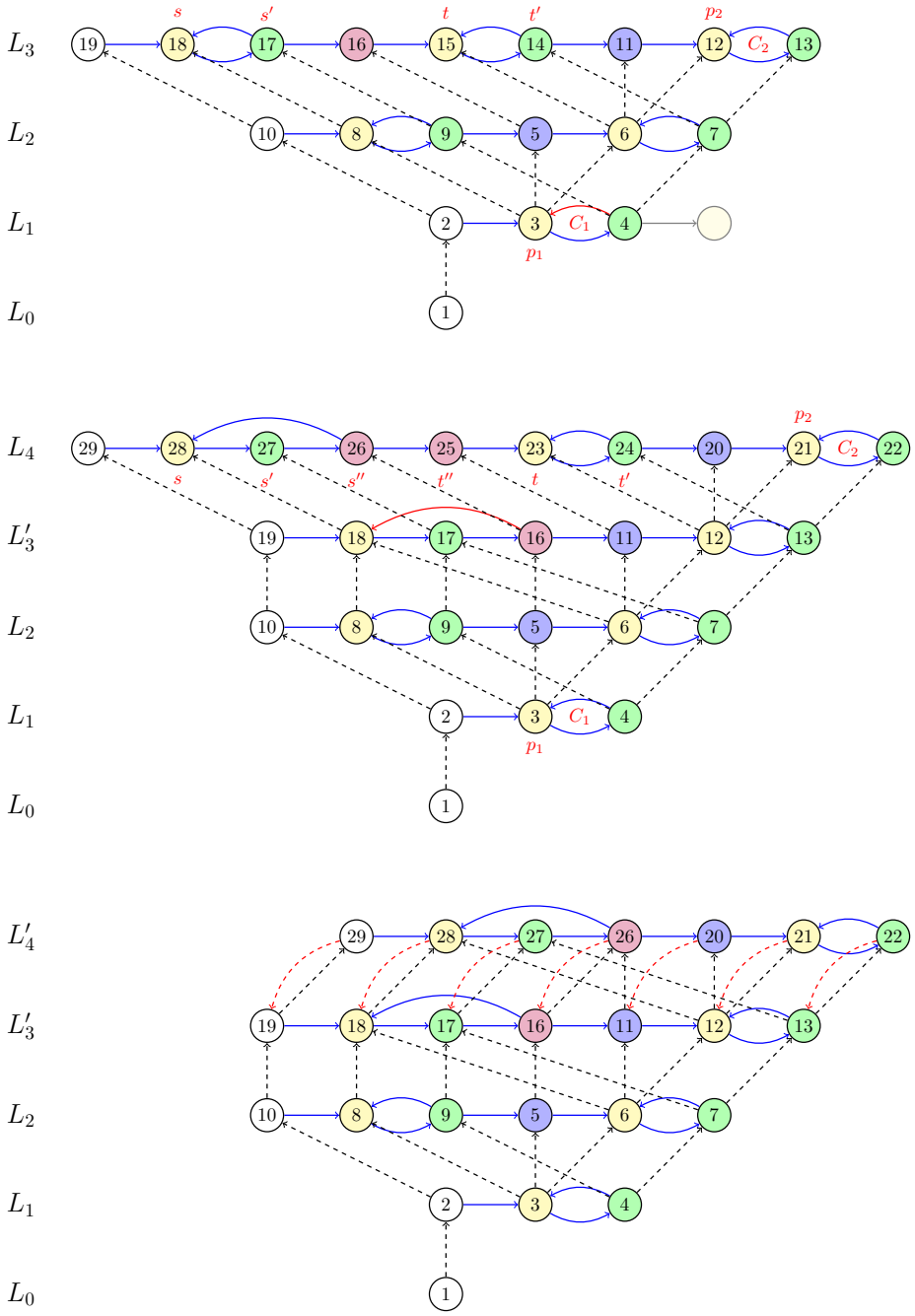


Fig. 12. Formula $\Box(\Diamond D \wedge \Diamond c) \supset \bot$, with $D = (a \supset b) \supset \bot$. A loop saturation step is applied to the topmost sequent, and the result of the saturation is represented in the middle sequent. Then, a loop saturation is also applied to the middle sequent, and the result is the lowermost sequent.

- $\mathfrak{G}$,x:$\Box B^\circ$: by happiness of x, there exist y and z such that $x \leq_{\mathfrak{G}} y$, $y R_{\mathfrak{G}} z$, and $\mathfrak{G}$,z:B°. By IH, $\mathcal{M}_{\mathfrak{G}}, z \not\Vdash B$. Therefore, $\mathcal{M}_{\mathfrak{G}}, x \not\Vdash \Box B$. □

C. Proofs from Section 6

We start with a proposition that is not in the main text but simplifies many arguments.

Proposition A.4. *Structurally saturated sequents $\mathfrak{G}$ are also saturated w.r.t. $R^{\leftrightarrow}_{\mathfrak{G}}$:*

(F) *if $x R^{\leftrightarrow}_{\mathfrak{G}} y$ and $x \leq_{\mathfrak{G}} z$, then there is u such that $y \leq_{\mathfrak{G}} u$ and $z R^{\leftrightarrow}_{\mathfrak{G}} u$.*

Proof. There must be a sequence $x_0, \ldots, x_n$ of labels with $n \geq 0$ such that $x_0 = x$, $x_n = y$, and for each $0 \leq i \leq n-1$, either $x_i R_{\mathfrak{G}} x_{i+1}$ or $x_{i+1} R_{\mathfrak{G}} x_i$. We use induction on n. If $n = 0$, i.e., $x = y$, then $u := z$ suffices. Otherwise, by IH, there is v such that $x_{n-1} \leq_{\mathfrak{G}} v$ and $z R^{\leftrightarrow}_{\mathfrak{G}} v$. If $x_n R_{\mathfrak{G}} x_{n-1}$, there is u by (F_1)-structural saturation such that $x_n \leq_{\mathfrak{G}} u$ and $u R_{\mathfrak{G}} v$. If $x_{n-1} R_{\mathfrak{G}} x_n$, there is u by (F_2)-structural saturation such that

$x_n \leq_{\mathfrak{G}} u$ and $v R_{\mathfrak{G}} u$. Either way, $y \leq_{\mathfrak{G}} u$ and $z R^{\leftrightarrow}_{\mathfrak{G}} u$. □

Proposition 6.3. *For a layered structurally saturated sequent $\mathfrak{G}$, relation $\leq$ is an order relation on its layers.*

Proof. For reflexivity, consider any layer L. For any label $x \in L$, by ($\leq$rf)-structural saturation, $x \leq_{\mathfrak{G}} x$. Hence, $L \leq L$ by Definition 6.2.

For transitivity, let $L_1 \leq L_2$ and $L_2 \leq L_3$ for layers L_1, L_2, and L_3. By Definition 6.2, $x \leq_{\mathfrak{G}} y'$ and $y \leq_{\mathfrak{G}} z$ for some labels $x \in L_1$, $y', y \in L_2$, and $z \in L_3$. Since $y' R^{\leftrightarrow}_{\mathfrak{G}} y$, by (F)-structural saturation (Proposition A.4), there is a label z' such that $y' \leq_{\mathfrak{G}} z'$ and $z' R^{\leftrightarrow}_{\mathfrak{G}} z$. The latter means that $z' \in L_3$. By ($\leq$tr)-structural saturation, $x \leq_{\mathfrak{G}} z'$. Hence, $L_1 \leq L_3$ by Definition 6.2.

For antisymmetry, let $L_1 \leq L_2$ and $L_1 \neq L_2$. By Definition 6.2, there are labels $x \in L_1$ and $x' \in L_2$ such that $x \leq_{\mathfrak{G}} x'$. For arbitrary labels $y \in L_1$ and $y' \in L_2$, we have $x R^{\leftrightarrow}_{\mathfrak{G}} y$ and $x' R^{\leftrightarrow}_{\mathfrak{G}} y'$. Since $L_1 \neq L_2$, we have $x \neq x'$. Hence, $y' \not\leq_{\mathfrak{G}} y$ by 2) of Definition 6.2. Since $y' \not\leq_{\mathfrak{G}} y$ for any $y' \in L_2$ and $y \in L_1$, it is not the case that $L_2 \leq L_1$. □

Proposition 6.7. *For a structurally saturated sequent $\mathfrak{G}$, relation $R_{\mathfrak{G}}$ is an order and $\leq_{\mathfrak{G}}$ is a preorder on its clusters. If $\mathfrak{G}$ is layered, then $\leq_{\mathfrak{G}}$ is also an order.*

Proof. For reflexivity of $R_{\mathfrak{G}}$, consider any cluster C. For any label $x \in C$, by (Rrf)-structural saturation, $x R_{\mathfrak{G}} x$. Hence, $C R_{\mathfrak{G}} C$ by Definition 6.6. For transitivity of $R_{\mathfrak{G}}$, let $C_1 R_{\mathfrak{G}} C_2$ and $C_2 R_{\mathfrak{G}} C_3$ for clusters C_1, C_2, and C_3. By Definition 6.6, $x R_{\mathfrak{G}} y$ and $u R_{\mathfrak{G}} z$ for some labels $x \in C_1$, $y, u \in C_2$, and $z \in C_3$. Since $y R_{\mathfrak{G}} u$, by (Rtr)-structural saturation $x R_{\mathfrak{G}} z$. Thus, $C_1 R_{\mathfrak{G}} C_3$ by Definition 6.6. For antisymmetry of $R_{\mathfrak{G}}$, let $C_1 R_{\mathfrak{G}} C_2$ and $C_2 R_{\mathfrak{G}} C_1$. By Definition 6.6, there are labels $x, y \in C_1$ and $x', y' \in C_2$ such that $x R_{\mathfrak{G}} x'$ and $y' R_{\mathfrak{G}} y$. For arbitrary labels $u \in C_1$ and $v \in C_2$, we have $u R_{\mathfrak{G}} x$, and $x' R_{\mathfrak{G}} v$, hence, by (Rtr)-structural saturation, $u R_{\mathfrak{G}} v$. Similarly, $v R_{\mathfrak{G}} u$ because $v R_{\mathfrak{G}} y'$ and $y R_{\mathfrak{G}} u$. Since both $u R_{\mathfrak{G}} v$ and $v R_{\mathfrak{G}} u$ for all $u \in C_1$ and $v \in C_2$, all labels in these two clusters form one equivalence class w.r.t. $R_{\mathfrak{G}} \cap R^{-1}_{\mathfrak{G}}$, i.e., $C_1 = C_2$.

For reflexivity of $\leq_{\mathfrak{G}}$, consider any cluster C. For every label $y \in C$, by ($\leq$rf)-structural saturation, $y \leq_{\mathfrak{G}} y$. Hence, $C \leq_{\mathfrak{G}} C$ by Definition 6.6. For transitivity of $\leq_{\mathfrak{G}}$, let $C_1 \leq_{\mathfrak{G}} C_2$ and $C_2 \leq_{\mathfrak{G}} C_3$ for clusters C_1, C_2, and C_3. By Definition 6.6, for every $z \in C_3$, there is $y \in C_2$ such that $y \leq_{\mathfrak{G}} z$. In its turn, for this y, there is $x \in C_1$ such that $x \leq_{\mathfrak{G}} y$. By ($\leq$tr)-structural saturation $x \leq_{\mathfrak{G}} z$ for this x. Thus, $C_1 \leq_{\mathfrak{G}} C_3$ by Definition 6.6. Hence, $\leq_{\mathfrak{G}}$ is a preorder.

Assume now additionally that $\mathfrak{G}$ is layered. For antisymmetry of $\leq_{\mathfrak{G}}$, let $C_1 \leq_{\mathfrak{G}} C_2$ and $C_1 \neq C_2$, i.e., $C_1 \cap C_2 = \varnothing$. To show that $C_2 \not\leq_{\mathfrak{G}} C_1$, we consider any label $x \in C_1$ and show that $y \not\leq_{\mathfrak{G}} x$ for all $y \in C_2$. By Definition 6.6, for each $y \in C_2$, there is some label $x' \in C_1$ such that $x' \leq_{\mathfrak{G}} y$. We have $x' R^{\leftrightarrow}_{\mathfrak{G}} x$ because they belong to the same cluster, $y R_{\mathfrak{G}} y$, and hence $y R^{\leftrightarrow}_{\mathfrak{G}} y$, by ($R$rf)-structural saturation, and $x' \neq y$ because they are from disjoint clusters. Hence, $y \not\leq_{\mathfrak{G}} x$ by Definition 6.2. Since y was chosen arbitrarily, it follows that $C_2 \not\leq_{\mathfrak{G}} C_1$ and $\leq_{\mathfrak{G}}$ is an order. □

D. Proofs from Section 7

For simplicity, when $w \leq v$, then we say that w is *a past of* v and that v is *a future of* w.

Lemma 7.2. *All the following statements hold:*

a) If a set $\mathfrak{S}$ is stable and $\mathfrak{S} \rightrightarrows_{\mathsf{s}} \mathfrak{S}'$, then $\mathfrak{S}'$ is stable.
b) On finite sets $\mathfrak{S}$, the rewrite relation $\rightrightarrows_{\mathsf{s}}$ is terminating.
c) A set $\mathfrak{S}$ is semi-saturated iff it is stable and in normal form w.r.t. $\rightrightarrows_{\mathsf{s}}$.

Proof. To prove a) assume that set $\mathfrak{S}$ is finite and all sequents $\mathfrak{G} \in \mathfrak{S}$ are stable, i.e., tree-layered and tree-clustered (in particular, layered and structurally saturated) with happy inner layers. The size of $\mathfrak{S}'$ is larger than that of $\mathfrak{S}$ by at most one sequent, hence, $\mathfrak{S}'$ is also finite. Since all labels in the inner layers of sequents from $\mathfrak{S}$ are happy, labels of all newly added labelled formulas, be it label x itself or labels $R_{\mathfrak{G}}$-accessible from x, must be in a topmost layer of some sequent from $\mathfrak{S}$. Given that this sequent is layered, the labels of all new labelled formulas have no non-reflexive futures, making (mon$^\bullet$) trivial for them. Since rewrites neither introduce new labels nor add relational atoms, all other conditions of structural saturation, as well as being being tree-layered and tree-clustered, remain true. Since no labelled formulas have been removed and all inner layers of all sequents remain unchanged, the inner layers remain happy (see Remark 6.10). Therefore, all sequents in $\mathfrak{S}'$ are stable, and so is $\mathfrak{S}'$ itself.

To prove b), observe that each rewrite adds a labelled formula to a sequent $\mathfrak{G} \in \mathfrak{S}$ if and only if the formula is not already in $\mathfrak{G}$. Moreover, the added formulas are subformulas of formulas in the sequent, and no new labels are introduced. Since each $\mathfrak{G}$ contains finitely many formulas and finitely many labels, only finitely many formulas can be added to the sequent. Thus, if there are finitely many sequents in $\mathfrak{S}$, relation $\rightrightarrows_{\mathsf{s}}$ is terminating.

Finally, to prove c), suppose first that $\mathfrak{S}$ is semi-saturated. Then, by definition, $\mathfrak{S}$ is finite, all sequents in $\mathfrak{S}$ are stable, in particular, have happy inner layers, and have naively happy topmost layers. Since all these finitely many sequents are stable, $\mathfrak{S}$ is stable. Since all their layers are (at least) naively happy, no rewrite step is applicable to $\mathfrak{S}$, so $\mathfrak{S}$ is in normal form w.r.t. $\rightrightarrows_{\mathsf{s}}$. Conversely, suppose that $\mathfrak{S}$ is stable and in normal form w.r.t. $\rightrightarrows_{\mathsf{s}}$. Then $\mathfrak{S}$ is a finite set of stable sequents. Since no rewrite is applicable to any sequent from $\mathfrak{S}$, all (topmost) layers of all these sequents are naively happy. Hence, all finitely many sequents from $\mathfrak{S}$ are semi-saturated, and so is $\mathfrak{S}$ itself. □

The following two lemmas are not in the main text and are needed to prove Lemma 7.6.

Lemma A.5. *The following statements hold:*

i) If a sequent $\mathfrak{G}$ is semi-saturated and $\mathfrak{G} \rightsquigarrow_{\Diamond} \mathfrak{G}'$, then the sequent $\mathfrak{G}'$ is stable.

ii) If a sequent $\mathfrak{G}$ is semi-saturated and $\mathfrak{G} \rightsquigarrow_\Diamond \mathfrak{G}'$ according to option 1, then the sequent $\mathfrak{G}'$ is semi-saturated.

Proof. To prove i) assume that a sequent $\mathfrak{G}$ is semi-saturated, in particular, stable (i.e., structurally saturated, tree-layered, and tree-clustered with happy inner layers). Let $\mathfrak{G} \rightsquigarrow_\Diamond \mathfrak{G}'$. For either option from Def. 7.5, the unhappy label y can only be in a topmost layer of $\mathfrak{G}$. Both labels x from option 1 or z from option 2 belong to the same topmost layer. Hence, in either case neither labels nor most labelled formulas from the inner layers are affected and, therefore, they remain happy by Remark 6.10. For option 2, no formulas or labels are removed so all white $A \supset B^\circ$ and $\Box A^\circ$ formulas from the inner layers remain happy. For option 1, formulas $A \supset B^\circ$ from inner layers remain happy because y has no past. Finally, it is possible that $\mathfrak{G}$,k:$\Box A^\circ$ from some inner layer is happy because of $\mathfrak{G}$,y:A°, $k \leq_{\mathfrak{G}} w$, and $w R_{\mathfrak{G}} y$. But since $x \sim y$, also $\mathfrak{G}'$,x:A°. Note that $w \neq y$ since y has no past. Thus, $k \leq_{\mathfrak{G}'} w$ and $w R_{\mathfrak{G}'} x$ means that $\mathfrak{G}'$,k:$\Box A^\circ$ remains happy. For option 1, there are no new labels and y has no past in $\mathfrak{G}$, hence, the only change in $\leq$-relational atoms is that $y \leq y$ turns into $x \leq x$; for option 2, the only new label is z with $z \leq z$ being the only new $\leq$-relational atom. Either way, the $(\leq\mathsf{rf})$- and $(\leq\mathsf{tr})$-structural saturation is ensured and the set of irreflexive $\leq$-relational atoms remains unchanged. The latter implies that the $(\mathsf{mon}^\bullet)$-, (F_1)-, and (F_2)-structural saturation of $\mathfrak{G}'$ follows from the same properties of $\mathfrak{G}$. The $(R\mathsf{tr})$-structural saturation is explicitly enforced. So is $(R\mathsf{rf})$-structural saturation for option 2, where it is necessary for the added label z. This completes the proof that $\mathfrak{G}'$ is structurally saturated. As the rewrite does not affect the layer structure of the sequent, $\mathfrak{G}'$ being tree-layered immediately follows from the same property of $\mathfrak{G}$. To show that $\mathfrak{G}'$ is tree-clustered we consider the two options separately. In option 1 of Def. 7.5, the rewrite performs a substitution of label x for label y and applies the transitive closure. By considering the shortest chains justifying new $R_{\mathfrak{G}'}$-links, and using the structural saturation of both $\mathfrak{G}$ and $\mathfrak{G}'$ along with $x R_{\mathfrak{G}} y$, it follows that:

$$u R_{\mathfrak{G}'} x \implies u R_{\mathfrak{G}} y; \tag{5}$$

$$x R_{\mathfrak{G}'} v \implies x R_{\mathfrak{G}} v; \tag{6}$$

$$u R_{\mathfrak{G}'} v \text{ but not } u R_{\mathfrak{G}} v \implies u R_{\mathfrak{G}} y \& x R_{\mathfrak{G}} v. \tag{7}$$

Let C'_x be the cluster in $\mathfrak{G}'$ that contains x. For any cluster C in $\mathfrak{G}$, either $y \in C$ and $C \setminus \{y\} \subseteq C'_x$ or $y \notin C$ and either $C \subseteq C'_x$ or C remains a cluster in $\mathfrak{G}'$. The case of $y \in C$ is easy. Suppose $y \notin C$, which is not a cluster in $\mathfrak{G}'$. Since all $R_{\mathfrak{G}}$-links unrelated to y are preserved in $\mathfrak{G}'$, cluster C can only grow. So there must be some labels $u \in C$ and $v \notin C$ such that $u R_{\mathfrak{G}'} v$, $v R_{\mathfrak{G}'} u$, but either not $u R_{\mathfrak{G}} v$ or not $v R_{\mathfrak{G}} u$. By (7), in the former case, $u R_{\mathfrak{G}} y$ and $x R_{\mathfrak{G}} v$, hence, $u R_{\mathfrak{G}'} x$ and $x R_{\mathfrak{G}'} v$, while in the latter case, $v R_{\mathfrak{G}} y$ and $x R_{\mathfrak{G}} u$, hence, $v R_{\mathfrak{G}'} x$ and $x R_{\mathfrak{G}'} u$. Either way, by structural saturation of $\mathfrak{G}'$, $u \in C'_x$, and $C \subseteq C'_x$. To show that $\mathfrak{G}'$ is tree-clustered (in option 1 of Def. 7.5), we consider three clusters C_1, C_2, and C' from $\mathfrak{G}'$ such that $C_1 R_{\mathfrak{G}'} C'$ and $C_2 R_{\mathfrak{G}'} C'$ and show that C_1 and C_2 are $R_{\mathfrak{G}'}$-comparable. If any two of them are equal, this is trivial, so assume they are pairwise distinct. If none of them is C'_x, then they are also clusters in $\mathfrak{G}$ and it is sufficient to show that there is some cluster C^* in $\mathfrak{G}$ such that $C_1 R_{\mathfrak{G}} C^*$ and $C_2 R_{\mathfrak{G}} C^*$. Then C_1 and C_2 are $R_{\mathfrak{G}}$-comparable because $\mathfrak{G}$ is tree-clustered and the connection remains in $\mathfrak{G}'$, meaning that they are also $R_{\mathfrak{G}'}$-comparable. If both $C_1 R_{\mathfrak{G}} C'$ and $C_2 R_{\mathfrak{G}} C'$, then $C^* = C'$. If neither $C_1 R_{\mathfrak{G}} C'$ nor $C_2 R_{\mathfrak{G}} C'$, then it follows from (7) that $C_1 R_{\mathfrak{G}} C_y$ and $C_2 R_{\mathfrak{G}} C_y$ where C_y is the cluster in $\mathfrak{G}$ that contains y. So $C^* = C_y$. Finally, if, w.l.o.g. $C_1 R_{\mathfrak{G}} C'$ but not $C_2 R_{\mathfrak{G}} C'$, then $C_2 R_{\mathfrak{G}} C_y$ and $C_x R_{\mathfrak{G}} C'$ by (7) where C_x is the cluster in $\mathfrak{G}$ that contains x. Since $\mathfrak{G}$ is tree-clustered, either $C_x R_{\mathfrak{G}} C_1$ or $C_1 R_{\mathfrak{G}} C_x$. In the latter case, $C_1 R_{\mathfrak{G}} C_y$, so $C^* = C_y$. In the former case, we derive the $R_{\mathfrak{G}'}$-connection $C_2 R_{\mathfrak{G}'} C_1$ directly from $C_2 R_{\mathfrak{G}} C_y$ and $C_x R_{\mathfrak{G}} C_1$. It remains to consider the case when exactly one of the three clusters is C'_x and the other two are clusters of $\mathfrak{G}$. If $C' = C'_x$, then $C_1 R_{\mathfrak{G}} C_y$ and $C_2 R_{\mathfrak{G}} C_y$ by (5), so the above argument with $C^* = C_y$ suffices. Finally, if w.l.o.g. $C_1 = C'_x$, then $C_x R_{\mathfrak{G}} C'$ by (6). If $C_2 R_{\mathfrak{G}} C'$, then C_x is $R_{\mathfrak{G}}$-comparable to C_2 because $\mathfrak{G}$ is tree-clustered, and C'_x is $R_{\mathfrak{G}'}$-comparable to C_2. If $C_2 R_{\mathfrak{G}'} C'$ but not $C_2 R_{\mathfrak{G}} C'$, then $C_2 R_{\mathfrak{G}} C_y$ by (7), and $C_2 R_{\mathfrak{G}'} C'_x$. The situation in option 2 of Def. 7.5 is much simpler: there a single new cluster $\{z\}$ is added to the clusters of $\mathfrak{G}$ and $C R_{\mathfrak{G}'} \{z\}$ iff $C = \{z\}$ or $C R_{\mathfrak{G}} C_y$. The only non-trivial new case to consider is when $C' = \{z\}$, $C_1 \neq \{z\}$, and $C_2 \neq \{z\}$, in which case $C_1 R_{\mathfrak{G}} C_y$ and $C_2 R_{\mathfrak{G}} C_y$, and C_1 and C_2 are $R_{\mathfrak{G}'}$-comparable because they are $R_{\mathfrak{G}}$-comparable. This completes the proof that $\mathfrak{G}'$ is tree-clustered and overall stable.

To prove ii), since most of the conditions were just shown in i), it is sufficient to additionally show that all topmost layers of $\mathfrak{G}'$ are naively happy. Since option 1 adds neither new formulas nor new labels, for most types of formulas their (naive) happiness is inherited from $\mathfrak{G}$. The only exceptions are $\Box^\bullet$- and $\Diamond^\circ$-formulas. The argument is the same for both. We show it for $\Box^\bullet$-formulas. Suppose $u R_{\mathfrak{G}'} v$ and $\mathfrak{G}'$,u:$\Box C^\bullet$. Since no formulas are added in option 1, also $\mathfrak{G}$,u:$\Box C^\bullet$. There are two possibilities: either $u R_{\mathfrak{G}} v$ already in $\mathfrak{G}$ or it was added to ensure the structural saturation of $\mathfrak{G}'$. In the former case, $\mathfrak{G}'$,v:$C^\bullet$ and $\mathfrak{G}'$,v:$\Box C^\bullet$ by the naive happiness of this layer in $\mathfrak{G}$. In the latter case, we have $u R_{\mathfrak{G}'} v$ but not $u R_{\mathfrak{G}} v$, hence, by (7), both $u R_{\mathfrak{G}} y$ and $x R_{\mathfrak{G}} v$. Therefore, $\mathfrak{G}$,y:$\Box C^\bullet$ naive happiness of the layer in $\mathfrak{G}$, so $\mathfrak{G}$,x:$\Box C^\bullet$ because $x \sim y$ in $\mathfrak{G}$, and $\mathfrak{G}$,v:$C^\bullet$ and $\mathfrak{G}$,v:$\Box C^\bullet$, again due to naive happiness. Both $C^\bullet$ and $\Box C^\bullet$ remain at v in $\mathfrak{G}'$. □

Lemma A.6. *Let $\mathfrak{G}$ be a semi-saturated sequent, let $\mathfrak{G} \rightsquigarrow_\Diamond \mathfrak{G}'$, and let $\mathfrak{S}^*$ be a semi-saturation of $\{\mathfrak{G}'\}$. Then all happy $\Diamond^\bullet$-formulas are preserved by the transition from $\mathfrak{G}$ to $\mathfrak{S}^*$, i.e., for all sequents $\mathfrak{G}^* \in \mathfrak{S}^*$ and for all labels u occurring in $\mathfrak{G}^*$, if $\mathfrak{G}$,u:$\Diamond C^\bullet$ is happy, then so is $\mathfrak{G}^*$,u:$\Diamond C^\bullet$.*

Proof. Note that semi-saturation does not remove formulas or labels, nor does it add R-relational atoms. Hence, semi-saturation in option 2 cannot spoil happiness of any $\Diamond^\bullet$-formula, and it is sufficient to show that no happy $\Diamond^\bullet$-formula of $\mathfrak{G}$ becomes unhappy in $\mathfrak{G}'$. For option 2, this is trivial as,

again, nothing is removed. For option 1, the only problems relate to the removal of y and could potentially occur if $\mathfrak{G}$,u:$\Diamond C^\bullet$ was happy because of $uR_{\mathfrak{G}}y$ and $\mathfrak{G}$,y:$C^\bullet$. However, in this case, $uR_{\mathfrak{G}'}x$ and $\mathfrak{G}'$,x:$C^\bullet$ because $x \sim y$ in $\mathfrak{G}$, thus, happiness persists. □

Lemma 7.6. *All the following statements hold:*

a) If a set $\mathfrak{S}$ is semi-saturated and $\mathfrak{S} \approx\!\!\!\Rightarrow_\Diamond \mathfrak{S}'$, then $\mathfrak{S}'$ is semi-saturated.

b) The rewrite relation $\approx\!\!\!\Rightarrow_\Diamond$ is terminating.

c) A set $\mathfrak{S}$ is saturated iff it is semi-saturated and in normal form w.r.t. $\approx\!\!\!\Rightarrow_\Diamond$.

Proof. To prove a) assume that set $\mathfrak{S}$ is finite and all sequents in $\mathfrak{S}$ are semi-saturated (in particular, stable). Observe that $\mathfrak{S}' = \big(\mathfrak{S} \setminus \{\mathfrak{G}\}\big) \cup \mathfrak{S}^*$ for some $\mathfrak{G} \rightsquigarrow_\Diamond \mathfrak{G}'$ where $\mathfrak{S}^*$ is as in Lemma A.6. In Option 1, we replace one semi-saturated sequent $\mathfrak{G}$ with another one sequent $\mathfrak{G}'$, which is semi-saturated by Lemma A.5.ii). Hence, the resulting set is semi-saturated. In option 2, sequent $\mathfrak{G}'$ is stable by Lemma A.5.i). By Lemma 7.2, set $\mathfrak{S}^*$ is semi-saturated. Hence, so is $\mathfrak{S}'$.

To prove b), consider a sequence of sets $\mathfrak{S} \approx\!\!\!\Rightarrow_\Diamond \mathfrak{S}_1 \approx\!\!\!\Rightarrow_\Diamond \mathfrak{S}_2 \approx\!\!\!\Rightarrow_\Diamond \dots$. Let us divide all labels occurring in this sequence into *old*, i.e., occurring in $\mathfrak{S}$, and *new*, i.e., introduced later. We also call unhappy $\Diamond^\bullet$-formulas old or new based on their label. The number of old unhappy $\Diamond^\bullet$-formulas can never increase, and each $\approx\!\!\!\Rightarrow_\Diamond$ transition makes at least one $\Diamond^\bullet$-formula (old or new) happy. Hence, it is sufficient to show that it is impossible to continue creating new labels indefinitely. Using tree-clusteredness, we define the *new-cluster-depth* of a new label x (from a topmost layer of a sequent from one of the sets) to be the length of the shortest chain $oR_{\mathfrak{G}}x_1R_{\mathfrak{G}}\dots R_{\mathfrak{G}}x_nR_{\mathfrak{G}}x$ such that o is an old label while labels $x_0, \dots, x_n$ are new. In particular, the new-cluster-depth of any old label o is 0. To prove termination we consider a Derschowitz–Manna-style ordering on multisets of new-cluster-depths of all new unhappy $\Diamond^\bullet$-formulas from $\mathfrak{S}_i$. Let us denote such a multiset of integers by $\mathsf{unh}_\Diamond(\mathfrak{S}_i)$ and pair it with the number of old unhappy $\Diamond^\bullet$-formulas from $\mathfrak{S}_i$. Let the $\mathfrak{S}_i \approx\!\!\!\Rightarrow_\Diamond \mathfrak{S}_{i+1}$ be performed based on an unhappy $\Diamond A^\bullet$ at label y with new-cluster-depth k. If $k = 0$, i.e., y is an old label, then the number of old unhappy $\Diamond^\bullet$-formulas decreases. It is easy to show by induction on i that for new y, i.e., for $k > 0$, label y forms a singleton cluster $\{y\}$ any non-trivial R-children of y must be R-leaves in the respective topmost layer. Thus, since by part A.6 all $\Diamond^\bullet$-formulas outside of y preserve happiness, all formulas in y disappear, and other clusters with unhappy $\Diamond^\bullet$-formulas preserve their new-cluster-depth, multiset $\mathsf{unh}_\Diamond(\mathfrak{S}_{i+1}) \subsetneq \mathsf{unh}_\Diamond(\mathfrak{S}_i)$, i.e., at least one unhappy $\Diamond^\bullet$-formula, $\Diamond A^\bullet$ at label y, disappeared, no additional unhappy $\Diamond^\bullet$-formulas appeared, and no unhappy $\Diamond^\bullet$-formulas decreased their new-cluster-depth. Alternatively, if $\mathfrak{S}'$ is obtained via option 2, then, again by A.6, no additional unhappy $\Diamond^\bullet$-formulas appear in labels present in $\mathfrak{S}_i$ and some unhappy ones may become happy. However, the semi-saturation can create new unhappy $\Diamond^\bullet$-formulas in the newly created label z, which has cluster-depth $k+1$ (it is easy to see that other labels are unaffected by the semi-saturation). Thus, $\mathsf{unh}_\Diamond(\mathfrak{S}_{i+1}) \subseteq \mathsf{unh}_\Diamond(\mathfrak{S}_i) \setminus \{k\} \cup \{k+1, \dots, k+1\}$ where the multiplicity of added $k+1$ depends on how many unhappy $\Diamond^\bullet$-formulas are in z after semi-saturation. To ensure termination, it remains to note that there is a global upper bound on new-cluster-depths of unhappy $\Diamond^\bullet$-formulas. Indeed, let M be the number of formulas that are subformulas of $\mathfrak{S}$. We show that no unhappy $\Diamond^\bullet$-formula can have new-cluster-depth exceeding $2^M + 1$. This follows from the fact that, due to the subformula property, there are at most 2^M distinct sets of formulas assigned to one label. All new labels z created in option 2 have no past, and $\Diamond^\bullet$-formulas in a child of such z are not used for saturation until z is almost happy. Consider two such new labels z_1 and z_2 such that $z_1R_{\mathfrak{G}}z_2$. Suppose an unhappy $\Diamond A^\bullet$ in z_2 was chosen for the next saturation step. Then $\{z_2\}$ is a singleton cluster and all its non-trivial parents, including z_1 are almost happy. If $z_2 \sim z_1$, then option 1 would dictate to identify z_2 with z_1. Thus, for z_2 to remain a separate cluster, it has to be different from all the labels in its strictly preceding clusters of new labels. Thus, for any sequence of distinct clusters $C_1R_{\mathfrak{G}}\dots R_{\mathfrak{G}}C_k$ of new labels, there must be labels $z_1R_{\mathfrak{G}}\dots R_{\mathfrak{G}}z_k$ such that $z_j \not\sim z_l$ for any $j \neq l$. It means that after at most $k \geq 2^M$ clusters of new labels in a row, the (semi-saturated) new label z would be equivalent to some of the previously created labels and option 1 would prevent extending the sequence further. Since all new unhappy $\Diamond^\bullet$-formulas are pushed in the direction of increasing their new-cluster-depth, which cannot exceed the global bound of $2^M + 1$ and the number of old unhappy $\Diamond^\bullet$-formulas only decreases, we conclude that $\approx\!\!\!\Rightarrow_\Diamond$ terminates.

To prove c), observe that $\mathfrak{S}$ being saturated by definition means that it is semi-saturated and all topmost layers of its sequents are almost happy. A layer is almost happy if and only if it is naively happy and all $\Diamond A^\bullet$-formulas occurring in it are happy. Saturation $\approx\!\!\!\Rightarrow_\Diamond$ can be applied to a semi-saturated set iff it has an unhappy $\Diamond A^\bullet$-formula (in its topmost layer). Hence, $\mathfrak{S}$ is saturated iff it is semi-saturated and in normal form w.r.t. $\approx\!\!\!\Rightarrow_\Diamond$. □

Lemma 7.15. *If $\mathfrak{G}$ is saturated, then $\mathfrak{G}{\uparrow}$ is stable.*

Proof. Assume that $\mathfrak{G}$ is saturated. Then it is stable, i.e., structurally saturated, tree-layered, and tree-clustered with happy inner layers, and, in addition, has almost happy topmost layers. By definition, $\mathfrak{G}{\uparrow}$ consists of $\mathfrak{G}$ to which we add one topmost layer $\hat{L}$ for each unhappy formula x:F° with $F = A \supset B$ or $F = \Box A$ occurring in some not simulated topmost layer L of $\mathfrak{G}$ (these formerly topmost layers become inner in $\mathfrak{G}{\uparrow}$). If $\{x\}$ is a singleton cluster, such a layer $\hat{L}$ is a copy of L (with an additional label z if $F = \Box A$); as can be seen in Fig. 5, if x belongs to a non-singleton cluster C_x, layer $\hat{L}$ additionally contains a copy of C_x and a label $\hat{x}$ (and an additional label z if $F = \Box A$). Since each of these added layers is topmost, all conditions can be verified for each of them independently. As mentioned earlier, Construction 7.8 (and Construction 7.9 for the additional label z) explicitly ensures that $\mathfrak{G}{\uparrow}$ is layered, that the new layers $\hat{L}$ of $\mathfrak{G}{\uparrow}$ enjoy the $(R\mathsf{rf})$- and $(R\mathsf{tr})$-

structural saturation, and that $\mathfrak{G}\uparrow$ satisfies the $(\leq\mathsf{rf})$-, $(\leq\mathsf{tr})$-, $(\mathsf{mon}^\bullet)$-, (F_1)-, and (F_2)-structural saturation. The fact that $\mathfrak{G}\uparrow$ is tree-layered follows from the fact that $\mathfrak{G}$ is tree-layered and, for each added topmost layer $\hat{L}$, we have $L' < \hat{L}$ iff $L' \leq L$. Showing that $\hat{L}$ is tree-clustered in case x belongs to a singleton cluster is immediate, as the R-structure of $\hat{L}$ is either isomorphic to that of the tree-clustered L or is obtained from it by adding one singleton cluster $\{z\}$ that is a leaf w.r.t. R. In case x belongs to a non-singleton cluster C_x, it is easy to see from Construction 7.8 and Construction 7.9 that $\hat{L}$ is partitioned into the following clusters: $\hat{C}'_x = \{\hat{x}'_j \mid x_j \in C_x\}$, $\hat{C}''_x = \{\hat{x}''_j \mid x_j \in C_x\}$, $\{\hat{x}\}$, which we call x-clusters, as well as clusters $\hat{C} := \{\hat{y}_j \mid y_j \in C\}$ for each cluster $C \neq C_x$ of L, which we call y-clusters and, in case $F = \Box A$, additionally cluster $\{z\}$ (see Fig. 5). It is equally easy to see for arbitrary y-clusters $\hat{C}$ and $\hat{C}^*$ that:

a) $\hat{C}R_{\mathfrak{G}\uparrow}\hat{C}^*$ iff $CR_{\mathfrak{G}}C^*$,
b) $\hat{C}R_{\mathfrak{G}\uparrow}\hat{C}'_x$ iff $\hat{C}R_{\mathfrak{G}\uparrow}\hat{C}''_x$ iff $\hat{C}R_{\mathfrak{G}\uparrow}\{\hat{x}\}$ iff $CR_{\mathfrak{G}}C_x$ (also iff $\hat{C}R_{\mathfrak{G}\uparrow}\{z\}$ in presence of z),
c) $\hat{C}'_xR_{\mathfrak{G}\uparrow}\hat{C}$ iff $\hat{C}''_xR_{\mathfrak{G}\uparrow}\hat{C}$ iff $\{\hat{x}\}R_{\mathfrak{G}\uparrow}\hat{C}$ iff $C_xR_{\mathfrak{G}}\hat{C}$ (but never $\{z\}R_{\mathfrak{G}\uparrow}\hat{C}$),
d) $\hat{C}'_xR_{\mathfrak{G}\uparrow}\hat{C}''_x$, $\hat{C}'_xR_{\mathfrak{G}\uparrow}\{\hat{x}\}$, $\{\hat{x}\}R_{\mathfrak{G}\uparrow}\hat{C}''_x$ are the only non-reflexive relations among x-clusters,
e) in presence of z: $\hat{C}'_xR_{\mathfrak{G}\uparrow}\{z\}$, $\{\hat{x}\}R_{\mathfrak{G}\uparrow}\{z\}$, and $\{z\}R_{\mathfrak{G}\uparrow}\{z\}$ are the only remaining relations involving $\{z\}$.

Consider any three of these clusters C_1, C_2, and C_3 from $\hat{L}$ such that $C_1R_{\mathfrak{G}\uparrow}C_3$ and $C_2R_{\mathfrak{G}\uparrow}C_3$. We need to show that C_1 and C_2 are comparable in $\mathfrak{G}\uparrow$. Omitting some trivial cases, assume that the three clusters are pairwise distinct. We distinguish cases depending on whether these clusters are x-clusters, y-clusters, etc. The case of $C_3 = \{z\}$ reduces to $C_3 = \{\hat{x}\}$ because $C_iR_{\mathfrak{G}\uparrow}\{z\}$ iff $C_iR_{\mathfrak{G}\uparrow}\{\hat{x}\}$ or $C_i = \{z\}$, the latter case being trivial. All other cases involving $\{z\}$ are also trivial. The case when both C_1 and C_2 are x-clusters is trivial because they are all pairwise comparable in $\mathfrak{G}\uparrow$. Thus, it is sufficient to consider the cases when none of the clusters is $\{z\}$ and, w.l.o.g., C_1 is a y-cluster. All non-$\{z\}$ clusters in $\hat{L}$ have origin clusters in L: C for a y-cluster $\hat{C}$ or C_x for each of the x clusters. Let us call the origins of C_1, C_2, and C_3 clusters C^o_1, C^o_2, and C^o_3 respectively. Considering all four possibilities on whether C_2 and C_3 are x-clusters or y-clusters, one can see that $C^o_1R_{\mathfrak{G}}C^o_3$ and $C^o_2R_{\mathfrak{G}}C^o_3$. Since $\mathfrak{G}$ is tree-clustered, C^o_1 and C^o_2 are comparable in $\mathfrak{G}$. And this implies that C_1 and C_2 are comparable in $\mathfrak{G}\uparrow$ in each of the four cases. This concludes the proof that $\mathfrak{G}\uparrow$ is tree-clustered.

Finally, since $\mathfrak{G}$ is saturated, all its topmost layers are almost happy, that is, all formulas that are not of the shape $A \supset B^\circ$ or $\Box A^\circ$ are happy. By adding new topmost layers without removing any formulas or any labels we make all of them happy, without making any other formula of $\mathfrak{G}$ unhappy (here we rely on the local formulation of happiness for $\mathfrak{G}$,x:$A \supset B^\bullet$ in Def. 5.3). Thus, all inner layers of $\mathfrak{G}\uparrow$ are happy, and $\mathfrak{G}\uparrow$ is stable. □

Lemma 7.19. *It holds that:*

a) If $\boldsymbol{\mathfrak{G}}$ is saturated and $\boldsymbol{\mathfrak{G}} \rightsquigarrow_{\circlearrowleft} \mathfrak{G}'$, then $\mathfrak{G}'$ is saturated.
b) The rewrite relation $\rightsquigarrow_{\circlearrowleft}$ is terminating.

Proof. To prove a) assume that $\boldsymbol{\mathfrak{G}}$ is saturated. Let us consider a sequent $\mathfrak{G} \in \boldsymbol{\mathfrak{G}}$ such that $\mathfrak{G} \rightsquigarrow_{\circlearrowleft} \mathfrak{G}'$. Then, $\mathfrak{G}$ contains either an unhappy R-triangle loop or an unhappy U-triangle loop. Sequent $\mathfrak{G}'$ is obtained from $\mathfrak{G}$ by substituting label s for all occurrences of label t and then closing R under transitivity (Definition 7.18). By assumption, $\mathfrak{G}$ is stable, i.e., it is structurally saturated, tree-layered, and tree clustered with happy inner layers. Irrespective of which kind of triangle loop occurs in $\mathfrak{G}$, labels s and t belong to the same topmost layer of the sequent. Hence, neither labels nor labelled formulas from inner layers are affected by the rewrite and, by Remark 6.10, their happiness is preserved for most formulas. There are two types of formulas that might be exceptions. We consider them separately. For the first type, if $\mathfrak{G}$,k:$A \supset B^\circ$ from some inner layer is happy because $k{\leq_{\mathfrak{G}}}t$, $\mathfrak{G}$,t:$A^\bullet$, and $\mathfrak{G}$,t:B°, then both $\mathfrak{G}'$,s:$A^\bullet$ and $\mathfrak{G}'$,s:B° because $s \sim t$. Thus, $\mathfrak{G}'$,k:$A \supset B^\circ$ remains happy because $k{\leq_{\mathfrak{G}'}}s$. For the other type, consider the situation where $\mathfrak{G}$,k:$\Box A^\circ$ from some inner layer is happy because $\mathfrak{G}$,v:A°, $k{\leq_{\mathfrak{G}}}w$, and $wR_{\mathfrak{G}}v$ where one or both of labels w and v are t. If only $v = t$, then $\mathfrak{G}'$,s:A° because $s \sim t$, thus, $k{\leq_{\mathfrak{G}'}}w$ and $wR_{\mathfrak{G}'}s$ make $\mathfrak{G}'$,k:$\Box A^\circ$ happy. If only $w = t$, then $k{\leq_{\mathfrak{G}'}}s$ and $sR_{\mathfrak{G}'}v$ make $\mathfrak{G}'$,k:$\Box A^\circ$ happy. Finally, if both $v = w = t$, then $\mathfrak{G}'$,s:A° because $s \sim t$ and $k{\leq_{\mathfrak{G}'}}s$ an $sR_{\mathfrak{G}'}s$ make $\mathfrak{G}'$,k:$\Box A^\circ$ happy. In all cases, all formulas in inner layers remain happy. The rewrite does not introduce new labels, and the only changes in $\leq$-relational atoms are that $t{\leq}t$ turns into $s{\leq}s$ and, for any $k \neq t$, if $k{\leq_{\mathfrak{G}}}t$, then $k{\leq_{\mathfrak{G}'}}s$. The $(R\mathsf{tr})$-structural saturation of $\mathfrak{G}'$ is explicitly enforced, while the $(\leq\mathsf{rf})$- and $(R\mathsf{rf})$-structural saturation are not affected by the removal of label t. The only non-trivial case for $(\leq\mathsf{tr})$-structural saturation is when $w{\leq_{\mathfrak{G}'}}k$ and $k{\leq_{\mathfrak{G}'}}s$ because $k{\leq_{\mathfrak{G}}}t$ where $k \neq s$. Then k and w belong to unaffected layers, hence, $w{\leq_{\mathfrak{G}}}k$. Thus, $w{\leq_{\mathfrak{G}}}t$ by $(\leq\mathsf{tr})$-structural saturation of $\mathfrak{G}$, and $w{\leq_{\mathfrak{G}'}}s$ by construction. The $(\mathsf{mon}^\bullet)$-structural saturation follows from $s \sim t$. Finally, for (F_1)-, and (F_2)-structural saturation of $\mathfrak{G}'$, the most non-trivial case is when, for one of the two, $kR_{\mathfrak{G}'}w$ and $k{\leq_{\mathfrak{G}'}}s$ because $k{\leq_{\mathfrak{G}}}t$ where $k \neq s$. Again k and w belong to unaffected layers, hence, $kR_{\mathfrak{G}}w$. By the same structural saturation property of $\mathfrak{G}$, there exists a label z such that $w{\leq_{\mathfrak{G}}}z$ and $tR_{\mathfrak{G}}z$. If $t \neq z$, then $sR_{\mathfrak{G}'}z$ and $w{\leq_{\mathfrak{G}'}}z$, so the same label z fulfills the conditions. If $t = z$, then $w{\leq_{\mathfrak{G}'}}s$, which together with $sR_{\mathfrak{G}'}s$ means that s fulfills the conditions. The remaining case is symmetric. This concludes the proof that $\mathfrak{G}'$ is structurally saturated. Since the rewrite does not affect the layer structure of the sequent, we immediately conclude that $\mathfrak{G}'$ is tree-layered. The proof that $\mathfrak{G}'$ is tree-clustered proceeds in the same way as the proof of case a) of Lemma 7.6, in the case of option 1 of Definition 7.5. This concludes the proof that $\mathfrak{G}'$ is stable. By assumption, $\mathfrak{G}$ is saturated. This means that all topmost layers of $\mathfrak{G}$ are almost happy, that is, all formulas that are not of the shape $A \supset B^\circ$ or $\Box A^\circ$ are happy. It remains to show that all topmost layers of $\mathfrak{G}'$ are almost happy. Almost happiness of labelled

propositional formulas of $\mathfrak{G}'$ is local and, hence, is not affected by the new R- and $\leq$-links. Happiness of $\Box^\bullet$- and $\Diamond^\circ$-formulas is shown as in the proof of case a) of Lemma 7.6.

To prove b), observe that, unlike in the case of $\rightsquigarrow_\Diamond$ (Definition 7.5), $\rightsquigarrow_{\circlearrowleft}$ never introduces new labels in the sequents on which it operates. Whenever $\rightsquigarrow_{\circlearrowleft}$ is applied to a sequent $\mathfrak{G}$, the number of labels occurring in one of the topmost layers of $\mathfrak{G}$ is reduced by one because $s \neq t$. Therefore, for any topmost layer L in $\mathfrak{G}$ containing $n \geq 1$ labels, transformation $\rightsquigarrow_{\circlearrowleft}$ can be applied at most $n-1$ times to the labels occurring in L. Since saturated set $\boldsymbol{\mathfrak{S}}$ is finite, the rewrite $\rightsquigarrow_{\circlearrowleft}$ is terminating. □

E. Proofs from Section 8

Theorem 8.1. *If the algorithm shown in Fig. 7 terminates in Step 4, then formula F is not a theorem of* IS4.

Proof. If the algorithm terminates in Step 4, then $\mathfrak{G}_i\uparrow = \mathfrak{G}_i$ for some non-axiomatic sequent $\mathfrak{G}_i \in \boldsymbol{\mathfrak{S}}_i$. It is easy to see that label r, the only label in $\boldsymbol{\mathfrak{S}}_0$, is never removed by the algorithm because it always remains in the root cluster of the root layer of every sequent in every set. Indeed: semi-saturation does not remove labels; saturation does not remove labels from root clusters; lifting saturation does not remove labels; finally, looping saturation does not remove labels from the root layer. Hence, $\mathfrak{G}_\mathfrak{i}, r{:}F^\circ$. For each unhappy topmost layer L of $\mathfrak{G}_\mathfrak{i}$, there is some inner layer L' simulating L (see Definition 7.11) via a simulation S_L. Let $\mathfrak{G}$ be obtained by first adding to $\mathfrak{G}_\mathfrak{i}$ all relational atoms $x{\leq}x'$ such that $x'\mathsf{S}_L x$ for some unhappy topmost layer L and then closing the result under transitivity of $\leq$. It is sufficient to show that $\mathfrak{G}$ is happy. Then by Theorem 5.8, $\mathcal{M}_\mathfrak{G}, r \not\Vdash F$. Therefore, by Theorem 2.5, F is not a theorem in IS4.

We need to show that $\mathfrak{G}$ is structurally saturated and all its formulas are happy. Since no new labels were added, $(\leq\mathsf{rf})$- and $(R\mathsf{rf})$-structural saturation is preserved. $(R\mathsf{tr})$-structural saturation is preserved because no new R-links were added. $(\leq\mathsf{tr})$-structural saturation is explicitly enforced. To show that $(\mathsf{mon}^\bullet)$-, (F_1)-, and (F_2)-structural saturation are preserved, it is sufficient to demonstrate them for each of the $\leq$-links added before the transitive closure. For the (F_1)-, and (F_2)-structural saturation these are exactly the simulation conditions S1 and S2 from Definition 7.11. For the $(\mathsf{mon}^\bullet)$-simulation this follows from the fact that $\mathsf{S}_L \subseteq \sim$ for all L, which means that whenever $\mathfrak{G}, x{:}A^\bullet$ and $x{\leq_\mathfrak{G}}x'$ because $x'\mathsf{S}_L x$ for some L, we have $x \sim x'$ and, hence, $\mathfrak{G}, x'{:}A^\bullet$. This completes the proof that $\mathfrak{G}$ is structurally saturated. Since no formulas or links were removed, all formulas happy in $\mathfrak{G}_\mathfrak{i}$ remain happy in $\mathfrak{G}$ (here we again rely on the local formulation of happiness for $\mathfrak{G}, x{:}A \supset B^\bullet$ in Def. 5.3). All formulas from the inner layers of saturated sequent $\mathfrak{G}_\mathfrak{i}$ were happy, as were all formulas apart from possibly some F° of one of the two shapes $A \supset B^\circ$ or $\Box A^\circ$ in topmost layers. For any such unhappy $\mathfrak{G}_\mathfrak{i}, x{:}F^\circ$ from some topmost layer L of $\mathfrak{G}_\mathfrak{i}$, there is an inner layer L' and, by Prop. 7.12, there is a label $x' \in L'$ such that $x'\mathsf{S}_L x$, hence, $x{\leq_\mathfrak{G}}x'$. At the same time $x \sim x'$ so $\mathfrak{G}_\mathfrak{i}, x'{:}F^\circ$ is happy in its inner layer. If $F^\circ = A \supset B^\circ$, we must have $\mathfrak{G}_\mathfrak{i}, z{:}A^\bullet$, $\mathfrak{G}_\mathfrak{i}, z{:}B^\circ$, and $x'{\leq_{\mathfrak{G}_\mathfrak{i}}}z$. Then $x{\leq_\mathfrak{G}}z$ by $(\leq\mathsf{tr})$-structural saturation of $\mathfrak{G}$, which makes $\mathfrak{G}, x{:}F^\circ$ happy. Finally, if $F^\circ = \Box A^\circ$, we must have $\mathfrak{G}_\mathfrak{i}, z{:}A^\circ$, $x'{\leq_{\mathfrak{G}_\mathfrak{i}}}u$, and $uR_{\mathfrak{G}_\mathfrak{i}}z$. Then $x{\leq_\mathfrak{G}}u$ by $(\leq\mathsf{tr})$-structural saturation of $\mathfrak{G}$ and $uR_\mathfrak{G}z$ by construction, which makes $\mathfrak{G}, x{:}F^\circ$ happy. This completes the proof that $\mathfrak{G}$ is a happy sequent. □

F. Proofs from Section 9

Proposition 9.4. *Every rule* r *in* $\mathsf{labIS4}^\star_\leq$ *is derivable in* $\mathsf{labIS4}'_\leq$*, and if* $\mathfrak{G}$ *is the conclusion of an instance of* r*, then each premise has the form* $\mathfrak{G} + \mathfrak{H}$ *for some* $\mathfrak{H}$*.*

Proof. Whenever the rule of $\mathsf{labIS4}_\leq$ that corresponds to r removes the principal formula, we use first the contraction rule to create a second copy. Then, rules of $\mathsf{labIS4}^\star_\leq$ adding formulas to the antecedent are emulated by the respective rule of $\mathsf{labIS4}_\leq$ followed by $\mathsf{mon}^\bullet$ to propagate the antecedent formula upward.

The reflexive relational atoms $x{\leq}x$ in $\supset^\bullet_\mathsf{k}$ and $\Box^\bullet$ are present in the conclusion by the assumption that the conclusion is a proper sequent.

The composite rule $\Diamond^\bullet_\star$ of $\mathsf{labIS4}^\star_\leq$ is emulated by using $\Diamond^\bullet$ to create label $\hat{x}$, followed by a sequence of F_2 rules to create other labels from $\hat{U}$ (it is important to perform these rules in the order compatible with the order on layers, i.e., without skipping intermediate layers), followed by $R\mathsf{rf}$ and $\leq\mathsf{rf}$ for all these created labels, followed by instances of $\leq\mathsf{tr}$ to add all requisite $\leq$-relational atoms (relying on the $\leq\mathsf{tr}$-structural saturation of the conclusion sequent), followed by instances of $R\mathsf{tr}$ to add all requisite R-relational atoms, followed by instances of admissible $\mathsf{mon}^\bullet$ to propagate A upward.

The composite rule $\supset^\circ_\star$ of $\mathsf{labIS4}^\star_\leq$ is emulated by using $\supset^\circ$ to create label $\hat{x}$, followed by a sequence of F_1 and F_2 rules to create other labels from $\hat{L}$ (it is important to perform these rules in the order of increasing distance from x in layer L, i.e., without skipping intermediate labels), followed by $R\mathsf{rf}$ and $\leq\mathsf{rf}$ for all these created labels, followed by instances of $R\mathsf{tr}$ to add all requisite R-relational atoms (relying on the $R\mathsf{tr}$-structural saturation in layer L), followed by instances of $\leq\mathsf{tr}$ to add all requisite $\leq$-relational atoms, followed by instances of admissible $\mathsf{mon}^\bullet$ to lift all succedent formulas of L into $\hat{L}$.

The composite rule $\Box^\circ_\star$ of $\mathsf{labIS4}^\star_\leq$ is emulated in the same way, only using $\Box^\circ$ to create $\hat{x}$ and $\hat{z}$ in place of $\supset^\circ$. Note that here instances of $R\mathsf{tr}$ would additionally add $\hat{u}R\hat{z}$ whenever uRx was present in the conclusion sequent because $\hat{u}R\hat{x}$ is created as in the case of $\supset^\circ_\mathsf{k}$, while $\hat{x}R\hat{z}$ is created by rule $\Box^\circ$. □

Proposition 9.8. *Let $\mathcal{D}$ be a tidy derivation. If the conclusion of $\mathcal{D}$ is proper, then every sequent in $\mathcal{D}$ is also proper.*

Proof. We also prove that for every instance of a unary or binary rule $\mathsf{r} \notin \{\supset^\circ_\star, \Box^\circ_\star\}$ in $\mathcal{D}$, the label of the principal formula is in a topmost layer (of both the conclusion and the premise(s)).

The proposition and this statement are proved by mutual induction on the given tidy derivation. It is sufficient to

consider each rule r of $\mathsf{labIS4}^{\star}_{\leq}$ separately. Let $\mathfrak{G}$ be the conclusion of an instance of r in $\mathcal{D}$ and $\mathfrak{H}$ be (one of) its premise(s).

- $\mathsf{r} \in \{\wedge^{\bullet}_{\mathsf{k}}, \wedge^{\circ}_{\mathsf{k}}, \vee^{\bullet}_{\mathsf{k}}, \vee^{\circ}_{\mathsf{k}}, \supset^{\bullet}_{\mathsf{k}}, \Box^{\bullet}, \Diamond^{\circ}, 4^{\bullet}, 4^{\circ}\}$ These rules create neither new labels nor new relational atoms, and have $f(\mathfrak{H}) = f(\mathfrak{G})$ by (4). Hence, most aspects of properness for $\mathfrak{H}$ are trivially inherited from $\mathfrak{G}$, including verticality, lack of non-singleton clusters, tree-clusteredness and tree-layeredness, as well as most aspects of structural completeness, with the exception of $\mathsf{mon}^{\bullet}$, which potentially could have been violated by adding formulas to inner layers. However, the tidiness of $\mathcal{D}$ means that all added formulas have been added to labels that are allowed, which in restricted sequents can only be in topmost layers. In other words, whenever rule r adds a labelled formula $x{:}A^{\bullet}$, label x has no non-trivial futures, i.e., there are no labels $y \neq x$ such that $x{\leq_{\mathfrak{H}}}y$. Thus, sequent $\mathfrak{H}$ is $\mathsf{mon}^{\bullet}$-saturated as well, and, hence, proper. Finally, since the label of the principal formula for all these rules is in the same layer as the label(s) to which formulas are added (the same label for $\wedge^{\bullet}_{\mathsf{k}}$, $\wedge^{\circ}_{\mathsf{k}}$, $\vee^{\bullet}_{\mathsf{k}}$, $\vee^{\circ}_{\mathsf{k}}$, and $\supset^{\bullet}_{\mathsf{k}}$ or a parent label for $\Box^{\bullet}$, $\Diamond^{\circ}$, $4^{\bullet}$, and 4°), the principal formula is also in a topmost layer.
- $\mathsf{r} = \Diamond^{\bullet}_{\star}$ Let $x{:}\Diamond A^{\bullet}$ be the principal formula of the rule. Although this rule creates a new label y forming a singleton cluster, it is added to the existing layer L_x and has no past in $\mathfrak{H}$ (nor non-trivial futures). Thus, tree-layeredness, verticality, and singleton-cluster conditions are all preserved. Note that by (4) the new label y must be allowed in $\mathfrak{H}$, which by definition of restrictedness implies that L_x is topmost (in both $\mathfrak{G}$ and $\mathfrak{H}$). The relational atoms added by the rule clearly ensure tree-clusteredness, $R\mathsf{tr}$-, $R\mathsf{rf}$-, and ${\leq}\mathsf{rf}$-structural completeness for the newly created label, whereas ${\leq}\mathsf{tr}$-structural completeness is trivial. $\mathsf{mon}^{\bullet}$-structural completeness is satisfied since no formulas have been added to labels existing in $\mathfrak{G}$, whereas the only new $\leq$-relational atom is $y{\leq_{\mathfrak{H}}}y$ for which $\mathsf{mon}^{\bullet}$-structural completeness is trivial. F_1- and F_2-structural completeness for this new atom $y{\leq_{\mathfrak{H}}}y$ are similarly trivial. As for the new R-atoms, they all are of the form $zR_{\mathfrak{H}}y$ where both z and y are in the topmost layer L_x of $\mathfrak{H}$. Since neither has non-trivial futures, F_1- and F_2-structural completeness is fulfilled.
- $\mathsf{r} \in \{\supset^{\circ}_{\star}, \Box^{\circ}_{\star}\}$ Let $x{:}A^{\circ}$ be the principal formula of the rule and x' be one of the new labels created by the rule such that $x{\leq_{\mathfrak{H}}}x'$. All the labels created by this rule are in the same layer $L_{x'}$ as x' so that $L_x \leq_{\mathfrak{H}} L_{x'}$ and this layer becomes a new topmost layer. All newly created labels (save, in the case of $\Box^{\circ}_{\star}$, for one label) have past. Clearly, tree-layeredness, verticality, F_1-, F_2-, $\mathsf{mon}^{\bullet}$-, ${\leq}\mathsf{rf}$- and ${\leq}\mathsf{tr}$-structural completeness for the newly created labels are all ensured. The tree-clusteredness for the new layer, as well as, $R\mathsf{tr}$- and $R\mathsf{rf}$-structural completeness for the new layer $L_{x'}$ is inherited from those for L_x (modulo an additional label y created by $\Box^{\circ}_{\star}$ in a way that ensures all these properties). Note that, since the rule only touches newly created labels, which are created in a new, topmost layer, it is not necessary that x is in a topmost layer. □

In order to proof Lemma A.9, we need another definition and an auxiliary lemma.

Definition A.7. Let a derivation $\mathcal{D}_n$ be an n-unfolding of a stable set $\boldsymbol{\mathfrak{S}}$, with premises $\hat{\mathfrak{G}}_1, \ldots, \hat{\mathfrak{G}}_k$ being n-unfoldings of a specific sequent $\mathfrak{G} \in \boldsymbol{\mathfrak{S}}$. Let $\mathsf{U}_i \subseteq \ell(\mathfrak{G}) \times \ell(\hat{\mathfrak{G}}_{\mathsf{i}})$ be n-unfolding relations witnessing that $\hat{\mathfrak{G}}_i$ is an n-unfolding of $\mathfrak{G}$ for each $i = 1..k$. We say that $\mathcal{D}_n$ is ***allowed*** for a label $x \in \ell(\mathfrak{G})$ iff for every $i = 1..k$, every label $\hat{x} \in \ell(\hat{\mathfrak{G}}_{\mathsf{i}})$ such that $x\mathsf{U}_i\hat{x}$ is allowed in $\hat{\mathfrak{G}}_i$.

Lemma A.8. *Assume that $\boldsymbol{\mathfrak{S}} \rightrightarrows_{\mathsf{s}} \boldsymbol{\mathfrak{S}}'$, that $\mathfrak{G} \in \boldsymbol{\mathfrak{S}}$ is the sequent replaced in the rewriting step, and that $x{:}C$ is the unhappy formula that is expanded according to Def. 7.1. If $\boldsymbol{\mathfrak{S}}$ is F-unfoldable such that for every $N \in \mathbb{N}$ there is an N-unfolding $\mathcal{D}_N$ of $\boldsymbol{\mathfrak{S}}$ that is allowed for x, then $\boldsymbol{\mathfrak{S}}'$ is also F-unfoldable.*

Proof. We need to show that $\boldsymbol{\mathfrak{S}}'$ is F-unfoldable, i.e., for every n there is a tidy derivation $\mathcal{D}'_n$ of $\mathfrak{G}_0(F)$ such that each premise of $\mathcal{D}'_n$ is an n-unfolding of some sequent from $\boldsymbol{\mathfrak{S}}'$.

It is sufficient to consider each way that $\boldsymbol{\mathfrak{S}}'$ can be obtained from $\boldsymbol{\mathfrak{S}}$ by steps listed in Def. 7.1. For each case of Def. 7.1, we pick some $N \geq n$ and extend $\mathcal{D}_N$ to $\mathcal{D}'_n$. The only sequent from $\boldsymbol{\mathfrak{S}}$ that is not present in $\boldsymbol{\mathfrak{S}}'$ is $\mathfrak{G}$, which is replaced in $\boldsymbol{\mathfrak{S}}'$ either with sequent $\mathfrak{G}'$ or with two sequents $\mathfrak{G}'$ and $\mathfrak{G}''$, depending on the step. Hence, the only premises of $\mathcal{D}_N$ that have to be extended are the N-unfoldings of $\mathfrak{G}$. All the other premises are N-unfoldings and, hence, n-unfoldings of some (stable) sequent from $\boldsymbol{\mathfrak{S}}$ that is still present in $\boldsymbol{\mathfrak{S}}'$.

Since $\mathcal{D}_N$ is allowed for x, for each premise $\hat{\mathfrak{G}}$ of $\mathcal{D}_N$ that is an N-unfolding of $\mathfrak{G}$, there is an N-unfolding relation $\mathsf{U} \subseteq \ell(\mathfrak{G}) \times \ell(\hat{\mathfrak{G}})$ such that $\hat{x} \notin f(\hat{\mathfrak{G}})$ whenever $x\mathsf{U}\hat{x}$. We will break up the steps of Def. 7.1 into substeps that only add one or two formulas to one particular label y with $xR_{\mathfrak{G}}y$.

The only unfolding condition from Def. 9.3 that prevents U from making $\hat{\mathfrak{G}}$ an N-unfolding of $\mathfrak{G}'$ (and of $\mathfrak{G}''$) is (U1), and only with regard to labels $\hat{y}$ such that $y\mathsf{U}\hat{y}$ because y is the only label of $\mathfrak{G}$ with formulas added. Additionally, when U is modified, (U5)–(U6) need to checked. Every time we extend the derivation $\mathcal{D}_N$, the set of forbidden labels for the premise of each added rule must be the same as for the rule's conclusion to preserve tidiness.

1) Suppose $\mathfrak{G}, x{:}A \wedge B^{\bullet}$ and the missing $x{:}A^{\bullet}$ and $x{:}B^{\bullet}$ were added to obtain $\mathfrak{G}'$. We use $N = n$, set $\mathsf{U}' := \mathsf{U}$, which trivially ensures (U5)–(U6). Using the fact that $x\mathsf{U}\hat{x}$ implies that $\hat{x}$ is allowed and that $\hat{\mathfrak{G}}, \hat{x}{:}A \wedge B^{\bullet}$ by (U1) for $\mathfrak{G}$, to obtain $\hat{\mathfrak{G}}'$ we extend premise $\hat{\mathfrak{G}}$ of $\mathcal{D}_n$ by rules
$$\wedge^{\bullet}_{\mathsf{k}} \frac{\mathcal{R}, \Gamma, \hat{x}{:}A \wedge B, \hat{x}{:}A, \hat{x}{:}B \Longrightarrow \Delta}{\mathcal{R}, \Gamma, \hat{x}{:}A \wedge B \Longrightarrow \Delta}$$
for each label $\hat{x}$ such that $x\mathsf{U}\hat{x}$. It is easy to see that U makes $\hat{\mathfrak{G}}'$ and n-unfolding of $\mathfrak{G}'$ because ${}_{\hat{\mathfrak{G}}'}\hat{x} \sim {}_{\mathfrak{G}'}x$ whenever $x\mathsf{U}\hat{x}$.

2) The case when $\mathfrak{G}$,x:$A \vee B^{\circ}$ and x:A° and x:B° were added is analogous, with $\vee^{\circ}_{\mathsf{k}}$ used instead of $\wedge^{\bullet}_{\mathsf{k}}$.

3) Suppose $\mathfrak{G}$,x:$\Box A^{\bullet}$. We further break this step into substeps consisting of adding the missing y:$A^{\bullet}$ and y:$\Box A^{\bullet}$ for only one label y such that $x R_{\mathfrak{G}} y$. Note that $u R_{\hat{\mathfrak{G}}} u$ for each label $u \in \ell(\hat{\mathfrak{G}})$ due to $\hat{\mathfrak{G}}$ being proper. The sequence of rules depends on the position of x relative to y:

a) If $x = y$, use $N = n$ and set $\mathsf{U}' := \mathsf{U}$. Again, the only problem to be fixed is (U1). Each $\hat{x}$ such that $x \mathsf{U} \hat{x}$ is allowed and, by (U1) for $\mathfrak{G}$, we have $\hat{\mathfrak{G}}$,$\hat{x}$:$\Box A^{\bullet}$. To obtain $\hat{\mathfrak{G}}'$, we extend premise $\hat{\mathfrak{G}}$ of $\mathscr{D}_n$ by rules

$$\Box^{\bullet}\,\frac{\mathcal{R}, \hat{x} R \hat{x}, \Gamma, \hat{x}{:}\Box A, \hat{x}{:}A \Longrightarrow \Delta}{\mathcal{R}, \hat{x} R \hat{x}, \Gamma, \hat{x}{:}\Box A \Longrightarrow \Delta}$$

for each $\hat{x}$ with $x \mathsf{U} \hat{x}$. Thus, ${}_{\hat{\mathfrak{G}}'}\hat{x} \sim {}_{\mathfrak{G}'}x$ whenever $x \mathsf{U} \hat{x}$.

b) If x and y are not in the same cluster, again use $N = n$ and set $\mathsf{U}' := \mathsf{U}$. By (U5)–(U6), there exists $\hat{x}$ such that $x \mathsf{U} \hat{x}$. Since $\mathscr{D}_n$ is allowed for x, label $\hat{x}$ is allowed in $\hat{\mathfrak{G}}$. By (U2), $\hat{x} R_{\hat{\mathfrak{G}}} \hat{y}$ for each $\hat{y}$ such that $y \mathsf{U} \hat{y}$. By Def. 9.6(a), all these $\hat{y}$ are also allowed in $\hat{\mathfrak{G}}$. By (U1) for $\mathfrak{G}$, we have $\hat{\mathfrak{G}}$,$\hat{x}$:$\Box A^{\bullet}$. Again (U1) is the only problem to be fixed. To obtain $\hat{\mathfrak{G}}'$, we extend premise $\hat{\mathfrak{G}}$ of $\mathscr{D}_n$ by rules

$$\Box^{\bullet},4^{\bullet}\,\frac{\mathcal{R}, \hat{x} R \hat{y}, \Gamma, \hat{x}{:}\Box A, \hat{y}{:}\Box A, \hat{y}{:}A \Longrightarrow \Delta}{\mathcal{R}, \hat{x} R \hat{y}, \Gamma, \hat{x}{:}\Box A \Longrightarrow \Delta}$$

for each label $\hat{y}$ such that $y \mathsf{U} \hat{y}$. This ensures that ${}_{\hat{\mathfrak{G}}'}\hat{y} \sim {}_{\mathfrak{G}'}y$ whenever $y \mathsf{U} \hat{y}$.

c) If $x \neq y$ are in the same cluster C, it is not a singleton cluster. Let $|C| = k \geq 2$. Use $N = n{+}1$. By (U6) for $\mathfrak{G}$, sequent $\hat{\mathfrak{G}}$ contains $n{+}1$ unfolded copies $\hat{C}_1, \ldots, \hat{C}_{n+1}$ of all labels from cluster C (the unfoldings are not themselves clusters as $\hat{\mathfrak{G}}$ is a proper sequent). In particular, there is $\hat{x}_1 \in \hat{C}_1 \subseteq \ell(\hat{\mathfrak{G}})$ such that $x \mathsf{U} \hat{x}_1$ and there are $\hat{y}_j \in \hat{C}_j \subseteq \ell(\hat{\mathfrak{G}})$ for $j = 2..n{+}1$ such that $y \mathsf{U} \hat{y}_j$ for $j = 2..n{+}1$. Since $\mathscr{D}_{n+1}$ is allowed for x, label $\hat{x}_1$ is allowed in $\hat{\mathfrak{G}}$. By (U1) for $\mathfrak{G}$, we have $\hat{\mathfrak{G}}$,$\hat{x}_1$:$\Box A^{\bullet}$. Since the proper sequent $\hat{\mathfrak{G}}$ must be Rtr-saturated, it follows from (U6) for $\mathfrak{G}$ that $\hat{x}_1 R_{\hat{\mathfrak{G}}} \hat{y}_j$ for $j = 2..n{+}1$. Hence, by Def. 9.6(a), all $\hat{y}_j$ for $j = 2..n+1$ are also allowed. Here, we restore (U1) only for n labels $\hat{y}_2, \ldots, \hat{y}_{n+1}$ unfolding y. To obtain $\hat{\mathfrak{G}}'$, we extend premise $\hat{\mathfrak{G}}$ of $\mathscr{D}_{n+1}$ by rules

$$\Box^{\bullet},4^{\bullet}\,\frac{\mathcal{R}, \hat{x}_1 R \hat{y}_j, \Gamma, \hat{x}_1{:}\Box A, \hat{y}_j{:}\Box A, \hat{y}_j{:}A \Longrightarrow \Delta}{\mathcal{R}, \hat{x}_1 R \hat{y}_j, \Gamma, \hat{x}_1{:}\Box A \Longrightarrow \Delta}$$

for each $j = 2..n{+}1$. Since $\hat{x}_1 R_{\hat{\mathfrak{G}}} \hat{y}_1$ need not generally be the case, for label $\hat{y}_1$, as well as for all other unfoldings of y, (U1) cannot be guaranteed, necessitating their removal from the unfolding relation, i.e.,

$$\mathsf{U}' := \mathsf{U} \setminus \Big\{ (y, \hat{u}) \mid \hat{u} \notin \{\hat{y}_2, \ldots, \hat{y}_{n+1}\} \Big\}.$$

For each singleton cluster of $\mathfrak{G}'$, (U5) for this U' follows from (U5) for the same cluster of $\mathfrak{G}$ for U. All non-singleton clusters of $\mathfrak{G}'$ other than C have $n{+}1$ unfolded copies by (U6) for the $(n{+}1)$-unfolding of $\mathfrak{G}$, and any n of them satisfy (U6) requisite for an n-unfolding of $\mathfrak{G}'$. Finally, for cluster C of $\mathfrak{G}'$, label sets $\hat{C}_2, \ldots, \hat{C}_{n+1}$ serve as the requisite n unfolded copies, in particular, ${}_{\hat{\mathfrak{G}}'}\hat{y}_j \sim {}_{\mathfrak{G}'}y$ for $j = 2..n{+}1$.

4) The case when $\mathfrak{G}$,x:$\Diamond A^{\circ}$ and y:A° and y:$\Diamond A^{\circ}$ were added for some label y such that $x R_{\mathfrak{G}} y$ is analogous, with $\Diamond^{\circ}$ and 4° used instead of $\Box^{\bullet}$ and $4^{\bullet}$ respectively.

5) Suppose $\mathfrak{G}$,x:$A \vee B^{\bullet}$ and x:$A^{\bullet}$ was added to obtain $\mathfrak{G}'$ and x:$B^{\bullet}$ was added to obtain $\mathfrak{G}''$. The sequence of rules depends on the size of the cluster that x belongs to in $\mathfrak{G}$:

a) If $\{x\}$ is a singleton cluster, use $N = n$ and set $\mathsf{U}' := \mathsf{U}$. By (U5) for $\mathfrak{G}$, there is a unique $\hat{x}$ such that $x \mathsf{U} \hat{x}$ and $\hat{x}$ is allowed in $\hat{\mathfrak{G}}$. By (U1) for $\mathfrak{G}$, we have $\hat{\mathfrak{G}}$,$\hat{x}$:$A \vee B^{\bullet}$ for this $\hat{x}$. Again (U1) is the only problem to be fixed, which for this rule can be done in two ways: by either adding $A^{\bullet}$ towards an n-unfolding of $\mathfrak{G}'$ or adding $B^{\bullet}$ towards an n-unfolding of $\mathfrak{G}''$. We extend premise $\hat{\mathfrak{G}}$ of $\mathscr{D}_n$ by one rule

$$\vee^{\bullet}_{\mathsf{k}}\,\frac{\mathcal{R}, \Gamma, \hat{x}{:}A \vee B, \hat{x}{:}A \Longrightarrow \Delta \quad \mathcal{R}, \Gamma, \hat{x}{:}A \vee B, \hat{x}{:}B \Longrightarrow \Delta}{\mathcal{R}, \Gamma, \hat{x}{:}A \vee B \Longrightarrow \Delta}$$

for the unique label $\hat{x}$. The resulting left premise $\hat{\mathfrak{G}}'$ and right premise $\hat{\mathfrak{G}}''$ can easily be seen to be n-unfoldings of $\mathfrak{G}'$ and $\mathfrak{G}''$ respectively, in particular,

$${}_{\hat{\mathfrak{G}}'}\hat{x} \sim {}_{\mathfrak{G}'}x \qquad \text{and} \qquad {}_{\hat{\mathfrak{G}}''}\hat{x} \sim {}_{\mathfrak{G}''}x.$$

b) If $x \in C$ for some cluster C with $|C| = k \geq 2$, use $N = 2n{-}1$. By (U6) for $\mathfrak{G}$, sequent $\hat{\mathfrak{G}}$ contains $2n{-}1$ unfolded copies $\hat{C}_1, \ldots, \hat{C}_{2n-1}$ of all labels from cluster C. In particular, there are $\hat{x}_j \in \hat{C}_j \subseteq \ell(\hat{\mathfrak{G}})$ for $j = 1..2n{-}1$ such that $x \mathsf{U} \hat{x}_j$ for $j = 1..2n{-}1$. Since $\mathscr{D}_{2n-1}$ is allowed for x, all labels $\hat{x}_j$ for $j = 1..2n{-}1$ are allowed in $\hat{\mathfrak{G}}$. By (U1) for $\mathfrak{G}$, we have $\hat{\mathfrak{G}}$,$\hat{x}_j$:$A \vee B^{\bullet}$ for $j = 1..2n{-}1$. We will not be able to fix (U1) for all $2n{-}1$ labels $\hat{x}_j$, but only for (at least) n of them. We set $\mathfrak{T}_0 := \{\hat{\mathfrak{G}}\}$ to be the set of $2^0 = 1$ premises of the current derivation. Note that $\hat{\mathfrak{G}}$ is the special case (with $l = 0$) of sequents of the form

$$\mathcal{R}, \Gamma, \hat{x}_1{:}D_1, \ldots, \hat{x}_l{:}D_l \Longrightarrow \Delta \tag{8}$$

where $D_i \in \{A, B\}$ for each $i = 1..l$ and $\hat{x}_j$:$A \vee B \in \Gamma$ for all $j = 1..2n{-}1$. Starting with $k = 0$ and repeating the process until $k = 2n{-}2$, we extend each of the 2^k derivation premises of form (8) for $l = k$ in $\mathfrak{T}_k$ as follows

$$\vee^{\bullet}_{\mathsf{k}}\,\frac{\mathcal{R}, \Gamma, , \hat{x}_1{:}D_1, \ldots, \hat{x}_k{:}D_k, \hat{x}_{k+1}{:}A \Longrightarrow \Delta \quad \mathcal{R}, \Gamma, , \hat{x}_1{:}D_1, \ldots, \hat{x}_k{:}D_k, \hat{x}_{k+1}{:}B \Longrightarrow \Delta}{\mathcal{R}, \Gamma, \hat{x}_1{:}D_1, \ldots, \hat{x}_k{:}D_k \Longrightarrow \Delta}$$

thus, obtaining 2^{k+1} derivation premises of form (8) for $l = k{+}1$ that form the set $\mathfrak{T}_{k+1}$. In the end, one premise $\hat{\mathfrak{G}}$ spawns the set $\mathfrak{T}_{2n-1}$ of 2^{2n-1} premises of form (8) for $l = 2n{-}1$. It remains to show how, for each of the premises from $\mathfrak{T}_{2n-1}$, to restrict relation U in a way that produces an n-unfolding of

either $\mathfrak{G}'$ or $\mathfrak{G}''$. Consider any $\mathfrak{H} \in \mathfrak{T}_{2n-1}$. It has the form

$$\mathcal{R}, \Gamma, \hat{x}_1{:}D_1, \ldots, \hat{x}_{2n-1}{:}D_{2n-1} \Longrightarrow \Delta$$

Since all $2n{-}1$ formulas D_j are repetitions of only two distinct formulas A and B, by the pigeonhole principle, one of these two formulas must be repeated at least n times. In other words there is $E \in \{A, B\}$ such that there are n indices $1 \leq i_1 < \cdots < i_n \leq 2n{-}1$ with $D_{i_j} = E$ for all $j = 1..n$. In this case, for this premise $\mathfrak{H}$, we set

$$\mathsf{U}_{\mathfrak{H}} := \mathsf{U} \setminus \Big\{(x, \hat{u}) \mid \hat{u} \notin \{\hat{x}_{i_1}, \ldots, \hat{x}_{i_n}\}\Big\}.$$

Let $\mathfrak{G}^{\mathfrak{H}}$ denote $\mathfrak{G}'$ if $E = A$ or $\mathfrak{G}''$ if $E = B$. For each singleton cluster of $\mathfrak{G}^{\mathfrak{H}}$, (U5) for $\mathsf{U}_{\mathfrak{H}}$ follows from (U5) for the same cluster of $\mathfrak{G}$ for U. All non-singleton clusters of $\mathfrak{G}^{\mathfrak{H}}$ other than C have $2n{-}1$ unfolded copies by (U6) for the $(2n{-}1)$-unfolding of $\mathfrak{G}$, and any n of them satisfy (U6) requisite for an n-unfolding of $\mathfrak{G}^{\mathfrak{H}}$. Finally, for cluster C of $\mathfrak{G}^{\mathfrak{H}}$, label sets $\hat{C}_{i_1}, \ldots, \hat{C}_{i_n}$ serve as the requisite n unfolded copies of C in $\mathfrak{G}^{\mathfrak{H}}$, in particular, ${}_{\mathfrak{H}}\hat{x}_{i_j} \sim {}_{\mathfrak{G}^{\mathfrak{H}}}x$ for $j = 1..n$. Thus, all 2^{2n-1} premises spawned by $\hat{\mathfrak{G}}$ are n-unfoldings of either $\mathfrak{G}'$ or $\mathfrak{G}''$.

6) The case when $\mathfrak{G}, x{:}A \wedge B^{\circ}$ and $x{:}A^{\circ}$ was added to obtain $\mathfrak{G}'$ and $x{:}B^{\circ}$ was added to obtain $\mathfrak{G}''$ is analogous, with $\wedge^{\circ}_{\mathsf{k}}$ used instead of $\vee^{\bullet}_{\mathsf{k}}$.
7) The case when $\mathfrak{G}, x{:}A \supset B^{\bullet}$ and $x{:}A^{\circ}$ was added to obtain $\mathfrak{G}'$ and $x{:}B^{\bullet}$ was added to obtain $\mathfrak{G}''$ is also analogous, with $\supset^{\bullet}_{\mathsf{k}}$ used instead of $\vee^{\bullet}_{\mathsf{k}}$.

For each of the steps, we have extended the given tidy derivation $\mathcal{D}_N$ to a derivation $\mathcal{D}'_n$ in such a way that no labels, forbidden labels, or relational atoms were modified. Since all added rules touch only allowed labels, the resulting derivation is tidy. Thus, derivation $\mathcal{D}'_n$ of $\mathfrak{G}_0(F)$ is an n-unfolding of $\boldsymbol{\mathfrak{G}}'$. Moreover, whenever $\mathcal{D}_N$ was allowed for some label z, so is $\mathcal{D}'_n$ because no label has changed its forbidden/allowed status in the premises. □

Lemma A.9. *(i) If $\boldsymbol{\mathfrak{G}}'_0 \approx\!\!\Rightarrow^*_{\mathsf{s}} \mathfrak{T}$, then $\mathfrak{T}$ is F-unfoldable.*
(ii) For each i, if $\boldsymbol{\mathfrak{G}}_i$ is F-unfoldable and $\boldsymbol{\mathfrak{G}}'_{i+1} \approx\!\!\Rightarrow^*_{\mathsf{s}} \mathfrak{T}$ for some semi-saturated $\mathfrak{T}$, then $\mathfrak{T}$ is F-unfoldable.

Proof.

(i) If $\boldsymbol{\mathfrak{G}}'_0 \approx\!\!\Rightarrow^*_{\mathsf{s}} \mathfrak{T}$, then $\boldsymbol{\mathfrak{G}}'_0 = \mathfrak{T}_0 \approx\!\!\Rightarrow_{\mathsf{s}} \mathfrak{T}_1 \approx\!\!\Rightarrow_{\mathsf{s}} \cdots \approx\!\!\Rightarrow_{\mathsf{s}} \mathfrak{T}_k = \mathfrak{T}$ for stable sets $\mathfrak{T}_0, \ldots, \mathfrak{T}_k$ for some $k \geq 0$. Note all sequents $\mathfrak{G}$ appearing in these sets are proper with $\ell(\mathfrak{G}) = \{r\}$ because semi-saturation steps neither add labels to the only sequent $\mathfrak{G}_0(F)$ of $\mathfrak{T}_0$ with the single label r nor create clusters.
To show the base of induction, i.e., that $\mathfrak{T}_0 = \{\mathfrak{G}_0(F)\}$ is F-unfoldable, it is sufficient to consider a tidy derivation $\mathcal{D}^0$ of the restricted version of $\mathfrak{G}_0(F)$ where label r is allowed and no rules are applied (note that the only label r in each of the sequents forms the only layer, which is, therefore, topmost and not inner). As discussed earlier, this one derivation is an n-unfolding of $\mathfrak{T}_0$ for each $n \in \mathbb{N}$ via the binary relation $\mathsf{U} := \{(r, r)\}$ that is an n-unfolding relation for each $n \in \mathbb{N}$. Note that $\mathcal{D}^0$ is clearly allowed for r.
Step of induction. Let derivations $\mathcal{D}^i_n$ of $\mathfrak{G}_0(F)$ for $n \in \mathbb{N}$ be n-unfoldings of $\mathfrak{T}_i$ that are allowed for r. For the semi-saturation step $\mathfrak{T}_i \approx\!\!\Rightarrow_{\mathsf{s}} \mathfrak{T}_{i+1}$ we can apply Lemma A.8 for $x = r$ to obtain derivations $\mathcal{D}^{i+1}_n$ of $\mathfrak{G}_0(F)$ for $n \in \mathbb{N}$ that are n-unfoldings of $\mathfrak{T}_{i+1}$ and are allowed for r.[5]

(ii) Recall that according to step 5) of the algorithm, Def. 7.14, and Constr. 7.10, $\boldsymbol{\mathfrak{G}}'_{i+1} = \big(\boldsymbol{\mathfrak{G}}_i \setminus \{\mathfrak{G}\}\big) \cup \{\mathfrak{G}{\uparrow}\}$ for some fully saturated non-axiomatic $\mathfrak{G} \in \boldsymbol{\mathfrak{G}}_i$, where

$$\mathfrak{G}{\uparrow} = \mathfrak{G} + \mathfrak{G}{\uparrow}^{x_1:F_1} + \cdots + \mathfrak{G}{\uparrow}^{x_h:F_h}$$

and $x_1{:}F^{\circ}_1, \ldots, x_h{:}F^{\circ}_h$ are all the unhappy formulas in some of the topmost layers of $\mathfrak{G}$ and each F°_j is either a $\supset$- or a $\Box$-formula. Also note that since $\boldsymbol{\mathfrak{G}}_i$ is semi-saturated, we only need to semi-saturate subset $\{\mathfrak{G}{\uparrow}\} \subseteq \boldsymbol{\mathfrak{G}}'_{i+1}$ to obtain $\mathfrak{T}$.
It might seem natural to split (ii) into several finer-grained statements stating that unfolding can be preserved by smaller steps, including many of the steps already covered by Lemma A.8 and adding to them new steps corresponding to Construction 7.9 that create one new layer based on one of $x_j{:}F^{\circ}_j$. Unfortunately, these intermediate stages cannot be formulated as unfoldings due to the sequent ceasing to be stable because as soon as the first of $x_j{:}F^{\circ}_j$ in a layer is made happy, this layer turns into an inner one, so the sequent does not become stable again until all remaining $x_j{:}F^{\circ}_j$ in this layer are happy.
At the same time, applying all requisite instances of $\supset^{\circ}_{\star}$ and $\Box^{\circ}_{\star}$ to obtain an n-unfolding of $\boldsymbol{\mathfrak{G}}'_{i+1}$ and then proceeding as in Case (i), repeatedly applying Lemma A.8 is met with another technical problem. It would violate the tidiness conditions (see Def. 9.7): each $\supset^{\circ}_{\star}$- and $\Box^{\circ}_{\star}$-instance creates a new layer and renders the layers created by the previous $\supset^{\circ}_{\star}$- or $\Box^{\circ}_{\star}$-instances forbidden, so that the rules needed to unfold the semi-saturation steps for those layers cannot be applied anymore. For this reason, we have to apply the necessary rules for unfolding the semi-saturation after each $\supset^{\circ}_{\star}$- or $\Box^{\circ}_{\star}$-instance.
Thus, we will still follow individual smaller steps in a specific order maintaining conditions (U1)–(U7) for all $n \in \mathbb{N}$ after each, but are forced to lump them together into (ii), at the end of which the result can once again be called an unfolding.
Let $L_1, \ldots, L_h$ be the layers of $\mathfrak{G}{\uparrow}$ that are introduced by $\mathfrak{G}{\uparrow}^{x_1:F_1}, \ldots, \mathfrak{G}{\uparrow}^{x_h:F_h}$ respectively. Since $\mathfrak{T}$ is semi-saturated and each rewriting step of $\approx\!\!\Rightarrow_{\mathsf{s}}$ is local to a layer, we can therefore without loss of generality assume that the rewrite sequence $\boldsymbol{\mathfrak{G}}'_{i+1} \approx\!\!\Rightarrow^*_{\mathsf{s}} \mathfrak{T}$ is adding formulas layerwise, i.e., all formulas that are added to layer L_j

[5] Since all sequents involved in sets $\mathfrak{T}_i$ are proper, a closer observation of Lemma A.8 shows that one derivation can serve as n-unfoldings of $\mathfrak{T}_i$ for all $n \in \mathbb{N}$.

are added before adding formulas to layer $L_{j'}$, whenever $j < j'$.

For each step semi-saturating the next layer L_j we preserve (U1)–(U7) for all $n \in \mathbb{N}$ the same way as in Case (i) using the fact that L_j is created of allowed labels that are not made forbidden while unfolding semi-saturation steps. However, formally we cannot rely on Lemma A.8 because of the lack of stability mentioned before. Instead, we simply repeat the arguments from the proof of Lemma A.8 noting that the only part of stability used in its proof was Rtr-structural completeness, which does hold upon creating layer L_j.

The only remaining steps to cover are the creation of layers L_j. This is done by simply applying the $\supset^\circ_\star$- or $\Box^\circ_\star$-rule to $x_j{:}F^\circ_j$. However, in order to preserve the (U5) and (U6) conditions, we have to start with an $(2n+1)$-unfolding of $\mathfrak{G}_i$ in order to obtain an n-unfolding of $\mathfrak{T}$. For every lifted cluster that does not contain x_j the additional $n+1$ copies in the unfolding are discarded. But when x_j is in a cluster, then L_j contains 2 copies of that cluster (see Figure 5), whose n repetitions are given to us by the copies $1..n$ and $n+1..2n+1$ of our lifting. In the nth copy, only the lifting of x_j is part of the unfolding relation.

It is easy to check that at the end of this process all the conditions of F-unfoldability are restored. □

The next step is to show that $\rightrightarrows_\Diamond$ preserves the property of being unfoldable. For this, we need to repeat parts of the proof. To simplify the process of repeating, we use the notion of embedding.

Definition A.10 (Embedding)**.** Let $\mathfrak{G}$ and $\mathfrak{H}$ be restricted proper sequents. An ***embedding*** $\mathsf{e}\colon \mathfrak{G} \to \mathfrak{H}$ is an injective function $\mathsf{e}\colon \ell(\mathfrak{G}) \to \ell(\mathfrak{H})$ obeying the following conditions for all $x, y \in \ell(\mathfrak{G})$:

(e1) $x \sim \mathsf{e}(x)$;
(e2) $xR_{\mathfrak{G}}y$ iff $\mathsf{e}(x)R_{\mathfrak{H}}\mathsf{e}(y)$;
(e3) $L_x \leq_{\mathfrak{G}} L_y$ iff $L_{\mathsf{e}(x)} \leq_{\mathfrak{H}} L_{\mathsf{e}(y)}$.
(e4) x has no past in $\mathfrak{G}$ iff $\mathsf{e}(x)$ has no past in $\mathfrak{H}$;
(e5) $x \in f(\mathfrak{G})$ iff $\mathsf{e}(x) \in f(\mathfrak{H})$;

Proposition A.11. *Embeddings are closed under composition and preserve unfoldings.*

Proof. First, we prove that a composition of two embeddings is an embedding. Let $\mathfrak{G}$, $\mathfrak{H}$, and $\mathfrak{J}$ be restricted proper sequents. Let $\mathsf{e}\colon \mathfrak{G} \to \mathfrak{H}$ and $\mathsf{e}'\colon \mathfrak{H} \to \mathfrak{J}$ be embeddings from $\mathfrak{G}$ to $\mathfrak{H}$ and from $\mathfrak{H}$ to $\mathfrak{J}$ respectively. We need to show that $\mathsf{e}'\mathsf{e}$ is an embedding from $\mathfrak{G}$ to $\mathfrak{J}$, where $(e'e)(x) := \mathsf{e}'\big(\mathsf{e}(x)\big)$ for any $x \in \ell(\mathfrak{G})$. Since $\mathsf{e}\colon \ell(\mathfrak{G}) \to \ell(\mathfrak{H})$ and $\mathsf{e}'\colon \ell(\mathfrak{H}) \to \ell(\mathfrak{J})$ are injective functions, it is immediate that $\mathsf{e}'\mathsf{e}\colon \ell(\mathfrak{G}) \to \ell(\mathfrak{J})$ is an injective function. We have for all $x, y \in \ell(\mathfrak{G})$:

(e1) ${}_{\mathfrak{G}}x \sim {}_{\mathfrak{H}}\mathsf{e}(x) \sim {}_{\mathfrak{J}}\mathsf{e}'\big(\mathsf{e}(x)\big)$.
(e2) $xR_{\mathfrak{G}}y$ iff $\mathsf{e}(x)R_{\mathfrak{H}}\mathsf{e}(y)$ iff $\mathsf{e}'\big(\mathsf{e}(x)\big)R_{\mathfrak{J}}\mathsf{e}'\big(\mathsf{e}(y)\big)$.
(e3) $L_x \leq_{\mathfrak{G}} L_y$ iff $L_{\mathsf{e}(x)} \leq_{\mathfrak{H}} L_{\mathsf{e}(y)}$ iff $L_{\mathsf{e}'(\mathsf{e}(x))} \leq_{\mathfrak{J}} L_{\mathsf{e}'(\mathsf{e}(y))}$.
(e4) x has no past in $\mathfrak{G}$ iff $\mathsf{e}(x)$ has no past in $\mathfrak{H}$ iff $\mathsf{e}'\big(\mathsf{e}(x)\big)$ has no past in $\mathfrak{J}$.
(e5) $x \in f(\mathfrak{G})$ iff $\mathsf{e}(x) \in f(\mathfrak{H})$ iff $\mathsf{e}'\big(\mathsf{e}(x)\big) \in f(\mathfrak{J})$.

Secondly, let $\mathsf{U} \subseteq \ell(\mathfrak{G}) \times \ell(\hat{\mathfrak{G}})$ be an n-unfolding relation witnessing that $\hat{\mathfrak{G}}$ is an n-unfolding of $\mathfrak{G}$. We show that binary relation $\mathsf{e}\mathsf{U} \subseteq \ell(\mathfrak{G}) \times \ell(\hat{\mathfrak{H}})$ witnesses that $\hat{\mathfrak{H}}$ is an n-unfolding of $\mathfrak{G}$, where

$$x(\mathsf{e}\mathsf{U})\hat{y} \qquad \Longleftrightarrow \qquad (\exists \hat{x})\Big(x\mathsf{U}\hat{x} \text{ and } \hat{y} = \mathsf{e}(\hat{x})\Big).$$

Let $x\,(\mathsf{e}\mathsf{U})\,\mathsf{e}(\hat{x})$ because $x\mathsf{U}\hat{x}$ and $x'\,(\mathsf{e}\mathsf{U})\,\mathsf{e}(\hat{x}')$ because $x'\mathsf{U}\hat{x}'$.

(U1) ${}_{\mathfrak{G}}x \sim {}_{\hat{\mathfrak{G}}}\hat{x}$ by (U1) for U and ${}_{\hat{\mathfrak{G}}}\hat{x} \sim {}_{\hat{\mathfrak{H}}}\mathsf{e}(\hat{x})$ by (e1) for e.
(U2) Let x and x' be from different clusters. Then $xR_{\mathfrak{G}}x'$ iff $\hat{x}R_{\hat{\mathfrak{G}}}\hat{x}'$ by (U2) for U and the latter iff $\mathsf{e}(\hat{x})R_{\hat{\mathfrak{H}}}\mathsf{e}(\hat{x}')$ by (e2) for e.
(U3) $L_x \leq_{\mathfrak{G}} L_{x'}$ iff $L_{\hat{x}} \leq_{\hat{\mathfrak{G}}} L_{\hat{x}'}$ by (U3) for U, and the latter iff $L_{\mathsf{e}(\hat{x})} \leq_{\hat{\mathfrak{H}}} L_{\mathsf{e}(\hat{x}')}$ by (e3) for e.
(U4) x has no past in $\mathfrak{G}$ iff $\hat{x}$ has no past in $\hat{\mathfrak{G}}$ by (U4) for U, and the latter iff $\mathsf{e}(\hat{x})$ has no past in $\hat{\mathfrak{H}}$ by (e4) for e.
(U5) If x forms a singleton cluster in $\mathfrak{G}$, then $\hat{x}$ is the only label in $\hat{\mathfrak{G}}$ such that $x\mathsf{U}\hat{x}$ by (U5) for U. Hence, $\mathsf{e}(\hat{x})$ is the only label in $\hat{\mathfrak{H}}$ such that $x\,(\mathsf{e}\mathsf{U})\,\mathsf{e}(\hat{x})$.
(U6) If C is a non-singleton cluster in $\mathfrak{G}$ with $|C| = k \geq 2$, then by (U6) for U there are $x_i \in \ell(\mathfrak{G})$ and $\hat{x}_{i,j} \in \ell(\hat{\mathfrak{G}})$ for $i = 1..k$ and $j = 1..n$ such that $C = \{x_1, \dots, x_k\}$, and $x_i\mathsf{U}\hat{x}_{i,j}$ for $i = 1..k$, $j = 1..n$, and in $\hat{\mathfrak{G}}$

$$\hat{x}_{1,1}R\cdots R\hat{x}_{k,1}R\hat{x}_{1,2}R\cdots R\hat{x}_{k,2}R\cdots R\hat{x}_{1,n}R\cdots R\hat{x}_{k,n}.$$

In this case, $x_i\,(\mathsf{e}\mathsf{U})\,\mathsf{e}(\hat{x}_{i,j})$ for $i = 1..k$, $j = 1..n$ and in $\hat{\mathfrak{H}}$ by (e2) for e

$$\mathsf{e}(\hat{x}_{1,1})\,R\cdots R\,\mathsf{e}(\hat{x}_{k,1})\,R\,\mathsf{e}(\hat{x}_{1,2})\,R\cdots R\,\mathsf{e}(\hat{x}_{k,2})\,R\cdots R\,\mathsf{e}(\hat{x}_{1,n})\,R\cdots R\,\mathsf{e}(\hat{x}_{k,n}).$$

Thus, $\mathsf{e}(\hat{x}_{i,j}) \in \ell(\hat{\mathfrak{H}})$ for $i = 1..k$, $j = 1..n$ provide the requisite n unfoldings for the labels of C in $\hat{\mathfrak{H}}$.
(U7) If $\mathsf{e}(\hat{x}) = \mathsf{e}(\hat{x}')$, then $\hat{x} = \hat{x}'$ by the injectivity of e, and $x = x'$ by (U7) for U. □

Lemma A.12 (Embedding Lemma)**.** *Assume we have a tidy derivation $\mathcal{D}$ with a restricted proper endsequent $\mathfrak{G}$ and premises $\mathfrak{G}_1, \dots, \mathfrak{G}_n$. If there is an embedding $\mathsf{e}\colon \mathfrak{G} \to \mathfrak{H}$ into a restricted proper sequent $\mathfrak{H}$, then there is a tidy derivation $\mathcal{D}'$ with conclusion $\mathfrak{H}$ and premises $\mathfrak{H}_1, \dots, \mathfrak{H}_n$ such that there are embeddings $\mathsf{e}_i\colon \mathfrak{G}_i \to \mathfrak{H}_i$ for $i = 1..n$.*

Proof. Let us first consider the case where $\mathcal{D}$ consists of only one tidy inference rule instance r. Let $\mathfrak{G}_i$ be a premise of r. Then, by Proposition 9.4, we have $\mathfrak{G}_i = \mathfrak{G} + \mathfrak{G}'_i$ for some $\mathfrak{G}'_i$. If r does not add new labels, then $\ell(\mathfrak{G}_i) = \ell(\mathfrak{G})$, and we have $\ell(\mathfrak{H}_i) = \ell(\mathfrak{H})$ and $\mathsf{e}_i = \mathsf{e}$. This gives us a correct rule application satisfying the lemma. Otherwise, if r adds new labels, then r is one of $\Diamond^\bullet_\star$, $\supset^\circ_\star$, or $\Box^\circ_\star$, there is only one premise, and the new labels only occur in $\mathfrak{G}'_1$.

- r is $\Diamond^\bullet_\star$: The rule applies to $x{:}\Diamond A^\bullet$ in $\mathfrak{G}$, creating a new label y, and we can apply the same rule to $\mathsf{e}(x){:}\Diamond A^\bullet$ in $\mathfrak{H}$, creating a new label y'. We can define $\mathsf{e}_1(y) = y'$, and for all other $u \in \ell(\mathfrak{G}_1)$ we define $\mathsf{e}_1(u) = \mathsf{e}(u)$.

- r is $\supset^\circ_\star$ or $\Box^\circ_\star$: The rule applies to $x{:}F^\circ$ in $\mathfrak{G}$, creating a new layer L' whose labels all occur only in $\mathfrak{G}'_1 = \mathfrak{G}{\uparrow^{x:F}}$. We can apply the same rule to $\mathsf{e}(x){:}F^\circ$ in $\mathfrak{H}$, creating a new layer L'' and adding $\mathfrak{H}'_1 = \mathfrak{H}{\uparrow^{\mathsf{e}(x):F}}$ to $\mathfrak{H}$. In general, L'' can contain more labels than L'. But e defines an embedding of L_x into $L_{\mathsf{e}(x)}$, which canonically defines a mapping $\mathsf{e}'\colon L' \to L''$ as follows: for $y' \in L'$, we define $\mathsf{e}'(y') = y'' \in L''$ iff either there is a $v \in L_x$ with $v \leq_{\mathfrak{G}_1} y'$ and $\mathsf{e}(v) \leq_{\mathfrak{H}_1} y''$, or y' has no past in $\mathfrak{G}_1$ and y'' has no past in $\mathfrak{H}_1$. We can then define our embedding $\mathsf{e}_1\colon \mathfrak{G}_1 \to \mathfrak{H}_1$ as $\mathsf{e}_1(u) = \mathsf{e}'(u)$ if $u \in L'$ and $\mathsf{e}_1(u) = \mathsf{e}(u)$ otherwise.

We can now prove the lemma for arbitrary $\mathcal{D}$ with a straightforward induction on the height of $\mathcal{D}$. □

Lemma A.13. *Whenever for a semi-saturated sequent $\mathfrak{G}$, the algorithm in step 1) applies Option 1) of Def. 7.5 by substituting label x for label y in $\mathfrak{G}$ and applying the transitive closure of R to obtain $\mathfrak{G}'$ such that $\mathfrak{G} \rightsquigarrow_\Diamond \mathfrak{G}'$, label y forms a singleton cluster in $\mathfrak{G}$.*

Proof. Since the application of Option 1) means that y has no past in $\mathfrak{G}$, it follows that the algorithm created y as a singleton cluster (unless $y = r$, in which case it was present from the very beginning, but was a singleton cluster at first). It is not hard to check that every time the algorithm creates a new nonsingleton cluster,either z have a past for all label z in the cluster or all labelled formulas $z{:}\Diamond B^\bullet$ become happy for all label z in the cluster. Since $y{:}\Diamond A^\bullet$ has to be unhappy and have no past to apply Option 1), label y is not part of a nontrivial cluster. □

Definition A.14. For a label x of a sequent $\mathfrak{G}$ we define

$$\overrightarrow{x} := \{z \mid xR_{\mathfrak{G}}z\}.$$

Lemma A.15. *If $\mathfrak{S} \rightrightarrows_\Diamond \hat{\mathfrak{S}}$ and $\mathfrak{S}$ is semi-saturated and F-unfoldable, then $\hat{\mathfrak{S}}$ is F-unfoldable.*

Proof. Recall that according to Def. 7.5, $\hat{\mathfrak{S}}$ here is a semi-saturation of a set $(\mathfrak{S}\setminus\{\mathfrak{G}\})\cup\{\mathfrak{G}'\}$ for some $\mathfrak{G} \in \mathfrak{S}$ such that $\mathfrak{G} \rightsquigarrow_\Diamond \mathfrak{G}'$. Note that $\mathfrak{G}'$ is a stable sequent by Lemma A.5(i)). The semi-saturation is handled as in Lemma A.9 using the fact that $\mathfrak{G}$ was semi-saturated and, hence, the only label that semi-saturation is going to touch is a new label, which is created as allowed. Thus, what remains is to explain how to turn N-unfoldings of $\mathfrak{G}$ into n-unfoldings of $\mathfrak{G}'$ for appropriate $N \geq n$. According to Definition 7.5 there are two options:

1) In Option 1, we have $\mathfrak{G}, y{:}\Diamond A^\bullet$ where y forms a singleton cluster by Lemma A.13, and we have some x from a topmost layer with $x \sim y$, $xR_{\mathfrak{G}}y$, and $x \neq y$ such that all $u \in \overrightarrow{x}$ have no past. Then $\mathfrak{G}'$ is obtained by substituting x for y and closing under R-transitivity. Note that the resulting $\mathfrak{G}'$ is semi-saturated, hence, $\hat{\mathfrak{S}} = (\mathfrak{S}\setminus\{\mathfrak{G}\})\cup\{\mathfrak{G}'\}$. Consider any premise $\hat{\mathfrak{G}}$ of $\mathcal{D}_N$ that is an N-unfolding of $\mathfrak{G}$ via U_1. By (U5), there is a unique $\hat{y}$ such that $y\mathsf{U}_1\hat{y}$. The choice of N and the extension of $\mathcal{D}_N$ depend on the size of the cluster that x belongs to in $\mathfrak{G}$:

a) If $\{x\}$ is a singleton cluster, set $N := n$. By (U5), there is a unique $\hat{x}$ such that $x\mathsf{U}_1\hat{x}$. By (U2), $\hat{x}R_{\hat{\mathfrak{G}}}\hat{y}$. Since x has no past by assumption, by (U4) neither has $\hat{x}$. It follows from tree-layeredness of $\hat{\mathfrak{G}}$ that all labels in $\overrightarrow{\hat{x}}$, including $\hat{y}$, have no past. Hence, all labels in $\overrightarrow{\hat{x}}$, with the possible exception of $\hat{x}$ itself, are created by rule $\Diamond^\bullet_\star$. Let $\mathfrak{K}_1$ be the premise of the first (i.e., lowermost) instance of rule $\Diamond^\bullet_\star$ on the branch of $\mathcal{D}_n$ leading to $\hat{\mathfrak{G}}$ that creates a non-trivial child of $\hat{x}$, let us call this child u_1. Since $\mathcal{D}_n$ is tidy, label u_1 is the only allowed label in $\mathfrak{K}_1$. Let $\hat{x}{:}\Diamond C^\bullet$ be the principal formula of this instance of $\Diamond^\bullet_\star$. Then $\mathfrak{K}_1, u_1{:}C^\bullet$ is the only labelled formula in u_1. Since $\mathfrak{K}_1, \hat{x}{:}\Diamond C^\bullet$, also $\hat{\mathfrak{G}}, \hat{x}{:}\Diamond C^\bullet$ higher up the branch, hence, $\mathfrak{G}, x{:}\Diamond C^\bullet$ by (U1). Therefore, $\mathfrak{G}, y{:}\Diamond C^\bullet$ because $x \sim y$ and $\hat{\mathfrak{G}}, \hat{y}{:}\Diamond C^\bullet$ by (U1). Since $\hat{y}$ is in a topmost layer of $\hat{\mathfrak{G}}$, a tidy rule $\Diamond^\bullet_\star$ can be applied to $\hat{\mathfrak{G}}$ with $\hat{y}{:}\Diamond C^\bullet$ as its principal formula. Let the premise of this rule be $\mathfrak{K}_2$ and u_2 be the new label created by this rule, the only allowed label in $\mathfrak{K}_2$. Consider the following function $\mathsf{e}_1\colon \ell(\mathfrak{K}_1) \to \ell(\mathfrak{K}_2)$:

$$\mathsf{e}_1 := \{(\hat{x}, \hat{y}), (u_1, u_2)\}\cup\{(w, w) \mid w \in \ell(\mathfrak{K}_1)\setminus\{\hat{x}, u_1\}\}.$$

It is easy to see that e_1 is an embedding. The tidy derivation subtree rooted at $\mathfrak{K}_1$ can be applied to $\mathfrak{K}_2$ via this embedding resulting in a tidy derivation by Lemma A.12 with all the premises of the extension being n-unfoldings of the same sequents from $\mathfrak{S}$ as the respective premises of $\mathcal{D}_n$ by Prop. A.11. In particular, for premise $\hat{\mathfrak{G}}$ of the subtree, embedding e_1 will be extended to en embedding into some premise $\hat{\mathfrak{G}}_2$ of the extended derivation with $\mathsf{e}_1(\hat{y}) = \hat{y}_2$ such that ${}_{\hat{\mathfrak{G}}_2}\hat{y}_2 \sim {}_{\hat{\mathfrak{G}}}\hat{y}$ (note that ${}_{\mathfrak{K}_2}\hat{y} \sim {}_{\hat{\mathfrak{G}}}\hat{y}$). By Prop. A.11, we can construct an n-unfolding U_2 of $\mathfrak{G}$ into $\hat{\mathfrak{G}}_2$ such that $w\mathsf{U}_1\hat{w}$ iff $w\mathsf{U}_2\hat{w}$ for all labels $\hat{w} \in \ell(\mathfrak{K}_1) \setminus \{\hat{x}, u_1\} = \ell(\hat{\mathfrak{G}}) \setminus \{\hat{x}\}$, including for all non-trivial parents of $\hat{x}$. Further, $x\mathsf{U}_2\hat{y}$ and, whenever $xR_{\mathfrak{G}}v$ and $v\mathsf{U}_2\hat{v}$ we have $\hat{y}R_{\hat{\mathfrak{G}}_2}\hat{v}$. In particular, $y\mathsf{U}_2\hat{y}_2$ for some $\hat{y}_2$ such that $\hat{y}R_{\hat{\mathfrak{G}}_2}\hat{y}_2$ and $\hat{\mathfrak{G}}_2, \hat{y}_2{:}\Diamond C^\bullet$.
We can apply $\Diamond^\bullet_\star$ to obtain $\mathfrak{K}_3$ with a new label u_3, take an embedding

$$\mathsf{e}_2 := \{(\hat{y}, \hat{y}_2), (u_2, u_3)\}\cup\{(w, w) \mid w \in \ell(\mathfrak{K}_2)\setminus\{\hat{y}, u_2\}\}.$$

from $\mathfrak{K}_2$ into $\mathfrak{K}_3$ and repeat the process.
After $n-1$ such repetitions, we will get a premise $\hat{\mathfrak{G}}_{n+1}$ and an n-unfolding U_{n+1} of $\mathfrak{G}$ into $\hat{\mathfrak{G}}_{n+1}$ such that $w\mathsf{U}_1\hat{w}$ iff $w\mathsf{U}_{n+1}\hat{w}$ for all labels $\hat{w} \in \ell(\hat{\mathfrak{G}}) \setminus \{\hat{x}\}$, including for all non-trivial parents of $\hat{x}$. Further, $x\mathsf{U}_{n+1}\hat{y}_n$ and, whenever $xR_{\mathfrak{G}}v$ and $v\mathsf{U}_{n+1}\hat{v}$ we have $\hat{y}_nR_{\hat{\mathfrak{G}}_{n+1}}\hat{v}$. In particular, $y\mathsf{U}_{n+1}\hat{y}_{n+1}$ for some y_{n+1} such that $\hat{y}_nR_{\hat{\mathfrak{G}}_{n+1}}\hat{y}_{n+1}$.
Once again, all other premises of this derivation expanded n times are n-unfoldings of sequents from $\mathfrak{S}$. It remains to turn $\hat{\mathfrak{G}}_{n+1}$ into an n-unfolding of $\mathfrak{G}'$. We do this by modifying U_{n+1} for the newly created cluster $C_x = \{u \neq y \mid xR_{\mathfrak{G}}uR_{\mathfrak{G}}y\}$ in $\mathfrak{G}'$ and keeping

unfoldings of all other labels of $\mathfrak{G}$ as in U_{n+1}. For each label $u \in C_x$ and each $j = 1..n$ there is at least one label $\hat{u}_j$ such that $u\mathsf{U}_j\hat{u}_j$ and $\hat{y}_{j-1}R_{\hat{\mathfrak{G}}_{n+1}}\hat{u}_jR_{\hat{\mathfrak{G}}_{n+1}}\hat{y}_j$. These labels $\hat{u}_1, \ldots, \hat{u}_n$ for each $u \in C_x$ can, thus, be taken as the requisite n unfolding copies of C_x. Indeed, whenever $wR_{\mathfrak{G}}x$ and $w\mathsf{U}_{n+1}\hat{w}$ for some $w \neq x$, as discussed, we have $\hat{w}R_{\hat{\mathfrak{G}}_{n+1}}\hat{x}R_{\hat{\mathfrak{G}}_{n+1}}\hat{u}_j$. On the other hand, whenever $xR_{\mathfrak{G}}v$ and $v\mathsf{U}_{n+1}\hat{v}$ for some $v \notin C_x \cup \{y\}$, as discussed, we have $\hat{u}_jR_{\hat{\mathfrak{G}}_{n+1}}\hat{y}_nR_{\hat{\mathfrak{G}}_{n+1}}\hat{v}$. Note that children of x that are not in C_x in $\mathfrak{G}'$ must be unfolded into labels that are children of all $\hat{u}_j$, which is why we needed to create $n+1$ successful unfoldings rather than n: the first n unfoldings are used to create n unfolded copies of C_x whereas the last $(n+1)$th provides unfoldings for children of x that are not in its cluster.

b) If x belongs to a non-singleton cluster C_x, we use a similar expansion to the case of singleton clusters just discussed. Hence, we will only outline the differences. Firstly, instead of $\mathcal{D}_n$ we start with $\mathcal{D}_N$ for $N = \max(n, 2)$. The fact that $N \geq n$ guarantees n-unfoldings for all premises other than $\mathfrak{G}$ and for all clusters of $\mathfrak{G}$ other than the newly created C'_x that contains C_x and all labels in between C_x and y excluding y.

Among the $N \geq 2$ unfolding copies of x in $\hat{\mathfrak{G}}$ existing by (U6), use the penultimate, $(N-1)$th unfolding copy as $\hat{x}$ and the unique unfolding of y as $\hat{y}$. For these $\hat{x}$ and $\hat{y}$ perform the expansion as in the case for a singleton cluster. The only difference to that case is how the final unfolding U_{n+1} is to be modified to account for the newly created cluster C'_x. For all labels outside of C_x, the construction remains the same. For each label $u \in C_x$, its last, Nth unfolding copy from U_1 is among labels repeated in the segment (note that the same need not be true about the $(N-1)$th unfolding copy, which is why we had to start one unfolding copy earlier). Hence, this Nth copy is now repeated $n+1$ times, and the first n of these are taken as new unfoldings of u.

2) In Option 2, we have $\mathfrak{G}, y{:}\Diamond A^\bullet$ and create a child z of y such that $\mathfrak{G}', z{:}A^\bullet$. This case is similar to the steps taken for the semi-saturation in Lemma A.8, which is also used to perform the semi-saturation after $\mathfrak{G}$ is replaced by $\mathfrak{G}'$, as discussed earlier. We provide an abbreviated description how to expand $\hat{\mathfrak{G}}$ that is an N-unfolding of $\mathfrak{G}$ into an n-unfolding of $\mathfrak{G}'$. Once again, the exact procedure depends on the size of the cluster that y belongs to in $\mathfrak{G}$:

a) If $\{y\}$ is a singleton cluster, use $N = n$. By (U5), there is a unique $\hat{y}$ such that $y\mathsf{U}\hat{y}$. By (U1), $\hat{\mathfrak{G}}, \hat{y}{:}\Diamond A^\bullet$ for this $\hat{y}$. It is sufficient to apply a tidy instance of $\Diamond^\bullet_\star$ to this $\hat{y}{:}\Diamond A^\bullet$ creating a fresh label $\hat{z}$, the only allowed label in the premise and set $\mathsf{U}' := \mathsf{U} \cup \{(z, \hat{z})\}$. It is easy to check that all unfolding conditions are satisfied for the resulting premise to be an n-unfolding of $\mathfrak{G}'$.

b) If $y \in C$ for some cluster C with $|C| = k \geq 2$, use $N = n+1$. By (U6), sequent $\hat{\mathfrak{G}}$ contains $n+1$ unfolded copies $\hat{y}_j$ for $j = 1..n+1$ in $\hat{\mathfrak{G}}$ such that $y = \mathsf{U}(\hat{y}_j)$ for $j = 1..n+1$. By (U1), $\hat{\mathfrak{G}}, \hat{y}_{n+1}{:}\Diamond A^\bullet$. Here we apply a tidy instance of $\Diamond^\bullet_\star$ to this $\hat{y}_{n+1}{:}\Diamond A^\bullet$ creating a fresh label $\hat{z}$, the only allowed label in the premise and modify U as follows:

- $\hat{z}$ is made the unique unfolding of z;
- unfoldings of all labels from $\ell(\mathfrak{G})\backslash C$ remain intact;
- for the unfoldings of C we only keep the first n unfoldings guaranteed by (U6) for U removing all others.

The removal of the extra unfoldings in the last item is necessary to make sure that $\hat{z}$ is a child of all unfoldings of cluster C. Once again, it is easy to check that all unfolding conditions are satisfied for the resulting premise to be an n-unfolding of $\mathfrak{G}'$. □

Lemma A.16. *If $\mathfrak{S} \rightrightarrows_{\circlearrowleft} \hat{\mathfrak{S}}$ and $\mathfrak{S}$ is saturated and F-unfoldable, then $\hat{\mathfrak{S}}$ is F-unfoldable.*

Proof. Assume that $\mathfrak{G} \rightsquigarrow_{\circlearrowleft} \mathfrak{G}'$ and $\mathfrak{G} \in \mathfrak{S}$ saturated and $\mathfrak{S}$ is F-unfoldable, and $\mathfrak{S}' = \mathfrak{S}\backslash\{\mathfrak{G}\}\cup\{\mathfrak{G}'\}$. We want to construct an n-unfolding for $\mathfrak{S}'$. $\mathfrak{G}'$ is obtained from $\mathfrak{G}$ by closing an unhappy triangle loop. Let $L_1, L_2, L', C_1, C_2, p_1, s, t$ as in Definitions 7.16 and 7.17. We also have a p_2 with $p_2 \in C_2$ and $p_2 \sim p_1$, that we will use in the proof.

If s and t are in the same cluster, then every n-unfolding of $\mathfrak{S}$ is also an n-unfolding of $\mathfrak{S}'$. Now, assume that s and t are not in the same cluster.

As in case 1b) of the previous lemma, we start from an N-unfolding $\mathcal{D}_N$ of $\mathfrak{S}$, where $N = \max(n, 2)$ and let $\hat{\mathfrak{G}}$ be a premise of $\mathcal{D}_N$ that is an unfolding of $\mathfrak{G}$. Let U be the corresponding unfolding relation and let $\hat{L}_1, \hat{L}_2, \hat{L}'$ be the unfoldings of L_1, L_2, L', respectively.

We now let $\mathcal{T}$ be the subtree of $\mathcal{D}_N$ rooted at the sequent where $\hat{L}'$ first occurs. Let $\hat{\mathfrak{G}}_0$ be that sequent, let r be the instance of the inference rule (an $\Box^\circ_\star$ or $\supset^\circ_\star$) that introduced $\hat{L}'$, let $\mathfrak{H}$ be the conclusion of that rule instance, and let $\hat{p}_1$ be the label in $\mathfrak{H}$ containing the formula to which that rule was applied. Then $p_1\mathsf{U}\hat{p}_1$. Let $\hat{p}_2$ be a label in $\hat{\mathfrak{G}}$ with $p_2\mathsf{U}\hat{p}_2$. If p_2 is a singleton, then $\hat{p}_2$ is uniquely defined. Otherwise, we pick for $\hat{p}_2$ the kth repetition of the cluster C_2, where k is the repetion of C_1 in which $\hat{p}_1$ occurred. We can apply the same rule r to $\hat{p}_2$ to obtain the layer $\hat{L}''$. Let the new sequent be $\hat{\mathfrak{G}}_1$.

We also let $\hat{p}'_1$ be the future of $\hat{p}_1$ in $\hat{L}'$, and let $\hat{p}'_2$ be the future of $\hat{p}_2$ in $\hat{L}''$. This allows us to define an embedding $\mathsf{e}\colon \hat{\mathfrak{G}}_0 \to \hat{\mathfrak{G}}_1$ acting like the identity on all layers, except $\hat{L}'$ which is embedded into $\hat{L}''$ mapping each label to its unique future, except for $\overrightarrow{\hat{p}_1}$ which is embedded into $\overrightarrow{\hat{p}_2}$. We are now going to apply the Embedding Lemma A.12 to repeat the proof tree $\mathcal{T}$. More precisely, let $\mathfrak{H}_1, \ldots, \mathfrak{H}_m$ be the leaves of $\mathcal{T}$ and let $\mathfrak{H}_1, \ldots, \mathfrak{H}_l$ for some $l \leq m$ be the leaves of $\mathcal{T}$ that are N-unfoldings of $\mathfrak{G}$ (and $\hat{\mathfrak{G}}$ is among them). Each of $\mathfrak{H}_i$ has a topmost layer $\hat{L}_{2,i}$ that is an N-unfolding of L_2, and that has

the labels $\hat{s}_i$, $\hat{t}_i$, and $\hat{p}_{2,i}$, corresponding to s, t, and p_2 in the unfolding (if s is in a cluster, we pick for $\hat{s}_i$ the first repetition; if t is in a cluster, we pick for $\hat{t}_i$ the last repetition).

We proceed now similarly to the proof of Lemma A.15 by plugging in a copy of the subtree $\mathcal{T}$ at each leaf $\mathfrak{H}_1, \ldots, \mathfrak{H}_l$. Note that whenever a new layer is created and the previous one is lifted using the $\supset^\circ_\star$ and $\Box^\circ_\star$-rules, also the gap between the old an the new triangle is lifted (see Fig. 13). We call $\hat{L}_{3,i}$ the new topmost layer (which is a copy of $\hat{L}_{2,i}$ with and additional part $\hat{L}^*_{3,i}$ that is a copy of labels in $\hat{L}_{2,i}$ but that have no counterpart in $\hat{L}_1$. If a rule instance in $\mathcal{T}$ is adding a formula to one of the labels u with $s\, R_{\mathfrak{h}_i}\, u\, R_{\mathfrak{h}_i}\, t$ then we need do the same to the corresponding label in $\hat{L}^*_{3,i}$. Formally, this can be achieved via a second embdding for each i, applying again Lemma A.12.

As in the previous Lemma A.15, we now repeat this construction n times, so that we obtain our desired n-unfolding of $\mathfrak{S}'$. □

Lemma 9.10 (Unfolding Lemma)**.** *All sets $\mathfrak{S}_i$ of sequents generated by the algorithm from Fig. 7 are F-unfoldable.*

Proof. The set $\mathfrak{S}'_0 = \{\mathfrak{G}_0(F)\}$ is trivially F-unfoldable. By the previous lemmas, this property is preserved by all the steps that are used to construct the sequence $\mathfrak{S}_0, \mathfrak{S}_1, \ldots$. □

Theorem 9.11. *If the algorithm shown in Fig. 7 terminates in Step 2, then the formula F is a theorem of* $\mathsf{IS4}$.

Proof. By the previous lemma, it follows that the axiomatic set $\mathfrak{S}_k$ that caused the algorithm to terminate is F-unfoldable. In particular, we have a derivation in $\mathsf{labIS4}^\star_\leq$ with conclusion $\mathfrak{G}_0(F)$ where all premises are 1-unfoldings of elements of $\mathfrak{S}_k$, which are all axiomatic. Therefore we can use instances of the id and $\bot^\bullet$ rules to give a complete proof of $\mathfrak{G}_0(F)$ in $\mathsf{labIS4}^\star_\leq$. By Proposition 9.4, there is also a proof of $\mathfrak{G}_0(F)$ in $\mathsf{labIS4}^r_\leq$. This we can append with rules $R\mathsf{rf}$ and $\leq\mathsf{rf}$ at the bottom to get the proof of $\Longrightarrow r{:}F$ in $\mathsf{labIS4}'_\leq$. By Corollary 4.3 and Theorem 4.1, we have that F is a theorem of $\mathsf{IS4}$. □

G. Proofs from Section 10

Lemma 10.1. *The size of a label occurring in a sequent of some $\mathfrak{S}_i$ is at most n. And there are 2^n many equivalence classes of labels with respect to $\sim$.*

Proof. Only subformulas of F can occur in a sequent. Furthermore, the position of each subformula occurrence in F determines if such a subformula can occur on the left or on the right of the $\Longrightarrow$. Moreover, the algorithm introduces in a sequent only labelled formulas which do not already occur in the sequent (Def. 7.1). Hence, we can have at most n formula occurrences at a given label. Also note that $x \sim y$ iff x and y contain the same set of subformulas of F. Hence, there are 2^n different equivalence classes for $\sim$ on labels which can occur in $\mathfrak{S}_i$. □

Lemma 10.2. *The size of a cluster occurring in a sequent of some $\mathfrak{S}_i$ is at most 2^n. And there are $2^{2^n}-1$ many equivalence classes of (non-empty) clusters with respect to $\sim$.*

Proof. We shall show that, by construction, all labels in a cluster are different with respect to $\sim$. There are two ways of introducing non-singleton clusters: (i) via unhappy $\Diamond^\bullet$-formulas (Option 1 of Definition 7.5): if there are equivalent labels in such a cluster, the cluster would have been created at an earlier step of the algorithm. (ii) via unhappy R-triangle or U-triangle loops (Definitions 7.16, 7.17 and 7.18): if at creation the chain from s to t contains duplicates, then they are collapsed immediately afterwards, as they are also valid choices of s and t, fulfilling the same unhappy triangle conditions. Hence, when $\rightrightarrows_{\circlearrowleft}$ terminates, in all new clusters all labels are different with respect to $\sim$. This limits the size of clusters to 2^n, and we have at most $2^{2^n} - 1$ different equivalence classes of non-empty clusters which can occur in $\mathfrak{S}_i$. □

Lemma 10.3. *The length of a branch in a layer in a sequent in a set $\mathfrak{S}_i$ is bounded, and the bound is determined by F.*

Proof. A branch M' is a *past* of a branch M in a sequent $\mathfrak{G}$, if for all $x' \in M'$ there is a $x \in M$ with $x' \leq_{\mathfrak{G}} x$. In this case we write $M' \leq_{\mathfrak{G}} M$. For a branch M we write $\check{M}$ for the labels in M that have a past, and $\bar{M}$ for the ones that do not have a past. Then $M = \check{M} \cup \bar{M}$ and $\check{M} \cap \bar{M} = \varnothing$. We now consider a branch M in a topmost layer L and all its pasts, i.e., branches M' with $M' \leq_{\mathfrak{G}} M$.

Let $M_0, M_1, \ldots, M_k$ be all the pasts of M for which $\bar{M}_i \neq \varnothing$, and such that $M_i \leq_{\mathfrak{G}} M_j$ whenever $0 \leq i \leq j \leq k$ (see Fig. 14). We have $\left|\bar{M}_i\right| \leq 2^n + 1$ for all i because every label in $\bar{M}_i$ is created by Option 2 of Definition 7.5, in correspondence to an unhappy $\Diamond^\bullet$-formula (except for the first label in $\bar{M}_i$, which could be created by layer lifting in correspondence to an unhappy $\Box^\circ$ formula (Construction 7.9), or could be the initial label created in Step 0 of the algorithm). But in any case, after at most 2^n such steps, we encounter a label which is equivalent to a previous one (see also the proof of Lemma 7.6.b)).

Now let $M^i_j = \{y \in M_i \mid \exists x \in \bar{M}_j.\, x \leq_{\mathfrak{G}} y\}$ be the set of futures of $\bar{M}_j$ in M_i for some $j < i$ (see Fig. 14). Because of the duplication of clusters in the layer lifting (Construction 7.8), we can have $\left|M^i_j\right| > \left|\bar{M}_j\right|$. We can restrict the size of M^i_j because there is only a limited amount of times a cluster can be duplicated before a U-triangle loops is encountered. Then, the length of the branch from s and t before and after the repetition of the cluster is again bound by $2 \cdot (2^n + 1)$, because otherwise the U-triangle loop would be unhappy. (We have to take $2^n + 1$ twice because the suricata label is not allowed to occur between s and t.) Since the size of a cluster is bound by 2^n, and the number of clusters in $\bar{M}_j$ is bound by 2^n, the size of M^i_j is bound by $2 \cdot 2 \cdot 2^n \cdot 2^n \cdot 2^n = 2^{3n+2}$. (This is a very naive argument, and the number could be restricted to a much smaller one.

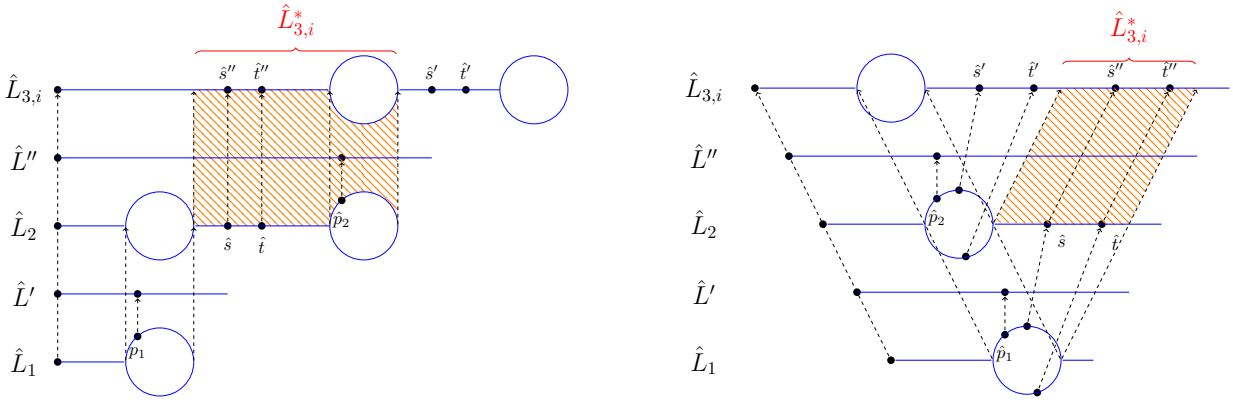


Fig. 13. Left: Unfolding of an unhappy R-triangle loop Right: Unfolding of an unhappy U-triangle loop

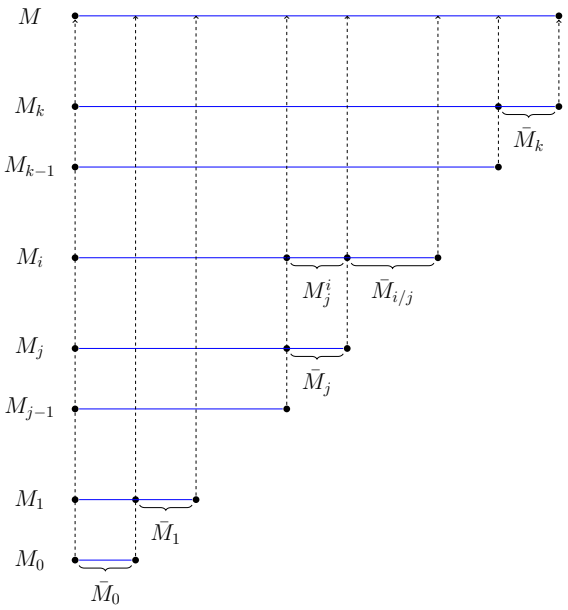


Fig. 14. A branch and its pasts

However, this paper is already technical enough, and we are here only interested in the fact that such a bound exists.)

Let us now put a bound to k. Very naively, we see an R-triangle loop after $k_{\max} = 2^{2^n}$ (as we need to repeat the cluster C_1, see Definition 7.16 and Lemma 10.2). Let M_j be the branch that sees a repetition of M_i. Define $\bar{M}_{i/j}$ (for $j < i$) to be the set of labels in M_i that do not have a past in M_j (see Fig. 14). Then the R-triangle loop is unhappy if $\left|\bar{M}_{i/j}\right| \geq 2 \cdot 2^n = 2^{n+1}$ (same argument as above). Since $2^{3n+2} > 2^{n+1}$, we have that $|M| \leq k_{\max} \cdot \left|M^i_j\right| = 2^n \cdot 2^{2^n} \cdot 2^{3n+2} = 2^{2^n+4n+2}$.

Note that as soon we observe an unhappy triangle loop, the size of the newly created cluster is $|C| \leq 2^n$. □

Theorem 10.4 (Termination)**.** *The proof search algorithm given in Fig. 7 is terminating.*

Proof. By the previous lemma, the size of a branch in a layer is bounded by the formula F in the end sequent. Hence the number of possible layers that can occur in a sequent in a set $\mathfrak{S}_i$ that is visited by the algorithm is also bounded. This puts a limit to the height of the tree of layers in the sequents, as eventually there will be a simulation by a previous layer. □