Cut-free Single-pass Tableaux for the Logic of Common Knowledge

Pietro Abate[1], Rajeev Goré[1], and Florian Widmann[2]

The Australian National University
Canberra ACT 0200, Australia
{Pietro.Abate|Rajeev.Gore|Florian.Widmann}@anu.edu.au

Abstract. We present a cut-free tableau calculus with histories and variables for the EXPTIME-complete multi-modal logic of common knowledge (*LCK*). Our calculus constructs the tableau using only one pass, so proof-search for testing theoremhood of φ does not exhibit the worst-case EXPTIME-behaviour for *all* φ as in two-pass methods. Our calculus also does not contain a "finitized ω-rule" so that it detects cyclic branches as soon as they arise rather than by worst-case exponential branching with respect to the size of φ. Moreover, by retaining the rooted-tree form from traditional tableaux, our calculus becomes amenable to the vast array of optimisation techniques which have proved essential for "practical" automated reasoning in very expressive description logics. Our calculus forms the basis for developing a uniform framework for the family of all fix-point logics of common knowledge. However, there is still no "free lunch" as, in the worst case, our method exhibits 2EXPTIME-behaviour. A prototype implementation can be found at `twb.rsise.anu.edu.au` via the web.

1 Introduction and Motivation

The logic of common knowledge *LCK* is a multi-modal logic where for a finite number of agents $a_1, \ldots, a_n$, the expressions $[i]\varphi$ captures that "agent a_i knows φ"; $[E]\varphi$ captures that "every agent knows φ"; and $[C]\varphi$ captures that "φ is common knowledge". *LCK* is a fix-point logic since the equivalence $[C]\varphi \leftrightarrow [E](\varphi \wedge [C]\varphi)$ is valid. The decision problem for *LCK* is known to be EXPTIME-complete [6] and automated reasoning in *LCK* is of fundamental importance in the area of intelligent agents.

The traditional decision procedure for *LCK* is a two-pass tableau method which *always* exhibit the worst-case EXPTIME behaviour. It should be possible to extend the optimal resolution-based methods [2] and optimal automata-based [10] methods for fix-point temporal logics to *LCK*. But we know of no actual resolution methods or automata-based methods for *LCK* itself, although the resolution method of Dixon and Fisher [4] comes close since it includes a fix-point modality $[C]\varphi$, but does not seem to actually allow $[C]\varphi$ as an object level connective. There is also a decision procedure for *LCK* using a cut-free Gentzen-style sequent calculus [9] with a "finitized ω-rule" which *always* has an exponential number of premises with respect to the size of the given formula φ, even when this is not necessary.

Thus, we know of no decision procedure for *LCK* which does not exhibit the worst-case exponential behaviour for *all* formulae.

We present a sound and complete tableau-based decision procedure for *LCK* with the following properties: it is cut-free; the average-case behaviour is not necessarily exponential in the size of the given formula; the tableau is a finite rooted tree rather than a cyclic graph; the decision procedure requires only a single pass of the tableau which can be done as it is created; there is potential for parallelisation where each branch is farmed out to a different processor; and the method is amenable to the vast array of optimisations [7] which have led to "practical" tableau-based decision procedures for very expressive description logics.

Our tableaux require standard extra devices like "histories" to detect cyclic branches and slightly non-standard "variables" to pass information from children to parents in the tableau. But the only extra operations are those of set membership, set intersection, and the usual min/max operations on sets of integers.

The price we pay is that, in the worst-case, our tableaux require $O(2^{2^{|\varphi|}})$-time rather than $O(2^{|\varphi|})$-time relative to the size $|\varphi|$ of the given formula φ. We have implemented our procedure using the Tableau WorkBench (`twb.rsise.anu.edu.au`) [1].

A longer version with full proofs can be found at `http://users.rsise.anu.edu.au/~rpg/submissions.html` via the web.

2 Syntax and Semantics

Definition 1. *Let* AP *denote the set* $\{p_0, p_1, p_2, \dots\}$ *of propositional variables and* AG *the finite set* $\{0, 1, \dots, N_{\mathrm{AG}}\}$ *of* agents. *The set* Fml *of all formulae of the logic LCK is inductively defined as follows:* $\mathrm{AP} \subset \mathrm{Fml}$*; if* φ *and* ψ *are in* Fml*, then so are* $\neg\varphi$, $\varphi \wedge \psi$, $\varphi \vee \psi$, $[C]\varphi$, *and* $\langle C\rangle\varphi$*; if* φ *is in* Fml*, then so are* $[a]\varphi$ *and* $\langle a\rangle\varphi$ *for every* $a \in \mathrm{AG}$. *Additionally, we define* $[E]\varphi := \bigwedge_{a\in\mathrm{AG}}[a]\varphi$ *and* $\langle E\rangle\varphi := \bigvee_{a\in\mathrm{AG}}\langle a\rangle\varphi$. *Let* $\mathrm{Fml}\langle \mathrm{C}\rangle$ *denote the set of all* $\langle C\rangle$*-formulae.*

Definition 2. *A* transition frame *is a pair* (W,R) *where* W *is a non-empty set of worlds and* R *is a function that assigns to each agent* $a \in \mathrm{AG}$ *a binary relation* R_a *over* W*. We define* $R_{\mathrm{AG}} := \bigcup_{a\in\mathrm{AG}} R_a$.

Definition 3. *Let* (W,R) *be a transition frame. A* transition sequence σ *in* (W,R) *is a finite or infinite sequence of worlds in* W *with the following properties: if* σ *is an infinite sequence* $\sigma_0, \sigma_1, \sigma_2, \dots$*, then* $\sigma_i R_{\mathrm{AG}} \sigma_{i+1}$ *for all* $i \in \mathbb{N}$*; if* σ *is a finite sequence* $\sigma_0, \sigma_1, \dots, \sigma_n$*, then* $\sigma_i R_{\mathrm{AG}} \sigma_{i+1}$ *for all* $i < n$ *and there is no* $w \in W$ *such that* $\sigma_n R_{\mathrm{AG}} w$*. For* $w \in W$*, a* w-sequence σ *in* (W,R) *is a transition sequence in* (W,R) *with* $\sigma_0 = w$*. For* $w \in W$*, let* $\mathcal{B}(w)$ *be the set of all* w*-sequences in* (W,R) *(we assume that* (W,R) *is clear from the context).*

Definition 4. *A* model $M = (W,R,L)$ *is a transition frame* (W,R) *and a labelling function* $L : W \longrightarrow 2^{\mathrm{AP}}$ *which associates with each world* w *a set* $L(w)$.

Definition 5. *Let* $M = (W,R,L)$ *be a model. The* satisfaction relation $\Vdash$ *is defined inductively as follows for each* $a \in \mathrm{AG}$ *where we give only the modal clauses:*

$$\begin{array}{lll}
M,w \Vdash [a]\varphi & \textit{iff} & \forall v \in W.\, wR_a v \Rightarrow M,v \Vdash \varphi \\
M,w \Vdash \langle a\rangle\varphi & \textit{iff} & \exists v \in W.\, wR_a v \,\&\, M,v \Vdash \varphi \\
M,w \Vdash [C]\varphi & \textit{iff} & \forall \sigma \in \mathcal{B}(w).\, \forall i \in \mathbb{N}_{>0}.\, M,\sigma_i \Vdash \varphi \\
M,w \Vdash \langle C\rangle\varphi & \textit{iff} & \exists \sigma \in \mathcal{B}(w).\, \exists i \in \mathbb{N}_{>0}.\, M,\sigma_i \Vdash \varphi \ .
\end{array}$$

Traditionally, the semantics of $[C]\varphi$ is given with the help of iterations of the formula $[E]\varphi$, but it is easy to show that both definitions are equivalent [5].

Definition 6. *A formula* $\varphi \in \mathrm{Fml}$ *is* satisfiable *iff there is a model* $M = (W, R, L)$ *and some* $w \in W$ *such that* $M, w \Vdash \varphi$. *A formula* $\varphi \in \mathrm{Fml}$ *is* valid *iff* $\neg\varphi$ *is not satisfiable.*

Definition 7. *A formula* $\varphi \in \mathrm{Fml}$ *is in* negation normal form *if the symbol* $\neg$ *appears only immediately before propositional variables. For every formula* $\varphi \in \mathrm{Fml}$, *we can obtain a formula* $\mathrm{nnf}(\varphi)$ *in negation normal form by pushing negations inward as far as possible (*e.g. *by using de Morgan's laws) such that* $\varphi \leftrightarrow \mathrm{nnf}(\varphi)$ *is valid.*

3 A One-pass Tableau Algorithm for *LCK*

Our tableaux store additional information with each node of the tableau using *histories* and *variables*. A history is a mechanism for collecting extra information during proof search and passing it from parents to children. A variable is a mechanism to propagate information from children to parents.

Definition 8. *A* tableau node *x is of the form* $(\Gamma :: \mathrm{HAg}, \mathrm{HCr} :: \mathrm{mrk}, \mathrm{uev})$ *where:*

Γ *is a set of formulae;*
HAg *is a partial function from formulae to agents;*
HCr *is a list of the formula sets of some designated ancestors of x;*
mrk *is a boolean valued variable indicating whether the node is marked; and*
uev *is a partial function from formulae to* $\mathbb{N}_{>0}$.

The list HCr and the partial function HAg are histories, *i.e.* their values in a node are determined by the parent node, whereas mrk and uev are variables, *i.e.* their values in a node are determined by the children. In the following we call tableau nodes just nodes when the meaning is clear.

Definition 9. *The partial function* $\mathrm{uev}_\perp : \mathrm{Fml} \rightharpoonup \mathbb{N}_{>0}$ *is the constant function that is undefined for all formulae (*i.e. $\mathrm{uev}_\perp(\psi) = \perp$ *for all* $\psi \in \mathrm{Fml}$*). Analogously, the partial function* $\mathrm{HAg}_\perp : \mathrm{Fml} \rightharpoonup \mathrm{AG}$ *is undefined everywhere.*

Definition 10. *A* tableau *for a formula* $\phi \in \mathrm{Fml}$ *is a tree of tableau nodes with root* $(\{\phi\} :: \mathrm{HAg}_\perp, [] :: \mathrm{mrk}, \mathrm{uev})$ *where the children of a node x are obtained by a single application of a rule to x (*i.e. *only one rule can be applied to a node). A tableau is* expanded *if no rules can be applied to any of its leaves. An expanded tableau is closed if its root has mrk* $=$ **true**, *and open otherwise.*

Note that mrk and uev in the definition are not given but are part of the result as they are determined by the children of the root. Intuitively, by soundness and completeness, ϕ is unsatisfiable iff the tableau for ϕ is closed: see the full paper for proofs.

Note 11. In the following, we use Λ to denote a set containing only propositional variables or their negations (*i.e.* $\varphi \in \Lambda \Rightarrow \exists p \in \mathrm{AP}.\ \varphi = p$ or $\varphi = \neg p$). To focus on the "important" parts of the rule, we use "$\cdots$" for the "unimportant" parts which are passed from node to node unchanged (*e.g.* $(\Gamma :: \cdots :: \cdots)$).

3.1 The Rules

Terminal Rule. $$(\mathit{id})\ \frac{(\Gamma :: \cdots :: \mathrm{mrk}, \mathrm{uev})}{}\ \{p, \neg p\} \subseteq \Gamma \text{ for some } p \in \mathrm{AP}$$

with mrk := **true** and uev := $\mathrm{uev}_\perp$. The node is "closed" so we pass this information up to the parent by putting mrk to **true**, and putting uev as undefined for all formulae.

Universal Branching (β) Rules.

$$(\vee)\ \frac{(\varphi_1 \vee \varphi_2 \ ; \ \Gamma :: \cdots :: \mathrm{mrk}, \mathrm{uev})}{(\varphi_1 \ ; \ \Gamma :: \cdots :: \mathrm{mrk}_1, \mathrm{uev}_1) \mid (\varphi_2 \ ; \ \Gamma :: \cdots :: \mathrm{mrk}_2, \mathrm{uev}_2)}$$

$$(\langle E \rangle)\ \frac{(\langle E \rangle \varphi \ ; \ \Gamma :: \mathrm{HAg}, \cdots :: \mathrm{mrk}, \mathrm{uev})}{\begin{matrix}\langle 1 \rangle \varphi \ ; \ \Gamma \\ :: \mathrm{HAg}_1, \cdots :: \mathrm{mrk}_1, \mathrm{uev}_1\end{matrix} \mid \ldots \mid \begin{matrix}\langle N_{\mathrm{AG}} \rangle \varphi \ ; \ \Gamma \\ :: \mathrm{HAg}_{N_{\mathrm{AG}}}, \cdots :: \mathrm{mrk}_{N_{\mathrm{AG}}}, \mathrm{uev}_{N_{\mathrm{AG}}}\end{matrix}}$$

$$(\langle C \rangle)\ \frac{(\langle C \rangle \varphi \ ; \ \Gamma :: \cdots :: \mathrm{mrk}, \mathrm{uev})}{(\langle E \rangle \varphi \ ; \ \Gamma :: \cdots :: \mathrm{mrk}_1, \mathrm{uev}_1) \mid (\langle E \rangle \langle C \rangle \varphi \ ; \ \Gamma :: \cdots :: \mathrm{mrk}_2, \mathrm{uev}_2)}$$

with:

$$\mathrm{HAg}_i(\psi) := \begin{cases} i & \text{if } \psi \in \mathrm{Fml}\langle \mathrm{C} \rangle \ \& \ \psi = \varphi \\ \mathrm{HAg}(\psi) & \text{otherwise} \end{cases}$$
$$\text{for } i = 1, \ldots, N_{\mathrm{AG}} \text{ in the } \langle E \rangle\text{-rule only;}$$

$$n := \begin{cases} N_{\mathrm{AG}} & \text{for the } \langle E \rangle\text{-rule} \\ 2 & \text{otherwise} \end{cases}$$

$$\mathrm{mrk} := \mathrm{mrk}_1 \ \& \ \ldots \ \& \ \mathrm{mrk}_n$$

$$\mathrm{excl}_\phi(f)(\psi) := \begin{cases} \perp & \text{if } \psi = \phi \\ f(\psi) & \text{otherwise} \end{cases}$$

$$\mathrm{uev}'_1 := \begin{cases} \mathrm{excl}_{\langle C \rangle \varphi}(\mathrm{uev}_1) & \text{for the } \langle C \rangle\text{-rule} \\ \mathrm{uev}_1 & \text{otherwise} \end{cases}$$

$$\mathrm{uev}'_i := \mathrm{uev}_i \text{ for } i = 2, \ldots, n$$

$$\min_\perp\{f_1, \ldots, f_k\}(\psi) := \begin{cases} l & \text{if } k > 0 \ \& \ \forall i \in \{1, \ldots, k\}.\ f_i(\psi) \neq \perp \ \& \\ & \quad l = \min\{f_1(\psi), \ldots, f_k(\psi)\} \\ \perp & \text{otherwise} \end{cases}$$

$$\mathrm{uev} := \min_\perp\{\mathrm{uev}'_i \mid i \in \{1, \ldots, n\} \ \& \ \mathit{mrk}_i = \mathbf{false}\}$$

The $\vee$-rule is standard and the $\langle E \rangle$-rule just unfolds the definition of $\langle E \rangle \varphi$. The $\langle C \rangle$-rule captures the fix-point nature of the $\langle C \rangle$-formulae. The intuitions of the definitions are:

HAg_i: in the $\langle E \rangle$-rule, if the principal formula $\langle E \rangle \varphi$ is $\langle E \rangle \langle C \rangle \chi$ (*i.e.* we have $\varphi \in \mathrm{Fml}\langle \mathrm{C} \rangle$), the value of φ in HAg_i is set to the corresponding agent a_i of the child to indicate that this child is "tracking" $\langle E \rangle \langle C \rangle \chi$;

mrk: the value of the variable mrk is **true** if the node is "closed", so the definition of mrk just captures the "universal" nature of these rules whereby the parent node is closed if all children are closed;

excl: the definition of $\mathrm{excl}_\phi(f)(\psi)$ just ensures that $\mathrm{excl}_\phi(f)(\phi)$ is undefined;

uev'_1: the definition of uev'_1 ensures that its value is undefined for the principal formulae of the $\langle C\rangle$-rule since the first child is charged with fulfilling $\langle C\rangle\varphi$; the other uev'_i for $i > 1$ are just there to avoid a case distinction in the definition of uev;

$\min_\perp$: the definition of $\min_\perp$ ensures that we take the minimum of all $f_i(\psi)$ only when all functions are defined for ψ;

uev: we only consider the uev'_i of unmarked (*i.e.* unclosed) children. If all children are "closed", then the argument to $\min_\perp$ is the empty set with $k = 0$, so uev is undefined everywhere. Otherwise, for a given formula ψ, we define $\mathrm{uev}(\psi)$ by first ensuring that $\mathrm{uev}'(\psi)$ is defined for *every* unclosed child, which means that none of the children "fulfil" ψ, and then assigning $\mathrm{uev}(\psi)$ to be their minimum value, which ensures that we keep the "highest" looping non-fulfilling path. In particular, if the first child is unmarked in the $\langle C\rangle$-rule, then the first child fulfils $\langle C\rangle\varphi$, so $\mathrm{uev}(\langle C\rangle\varphi)$ is undefined for the principal formula $\langle C\rangle\varphi$ via the definition of uev'_i.

Existential Branching Rule.

$$(\langle a\rangle)\ \frac{\begin{array}{l}\langle a_1\rangle\varphi_1;\ldots;\langle a_n\rangle\varphi_n\ ;\ \langle a_{n+1}\rangle\varphi_{n+1};\ldots;\langle a_{n+m}\rangle\varphi_{n+m}\ ;\\ [1]\Delta_1;\ldots;[N_{\mathrm{AG}}]\Delta_{N_{\mathrm{AG}}}\ ;\ \Lambda\\ ::\mathrm{HAg},\mathrm{HCr}::\mathrm{mrk},\mathrm{uev}\end{array}}{\begin{array}{l}\varphi_1\ ;\ \Delta_{a_1}\\ ::\mathrm{HAg}_\perp,\mathrm{HCr}_1::\mathrm{mrk}_1,\mathrm{uev}_1\end{array}\ \Big|\cdots\Big|\ \begin{array}{l}\varphi_n\ ;\ \Delta_{a_n}\\ ::\mathrm{HAg}_\perp,\mathrm{HCr}_n::\mathrm{mrk}_n,\mathrm{uev}_n\end{array}}$$

where:

(1) $\{p,\neg p\}\not\subseteq\Lambda$
(2) $n+m\geq 0$
(3) $\forall i\in\{1,\ldots,n+m\}.\,a_i\in\mathrm{AG}$
(4) $\forall i\in\{1,\ldots,n\}.\,\forall j\in\{1,\ldots,\mathrm{len}(\mathrm{HCr})\}.\,\{\varphi_i\}\cup\Delta_{a_i}\neq\mathrm{HCr}[j]$
(5) $\forall k\in\{n+1,\ldots,n+m\}.\,\exists j\in\{1,\ldots,\mathrm{len}(\mathrm{HCr})\}.\,\{\varphi_k\}\cup\Delta_{a_k}=\mathrm{HCr}[j]$

with:

$$\mathrm{HCr}_i := \mathrm{HCr}\ @\ [\{\varphi_i\}\cup\Delta_{a_i}]\ \text{for } i=1,\ldots,n$$

$$\begin{aligned}\mathrm{mrk} := {}& \textstyle\bigvee_{i=1}^n \mathrm{mrk}_i \text{ or}\\ & \exists i\in\{1,\ldots,n\}.\,\exists\psi\in\{\varphi_i\}\cup\Delta_{a_i}.\,\perp\neq\mathrm{uev}_i(\psi)>\mathrm{len}(\mathrm{HCr})\end{aligned}$$

$$\begin{aligned}\mathrm{uev}_k(\cdot) := {}& j\in\{1,\ldots,len(\mathrm{HCr})\}\text{ such that }\{\varphi_k\}\cup\Delta_{a_k}=\mathrm{HCr}[j]\\ & \text{for } k=n+1,\ldots,n+m\end{aligned}$$

$$\mathrm{uev}(\psi) := \begin{cases}\mathrm{uev}_i(\psi) & \text{if } \psi\in\mathrm{Fml}\langle\mathrm{C}\rangle\ \&\ \psi=\varphi_i\ \&\ \mathrm{HAg}(\psi)=a_i\\ & \quad\text{for an } i\in\{1,\ldots,n+m\}\\ \perp & \text{otherwise}\end{cases}$$

Some intuitions are in order:

(1) The $\langle a\rangle$-rule is applicable if the parent node contains no α- or β-formulae and Λ, which contains atoms and their negations only, contains no atomic contradictions.

(2) Both n and m can be zero.

(4) If $n > 0$ then each $\langle a_i \rangle \varphi_i$ for $1 \leq i \leq n$ is not "blocked" by an ancestor, and has a child containing $\varphi_i ; \Delta_{a_i}$, thereby generating the required successor.

(5) If $m > 0$ then each $\langle a_k \rangle \varphi_k$ for $n+1 \leq k \leq m$ is "blocked" from creating its required child $\varphi_k ; \Delta_{a_k}$ because some ancestor does the job.

HAg_i: we reset the partial functions HAg_i of the children so that they are undefined for all formulae since we no longer need to "track" them.

HCr_i: is just the HCr of the parent but with an extra entry to extend the "history" of nodes on the path from the root down to the i^{th} child.

mrk: captures the "existential" nature of this rule whereby the parent is marked if some child is closed or if some child contains a formula whose uev is defined and "loops" lower than the parent. Moreover, if n is zero, then mrk is set to **false** to indicate that this branch is "open" (*i.e.* not "closed").

uev_k: for $n+1 \leq k \leq n+m$ the k^{th} child is blocked by a proxy child higher in the branch. For every such k we set uev_k to be the *constant* function which maps every formula to the level of this proxy child. Note that this is just a temporary function used to define uev as explained next.

$\mathrm{uev}(\psi)$: for a $\langle C \rangle$-formula $\psi = \langle C \rangle \chi$, we check whether there is a principal formula $\langle a_i \rangle \varphi_i$ so that $\varphi_i = \psi$ and $a_i = \mathrm{HAg}(\psi)$ indicating that ψ is being tracked by the i^{th} child and hence this i^{th} child has the responsibility to fulfil $\langle C \rangle \chi$. If no such formulae exist, we put uev to be undefined. Otherwise, we take uev of ψ from the corresponding child if $\langle a_i \rangle \varphi_i$ is "unblocked", or set it to be the *level* of the proxy child higher in the branch if it is "blocked". For all other formulae, put uev to be undefined. The intuition is that a defined $\mathrm{uev}(\psi)$ tells us that there is a "loop" which starts at the parent and eventually "loops" up to some blocking node higher up on the current branch. The value of $\mathrm{uev}(\psi)$ is the level of the proxy as we cannot tell whether this "loop" is "good" or "bad" until we backtrack up to that level.

Linear (α) Rules.

$$\textbf{(}[E]\textbf{)} \; \frac{([E]\varphi \,;\, \Gamma :: \cdots :: \cdots)}{([1]\varphi \,;\, [2]\varphi \,;\, \ldots \,;\, [N_{\mathrm{AG}}]\varphi \,;\, \Gamma :: \cdots :: \cdots)}$$

$$\textbf{(}\wedge\textbf{)} \; \frac{(\varphi \wedge \psi \,;\, \Gamma :: \cdots :: \cdots)}{(\varphi \,;\, \psi \,;\, \Gamma :: \cdots :: \cdots)} \qquad \textbf{(}[C]\textbf{)} \; \frac{([C]\varphi \,;\, \Gamma :: \cdots :: \cdots)}{([E]\varphi \,;\, [E][C]\varphi \,;\, \Gamma :: \cdots :: \cdots)}$$

The $\wedge$-rule is standard and the $[E]$-rule just unfolds the definition of $[E]\varphi$. The $[C]$-rule capture the fix-point nature of the corresponding formulae. These rules do not modify the histories or variables at all.

3.2 A Fully Worked Example

Consider the obviously valid formula $[C](p \vee \neg p)$. Its negation $\langle C \rangle (p \wedge \neg p)$ is not satisfiable and the root of an expanded tableau for $\langle C \rangle (p \wedge \neg p)$ should be marked.

Figure 1 shows a tableau with root node (1) and Figure 2 shows the sub-tableau rooted at node (4). Each node is classified as a ρ-node if rule ρ is applied to that node in the tableau. The edges labelled with $\langle 1 \rangle$ and $\langle 2 \rangle$ go from states to pre-states. Dotted frames indicate that the sub-tableaux at these nodes are not shown because they are very similar to sub-tableaux of other nodes: that is node $(5a)$ and $(4')$ are treated the same way as node $(1a)$ and (4), respectively. We have also omitted the history HAg as it does not play a role in this simple example. Dots "$\cdots$" indicate that the corresponding values are not important for this example. The partial function UEV maps $\langle C\rangle(p \wedge \neg p)$ to 1 and is undefined otherwise as explained below.

The marking of the nodes in Figure 1 with **true** is straightforward, starting from the leaves (2b) and (2c) and flowing upwards, to node (1a). For nodes (1b) and node (4), our procedure constructs the tableau shown in Figure 2 for node (4). The leaf (6) is an $\langle a\rangle$-node, but it is "blocked" from creating the desired successor containing $\{\langle C\rangle(p \wedge \neg p)\}$ because there is a j such that $\mathrm{HCr}_6[j] = \{\langle C\rangle(p \wedge \neg p)\}$: namely $j = 1$. Thus the $\langle a\rangle$-rule computes $UEV(\langle C\rangle(p \wedge \neg p)) = 1$ as stated above and also puts mrk := **false**. The leaf node (6') behaves in exactly the same way, so node (5b) does not change the value of uev in taking their minimum, and also computes mrk $=$ **false** $\vee$ **false** $=$ **false**. Node (5a) has a dotted border since it behaves like node (1a), meaning that the β_1-child of node (5) is "closed". Since node (5b) is not "closed", node (5) inherits its uev and mrk values.

The crux of our procedure happens at node (4) which is an $\langle a\rangle$-node with $\mathrm{HCr}_4 = []$ and hence $len(\mathrm{HCr}_4) = 0$. The $\langle a\rangle$-rule therefore finds a child node (5) and a formula $\langle C\rangle(p \wedge \neg p)$ in it, which jointly satisfy $1 = \mathrm{uev}_5(\langle C\rangle(p \wedge \neg p)) > len(\mathrm{HCr}_4) = 0$. That is, node (4) "sees" a child that "loops lower", meaning that node (5) is the root of an "isolated" subtree which does not fulfil its eventuality $\langle C\rangle(p \wedge \neg p)$. Thus the $\langle a\rangle$-rule sets $\mathrm{mrk}_4 =$ **true**, marking (4) as "closed".

Returning to Figure 1, we find that the node (4') behaves in exactly the same way as does (4), which forces node (1b) to evaluate $\mathrm{mrk}_{1b} =$ **true**. Since node (1) now has two "closed" β-children, it also gets marked as "closed".

References

1. P. Abate. The Tableau Workbench: a framework for building automated tableau-based theorem provers. PhD thesis, The Australian National University, 2006.
2. A. Basukoski and A. Bolotov. A clausal resolution method for branching time logic ECTL+. *Annals of Mathematics and Artificial Intelligence*, 46(3):235–263, 2006.
3. M. Ben-Ari, Z. Manna, and A. Pnueli. The temporal logic of branching time. In *Proceedings of Principles of Programming Languages*, 1981.
4. C. Dixon and M. Fisher. Resolution-based proof for multi-modal temporal logics of knowledge. In *Proc. TIME 2000*, pages 69-78.
5. R. Fagin, J. Y. Halpern, Y. Moses, and M. Y. Vardi. Reasoning about Knowledge. The MIT Press, Cambridge, Massachusetts, 1995.
6. J. Y. Halpern and Y. Moses. A guide to completeness and complexity for modal logics of knowledge and belief. In *Artificial Intelligence 54*, pages 319–379, 1992.
7. I. Horrocks and P. F. Patel-Schneider. Optimising description logic subsumption. *Journal of Logic and Computation*, 9(3):267–293, 1999.
8. U. Hustadt and B. Konev. TRP++: A temporal resolution prover. In *Collegium Logicum*, pages 65–79. Kurt Gödel Society, 2004.
9. G. Jäger, M. Kretz, and T. Studer. Cut-free common knowledge. To appear `http://www.iam.unibe.ch/til/publications/to_appear/jaeger_kretz_studer_06`
10. M. Y. Vardi and P. Wolper. Automata-theoretic techniques for modal logics of programs. *Journal of Computer Systems and Science*, 1986.

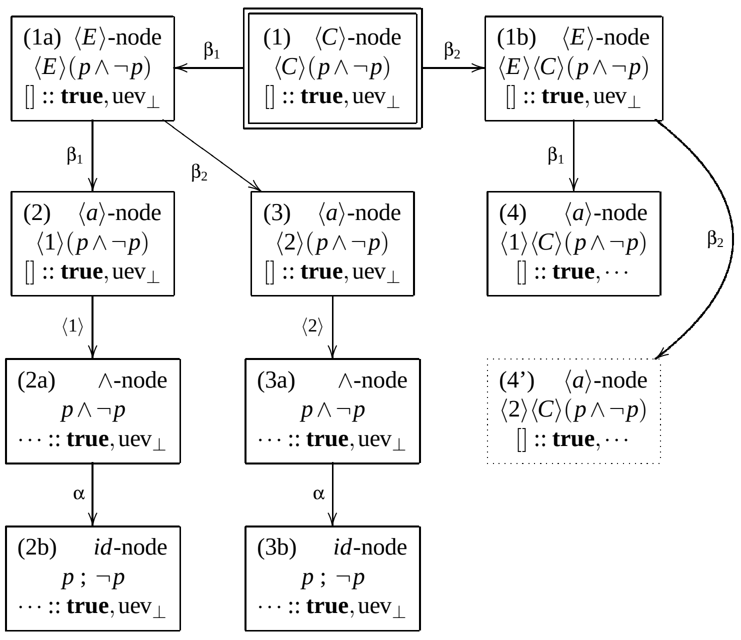
(1) ⟨C⟩-node
$\langle C\rangle(p\wedge\neg p)$
$[] :: \textbf{true}, \mathrm{uev}_\perp$
(1a) ⟨E⟩-node
$\langle E\rangle(p\wedge\neg p)$
$[] :: \textbf{true}, \mathrm{uev}_\perp$
(1b) ⟨E⟩-node
$\langle E\rangle\langle C\rangle(p\wedge\neg p)$
$[] :: \textbf{true}, \mathrm{uev}_\perp$
β_1
β_2
(2) ⟨a⟩-node
$\langle 1\rangle(p\wedge\neg p)$
$[] :: \textbf{true}, \mathrm{uev}_\perp$
(3) ⟨a⟩-node
$\langle 2\rangle(p\wedge\neg p)$
$[] :: \textbf{true}, \mathrm{uev}_\perp$
(4) ⟨a⟩-node
$\langle 1\rangle\langle C\rangle(p\wedge\neg p)$
$[] :: \textbf{true}, \cdots$
$\langle 1\rangle$
$\langle 2\rangle$
(2a) ∧-node
$p\wedge\neg p$
$\cdots :: \textbf{true}, \mathrm{uev}_\perp$
(3a) ∧-node
$p\wedge\neg p$
$\cdots :: \textbf{true}, \mathrm{uev}_\perp$
(4’) ⟨a⟩-node
$\langle 2\rangle\langle C\rangle(p\wedge\neg p)$
$[] :: \textbf{true}, \cdots$
α
(2b) *id*-node
$p\,;\,\neg p$
$\cdots :: \textbf{true}, \mathrm{uev}_\perp$
(3b) *id*-node
$p\,;\,\neg p$
$\cdots :: \textbf{true}, \mathrm{uev}_\perp$

Fig. 1. An example: a tableau for $\langle C\rangle(p\wedge\neg p)$

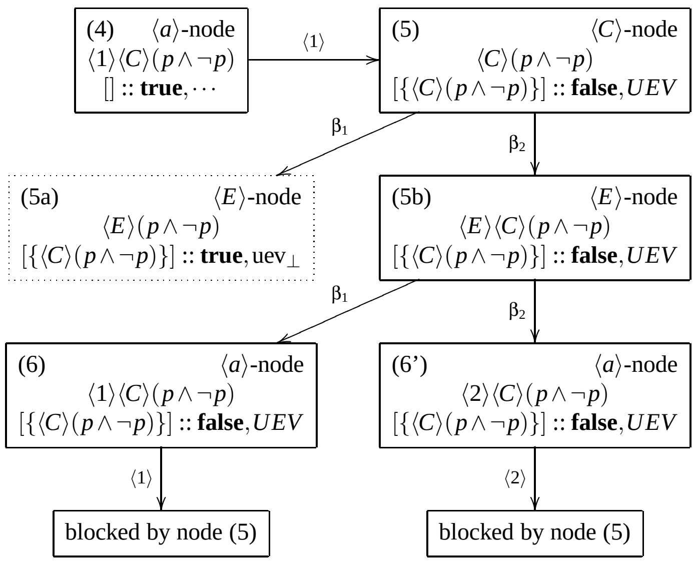
(4) ⟨a⟩-node
$\langle 1\rangle\langle C\rangle(p\wedge\neg p)$
$[] :: \textbf{true}, \cdots$
$\langle 1\rangle$
(5) ⟨C⟩-node
$\langle C\rangle(p\wedge\neg p)$
$[\{\langle C\rangle(p\wedge\neg p)\}] :: \textbf{false}, UEV$
β_1
β_2
(5a) ⟨E⟩-node
$\langle E\rangle(p\wedge\neg p)$
$[\{\langle C\rangle(p\wedge\neg p)\}] :: \textbf{true}, \mathrm{uev}_\perp$
(5b) ⟨E⟩-node
$\langle E\rangle\langle C\rangle(p\wedge\neg p)$
$[\{\langle C\rangle(p\wedge\neg p)\}] :: \textbf{false}, UEV$
(6) ⟨a⟩-node
$\langle 1\rangle\langle C\rangle(p\wedge\neg p)$
$[\{\langle C\rangle(p\wedge\neg p)\}] :: \textbf{false}, UEV$
(6’) ⟨a⟩-node
$\langle 2\rangle\langle C\rangle(p\wedge\neg p)$
$[\{\langle C\rangle(p\wedge\neg p)\}] :: \textbf{false}, UEV$
$\langle 2\rangle$
blocked by node (5)
blocked by node (5)

Fig. 2. An example: a tableau for $\langle C\rangle(p\wedge\neg p)$ (continued)