A tableau method for public announcement logics

Philippe Balbiani[1], Hans van Ditmarsch[2],
Andreas Herzig[1], and Tiago de Lima[1]

[1] Institut de Recherche en Informatique de Toulouse, France
[2] Computer Science, University of Otago, New Zealand

Abstract. Public announcement logic is an extension of multi-agent epistemic logic with dynamic operators to model the informational consequences of announcements to the entire group of agents. We propose a labelled tableau-calculus for this logic. We also present an extension of the calculus for a logic of arbitrary announcements.

1 Introduction

Public announcement logic (PAL) was originally proposed in [1]. This is one from a series of logics developed with the aim of modelling dynamics of knowledge and belief in multi-agent settings. These logics, sometimes called dynamic epistemic logics (DELs), deal with a number of epistemic scenarios and puzzles (see [2–5] for some examples). PAL is the simplest of them. It extends epistemic logic (EL) with dynamic operators $[\varphi]$. The formula $[\varphi]\psi$ stands for 'ψ is true after the public announcement of φ'. Being the simplest form of agent communication, public announcements are present in all DELs. Some recent works, such as [6–8], show that they suffice to model dynamics of knowledge and belief in several cases.

Traditionally, axiomatisations for DELs are obtained by means of *reduction axioms*. In the particular case of PAL, they permit the translation of each PAL-formula into an equivalent EL-formula. The well-known axiomatisation for PAL is therefore obtained by just extending that of EL by the former reduction axioms and by a necessitation inference rule for announcements operators.

It follows that both logics have the same expressivity. Nevertheless the translated formula is exponentially larger than the original one. That is, PAL is strictly more succinct. This is the reason why PAL is considered to be more convenient for reasoning about knowledge [5]. Curiously however, satisfiability check in PAL is also PSPACE-complete [9].

In this paper, we present a tableau-calculus for PAL. The method decides satisfiability without reducing PAL-formulas to another language. We also extended the calculus to deal with *arbitrary public announcement logic* (APAL). It extends PAL by a modal operator $\Diamond$. The formula $\Diamond\varphi$ stands for 'there is a public announcement after which φ is true'. Note that while all other DELs deal with the question 'what becomes true after the execution of a given action?', APAL

deals with the question 'is there an action whose execution makes a given formula true?'. This was originally proposed in our technical report [10], including the tableau calculi presented in this paper.

The organisation of the paper is as follows. In Section 2 we define syntax and semantics of PAL. The tableau-calculus for this logic in presented in Section 3. In Section 4 we transform it into a decision procedure. In Section 5 we expand it to a tableau-calculus for APAL. Section 6 is dedicated to some related works and discussion.

2 Syntax and semantics of public announcement logic

Assume a finite set of agents A and a countably infinite set of atoms P.

Definition 1 (Language). *The language $\mathcal{L}_{\mathrm{PAL}}$ is inductively defined by the following BNF:*

$$\varphi ::= p \mid \neg\varphi \mid (\varphi \wedge \varphi) \mid K_a\varphi \mid [\varphi]\varphi$$

where a ranges over A and p ranges over P.

For $K_a\varphi$, read 'agent a knows that φ'. For $[\varphi]\psi$, read 'after public announcement of φ, ψ is true'. Other propositional and epistemic connectives are defined by usual abbreviations. The dual of K_a is $\widehat{K}_a$ and the dual of $[\varphi]$ is $\langle\varphi\rangle$. For $\widehat{K}_a\varphi$ read 'agent a considers it possible that φ', for $\langle\varphi\rangle\psi$ read 'the announcement φ can be made and after that ψ is true'.

Definition 2 (Structures). *An* epistemic model $M = \langle W, R, V\rangle$ *consists of a non-empty set W of (factual)* states *(or 'worlds'),* accessibility $R : A \to \wp(W \times W)$*, and a* valuation $V : P \to \wp(W)$*. For $w \in W$, (M, w) is an* epistemic state *(also known as a pointed Kripke model).*

For $R(a)$ we write R_a and for $V(p)$ we write V_p. Given two states w, w' in W, the intuitive meaning of wR_aw' is that at w, the agent a considers it possible that the real world is w'.

If no restriction is imposed over the accessibility relations in R, then we call the resultant logic K-PAL. If each R_a is reflexive, then the resultant logic is called KT-PAL. If each R_a is reflexive and transitive, then we call the resultant logic S4-PAL. And finally, if each R_a is reflexive, transitive and symmetric, then we call the resultant logic S5-PAL. We continue with the semantics.

Definition 3. *Assume an epistemic model $M = \langle W, R, V\rangle$. The interpretation of an arbitrary $\varphi \in \mathcal{L}_{\mathrm{PAL}}$ is defined by induction as follows:*

$$\begin{array}{ll}
M, w \models p & \textit{iff } w \in V_p \\
M, w \models \neg\varphi & \textit{iff } M, w \not\models \varphi \\
M, w \models \varphi \wedge \psi & \textit{iff } M, w \models \varphi \textit{ and } M, w \models \psi \\
M, w \models K_a\varphi & \textit{iff for all } v \in W,\ wR_av \textit{ implies } M, v \models \varphi \\
M, w \models [\varphi]\psi & \textit{iff } M, w \models \varphi \textit{ implies } M|\varphi, w \models \psi
\end{array}$$

In the last clause, the epistemic model $M|\varphi = \langle W^\varphi, R^\varphi, V^\varphi \rangle$ *is defined as follows:*

$$\begin{aligned} W^\varphi &= \{w' \in W \mid M, w' \models \varphi\} \\ R_a^\varphi &= R_a \cap (W^\varphi \times W^\varphi) \\ V_p^\varphi &= V_p \cap W^\varphi \end{aligned}$$

Formula φ *is valid in model* M*, notation* $M \models \varphi$*, if and only if for all* $w \in W$*,* $M, w \models \varphi$*. Let* C *be a class of models in* $\{\mathrm{K}, \mathrm{KT}, \mathrm{S4}, \mathrm{S5}\}$*. Formula* φ *is valid in* C-PAL*, notation* $\models_{C\text{-PAL}} \varphi$*, if and only if for all epistemic models* $M \in C$*,* $M \models \varphi$*.*

The dynamic modal operator $[\varphi]$ is interpreted as an epistemic state transformer. Announcements are assumed to be truthful, and this is commonly known by all agents. Therefore, the model $M|\varphi$ is the model M restricted to all the states where φ is true, including access between states. For the semantics of the dual operators, we have that $M, w \models \langle\varphi\rangle\psi$ if and only if $M, w \models \varphi$ and $M|\varphi, w \models \psi$.

3 A tableau method for public announcement logic

We present in this section a proof method for public announcement logic that uses tableaux. Exactly in the same way as all other tableau methods, given a formula φ, it systematically tries to construct a model for it. When it fails, φ is inconsistent and thus its negation is valid.

In our representation formulas are prefixed by a number that represents possible worlds in the model (similar to [11, Chapter 8]). Formulas are also prefixed by finite sequences of announcements corresponding to successive model restrictions. Given a finite sequence of formulas $\psi^k = (\psi_1 \dots \psi_k)$, for each $1 \leq i \leq k$, the sequence $(\psi_1 \dots \psi_i)$ is noted ψ^i whereas $\psi^0 = \epsilon$ denotes the empty sequence. In addition, we write $M|\psi^k$ for $M|\psi_1| \dots |\psi_k$.

Definition 4. *A* labelled formula *is a triple* $\lambda = (\psi^k, x, \varphi)$ *where*

- ψ^k *is a finite sequence* $(\psi_1 \dots \psi_k)$ *of formulas in* $\mathcal{L}_{\mathrm{PAL}}$*;*
- $x \in \mathbb{N}$*; and*
- $\varphi \in \mathcal{L}_{\mathrm{PAL}}$*.*

The part ψ^k, x *is the* label *of the formula* φ*. It represents a possible world* x *in the epistemic model that is successively restricted by the formulas in* ψ^k*.*

Definition 5. *A* skeleton *is a ternary relation* $\Sigma \subseteq (A \times \mathbb{N} \times \mathbb{N})$ *that represents the accessibility relations. A* branch *is a pair* $b = (\Lambda, \Sigma)$ *where* Λ *is a set of labelled formulas and* Σ *is a skeleton.*

Definition 6 (Tableau). *A* tableau *is a set* $T^i = \{b_1^i, b_2^i, \dots\}$ *of branches. A tableau* T^{i+1} *is* obtained from *a tableau* T^i *if and only if* $T^{i+1} := (T^i \setminus \{b_j^i\}) \cup B$ *for some* $b_j^i = (\Lambda, \Sigma) \in T^i$ *and some finite set* B *of branches generated from* b_j^i *by the application of one of the* tableau rules *defined below.*

$\neg$: *if* $(\psi^k, x, \neg\neg\varphi) \in \Lambda$, *then* $B = \{(\Lambda \cup \{(\psi^k, x, \varphi)\}, \Sigma)\}$.
$\wedge$: *if* $(\psi^k, x, \varphi_1 \wedge \varphi_2) \in \Lambda$, *then* $B = \{(\Lambda \cup \{(\psi^k, x, \varphi_1), (\psi^k, x, \varphi_2)\}, \Sigma)\}$.
$\vee$: *if* $(\psi^k, x, \neg(\varphi_1 \wedge \varphi_2)) \in \Lambda$, *then* $B = \{(\Lambda \cup \{(\psi^k, x, \neg\varphi_1)\}, \Sigma), (\Lambda \cup \{(\psi^k, x, \neg\varphi_2)\}, \Sigma)\}$.
K: *if* $(\psi^k, x, K_a\varphi) \in \Lambda$ *and* $(a, x, x') \in \Sigma$, *then* $B = \{(\Lambda_0, \Sigma), \ldots (\Lambda_k, \Sigma)\}$, *where*

$$\begin{aligned}
\Lambda_0 &= \Lambda \cup \{(\psi^0, x', \neg\psi_1)\}\\
\Lambda_1 &= \Lambda \cup \{(\psi^0, x', \psi_1), (\psi^1, x', \neg\psi_2)\}\\
\Lambda_2 &= \Lambda \cup \{(\psi^0, x', \psi_1), (\psi^1, x', \psi_2), (\psi^2, x', \neg\psi_3)\}\\
&\vdots\\
\Lambda_{k-1} &= \Lambda \cup \{(\psi^0, x', \psi_1), \ldots, (\psi^{k-2}, x', \psi_{k-1}), (\psi^{k-1}, x', \neg\psi_k)\}\\
\Lambda_k &= \Lambda \cup \{(\psi^0, x', \psi_1), \ldots, (\psi^{k-1}, x', \psi_k), (\psi^k, x', \varphi)\}.
\end{aligned}$$

T: *if* $(\psi^k, x, K_a\varphi) \in \Lambda$, *then* $B = \{(\Lambda \cup \{(\psi^k, x, \varphi)\}, \Sigma)\}$.
4: *if* $(\psi^k, x, K_a\varphi) \in \Lambda$ *and* $(a, x, x') \in \Sigma$, *then* $B = \{(\Lambda_1, \Sigma), \ldots (\Lambda_{k+1}, \Sigma)\}$, *where*

$$\begin{aligned}
\Lambda_0 &= \Lambda \cup \{(\psi^0, x', \neg\psi_1)\}\\
\Lambda_1 &= \Lambda \cup \{(\psi^0, x', \psi_1), (\psi^1, x', \neg\psi_2)\}\\
\Lambda_2 &= \Lambda \cup \{(\psi^0, x', \psi_1), (\psi^1, x', \psi_2), (\psi^2, x', \neg\psi_3)\}\\
&\vdots\\
\Lambda_{k-1} &= \Lambda \cup \{(\psi^0, x', \psi_1), \ldots, (\psi^{k-2}, x', \psi_{k-1}), (\psi^{k-1}, x', \neg\psi_k)\}\\
\Lambda_k &= \Lambda \cup \{(\psi^0, x', \psi_1), \ldots, (\psi^{k-1}, x', \psi_k), (\psi^k, x', K_a\varphi)\}.
\end{aligned}$$

5: *if* $(\psi^k, x, K_a\varphi) \in \Lambda$ *and* $(a, x', x) \in \Sigma$, *then* $B = \{(\Lambda_1, \Sigma), \ldots (\Lambda_{k+1}, \Sigma)\}$, *where*

$$\begin{aligned}
\Lambda_0 &= \Lambda \cup \{(\psi^0, x', \neg\psi_1)\}\\
\Lambda_1 &= \Lambda \cup \{(\psi^0, x', \psi_1), (\psi^1, x', \neg\psi_2)\}\\
\Lambda_2 &= \Lambda \cup \{(\psi^0, x', \psi_1), (\psi^1, x', \psi_2), (\psi^2, x', \neg\psi_3)\}\\
&\vdots\\
\Lambda_{k-1} &= \Lambda \cup \{(\psi^0, x', \psi_1), \ldots, (\psi^{k-2}, x', \psi_{k-1}), (\psi^{k-1}, x', \neg\psi_k)\}\\
\Lambda_k &= \Lambda \cup \{(\psi^0, x', \psi_1), \ldots, (\psi^{k-1}, x', \psi_k), (\psi^k, x', K_a\varphi)\}.
\end{aligned}$$

$\widehat{\mathbf{K}}$: *if* $(\psi^k, x, \neg K_a\varphi) \in \Lambda$, *then* $B = \{(\Lambda \cup \{(\psi^0, x', \psi_1), \ldots, (\psi^{k-1}, x', \psi_k), (\psi^k, x', \neg\varphi)\}, \Sigma \cup \{(a, x, x')\})\}$ *for some* x' *that does not appear in* Λ.
$[\cdot]$: *if* $(\psi^k, x, [\varphi_1]\varphi_2) \in \Lambda$, *then* $B = \{(\Lambda \cup \{(\psi^k, x, \neg\varphi_1)\}, \Sigma), (\Lambda \cup \{(\psi^k, x, \varphi_1), (\psi^k\varphi_1, x, \varphi_2)\}, \Sigma)\}$.
$\langle\cdot\rangle$: *if* $(\psi^k, x, \neg[\varphi_1]\varphi_2) \in \Lambda$, *then* $B = \{(\Lambda \cup \{(\psi^k, x, \varphi_1), (\psi^k\varphi_1, x, \neg\varphi_2)\}, \Sigma)\}$.

Given a formula $\varphi \in \mathcal{L}_{\mathrm{PAL}}$, *the tableau* $T^0 := \{b_1^0\} := \{(\{(\epsilon, 0, \varphi)\}, \emptyset)\}$ *is the* initial tableau *for* φ. *A* tableau for φ *is a tableau that can be obtained from the initial tableau for* φ *by successive applications of tableau rules.*

The rules $\neg$, $\wedge$ and $\vee$ are standard. The intuition behind rules $[\cdot]$ and $\langle\cdot\rangle$ reflects the semantics of public announcements. The model (M, w) satisfies $[\psi]\varphi$ if and only if (M, w) satisfies $\neg\psi$ or it satisfies ψ and the restricted model satisfies φ. The rule $\langle\cdot\rangle$ is the dual of rule $[\cdot]$. However, rules for the knowledge operators are quite different from their correspondent in EL. When we create a new world x in $M|\psi_0|\psi_1|\ldots|\psi_k$ by the rule $\widehat{\mathbf{K}}$, we must be sure that this world can consistently belong to M and that it is not deleted by one of the announcements of

the sequence. In the case of rules **K**, **4** and **5**, the world x' was already created, but possibly in a model restricted by a different sequence of announcements. We therefore must be sure that x' would also be present in a model generated by the sequence of announcements we have in hand.

The tableau method for K-PAL consists on rules $\neg$, $\wedge$, $\vee$, **K**, $\widehat{\mathbf{K}}$, $[\cdot]$ and $\langle\cdot\rangle$. For KT-PAL, we also have rule **T**. For S4-PAL, we have all rules for KT-PAL plus rule **4**. And for S5-PAL, we have all rules for S4-PAL plus rule **5**.

Definition 7. *Let $b = (\Lambda, \Sigma)$ be a branch. The set Λ is* blatantly inconsistent *if and only if $\{(\psi^k, x, \varphi), (\psi^k, x, \neg\varphi)\} \subseteq \Lambda$ or $\{(\psi^k, x, p), (\chi^\ell, x, \neg p)\} \subseteq \Lambda$. The branch b is* closed *if and only if Λ is blatantly inconsistent. The branch b is* open *if and only if it is not closed. A tableau is* closed *if and only if all its branches are closed. A tableau is* open *if and only if it has at least one open branch.*

Note that (ψ^k, x, p) and $(\chi^\ell, x, \neg p)$ are inconsistent because boolean formulas are preserved through announcements.

Example 1. Consider the formula $[p\wedge\neg K_a p]\neg(p\wedge\neg K_a p)$. In Figure 1 the tableau method is used to show its validity in K-PAL. Note that in this formula the announcement corresponds to the so-called Moore sentence [6]: "p is true and agent a does not know it". When it is true and publicly announced, all the agents, in particular agent a, become aware of it. Then the sentence becomes false just after being announced. ∎

1.	$\epsilon, 0, \neg[p \wedge \neg K_a p]\neg(p \wedge \neg K_a p)$		
2.	$\epsilon, 0, p \wedge \neg K_a p$		$(\langle\cdot\rangle : 1)$
3.	$p \wedge \neg K_a p, 0, \neg\neg(p \wedge \neg K_a p)$		$(\langle\cdot\rangle : 1)$
4.	$p \wedge \neg K_a p, 0, p \wedge \neg K_a p$		$(\neg : 3)$
5.	$p \wedge \neg K_a p, 0, p$		$(\wedge : 4)$
6.	$p \wedge \neg K_a p, 0, \neg K_a p$		$(\wedge : 4)$
7.	$\epsilon, 1, p \wedge \neg K_a p$	$(a, 0, 1) \in \Sigma$	$(\widehat{\mathbf{K}} : 6)$
8.	$p \wedge \neg K_a p, 1, \neg p$		$(\widehat{\mathbf{K}} : 6)$
9.	$\epsilon, 1, p$		$(\wedge : 7)$
10.	$\epsilon, 1, \neg K_a p$		$(\wedge : 7)$
	closed		$(8, 9)$

Fig. 1. Closed tableau for the formula $[p \wedge \neg K_a p]\neg(p \wedge \neg K_a p)$.

Theorem 1 (Soundness and completeness). *For $C \in \{\mathrm{K}, \mathrm{KT}, \mathrm{S4}, \mathrm{S5}\}$, there is a closed C-*PAL*-tableau for $\neg\varphi$ if and only if φ is C-*PAL*-valid.*

Proof (Sketch:). From the left to the right. We prove that if φ is satisfiable, then there is no closed tableau for φ. We do this by showing that all tableau rules preserve satisfiability. In other words, let T^i be a tableau for a given formula

that contains a branch $b = (\Lambda, \Sigma)$, we show that if b is satisfiable, then the set of branches B generated by any tableau rule has also at least one satisfiable branch (details are omitted).

From the right to the left. We show that if a saturated tableau for a given formula φ is open, then φ is satisfiable. Suppose that T^∞ is an open saturated tableau for a S5-PAL-formula φ. Then, it contains at least one open branch $b = \langle \Lambda, \Sigma \rangle$. We use this branch to construct an epistemic structure $M = \langle W, R, V \rangle$ that satisfies φ as follows:

- $W = \{x \in \mathbb{N} \mid x \text{ occurs in } \Lambda\}$;
- $R_a =$ reflexive, transitive and symmetric closure of $\{(x, x') \mid (a, x, x') \in \Sigma\}$; and
- $V_p = \{x \mid (\psi^k, x, p) \in \Lambda \text{ for some } \psi^k\}$.

And we also define a function $f(x) = x$ for all x occurring in Λ.

Clearly, W is a non-empty set, R_a is an equivalence relation, V_p assigns a subset of W to each proposition that appears on the tableau and if $(a, x, x') \in \Sigma$, then $f(x')R_a f(x)$. Thus, we now show that for all labelled formulas $\lambda = (\psi^k, x, \varphi) \in \Lambda$, we have $\mathcal{P}(\lambda)$ defined as follows:

$$\mathcal{P}(\lambda) = \begin{cases} M|\psi^0, f(x) \models \psi_1 & \text{and} \\ \quad \vdots & \\ M|\psi^{k-1}, f(x) \models \psi_k & \text{and} \\ M|\psi^k, f(x) \models \varphi. & \end{cases}$$

We do this by induction on the structure of λ (details are omitted). ∎

4 Decision procedures for public announcement logics

In the way the method is defined, redundant applications of tableau rules are allowed. In particular, they can be applied indefinitely often. Therefore it may never stop. In this section we define strategies for the application of the tableau rules. They are inspired by the "tableau construction" defined in [12].

When a set of labelled formulas Λ is not saturated under one or more tableau rules, we say that $\lambda \in \Lambda$ is a *witness* to this fact if the given rule, or rules, were still not applied to λ. For convenience, we further use notation $\Lambda(x)$ for the set of *labelled formulas of* x, defined by $\{(\psi^k, \varphi) \mid (\psi^k, x, \varphi) \in \Lambda\}$. And we also use notation $\Lambda(x, a)$ for the set of *labelled formulas of agent* a *in* x, defined by $\{(\psi^k, K_a\varphi) \mid (\psi^k, x, K_a\varphi) \in \Lambda\} \cup \{(\psi^k, \neg K_a\varphi) \mid (\psi^k, x, \neg K_a\varphi) \in \Lambda\}$.

The strategy defined below is for S5-PAL. It constructs a tree of nodes s whose labels are tableau branches $L(s) = (\Lambda_s, \Sigma_s)$ generated by the application of tableau rules to their antecedents.

Strategy 1 *Let $\varphi_0 \in \mathcal{L}_{\mathrm{PAL}}$ be given. Construct a tree as follows.*

1. *Start with a single node s_0 (the root of the tree) whose label is the initial branch for φ_0, i.e., the pair $L(s_0) = (\Lambda_{s_0}, \Sigma_{s_0})$, where $\Lambda_{s_0} = \{(\epsilon, 0, \varphi_0)\}$ and $\Sigma_{s_0} = \emptyset$.*

2. *Repeat until neither step 2(a) nor step 2(b) below applies.*
   (a) World saturation: *if s is a leaf with label $L(s)$ such that $L(s)$ is open and not saturated under rules $\neg$, $\wedge$, rt, $\langle\cdot\rangle$, $\vee$, $[\cdot]$, $\mathbf{K}$, $\mathbf{4}$ and $\mathbf{5}$, and $\lambda \in \Lambda_s$ is a witness to this fact, then do:*
      i. *if $\lambda = (\psi^k, x, \neg\neg\varphi)$ then create a successor s' such that $\Lambda_{s'} = \Lambda_s \cup \{(\psi^k, x, \varphi)\}$ and $\Sigma_{s'} = \Sigma_s$. And then go to step 2.*
      ii. *if $\lambda = (\psi^k, x, \varphi_1 \wedge \varphi_2)$ then create a successor s' such that $\Lambda_{s'} = \Lambda_s \cup \{(\psi^k, x, \varphi_1), (\psi^k, x, \varphi_2)\}$ and $\Sigma_{s'} = \Sigma_s$. And then go to step 2.*
      iii. *if $\lambda = (\psi^k, x, K_a\varphi)$ then create a successor s' such that $\Lambda_{s'} = \Lambda_s \cup \{(\psi^k, x, \varphi)\}$ and $\Sigma_{s'} = \Sigma_s$. And then go to step 2.*
      iv. *if $\lambda = (\psi^k, x, \neg[\varphi_1]\varphi_2)$ then create a successor s' such that $\Lambda_{s'} = \Lambda_s \cup \{(\psi^k, x, \varphi_1), (\psi^k\varphi_1, x, \varphi_2)\}$ and $\Sigma_{s'} = \Sigma_s$. And then go to step 2.*
      v. *if $\lambda = (\psi^k, x, \neg(\varphi_1 \wedge \varphi_2))$ then create two successors s_1 and s_2 such that $\Lambda_{s_1} = \Lambda_s \cup \{(\psi^k, x, \neg\varphi_1)\}$ and $\Sigma_{s_1} = \Sigma_s$, and $\Lambda_{s_2} = \Lambda_s \cup \{(\psi^k, x, \neg\varphi_2)\}$ and $\Sigma_{s_2} = \Sigma_s$. And then go to step 2.*
      vi. *if $\lambda = (\psi^k, x, [\varphi_1]\varphi_2)$ then create two successors s_1 and s_2 such that $\Lambda_{s_1} = \Lambda_s \cup \{(\psi^k, x, \neg\varphi_1)\}$ and $\Sigma_{s_1} = \Sigma_s$, and $\Lambda_{s_2} = \Lambda_s \cup \{(\psi^k, x, \varphi_1), (\psi^k\varphi_1, x, \varphi_2)\}$ and $\Sigma_{s_2} = \Sigma_s$. And then go to step 2.*
      vii. *if $\lambda = (\psi^k, x, K_a\varphi)$ and $(a, x, x') \in \Sigma$, then for each $i \in \{0, 1, \ldots, k-1\}$, create a successor s_i such that $\Lambda_{s_i} = \Lambda_s \cup \{(\psi^j, x', \psi_{j+1}) \mid 0 \leq j < i\} \cup \{(\psi^i, x', \neg\psi_{i+1})\}$ and $\Sigma_{s_i} = \Sigma_s$, and also create a successor node s_k such that $\Lambda_{s_k} = \Lambda_s \cup \{(\psi^j, x', \psi_{j+1}) \mid 0 \leq j < k\} \cup \{(\psi^k, x', \varphi)\}$ and $\Sigma_{s_k} = \Sigma_s$. And then go to step 2.*
      viii. *if $\lambda = (\psi^k, x, K_a\varphi)$ and $(a, x, x') \in \Sigma$, then for each $i \in \{0, 1, \ldots, k-1\}$, create a successor s_i such that $\Lambda_{s_i} = \Lambda_s \cup \{(\psi^j, x', \psi_{j+1}) \mid 0 \leq j < i\} \cup \{(\psi^i, x', \neg\psi_{i+1})\}$ and $\Sigma_{s_i} = \Sigma_s$, and also create a successor node s_k such that $\Lambda_{s_k} = \Lambda_s \cup \{(\psi^j, x', \psi_{j+1}) \mid 0 \leq j < k\} \cup \{(\psi^k, x', K_a\varphi)\}$ and $\Sigma_{s_k} = \Sigma_s$. And then go to step 2.*
      ix. *if $\lambda = (\psi^k, x, K_a\varphi)$ and $(a, x', x) \in \Sigma$, then for each $i \in \{0, 1, \ldots, k-1\}$, create a successor s_i such that $\Lambda_{s_i} = \Lambda_s \cup \{(\psi^j, x', \psi_{j+1}) \mid 0 \leq j < i\} \cup \{(\psi^i, x', \neg\psi_{i+1})\}$ and $\Sigma_{s_i} = \Sigma_s$, and also create a successor node s_k such that $\Lambda_{s_k} = \Lambda_s \cup \{(\psi^j, x', \psi_{j+1}) \mid 0 \leq j < k\} \cup \{(\psi^k, x', K_a\varphi)\}$ and $\Sigma_{s_k} = \Sigma_s$. And then go to step 2.*
   (b) Create a new world: *if s is a leaf with label $L(s)$ such that $L(s)$ is open, world-saturated and not saturated under rule $\widehat{\mathbf{K}}$ and $\lambda = (\psi^k, x, \neg K_a\varphi)$ is a witness to this fact, then do steps i, ii and iii below. And then go to step 2.*
      i. *generate a new natural number x' that does not appear in Σ_s and create a label $L' = (\Lambda', \Sigma')$, where $\Lambda' = \{(\psi^j, x', \psi_{j+1}) \mid 0 \leq j < k\} \cup \{(\psi^k, x', \neg\varphi)\}$ and $\Sigma' = \{(a, x, x')\}$.*
      ii. *if there is a sequence of natural numbers $y_0, y_1, \ldots, y_n$ such that $y_n = x$ and for all $0 \leq i < n$, $(a, y_i, y_{i+1}) \in \Sigma_s$ and $\Lambda_s(x, a) = \Lambda_s(y_0, a)$ and $\Lambda' \subseteq \Lambda_s(y_1)$, then create a successor s' such that $\Lambda_{s'} = \Lambda_s$ and $\Sigma_{s'} = \Sigma_s \cup \{(a, x, y_1)\}$.*

iii. if step 2(b)ii does not apply, then create a successor s' such that $\Lambda_{s'} = \Lambda_s \cup \Lambda'$ and $\Sigma_{s'} = \Sigma_s \cup \Sigma'$.

3. If s is a leaf and its label $L(s)$ is open, then return `true`*, else return* `false`*.*

Simple modifications of Strategy 1 above give us strategies for the other logics we consider here. A strategy for S4-PAL can be obtained by removing step 2(a)ix. By removing steps 2(a)viii and 2(a)ix, we obtain a strategy for KT-PAL. And by removing steps 2(a)iii, 2(a)viii and 2(a)ix we obtain a strategy for K-PAL.

Note that step 2(b)ii has a *loop test*. This is crucial to guarantee that the process halts for S4-PAL and S5-PAL. Before applying rule $\widehat{\mathbf{K}}$, which means that a new "world" x' will be created, it verifies if there is no loop.

We continue by proving termination. After that we prove soundness and completeness for S5-PAL only. Proofs for the other logics are similar and left to the reader. We first need a definition and a lemma.

Definition 8. *The* set of labelled sub-formulas of φ, $\mathrm{Sub}(\varphi)$, *and* the set of labelled sub-formulas of φ and its negations, $\mathrm{Sub}^+(\varphi)$, *are recursively defined as follows:*

$$\begin{aligned}
\mathrm{Sub}(p) &:= \{(\epsilon, p)\} \\
\mathrm{Sub}(\neg\varphi) &:= \mathrm{Sub}(\varphi) \cup \{(\epsilon, \neg\varphi)\} \\
\mathrm{Sub}(\varphi \wedge \psi) &:= \mathrm{Sub}(\varphi) \cup \mathrm{Sub}(\psi) \cup \{(\epsilon, \varphi \wedge \psi)\} \\
\mathrm{Sub}(K_a\varphi) &:= \mathrm{Sub}(\varphi) \cup \{(\epsilon, K_a\varphi)\} \\
\mathrm{Sub}([\psi]\varphi) &:= \mathrm{Sub}(\psi) \cup \{(\psi\chi^k, \varphi') \mid (\chi^k, \varphi') \in \mathrm{Sub}(\varphi)\} \cup \{(\epsilon, [\psi]\varphi)\}
\end{aligned}$$

$$\mathrm{Sub}^+(\varphi) := \mathrm{Sub}(\varphi) \cup \{(\psi^k, \neg\varphi') \mid (\psi^k, \varphi') \in \mathrm{Sub}(\varphi)\}$$

Lemma 1.

1. $|\mathrm{Sub}(\varphi_0)| \leq \mathrm{len}(\varphi_0)$.
2. *For all* $(\psi^k, \varphi) \in \mathrm{Sub}(\epsilon, \varphi_0)$, $k \leq \mathrm{len}(\varphi_0)$.
3. $|\mathrm{Sub}^+(\varphi)| \leq 2 \times \mathrm{len}(\varphi)$.

Items 1 and 2 are proved in [9] and 3 is an obvious consequence of them.

Theorem 2. *For all $\varphi \in \mathcal{L}_{\mathrm{PAL}}$, Strategy 1 creates a finite tree for φ.*

Proof. Let a $\mathcal{L}_{\mathrm{PAL}}$-formula φ be given. Because $\mathrm{Sub}^+(\varphi)$ is finite, each step generates a finite number of immediate successors. Then, by the fact that the initial tree for φ is a single node (and, in particular, it is finite), each step of the strategy generates a finite tree.

We now show that each step is applied finitely often. Let $\mathrm{len}(\varphi) = n$. By Lemma 1, the number of labelled sub-formulas of φ and its negations is bounded by $2n$. Then after $2n$ applications of step 2(a) all the leafs of the tree are world-saturated. This means that there can be at most $2n$ applications of step 2(a) between two subsequent applications of step 2(b).

Now, note that there exists at most 2^{2n} different subsets of $\mathrm{Sub}^+(\varphi)$. This means that the loop tests can fail at most 2^{2n} times. It immediately follows that step 2(b) can be applied at most $\mathcal{O}(2^n)$ times. Therefore, Strategy 1 always creates a finite tree and thus always halts. ∎

Theorem 3. *For all $\varphi \in \mathcal{L}_{\mathrm{PAL}}$, φ is* S5-PAL*-satisfiable if and only if Strategy 1 for φ returns* `true`*.*

Proof. From the left to the right. We show that if φ is S5-PAL-satisfiable, then the tree for φ generated by Strategy 1 will have at least one leaf whose label is an open tableau branch. We do this by showing that all steps preserve satisfiability. This proof is along the lines of the first part of the proof of Theorem 1. The only remarkable difference is the step 2(b)ii: suppose that $(\psi^k, x, \neg K_a\varphi) \in \Lambda_s$ and that the loop test succeeds. This means that there is a sequence y_0,y_1,...,y_n such that $y_n = x$ and for all $0 \leq i < n$, $(a, y_i, y_{i+1}) \in \Sigma_s$, and s has a successor s' such that $\Lambda_{s'} = \Lambda_s$ and $\Sigma_{s'} = \Sigma_s \cup \{(a, x, y_1)\}$. We then consider the *unfolded* tableau branch $L' = (\Lambda', \Sigma')$ such that $\Lambda' = \Lambda_{s'} \cup \{(\chi^\ell, x', \varphi') \mid (\chi^\ell, y_1, \varphi') \in \Lambda_{s'}\}$ and $\Sigma' = (\Sigma_s \setminus \{(a, x, y_1)\}) \cup \{(a, x, x'), (a, x', y_2)\}$. Clearly, L' is satisfiable if and only if $L(s')$ is satisfiable. By hypothesis, there is an epistemic structure $M = \langle W, R, V\rangle$ and a function $f : \mathbb{N} \to W$ that satisfy $L(s)$. Then there exists $w \in W^{\psi^k}$ such that $f(x)R_a^{\psi^k}w$. We thus consider the function $f' : \mathbb{N} \to W$ such that for all integer x that occur in Λ', $f'(x) := f(x)$ and $f'(x') := w$. Therefore $L(s')$ is satisfiable.

From the right to the left. If Strategy 1 for φ returns `true`, then the tree for φ has a leaf s such that $L(s)$ is open and saturated. Then we use this node to construct a model $M = \langle W, R, V\rangle$ that satisfies φ as follows. W contains all x that appear in Σ_s; R is the reflexive, transitive and symmetric closure of all triples $(a, x, x') \in \Sigma_s$; and each V_p contains all x such that $(\psi^k, x, p) \in \Lambda_s$ for some ψ^k. We then proceed by induction on the length of labelled formulas where the induction hypothesis is: if $L(s)$ is an open saturated branch that contains (ψ^k, x, φ') and $\mathrm{len}(\psi^k, x, \varphi') < n$, then $M|\psi^0, x \models \psi_1, \ldots, M|\psi^{k-1}, x \models \psi_k$, and $M|\psi^k, x \models \varphi'$. This is done along the lines of the second part of the proof of Theorem 1. The details are left to the reader. ∎

The depth of the tree created in Strategy 1 is exponential on the size of the input formula. Therefore, this algorithm is not optimal. Below, we present optimal strategies for logics K-PAL and KT-PAL.

Similarly to the "tableau construction" defined in [12], instead of labelling the nodes of the tree with entire tableau branches, in our next strategy, node labels contain formulas of only one world x. Hence, we now use pairs of the form $\lambda = (\psi^k, \varphi)$ that, for convenience, are called *labelled formulas* as well (note that x is no longer necessary). But our algorithm differs from that of [12] in a crucial point: suppose that $L(s)$ contains the formula $(\psi^k, K_a\varphi)$. When an a-successor node s' of s is created, one cannot immediately add the labelled formula (ψ^k, φ) to $L(s')$. The reason is that the world s' can have being deleted by some announcement in the sequence ψ^k (Cf. tableau rules **K**, **4** and **5**). Then,

in step 2(b), before adding this labelled formula to $L(s')$, the algorithm "verifies" that each ψ_i is true in s'. This is implemented with an auxiliary set $\Gamma_{s'}$.

Strategy 2 *Let $\varphi_0 \in \mathcal{L}_{\mathrm{PAL}}$ be given. Construct a tree as follows.*

1. *Start with a single node s_0 (the root of the tree) whose label is the pair $L(s_0) = (\Lambda_{s_0}, \Gamma_{s_0})$, where $\Lambda_{s_0} = \{(\epsilon, \varphi_0)\}$ and $\Gamma_{s_0} = \emptyset$.*
2. *Repeat until neither (a) nor (b) below applies:*
   (a) Local saturation: *if s is a leaf with label $L(s)$ such that $L(s)$ is open and not saturated under rules $\neg$, $\wedge$, $\vee$, $\mathbf{K}$ and rt, and $\lambda \in \Lambda_s$ is a witness to this fact, then do:*
      i. if $\lambda = (\psi^k, \neg\neg\varphi) \in \Lambda_s$ then create a successor s' such that $\Lambda_{s'} = \Lambda_s \cup \{(\psi^k, \varphi)\}$ and $\Gamma_{s'} = \Gamma_s$. And then go to step 2.
      ii. if $\lambda = (\psi^k, \varphi_1 \wedge \varphi_2) \in \Lambda_s$ then create successor s' such that $\Lambda_{s'} = \Lambda_s \cup \{(\psi^k, \varphi_1), (\psi^k, \varphi_2)\}$ and $\Gamma_{s'} = \Gamma_s$. And then go to step 2.
      iii. if $\lambda = (\psi^k, K_a\varphi) \in \Lambda_s$ then create successor s' such that $\Lambda_{s'} = \Lambda_s \cup \{(\psi^k, \varphi)\}$ and $\Gamma_{s'} = \Gamma_s$. And then go to step 2.
      iv. if $\lambda = (\psi^k, \neg[\varphi_1]\varphi_2) \in \Lambda_s$ then create a successor s' such that $\Lambda_{s'} = \Lambda_s \cup \{(\psi^k, \varphi_1), (\psi^k\varphi_1, \varphi_2)\}$ and $\Gamma_{s'} = \Gamma_s$. And then go to step 2.
      v. if $\lambda = (\psi^k, \neg(\varphi_1 \wedge \varphi_2)) \in \Lambda_s$ then create two successors s_1 and s_2 such that $\Lambda_{s_1} = \Lambda_s \cup \{(\psi^k, \varphi_1)\}$ and $\Gamma_{s_1} = \Gamma_s$, and $\Lambda_{s_2} = \Lambda_s \cup \{(\psi^k, \varphi_2)\}$ and $\Gamma_{s_2} = \Gamma_s$. And then go to step 2.
      vi. if $\lambda = (\psi^k, [\varphi_1]\varphi_2) \in \Lambda_s$ then create two successors s_1 and s_2 such that $\Lambda_{s_1} = \Lambda_s \cup \{(\psi^k, \neg\varphi_1)\}$ and $\Gamma_{s_1} = \Gamma_s$, and $\Lambda_{s_2} = \Lambda_s \cup \{(\psi^k, \varphi_1), (\psi^k\varphi_1, \varphi_2)\}$ and $\Gamma_{s_2} = \Gamma_s$. And then go to step 2.
      vii. if $\lambda = (\psi^k, \varphi) \in \Gamma_s$ then for each $i \in \{0, 1, \ldots, k-1\}$, create a successor s_i such that $\Lambda_{s_i} = \Lambda_s \cup \{(\psi^j, \psi_{j+1}) \mid 0 \leq j < i\} \cup \{(\psi^i, \neg\psi_{i+1})\}$ and $\Gamma_{s_i} = \Gamma_s \setminus \{\lambda\}$, and also create a successor s_k such that $\Lambda_{s_k} = \Lambda_s \cup \{(\psi^j, \psi_{j+1}) \mid 0 \leq j < k\} \cup \{(\psi^k, \varphi)\}$ and $\Gamma_{s_k} = \Gamma_s \setminus \{\lambda\}$. And then go to step 2.
   (b) Create new worlds: *if s is a leaf with label $L(s)$ which is local saturated and not saturated under rule $\widehat{\mathbf{K}}$, then for each labelled formula of the form $\lambda = (\psi^k, \neg K_a\varphi) \in \Lambda_s$ that is witness to this, create an a-successor s' such that $\Lambda_{s'} = \{(\psi^j, \psi_{j+1}) \mid 0 \leq j < k\} \cup \{(\psi^k, \neg\varphi)\}$ and $\Gamma_{s'} = \{(\chi^{k'}, \varphi') \mid \{(\chi^{k'}, K_a\varphi')\} \in \Lambda_s\})$. And then go to step 2.*
   (c) Mark nodes: *if the node s with label $L(s)$ is not marked* `sat`*, then mark it* `sat` *if either:*
      – *Λ_s is not local saturated and one of its successor is marked* `sat`*;*
      – *Λ_s is local saturated, it is not blatantly inconsistent and Λ_s does not contain labelled formulas of the form $(\psi^k, \neg K_a\varphi)$; or*
      – *Λ_s is local saturated and s has successors and all of them are marked* `sat`*.*
3. *If the root of the tree is marked* `sat`*, then return* `true`*, else return* `false`*.*

The strategy above can be modified for deal with other two logics we consider here. For K-PAL we remove step 2(a)iii, and for S4-PAL, we replace step 2(b) by the following:

2(b') Create new worlds: *if s is a leaf with label $L(s)$ which is local saturated and not saturated under rule $\widehat{\mathbf{K}}$, then for each labelled formula of the form $\lambda = (\psi^k, \neg K_a\varphi) \in \Lambda_s$ that is a witness to this, then do steps i, ii and iii below. And then go to step 2.*

i. create a label $L' = (\Lambda', \Gamma')$, where $\Lambda' = \{(\psi^j, \psi_{j+1}) \mid 0 \leq j < k\} \cup \{(\psi^k, \neg\varphi)\}$ and $\Gamma' = \{(\chi^{k'}, \varphi'), (\chi^{k'}, K_a\varphi') \mid (\chi^{k'}, K_a\varphi') \in \Lambda_s\}$).

ii. if there is no node s'' in the path from the root to s such that $L_{s''} = L'$, then create an a-successor node s' with label $L(s') = L'$.

iii. if step 2(b)ii does not apply, then create an a-arrow to the node s'' such that $L(s'') = L'$.

Note that we also have a loop test in step 2(b')ii. The idea is the same as in Strategy 1, but here, we also compare the sets Γ. We also remark that it is not possible to use the same idea to define a strategy for S5-PAL. We address this question in Section 6. In the sequel, we prove termination, soundness and completeness for S4-PAL only. We leave other cases to the reader.

Theorem 4. *For all $\varphi \in \mathcal{L}_{\mathrm{PAL}}$, Strategy 2 creates a finite tree for φ.*

Proof. The proof is essentially the same as for Theorem 2. ∎

Theorem 5. *For all $\varphi \in \mathcal{L}_{\mathrm{PAL}}$, φ is* S4-PAL*-satisfiable if and only if Strategy 2 for φ returns* `true`*.*

Proof. From the left to the right. We show that if φ is S4-PAL-satisfiable, then the tree for φ generated by Strategy 2 has its root marked `sat`. We do this by showing that all steps preserve satisfiability. This proof is along the lines of the first part of proof of Theorem 1. The differences are steps 2(a)vii and 2(b'). Note that step 2(a)vii performs essentially the same task as steps 2(a)vii and 2(a)viii of Strategy 1. So their proof of soundness is very similar. For step 2(b'), note that it is similar to step 2(b) of Strategy 1. So its proof of soundness is also similar. We omit details here.

From the right to the left. If Strategy 2 for φ returns `true`, then the root s_0 is marked `sat`. Then, there is a sub-tree such that all its nodes are marked `sat`. We use this tree to construct a model $M = \langle W, R, V\rangle$ that satisfies φ in the following way. W contains all s in the sub-tree such that s is local saturated; each R_a is the reflexive and transitive closure of pairs (s, s') of nodes in W such that s' is a descendent of an a-successor of s; and each V_p contains all $s \in W$ such that $(\psi^k, p) \in \Lambda_s$ for some ψ^k. The proof continues along the lines of the second part of the proof of Theorem 1. We omit details here. ∎

We continue by showing computational complexity of Strategy 2 for K-PAL and KT-PAL. Remark that no optimal procedure is achieved for S4-PAL. We discuss this in Section 6.

Theorem 6 (Complexity). *The tableau system for* K-PAL *and* KT-PAL *can be implemented in polynomial space.*

Proof. We first show that the height of the trees, generated by Strategy 2, are polynomial in $\mathrm{len}(\varphi)$. The tree construction starts with the root node s_0 whose label is $L(s_0) = (\{\lambda_0\}, \emptyset)$. Suppose that $\mathrm{len}(\lambda_0) = n$. Note that by Lemma 1 step 2(a) can be applied at most $2n$ times before reach a node s such that Λ_s is either closed or saturated. Also note that at this stage Γ_s is empty. Now, suppose that Λ_s is local saturated. Also suppose that s' is a local saturated descendent of an a-successor of s. Note that the K-modal depth of $\Lambda_{s'}$ is less than that of Λ_s. Because the number of K_a operators in φ is at most n, the root of the tree can have at most n a-descendents in the same branch of the tree. From this fact and the observation made before, it follows that the tree has height at most $\mathcal{O}(n^2)$.

We now prove that a depth first exploration of the trees can be made using a polynomial amount of memory. To see this, remember that by Lemma 1 we have that for all nodes s in the tree, $|L(s)| \leq 4n$. Then we can use a vector of $4n$ bits to encode the label of each node in the tree. We do this by setting to 1 the bit that corresponds to the formulas that are present in $L(s)$. Each step of Strategy 2 can produce at most $2n$ different immediate successors. Then, for each node, we can use a vector of $4n^2$ bits to memorise all the choices to be explored after the backtrack. It follows that we need at most $\mathcal{O}(n^5)$ bits of memory to explore the entire tree. ∎

5 A tableau method for arbitrary announcement logic

Consider the extension of public announcement logic proposed in [10] wherein we can express what becomes true, whether known or not, without explicit reference to announcements realising that. For example, when p is true, it becomes known by announcing it:

$$\langle p \rangle K_a p$$

We can also describe 'there is an announcement after which the agent knows p' straightforwardly as

$$\Diamond K_a p$$

In case p is false, we can achieve $\Diamond K_a \neg p$ instead. The formula $\Diamond(K_a p \vee K_a \neg p)$ is valid. We call the logic *arbitrary public announcement logic*. For more information, see [10, 13].

Definition 9 (Language and semantics). *The language $\mathcal{L}_{\mathrm{APAL}}$ of arbitrary public announcement logic is inductively defined as*

$$\varphi ::= p \mid \neg\varphi \mid \varphi \wedge \varphi \mid K_a\varphi \mid [\varphi]\varphi \mid \Box\varphi$$

where $a \in A$ and $p \in P$. The extra clause needed for the semantics is as follows (note the restriction to the language of PAL*):*

$$M, w \models \Box\varphi \quad \textit{iff} \quad \textit{for all } \psi \in \mathcal{L}_{\mathrm{PAL}},\ M, w \models [\psi]\varphi$$

For $\Box\varphi$, read 'after every public announcement, φ is true'. The dual of $\Box$ is $\Diamond$. For $\Diamond\varphi$, read 'there is an announcement after which φ'. For the semantics of the dual operator, we have that: $M, w \models \Diamond\varphi$ iff there is a $\psi \in \mathcal{L}_{\mathrm{PAL}}$ such that $M, w \models \langle\psi\rangle\varphi$.

Similarly as done in Section 2, we define K-APAL, KT-APAL, S4-APAL and S5-APAL.

Example 2. A valid formula of the logic is $\Diamond(K_a p \vee K_a \neg p)$. To prove this, let (M, w) be arbitrary. Either $M, w \models p$ or $M, w \models \neg p$. In the first case, $M, w \models \Diamond(K_a p \vee K_a \neg p)$ because $M, w \models \langle p\rangle(K_a p \vee K_a \neg p)$ – the latter is true because $M, w \models p$ and $M|p, w \models K_a p$; in the second case, we analogously derive $M, w \models \Diamond(K_a p \vee K_a \neg p)$ because $M, w \models \langle\neg p\rangle(K_a p \vee K_a \neg p)$. ∎

We now provide an extension of the tableau method for public announcement logic to a tableau method for arbitrary public announcement logic. We reuse labelled formulas, skeleton and branch as introduced in Definitions 4 and 5, as well as the notions of closed and open branch as in Definition 7.

Definition 10 (Tableau (continuation)). *A* tableau *for the formula $\varphi \in \mathcal{L}_{\mathrm{APAL}}$ is defined as in Definition 6. The tableau rules are the same, plus the following ones.*

$\Box$: *If $(\psi^k, n, \Box\varphi) \in \Lambda$, then $B = \{\langle \Lambda \cup \{(\psi^k : n : [\chi]\varphi)\}, \Sigma\rangle\}$ for any $\chi \in \mathcal{L}_{\mathrm{PAL}}$.*
$\Diamond$: *If $(\psi^k, n, \neg\Box\varphi) \in \Lambda$, then $B = \{\langle \Lambda \cup \{(\psi^k : n : \neg[p]\varphi)\}, \Sigma\rangle\}$ for some $p \in P$ that does not occur in Λ.*

These rules are similar to Smullyan's tableau rules for closed first-order formulas [14, 15]. They reflect that the operator $\Box$ quantifies over announcements. In tableau rule $\Box$, this operator is eliminated by replacing it by an arbitrary $\mathcal{L}_{\mathrm{PAL}}$-formula. Tableau rule $\Diamond$ is more curious though: instead of replacing the operator by an announcement of a $\mathcal{L}_{\mathrm{PAL}}$-formula ψ, we replace it by an announcement of a new propositional letter. The intuitive argument here is the following. Since this new propositional letter does not occur in the branch, we are free to give it an arbitrary interpretation to represent a specific restriction in the model. In this way, we make the calculus simpler because it is not necessary to make a 'good choice' at the moment of the application of rule $\Diamond$.

Example 3. Consider the formula $[\Diamond K_a p]K_a p$. Note that it is valid in S5-APAL since its announcement corresponds to the sentence: 'there is an announcement after which agent a knows that p'. That is, it is publicly announced that p can be known. This means that p is true and thus now agent a knows it. In Figure 2 we use the tableau method to show that this formula is S5-APAL-valid. ∎

Theorem 7 (Soundness and completeness (continuation)). *For all $\varphi \in \mathcal{L}_{\mathrm{APAL}}$, there is a closed tableau for $\neg\varphi$ if and only if φ is* S5-APAL*-valid.*

Proof. This is an easy extention to proof of theorem 1 (details are omitted).

1.	$\epsilon, 0, \neg[\neg\Box\neg K_a p]K_a p$		
2.	$\epsilon, 0, \neg\Box\neg K_a p$		$(\langle\cdot\rangle : 1)$
3.	$\neg\Box\neg K_a p, 0, \neg K_a p$		$(\langle\cdot\rangle : 1)$
4.	$\epsilon, 0, \neg[p]\neg K_a p$		$(\Diamond : 2)$
5.	$\epsilon, 0, p$		$(\langle\cdot\rangle : 4)$
6.	$p, 0, \neg\neg K_a p$		$(\langle\cdot\rangle : 4)$
7.	$p, 0, K_a p$		$(\neg : 6)$
8.	$\epsilon, 1, \neg\Box\neg K_a p$	$(a, 0, 1) \in \Sigma$	$(\widehat{\mathbf{K}} : 3)$
9.	$\neg\Box\neg K_a p, 1, \neg p$		$(\widehat{\mathbf{K}} : 3)$
10.	$p, 1, p$		$(\mathbf{K} : 7)$
	closed		$(9, 10)$

Fig. 2. Closed tableau for the formula $[\Diamond K_a p]K_a p$.

This tableau method can be used for giving us a proof of semi-decidability of this logic.

Theorem 8. *The set of* S5-APAL*-valid formulas of* $\mathcal{L}_{\mathrm{APAL}}$ *is recursively enumerable.*

Proof. First note that the same argument used in the proof of Theorem 2 can be used to show that each tableau rule generates a finite tableau. Then, by completeness, we have that for all formulas φ, all closed tableaux for φ are finite. Then, consider a procedure that enumerates all pairs (φ, T) such that T is a closed tableau. For each pair, the procedure verifies if T is a tableau for $\neg\varphi$. When the checking is finished, it generates another pair and performs another round of checking, and so on ad infinitum. ∎

6 Related work and discussion

We considered versions of PAL where the underlying epistemic logic obeys combination of principles T, 4 and 5. We did not consider the axiom D ($K_a\varphi \to \neg K_a\neg\varphi$) alone, i.e., epistemic logics such as KD and KD45. The reason is that in both systems the axiom T ($K_a\varphi \to \varphi$) is derivable for any boolean formula φ. To see this, note that if we have axiom D, then $(K_a\varphi \wedge \neg\varphi) \to \langle\neg\varphi\rangle\bot$ is valid for any boolean formula φ.

Recently, another tableau method for S5-PAL was proposed by [16]. Apart from some aesthetical differences, this method is very similar to ours. However, no proof of decidability is provided.

Our Strategy 2 is based in the optimal strategy for EL presented in [12]. Note however that instead of our rule **5**, Halpern and Moses use a rule that propagates all formulas prefixed by K_a and $\widehat{K}_a$ operators to the a-successors. As this rule alone is not complete for S5, they also need to saturate the nodes under sub-formulas (which is called full propositional tableau). But note that such a rule would not be sound in our setting. For example, suppose that in node s with label $L(s)$ we have that $(\psi, \neg K_a\varphi) \in \Lambda_s$. Because it may be the

case that Λ_s also contains $(\epsilon, \neg\psi)$, we cannot add neither (ψ, φ) nor $(\psi, \neg\varphi)$ to Λ_s under the risk of making it blatantly inconsistent. Then, we cannot have our set of formulas saturated under sub-formulas in this way. An optimal strategy for S4-PAL seems to be impossible too. An example is the formula $\neg K_a p_0 \wedge K_a[q_1]K_a p_1 \wedge K_a[q_2]K_a p_2 \wedge \cdots \wedge K_a[q_i]K_a p_i$, for which Strategy 2 generates a tree containing a branch with 2^i different a-successors.

7 Conclusion

We provided a proof system for PAL with and without positive introspection that avoids translation to other languages as done by [9]. It also extends to APAL.

As mentioned before, proof systems for DELs are usually built from reduction axioms. In all cases, the dynamic operators can be eliminated and the formula is translated into a simpler language. However, the price is an exponentially larger formula to be evaluated. As proved here and also in [9], there are cases where this price is not mandatory. Therefore, one of the raised questions is whether direct methods, like the method presented here, can be also applied to the other languages. In this direction, we envisage some "natural" extensions to our approach. For instance, PAL with common knowledge.

Acknowledgements

Hans van Ditmarsch appreciate support from the NIAS (Netherlands Institute for Advanced Study in the Humanities and Social Sciences) project 'Games, Action, and Social Software' and the NWO (Netherlands Organisation for Scientific Research) Cognition Program for the Advanced Studies grant NWO 051-04-120.

Tiago de Lima is supported by the Programme Alßan, the European Union Programme of High Level Scholarships for Latin America, scholarship number E04D041703BR.

All the authors thank the anonymous reviewers for their useful comments.

References

1. Plaza, J.: Logics of public communications. In Emrich, M.L., et al., eds.: Proceedings of ISMIS. (1989) 201–216
2. Baltag, A., Moss, L. Solecki, S.: The logic of public announcements, common knowledge, and private suspicious. Technical Report SEN-R9922, Centrum voor Wiskunde en Informatica, Amsterdam, Netherlands (1999)
3. Gerbrandy, J.: Bisimulations on Planet Kripke. PhD thesis, University of Amsterdam, Amsterdam, Netherlands (1999)
4. Kooi, B.: Expressivity and completeness for public update logic via reduction axioms. Journal of Applied Non-Classical Logics (2007) To appear.
5. van Benthem, J., van Eijck, J., Kooi, B.: Logics for communication and change. Manuscript (2005)

6. van Ditmarsch, H., Kooi, B.: The secret of my success. Synthese (151) (2006) 201–232
7. van Ditmarsch, H., van der Hoek, W., Kooi, B.: Playing cards with Hintikka. Australasian Journal of Logic (3) (2005) 108–134
8. Herzig, A., De Lima, T.: Epistemic actions and ontic actions: A unified logical framework. In Sichman, J., et al., eds.: IBERAMIA-SBIA. Volume 4140 of Lecture Notes in Artificial Intelligence., Spriniger-Verlag (2006) 409–418
9. Lutz, C.: Complexity and succintness of public announcement logic. In Stone, P., Weiss, G., eds.: Proceedings of AAMAS. (2006) 137–144
10. Balbiani, P., van Ditmarsch, H., Herzig, A., De Lima, T.: What becomes true after arbitrary announcements. Technical Report OUCS-2006-06, Department of Computer Science, University of Otago, Dunedin, New Zealand (2006)
11. Fitting, M.: Proof Methods for Modal and Intuitionistic Logics. Reidel Publishing Company (1983)
12. Halpern, J., Moses, Y.: A guide to compleness and complexity for modal logics of knowledge and belief. Artificial Intelligence **54** (1992) 311–379
13. Balbiani, P., Baltag, A., van Ditmarsch, H., Herzig, A., Hoshi, T., de Lima, T.: What can we achieve by arbitrary announcements? A dynamic take on Fitch's knowability. (2007) submitted.
14. Smullyan, R.M.: First-Order Logic. Springer-Verlag (1968)
15. Letz, R.: Tableau methods for modal and temporal logics. In D'Agostino, M., et al., eds.: Handbook of Tableau Methods. Kluwer Academic Publishers (1999)
16. de Boer, M.: Praktische bewijzen in public announcement logica (Practical proofs in public announcement logic). Master's thesis, Department of Artificial Intelligence, University of Groningen (2006) in Dutch.