Defining and Model-Checking an Epistemic Temporal Logic with Changes of Observations

Aurèle Barrière

École Normale Supérieure de Rennes, France
aurele.barriere@ens-rennes.fr

Supervisors:
Aniello Murano
Bastien Maubert
Sasha Rubin

Università degli Studi di Napoli Federico II

September 11th, 2017 - December 11th, 2017

Abstract. We define a logic for systems with imperfect information, CTL*KΔ. It inherits operators from temporal and epistemic logics, as well as a new one that allows agents to dynamically change their observation of the system. We first define it for single-agent settings with synchronous perfect recall. We define an equivalent semantics, that is used later for model-checking the logic, and proved to be equivalent. We give an algorithm for model-checking. Finally, we extend our logic to multi-agent settings.

Keywords: Epistemic Temporal Logic, Model Checking, Changing Observations

1 Introduction

Epistemic Logics are well known to be a required formalism to describe and reason about knowledge in distributed systems. Distributed algorithms (also called *protocols*) involve agents without complete knowledge of the state of the system. Therefore, "Any logic of protocols must include as part of it a logic of knowledge", as said by Ladner and Reif in [8]. Such systems have been found to be useful for in many areas, including Game Theory and Artificial Intelligence. A popular extension of Epistemic Logics is to combine them with Temporal Logics. Reasoning about the evolution of agents' knowledge inside a system becomes possible. For each of these logics, model-checking (deciding if a formula is true in a given model) is an important problem, as it allows to confront the model of a system to its specification.

In these settings, agents are usually given a fixed observation for the whole evolution. Observations describe an agent's point of view, to model imperfect information. In this paper, to deal with dynamic changes of observations during the evolution in a system with imperfect information, we introduce a new logic, CTL*KΔ. To the best of our knowledge, this is the first time that such changes are studied. This logic includes branching-time temporal operators, epistemic operators, and a new one, Δ^o, to represent changes of observation. For instance, the formula $\Delta^o KAXp$ states that after changing to an observation o, the agent knows that, on the next step, the proposition p holds.

This logic could be useful for any system where agents can change their observational power of the system. For instance, in a scenario where there exists different "security levels" where different levels have access to different information. With our logic, it becomes possible to express statements such as "For an agent with initial observation o_1, there exists a point in time where, if the agent changes his observation to o_2, he knows whether or not some proposition holds" ($\Delta^{o_1} F(\Delta^{o_2}(Kp \vee K\neg p))$). Another main motivation to define such a logic is the work that has been done on Strategy Logic with Imperfect Information [2], an extension of Strategy Logic [4]. In this logic, agents can change observation when changing strategies. Before investigating it in the full framework of Strategy Logic, the natural first step was to study the interactions of observation changes, knowledge and time in a simpler setting, without the strategic aspects.

In Section 2, we first define the logic CTL*KΔ. To begin, we only define it for single-agent synchronous perfect recall settings. Then, in Section 3, we introduce an alternative, finitary semantics for the same formulas, that we later prove to be equivalent to the first one, and easier to model-check. In Section 4, we describe an algorithm to model-check a formula of CTL*KΔ. Finally, we extend the logic for multi-agent settings in Section 5.

Related Works Works on epistemic logics are numerous. The first ones started to investigate logics of knowledge in a static setting [13]. Later, many works have studied combinations of epistemic and temporal logics. Such logics are said to be Epistemic Temporal Logics [6].

In [14], the logic LTLK is defined and model-checked. We use the same k-tree structure in Section 5 to represent the knowledge of multiple agents.

Without our changes of observations, the combination of CTL with epistemic logics has been studied before [11][10]. CTL*, the extension of CTL, has also been studied with epistemic operators. CTL*K has been investigated and model-checked in [7][12][3], in both memoryless and perfect recall settings.

The possibility of dynamically changing observation has been introduced in Strategy Logic with Imperfect Information [2]. In this logic, an operator (a, x) allows to assign a strategy x to a player a. Strategies are defined with the operator $\ll x \gg^o$, where o is an observation, because strategies for imperfect information systems can only be defined with regards to some observation. Whenever a strategy is assigned to a player, this player will behave as if he sees the system with the observation that his strategy was defined with. In that sense, it is possible for a player to change his observation power if he changes his strategy to another one using a different observation. A natural extension to Strategy Logic with imperfect information being epistemic operators, we decided to study how such changes of observation would interact with the agents' knowledge, by defining a dedicated operator.

2 CTL*KΔ

2.1 Syntax

We begin by introducing the syntax of our logic. It contains operators of branching-time logics, temporal logics and epistemic logics, as well as a new one to indicate changes of observations. At first, we will study the case where there is only one agent (and thus only one knowledge operator).

We consider $\mathcal{O}$ to be a set of *observations*, that each represent a possible observational power of the agent. AP is a set of atomic propositions. Formulas of CTL*KΔ can be *history formulas* φ or *path formulas* ψ.

$$\varphi := p \mid \neg\varphi \mid \varphi \wedge \varphi \mid A\psi \mid K\varphi \mid \Delta^o\varphi$$
$$\psi := \varphi \mid \neg\psi \mid \psi \wedge \psi \mid X\psi \mid \psi U \psi$$

Where $p \in AP$ and $o \in \mathcal{O}$. The temporal operators X and U are meant to represent the typical *next* and *until* operators of temporal logics. A is a path quantifier, similar to those found in branching-time logics. Intuitively, $A\psi$ should hold for a history if ψ is true in every possible future. K is an epistemic operator. Intuitively, $K\varphi$ should be true whenever the agent knows that φ is true. We introduce a new one, Δ^o, to represent a change of observation. φ is called a history formula, as we only need to know what happened in the past to decide if the formula is true. ψ is called a path formula as it also requires the future evolution of the system.

We can also define the temporal operator R (the *release*), dual of U. The path quantifier E, dual of A. The knowledge operator $\bar{K}$ (*possibility*), dual of K. With negation and $\wedge$, we can also define the classical Boolean operators $\vee$ and $\rightarrow$.

2.2 Semantics

The models on which such formulas can be interpreted are classical Kripke Structure, with several equivalence relations between states (one for each observation). Let $\mathcal{O} = \{o_1, o_2, \ldots, o_m\}$. A Kripke Structure with observations is a structure $M = (S, I_s, o_I, T, V, \sim_{o_1}, \ldots, \sim_{o_m})$ where

S is a set of states.
$I_s \subseteq S$ is a set of initial states.
o_I is the initial observation.
$T \subseteq S \times S$ is a transition relation between states.
$V : S \to 2^{AP}$ is a valuation function.
$\forall o_i, \sim_{o_i}$ is an equivalence relation between states.

A *run* or *path* is an infinite sequence of states $\pi = \pi_0\pi_1 \ldots$. A *history* is a finite sequence of states $h = h_1 \ldots h_n$.

Observation records Given $\mathcal{O}$ a set of observations, we define *observations records* to be ordered lists of pairs of observations and natural numbers. Intuitively, an observation record represents changes of observations.

Example: $r = [(o_1, 0), (o_2, 3), (o_3, 3)]$ means that the player starts at time 0 with observation o_1. It keeps this observation, then at time 3, it first changes to o_2 and then to o_3. We use observation records in the semantics to remember the previous observations of the agent.

We write $r[(o, n)]$ to append a new pair (o, n) to the observation record r. We write $r_{\leq n}$ the record r without the pairs (o, m) where $m > n$. We write r_n the record r without the pairs (o, m) where $m \neq n$. We define a function $O(r, n)$ which gives a tuple of the observations at time n. $O(r, n)$ includes every observation of r_n, as well as the previous one.

Example: Let $r = [(o_1, 0), (o_2, 3), (o_3, 3)]$. $O(r, 0) = (o_I, o_1)$, $O(r, 1) = O(r, 2) = (o_1)$, $O(r, 3) = (o_1, o_2, o_3)$ and $O(r, 4) = (o_3)$.

On a given model, with an observation record we can define an equivalence relation between histories (finite sequences of states), with regard to a record. Two histories are equivalent with regard to the record if the player can't distinguish them by using the observations in the record.
$h \approx_r h'$ *iff* $\forall i < |h|, \forall o \in O(r, i), h(i) \sim_o h'(i)$ *and* $|h| = |h'|$.

Record semantics We first define the intuitive semantics of CTL*KΔ. History formulas need a history h (finite sequence of previous states) and an observation record r to be interpreted, to know which history might be considered possible for the agent. Path formulas are interpreted on a run π (infinite sequence of states of the model), a point in time (natural number), and an observation record.

$$
\begin{array}{lll}
M,h,r \models p & \textit{iff} & p \in V(\textit{last}(h)) \\
M,h,r \models \neg\varphi & \textit{iff} & M,h,r \not\models \varphi \\
M,h,r \models \varphi_1 \wedge \varphi_2 & \textit{iff} & (M,h,r \models \varphi_1 \text{ and } M,h,r \models \varphi_2) \\
M,h,r \models A\psi & \textit{iff} & \forall \pi \text{ that extends } h, \text{ we have } M,\pi,|h|-1,r \models \psi \\
M,h,r \models K\varphi & \textit{iff} & \forall h' \text{ such that } h' \approx_r h, \text{ we have } M,h',r \models \varphi \\
M,h,r \models \Delta^o\varphi & \textit{iff} & M,h,r[(o,|h|-1)] \models \varphi \\
M,\pi,n,r \models \varphi & \textit{iff} & M,(\pi_0 \dots \pi_n),r \models \varphi \\
M,\pi,n,r \models \neg\psi & \textit{iff} & M,\pi,n,r \not\models \psi \\
M,\pi,n,r \models \psi_1 \wedge \psi_2 & \textit{iff} & (M,\pi,r,n \models \psi_1 \text{ and } M,\pi,r,n \models \psi_2) \\
M,\pi,n,r \models X\psi & \textit{iff} & M,\pi,(n+1),r \models \psi \\
M,\pi,n,r \models \psi_1 U \psi_2 & \textit{iff} & \exists m \geq n \text{ such that } \forall k \in [n,m[, M,\pi,k,r \models \psi_1 \\
 & & \text{and } M,\pi,m,r \models \psi_2
\end{array}
$$

Finally, we say that $M = (S, I_s, o_I, T, V, \sim_{o_1}, \dots, \sim_{o_m})$ models φ (written $M \models \varphi$), if $\forall s \in I_s, M, s, [(o_I, 0)] \models \varphi$. This definition corresponds to the model-checking problem that we solve in Section 4.

A few validities of CTL*KΔ

$$
\begin{array}{lcll}
\Delta^o(\varphi_1 \wedge \varphi_2) & \leftrightarrow & (\Delta^o\varphi_1 \wedge \Delta^o\varphi_2) & \text{(distributivity)} \\
\Delta^o\neg\varphi & \leftrightarrow & \neg\Delta^o\varphi & \text{(self-duality)} \\
\Delta^o AX\Delta^o K\varphi & \leftrightarrow & \Delta^o AXK\varphi & \text{(redundant change of observation)} \\
\Delta^o K\varphi & \rightarrow & \Delta^o K\Delta^o K\varphi & \text{(positive introspection)}
\end{array}
$$

Our Model-Checking approach Once CTL*KΔ is defined, our goal is to solve the model-checking problem. Because of perfect-recall semantics, it may seem that we have to remember the complete history and records when evaluating a formula. However, our main idea is that we can extract some information from the history that is sufficient for the evaluation of the formula. Intuitively, to evaluate a history formula, it is enough to know the current state, the current observation and the set of states that the agent believes the system might be in (this set is later called the *Information Set*). This new structure to represent the knowledge is more succinct than remembering entire histories and records, as there is a finite number of information sets. We start by defining a new semantics for the same formulas. This semantics uses information sets. Then, we will present an algorithm to model-check a formula according to these new semantics.

3 Alternative Semantics

We now introduce another semantics to interpret formulas of CTL*KΔ. In this semantics, we aim to replace histories and observation records with information sets (intuitively, the set of states the agent believe it is possible to be in).

3.1 Information sets

In this semantics, history formulas are interpreted on s a state of the model, I an information set (set of model states) and o the current observation. Path

formulas are interpreted on π an infinite sequence of states of the model starting in the current one (we don't remember the past states), I the information set and o the current observation.

Preliminary definitions If $\pi = \pi_0\pi_1\dots$ is an infinite sequence of states, we write $\pi_{n\dots}$ the infinite sequence $\pi_n\pi_{n+1}\dots$.

We define two functions to update information sets. U_Δ updates the set when a player goes through a change of observation and U_T updates the set when the player moves to a new state. Finally, we define a function I_I to get the initial information set from an initial state and an initial observation.

$U_\Delta(I,s,o) = \{x \in I \mid x \sim_o s\}$
$U_T(I,s,o) = \{x \in S \mid \exists t \in I, t \to x \text{ and } x \sim_o s\}$
$I_I(s_I,o_I) = \{s \in S \mid s \sim_{o_I} s_I\}$

When we are in state s with information set I and the observation changes to o, the new information set is $U_\Delta(I,s,o)$. When we move to a new state s with information set I and observation o, the new information set is $U_T(I,s,o)$

For a path π and $n \in \mathbb{N}$, we write $U_T^n(I,\pi,o)$ the successive temporal updates from π_0 to π_n.

$U_T^0(I,\pi,o) = I$ and $U_T^{n+1}(I,\pi,o) = U_T(U_T^n(I,\pi,o),\pi_{n+1},o)$

3.2 Information Set Semantics

$M,s,I,o \models p$	*iff*	$p \in V(s)$
$M,s,I,o \models \neg\varphi$	*iff*	$M,s,I,o \not\models \varphi$
$M,s,I,o \models \varphi_1 \wedge \varphi_2$	*iff*	$(M,s,I,o \models \varphi_1$ and $M,s,I,o \models \varphi_2)$
$M,s,I,o \models A\psi$	*iff*	$\forall\pi$ such that $\pi_0 = s$, we have $M,\pi,I,o \models \psi$
$M,s,I,o \models K\varphi$	*iff*	$\forall s' \in I$, we have $M,s',I,o \models \varphi$
$M,s,I,o' \models \Delta^o\varphi$	*iff*	$M,s,U_\Delta(I,s,o),o \models \varphi$
$M,\pi,I,o \models \varphi$	*iff*	$M,\pi_0,I,o \models \varphi$
$M,\pi,I,o \models \neg\psi$	*iff*	$M,\pi,I,o \not\models \psi$
$M,\pi,I,o \models \psi_1 \wedge \psi_2$	*iff*	$(M,\pi,I,o \models \psi_1$ and $M,\pi,I,o \models \psi_2)$
$M,\pi,I,o \models X\psi$	*iff*	$M,\pi_{1\dots},U_T(I,\pi_1,o),o \models \psi$
$M,\pi,I,o \models \psi_1 U \psi_2$	*iff*	$\exists n \geq 0, \forall m \leq n, M,\pi_{m\dots},U_T^m(I,\pi,o),o \models \psi_1$ and $M,\pi_{n\dots},U_T^n(I,\pi,o),o \models \psi_2$

3.3 Reduction Theorem

We now have different semantics, that share the same syntax. We will now prove that these two semantics are equivalent.

To do so, we first notice that, from a history h and an observation record r, we can get corresponding current state, information set and current observation. Similarly, from an infinite sequence π, a time n and a record, we can get corresponding sequence π' that starts at the current state, information set and current observation.

We accordingly define two partial functions, *FH* and *FP* as follows:

$FH(h,r) = (s,I,o)$ where
I) $s = last(h)$
II) $o = last(r_{\leq |h|-1})$
III) $I = f(h,r)$

$FP(\pi,n,r) = (\pi',I,o)$ where
IV) $\pi' = \pi_{n\ldots}$
V) $o = last(r_{\leq n})$
VI) $I = f((\pi_0,\ldots,\pi_n),r)$

FH and FP are partial functions, because $FH(h,r)$ is defined only if $r = r_{\leq |h|-1}$ and $FP(\pi,n,r)$ is defined only if $r = r_{\leq n}$.

f is defined inductively as follows: (s is a state, h a history)
$f(s,r) = I_I(s,o_I)$ if r_0 is empty
$f(s,r) = U_\Delta(U_\Delta(\ldots U_\Delta(I_I(s,o_I),s,o_1),s,o_2)\ldots,s,o_m)$ if $r_0 = [o_1,o_2,\ldots,o_m]$
$f(h.s,r) = U_T(f(h,r),s,last(r_{\leq |h|-1}))$ if $r_{|h|}$ is empty
$f(h.s,r) = U_\Delta(\ldots U_\Delta(U_T(f(h,r),s,last(r_{\leq |h|-1})),s,o_1)\ldots,s,o_m)$ if $r_{|h|} = [o_1,\ldots,o_m]$

Lemmas

Information Set Lemma If $f(h,r) = I$, then
$I = \{s \in S \mid \exists h', h' \approx_r h \text{ and } last(h') = s\}$.

Proof: This is proved inductively.

- Let $s \in S$ and r a record. If r_0 is empty, then $\{s' \in S \mid \exists h', h' \approx_r s \text{ and } last(h') = s'\} = \{s' \in S \mid s' \sim_{o_I} s\} = I_I(s,o_I)$. This is the definition of $f(s,r)$.
- If $r_0 = [o_1 \ldots o_m]$, then $\{s' \in S \mid \exists h', h' \approx_r s \text{ and } last(h') = s'\} = \{s' \in S \mid s' \sim_{o_i} s, \forall o_i \in r_0 \cup \{o_I\}\}$. This is the definition of $f(s,r)$.
- Let h be a history and r a record. Let $o_r = last(r_{\leq |h|-1})$. If $r_{|h|}$ is empty
  $\{s' \in S \mid \exists h', h' \approx_r h.s \text{ and } last(h') = s'\}$
  $= \{s' \mid \exists h', h' \approx_r h, last(h') \to s' \text{ and } s' \sim_{o_r} s\}$
  $= \{s' \mid \exists t \in f(h,r), t \to s' \text{ and } s' \sim_{o_r} s\}$ by induction hypothesis
  $= U_T(f(h,r),s,o_r)$.
- Let $o_r = last(r_{\leq |h|-1})$. If $r_{|h|} = [o_1 \ldots o_m]$, then
  $\{s' \in S \mid \exists h', h' \approx_r h.s \text{ and } last(h') = s'\}$
  $= \{s' \mid \exists h', h' \approx_r h, last(h') \to s' \text{ and } s' \sim_o s \forall o \in [o_1 \ldots o_m] \cup \{o_r\}\}$
  $= \{s' \mid \exists t \in f(h,r), t \to s' \text{ and } s' \sim_o s \forall o \in [o_1 \ldots o_m] \cup \{o_r\}\}$ by induction hypothesis
  $= U_\Delta(\ldots U_\Delta(U_T(f(h,r),s,o_r),s,o_1)\ldots,s,o_m)$. ■

We now present useful results about the functions FH and FP. The proofs of the following lemmas can be found in appendix, Section 7.1.

Lemma 1 Let h, h' and r such that $h \approx_r h'$. Let $(s,I,o) = FH(h,r)$ and $(s',I',o') = FH(h',r)$. We have $I = I'$ and $o = o'$.

Lemma 2 Let $(s,I,o) = FH(h,r)$.
Then, $\forall \pi$ such that $\pi_0 = s$, $FP(h.\pi_{1\ldots}, |h|-1, r) = (\pi,I,o)$.

Lemma 3 Let $(s,I,o) = FH(h,r)$. Then, $FH(h, r[(o', |h|-1)]) = (s, U_\Delta(I,s,o'), o')$.

Lemma 4 Let $(\pi', I, o) = FP(\pi, n, r)$. Then, $FH(\pi_0 \dots \pi_n, r) = (\pi'_0, I, o)$.

Lemma 5 Let $(\pi', I, o) = FP(\pi, n, r)$. Then, $FP(\pi, n{+}1, r) = (\pi'_{1\dots}, U_T(I, \pi'_1, o), o)$.

Lemma 6 Let $(\pi', I, o) = FP(\pi, n, r)$.
Then, $\forall k \geq 0, FP(\pi, n+k, r) = (\pi'_{k\dots}, U^k_T(I, \pi', o), o)$.

Reduction Theorem $\forall \phi$ formula of CTL*KΔ,
if $\phi = \varphi$ is a history formula, $\forall h, r, s, I, o$ such that $FH(h, r) = (s, I, o)$,
$M, h, r \models \varphi$ *iff* $M, s, I, o \models \varphi$,
if $\phi = \psi$ is a path formula, $\forall \pi, n, r, \pi', I, o$ such that $FP(\pi, n, r) = (\pi', I, o)$,
$M, \pi, n, r \models \psi$ *iff* $M, \pi', I, o \models \psi$.

Proof By induction on ϕ. Let h, r, s, I, o such that $FH(h, r) = (s, I, o)$. For each case, we prove **1:** if $M, h, r \models \varphi$ then $M, s, I, o \models \varphi$, and **2:** if $M, s, I, o \models \varphi$ then $M, h, r \models \varphi$,

- p **1: and 2:** We have $p \in V(last(h))$ *iff* $p \in V(s)$, because $FH(h, r) = (s, I, o)$. Thus, $M, h, r \models p$ *iff* $M, s, I, o \models p$.
- $\neg\varphi$ **1: and 2:** We have $M, h, r \not\models \varphi$ *iff* $M, s, I, o \not\models \varphi$ by induction hypothesis.
- $\varphi_1 \wedge \varphi_2$ **1 and 2:** We have $M, h, r \models \varphi_1$ and $M, h, r \models \varphi_2$ *iff* $M, s, I, o \models \varphi_1$ and $M, s, I, o \models \varphi_2$ by induction hypothesis.
- $A\psi$ **1:** We have $\forall \pi'$ extending h, $M, \pi', |h|-1, r \models \psi$. Let π such that $\pi_0 = s$. Let us prove that $M, \pi, I, o \models \psi$. We have $FP(h.\pi_{1\dots}, |h| - 1, r) = (\pi, I, o)$, because $FH(h, r) = (s, I, o)$ (**Lemma 2**). Then by induction, because $h.\pi_{1\dots}$ extends h and $M, h.\pi_{1\dots}, |h| - 1, r \models \psi$, we have $M, \pi, I, o \models \psi$.
- $A\psi$ **2:** Assume $\forall \pi'$ such that $\pi'_0 = s$, $M, \pi', I, o \models \psi$. Let π an infinite sequence of states. Let us prove that $M, h.\pi, |h| - 1, r \models \psi$. By induction, it suffices to prove that $M, s.\pi, I, o \models \psi$, because $FP(h.\pi, |h|-1, r) = (s.\pi, I, o)$ (**Lemma 2**). $s.\pi$ is a run π' such that $\pi'_0 = s$, and thus $M, h, r \models A\psi$.
- $K\varphi$ **1:** Assume that $\forall h_1 \approx_r h, M, h_1, r \models \varphi$. Let $s' \in I$. Let us prove that $M, s', I, o \models \varphi$. Using the **Information Set Lemma**, there exists some $h_1 \approx_r h$ with $last(h_1) = s'$. We then have $FH(h_1, r) = (s', I, o)$ (**Lemma 1**). As $M, h_1, r \models \varphi$, we conclude by induction that $M, s', I, o \models \varphi$ and thus $M, s, I, o \models K\varphi$.
- $K\varphi$ **2:** Assume that $\forall s' \in I, M, s', I, o \models \varphi$. Let h' such that $h' \approx_r h$. Let $s' = last(h')$. We have that $FH(h', r) = (s', I, o)$ (**Lemma 1**). Because $s' \in I$ (**Information Set Lemma**), by induction we conclude $M, h', r \models \varphi$, and then $M, h, r \models K\phi$.
- $\Delta^{o'}\varphi$ **1: and 2:** Using **Lemma 3**, we have $FH(h, r[(o', |h|-1)]) = (s, U_\Delta(I, s, o'), o')$. Then, by induction, $M, h, r[(o', |h| - 1)] \models \varphi$ *iff* $M, s, U_\Delta(I, s, o'), o' \models \varphi$.

Let π, n, r, π', I, o such that $FP(\pi, n, r) = (\pi', I, o)$.

- φ With **Lemma 4**, we have $FH(\pi_0 \dots \pi_n, r) = (\pi'_0, I, o)$ and thus by induction, $M, \pi_0 \dots \pi_n, r \models \varphi$ *iff* $M, \pi'_0, I, o \models \varphi$. Finally, $M, \pi, n, r \models \varphi$ *iff* $M, \pi', I, o \models \varphi$.
- $\neg\psi$ and $\psi_1 \wedge \psi_2$ Apply the induction hypothesis.

- $\underline{X\psi}$ With **Lemma 5**, we have $FP(\pi, n+1, r_{\leq n}) = (\pi'_{1\ldots}, U_T(\pi'_1, I, o), o)$ and thus by induction, $M, \pi, n, r \models X\psi$ *iff* $M, \pi', I, o \models X\psi$.
- $\underline{\psi_1 U \psi_2}$ According to the definitions, it suffices to prove that $\forall k \geq 0$ and ψ subformula of $\psi_1 U \psi_2$, $M, \pi'_k, U_T^k(I, \pi', o) \models \psi$ *iff* $M, \pi, n+k, r \models \psi$. This result comes from the induction hypothesis and **Lemma 6**. ■

4 Model-Checking CTL*KΔ

Model-Checking problem Given a model $M = (S, I_s, o_I, T, V, \sim_{o_1}, \ldots, \sim_{o_m})$ and a history formula φ, return **yes** if $M \models \varphi$, and **no** otherwise. Thanks to the reduction theorem, it suffices to show that, for each $s \in I_s$, $M, s, I_I(s, o_I), o_I \models \varphi$. The algorithm will check the formula according to the information set semantics.

Augmented Model We define an augmented model in which the states are tuples (s, I, o). Because there is a finite number of states, information sets and observations, this model is finite. According to the information set semantics, history formulas can be viewed on this model as state formulas.

From $M = (S, I, o_I, T, V, \sim_{o_1}, \ldots, \sim_{o_m})$ we define the augmented model $\hat{M} = (S', T', V')$, a Kripke Structure.

- $S' = S \times 2^S \times \mathcal{O}$: states are state of the original model, an observation set and an observation. There is a finite number of such states.
- $(s, I, o)\ T'\ (s', I', o)$ *iff* $s\ T\ s'$ and $I' = U_T(I, s', o)$
- $V'(s, I, o) = V(s)$. As the algorithm is executed, new atomic propositions will appear. We will update V' accordingly.

We write M_o the Kripke Structure obtained by keeping only the states where the observation is o. The different M_o are disjoint with regards to T'.

Model checking a state formula On this model $\hat{M}$, we define the function CheckCTL*KΔ to determine if a history formula is true in a given state (**Algorithm** 1). We assume that we can check if a CTL* state formula φ is true in a state s of the Kripke Structure K with the function CheckCTL*(K, s, φ) [5].

Algorithm Correctness To be convinced of the correctness of the algorithm, it suffices to prove the following properties:

- If φ is a formula of CTL*, CheckCTL*$(M_o, (s, I, o), \varphi)$ returns **true** *iff* $M, s, I, o \models \varphi$.
- For each formula $K\varphi_1$ chosen by the algorithm, after the `for` loop, $p_\phi \in V'(s, I, o)$ *iff* $M, s, I, o \models K\varphi_1$
- For each formula $\Delta^{o'}\varphi_1$ chosen by the algorithm, after the `for` loop, $p_\phi \in V'(s, I, o)$ *iff* $M, s, I, o \models \Delta^{o'}\varphi_1$

Algorithm 1 Model Checking CTL*KΔ

function CheckCTL*K$\Delta(\hat{M}, (s_c, I_c, o_c), \varphi)$
  if There exists $\phi = K\varphi_1$ or $\phi = \Delta^{o'}\varphi_1$ a subformula of φ such that φ_1 is a formula of CTL* **then**
    Let p_{φ_1} be a new Atomic Proposition
    for $(s, I, o) \in S'$ **do**
      if CheckCTL$^*(M_o, (s, I, o), \varphi_1)$ **then**
        $V'(s, I, o) := V'(s, I, o) \cup \{p_{\varphi_1}\}$
      end if
    end for
    Let p_ϕ be a new Atomic Proposition
    if $\phi = K\varphi_1$ **then**
      for $(s, I, o) \in S'$ **do**
        if $p_{\varphi_1} \in V'(s', I, o)$ for each $s' \in I$ **then**
          $p_\phi \in V'(s, I, o)$
        end if
      end for
    end if
    if $\phi = \Delta^{o'}\varphi_1$ **then**
      for $(s, I, o) \in S'$ **do**
        if $p_{\varphi_1} \in V'(s, U_\Delta(I, s, o'), o')$ **then**
          $V'(s, I, o) := V'(s, I, o) \cup \{p_\phi\}$
        end if
      end for
    end if
    CheckCTL*K$\Delta(\hat{M}, (s_c, I_c, o_c), \varphi[\phi \leftarrow p_\phi])$
  else
    CheckCTL$^*((M_{o_c}, (s_c, I_c, o_c), \varphi))$
  end if
end function

The first property comes from the fact that, once restricted to the CTL* operators, the semantics of CTL*KΔ is identical to CTL*. We thus have that CheckCTL$^*(M_o, (s, I, o), \varphi)$ returns **true** *iff* $M_o, s, I, o \models \varphi$. Finally, $M_o, s, I, o \models \varphi$ is equivalent to $M, s, I, o \models \varphi$ because φ is a formula of CTL* and the different M_o are disjoints.

For the second and third properties, we use the previous one to know that after the first `for` loop, $p_{\varphi_1} \in V'(s, I, o)$ *iff* $M, s, I, o \models \varphi_1$. Then, after the second loop, $p_\phi \in V'(s, I, o)$ *iff* $\forall s' \in I, p_{\varphi_1} \in V'(s', I, o)$, which is equivalent to $M, s, I, o \models K\varphi_1$. Similarly for he third property, $p_\phi \in V'(s, I, o)$ *iff* $p_{\varphi_1} \in V'(s, U_\Delta(I, s, o'), o')$ which is equivalent to $M, s, I, o \models \Delta^{o'}\varphi_1$.

Finally, every formula of CTL*KΔ is either a formula of CTL* or contains a formula $K\varphi_1$ or $\Delta^{o'}\varphi_1$ such that φ_1 is a formula of CTL*. Because every recursive call removes one operator K or $\Delta^{o'}$ from the formula, the algorithm eventually finishes for every formula of CTL*KΔ. We know that the model-checking of CTL* is in PSPACE [5]. Because this function is called for each state of the

augmented model, for each subformula $K\varphi$ or $\Delta^o\varphi$, the overall complexity of the model-checking algorithm for a single player is in EXPTIME.

Example Let $M = (S, I, o_I, T, V, \sim_{o_1}, \sim_{o_2})$ where $S = I = \{s_1, s_2\}$, $\mathcal{O} = \{o_1, o_2\}$, $o_I = o_1$, $AP = \{q\}$, o_1 is the blind observation ($s_1 \sim_{o_1} s_2$) and o_2 is the perfect observation ($s \sim_{o_2} s'$ *iff* $s = s'$). $V(s_1) = \{q\}$ and $V(s_2) = \emptyset$. The transitions T are pictured on **Fig.** 1. The augmented model $\hat{M}$ is pictured **Fig.** 2. We only drew the reachable states (reachable with T' or U_Δ).

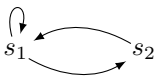

Fig. 1. M

Consider that the formula to model-check is $\varphi = \Delta^{o_2}(Kq \vee \Delta^{o_1} KAXq)$. Intuitively, it means that if the agent changes to the perfect observation, then either the agent knows that p holds, or even after changing to the blind observation he knows that in every possible next step, p holds. After running the algorithm, we get the following valuation:

$V'(s_1, \{s_1, s_2\}, o_1) = \{q, p_\varphi\}$
$V'(s_2, \{s_1, s_2\}, o_1) = \{p_\varphi\}$
$V'(s_1, \{s_1\}, o_1) \quad = \{q, p_{(Kq)}, p_{(Kq \vee \Delta^{o_1} KAXq)}, p_\varphi\}$
$V'(s_2, \{s_2\}, o_1) \quad = \{p_{(\Delta^{o_1} KAXq)}, p_{(Kq \vee \Delta^{o_1} KAXq)}, p_\varphi\}$
$V'(s_1, \{s_1\}, o_2) \quad = \{q, p_{(Kq)}, p_{(Kq \vee \Delta^{o_1} KAXq)}, p_\varphi\}$
$V'(s_2, \{s_2\}, o_2) \quad = \{p_{(\Delta^{o_1} KAXq)}, p_{(Kq \vee \Delta^{o_1} KAXq)}, p_\varphi\}$

For instance, $q \in V'(s_1, \{s_1\}, o_1)$ because $s \in V(s_1)$, $p_{(Kq)} \in V'(s_1, \{s_1\}, o_1)$ because $\forall s' \in \{s_1\}, q \in V'(s', \{s_1\}, o_1)$. $p_{(Kq \vee \Delta^{o_1} KAXq)} \in V'(s_1, \{s_1\}, o_1)$ because $p_{(Kq)} \in V'(s_1, \{s_1\}, o_1)$ and finally, $p_\varphi \in V'(s_1, \{s_1\}, o_1)$ because $p_{(Kq \vee \Delta^{o_1} KAXq)} \in V'(s_1, \{s_1\}, o_2)$.

We see that p_φ is true in every state, and therefore true in the initial states. We conclude, as expected, that $M \models \varphi$.

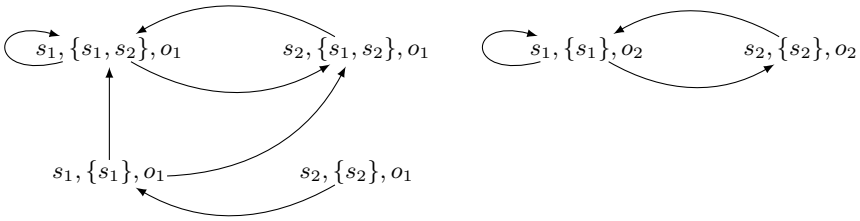

Fig. 2. $\hat{M}$, the augmented model

5 Multi-agent setting

We now define CTL*KΔ for multiple agents. We consider $\mathcal{A}$ to be the (finite) set of agents. We write g the number of agents. In this logic, changes of observation are public, meaning that every player knows the changes of observations of all players. This allows agents to reason about another agent's knowledge.

5.1 Syntax and intuitive Semantics

Syntax The syntax has to be modified. There is now one knowledge operator and one change operator for each agent $a \in \mathcal{A}$: K_a and Δ_a^o.

$$\varphi := p \mid \neg\varphi \mid \varphi \wedge \varphi \mid A\psi \mid K_a\varphi \mid \Delta_a^o\varphi$$
$$\psi := \varphi \mid \neg\psi \mid \psi \wedge \psi \mid X\psi \mid \psi U \psi$$

Where $p \in AP, a \in \mathcal{A}$ and $o \in \mathcal{O}$.

Record Semantics We now need one observation record for each player to interpret a formula. For most operators, the semantics remains unchanged. We make the following modifications:

$M, h, r_1, \ldots, r_g \models K_a\varphi$ *iff* $\forall h'$ s.t. $h' \approx_{r_a} h$, we have $M, h', r_1, \ldots, r_g \models \varphi$
$M, h, r_1, \ldots, r_g \models \Delta_a^o\varphi$ *iff* $M, h, r_1, \ldots, r_a[(o, |h|-1)], \ldots, r_g \models \varphi$

5.2 *k*-trees semantics

***k*-trees** We want to define another semantics, similarly to the Information set semantics defined in Section 3. However, Information sets no longer contain sufficient information to represent the epistemic situation of a multi-agent system. Indeed, agents need not only to remember what states they believe the system might be in, but also the states that other agents believe the system might be in, in case of formulas with nested knowledge operators. For a formula ϕ of CTL*KΔ with multiple agent, we write $depth(\phi)$ the maximal number of nested knowledge operators in ϕ. For a finite number of nested operators k, this information can conveniently be stored in k-trees, as defined in [14]. The set $\mathcal{T}_k$ of k-trees over the set of states S for g agents is defined inductively as follows:

$\mathcal{T}_0 = \{(s, \emptyset, \ldots, \emptyset) \mid (g+1)\text{-tuple, with } s \in S\}$
$\mathcal{T}_{k+1} = \{(s, U_1, \ldots, U_g) \mid s \in S \text{ and } \forall i, U_i \subseteq \mathcal{T}_k\}$

Intuitively, if $(s, U_1, \ldots, U_g)$ is a k-tree, then s (the root) is the real state of the system, and each U_i represents the knowledge of agent i. This U_i (the set of i-children) is itself a set of $(k-1)$-trees including the knowledge of other agents.

Updating *k*-trees Similarly to U_Δ and U_T, we define inductively $U_{\Delta k}$ and U_{Tk} to update k-trees.

$U_{Tk} : \mathcal{T}_k \times S \times \mathcal{O}^g \to \mathcal{T}_k$

$$U_{T0}(t, s, o_1, \ldots, o_g) = (s, \emptyset, \ldots, \emptyset)$$
$$U_{Tk+1}((s_t, U_{t,1}, \ldots, U_{t,g}), s, o_1, \ldots, o_g) = (s, U_1, \ldots, U_g)$$
$$\text{Where } U_i = \{U_{Tk}(t, s', o_1, \ldots, o_g) \mid t \in U_{t,i}, \quad s' \sim_{o_i} s, \quad root(t)\ T\ s'\}$$

$U_{\Delta k} : \mathcal{T}_k \times \mathcal{O}^g \to \mathcal{T}_k$

$$U_{\Delta 0}(t, o_1, \ldots, o_g) = t$$
$$U_{\Delta k+1}((s_t, U_{t,1}, \ldots, U_{t,g}), o_1, \ldots, o_g) = (s_t, U_1, \ldots, U_g)$$
$$\text{Where } U_i = \{U_{\Delta k}(t, o_1, \ldots, o_g) \mid t \in U_{t,i}, \quad root(t) \sim_{o_i} s_t\}$$

Intuitively, when moving to a new state s, each agent i has to update his knowledge (U_i). The next possible trees will be the update of every tree he considered possible with a state that is equivalent to the new one using the current observation (o_i). When changing observation, we simply remove the trees that are no longer equivalent to the actual state.

***k*-trees Semantics** Let $t = (s, U_1, \ldots, U_g)$. We show here the differences with the Information Set semantics:

$$
\begin{array}{lll}
M, t, o_1, \ldots, o_g \models p & \textit{iff} & p \in V(root(t)) \\
M, t, o_1, \ldots, o_g \models A\psi & \textit{iff} & \forall \pi \text{ such that } \pi_0 = root(t), \\
 & & \text{we have } M, \pi, t, o_1, \ldots, o_g \models \psi \\
M, t, o_1, \ldots, o_g \models K_i\varphi & \textit{iff} & \forall t' \in U_i, \text{ we have } M, t', o_1, \ldots, o_g \models \varphi \\
M, t, o_1, \ldots, o'_g \models \Delta_i^{o'}\varphi & \textit{iff} & M, U_\Delta(t, o_1, \ldots, o', \ldots, o_g), o_1, \ldots, o', \ldots, o_g \models \varphi \\
M, \pi, t, o_1, \ldots, o_g \models \varphi & \textit{iff} & M, t, o_1, \ldots, o_g \models \varphi \\
M, \pi, t, o_1, \ldots, o_g \models X\psi & \textit{iff} & M, \pi_{1\ldots}, U_T(t, \pi_1, o_1, \ldots, o_g), o_1, \ldots, o_g \models \psi \\
M, \pi, t, o_1, \ldots, o_g \models \psi_1 U \psi_2 & \textit{iff} & \exists n \geq 0, \forall m \leq n, \\
 & & M, \pi_{m\ldots}, U_T^m(t, \pi, o_1, \ldots, o_g), o_1, \ldots, o_g \models \psi_1 \\
 & & \text{and } M, \pi_{n\ldots}, U_T^n(t, \pi, o_1, \ldots, o_g), o_1, \ldots, o_g \models \psi_2
\end{array}
$$

An adaptation of the reduction theorem can be found in Section 7.2. The modified model-checking algorithm can be found in Section 7.3. Now that we successfully defined the data structure that holds the epistemic information and how to update it, the way to model-checking is similar to the single-agent setting, with k-trees instead of information sets.

6 Conclusion and Future Works

We successfully defined a logic to capture the dynamic changes of observation. We introduced an alternative semantics that we proved to be equivalent and is easier to model-check. This logic can be model-checked using a marking algorithm. We have shown that it can be extended to multi-agent settings with nested knowledge operators using the k-tree structure defined in [14].

We believe that our work could be insightful for future works on Strategy Logic with Imperfect Information [2][1], and provide elements for model-checking epistemic extensions.

Another possible future work could be extending CTL*KΔ. Allowing $\Delta^o\psi$ to be a path formula instead of a history formula would define a logic that we suspect to be more expressive than CTL*KΔ. We haven't solved the model-checking problem for this extension yet. The marking algorithm cannot be applied anymore because $\Delta^o\psi$ cannot be marked on the states of the augmented model, as it also requires a run to be interpreted. We believe that we could define a *focus game*, as defined in [9]. These games are defined such that there exists a winning strategy for some player if and only if a model satisfy a given formula. This work has yet to be done.

References

1. Francesco Belardinelli. A logic of knowledge and strategies with imperfect information.
2. Raphaël Berthon, Bastien Maubert, Aniello Murano, Sasha Rubin, and Moshe Y. Vardi. Strategy logic with imperfect information. In *32nd Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2017, Reykjavik, Iceland, June 20-23, 2017*, pages 1–12, 2017.
3. Laura Bozzelli, Bastien Maubert, and Sophie Pinchinat. Uniform strategies, rational relations and jumping automata. *Information and Computation*, 242(Supplement C):80 – 107, 2015.
4. Krishnendu Chatterjee, Thomas A. Henzinger, and Nir Piterman. Strategy logic. *Inf. Comput.*, 208(6):677–693, 2010.
5. Edmund M. Clarke, E. Allen Emerson, and A. Prasad Sistla. Automatic verification of finite state concurrent systems using temporal logic specifications: A practical approach. In *Conference Record of the Tenth Annual ACM Symposium on Principles of Programming Languages, Austin, Texas, USA, January 1983*, pages 117–126, 1983.
6. Cătălin Dima. *Revisiting Satisfiability and Model-Checking for CTLK with Synchrony and Perfect Recall*, pages 117–131. Springer Berlin Heidelberg, Berlin, Heidelberg, 2009.
7. Jeremy Kong and Alessio Lomuscio. Symbolic model checking multi-agent systems against ctl*k specifications. In *Proceedings of the 16th Conference on Autonomous Agents and MultiAgent Systems, AAMAS 2017, São Paulo, Brazil, May 8-12, 2017*, pages 114–122, 2017.
8. Richard E. Ladner and John H. Reif. The logic of distributed protocols. In *Proceedings of the 1st Conference on Theoretical Aspects of Reasoning about Knowledge, Monterey, CA, March 1986*, pages 207–222, 1986.
9. Martin Lange and Colin Stirling. Model checking games for branching time logics. *J. Log. Comput.*, 12(4):623–639, 2002.
10. Alessio Lomuscio, T. Lasica, and Wojciech Penczek. Bounded model checking for interpreted systems: Preliminary experimental results. In *Formal Approaches to Agent-Based Systems, Second International Workshop, FAABS 2002, Greenbelt, MD, USA, October 29-31, 2002, Revised Papers*, pages 115–125, 2002.
11. Alessio Lomuscio and Wojciech Penczek. Symbolic model checking for temporal-epistemic logic. In *Logic Programs, Norms and Action - Essays in Honor of Marek J. Sergot on the Occasion of His 60th Birthday*, pages 172–195, 2012.
12. Bastien Maubert. *Logical foundations of games with imperfect information : uniform strategies. (Fondations logiques des jeux à information imparfaite : stratégies uniformes)*. PhD thesis, University of Rennes 1, France, 2014.
13. Masahiko Sato. A study of kripke-type models of some modal logics by gentzen's sequential method. 13, 01 1977.
14. Ron van der Meyden and Nikolay V. Shilov. Model checking knowledge and time in systems with perfect recall (extended abstract). In *Foundations of Software Technology and Theoretical Computer Science, 19th Conference, Chennai, India, December 13-15, 1999, Proceedings*, pages 432–445, 1999.

7 Appendix

7.1 Lemmas Proofs

Lemma 1 Let h, h' and r such that $h \approx_r h'$. Let $(s, I, o) = FH(h, r)$ and $(s', I', o') = FH(h', r)$. We have $I = I'$ and $o = o'$.
Proof: $I = I'$ according to the **Information Set Lemma**. In II), we see that the observation only depends on the record and the length of the history. $|h| = |h'|$ because $h \approx_r h'$ and thus $o = o'$. ■

Lemma 2 Let $(s, I, o) = FH(h, r)$. Then, $\forall \pi$ such that $\pi_0 = s$, $FP(h.\pi_{1\ldots}, |h| - 1, r) = (\pi, I, o)$.
Proof:

- IV) $(h.\pi_{1\ldots})_n = s = \pi_0$ and $\forall i \in \mathbb{N}, (h.\pi_{1\ldots})_{n+i} = \pi_i$.
- V) $o = last(r_{\leq |h|-1})$ because $(s, I, o) = FH(h, r)$ (II).
- VI) Because $(s, I, o) = FH(h, r)$ (III). ■

Lemma 3 Let $(s, I, o) = FH(h, r)$. Then, $FH(h, r[(o', |h|-1)]) = (s, U_\Delta(I, s, o'), o')$.
Proof:

- I) Because $(s, I, o) = FH(h, r)$ (I).
- II) Because $o' = last(r[(o', |h| - 1)]_{\leq |h|-1})$.
- III) We have $f(h, r[(o', |h|-1)]) = U_\Delta(f(h, r), last(h), o')$. Because $(s, I, o) = FH(h, r)$ (III), $f(h, r) = I$ and (I) $last(h) = s$. Thus, $f(h, r[(o', |h| - 1)]) = U_\Delta(I, s, o')$. ■

Lemma 4 Let $(\pi', I, o) = FP(\pi, n, r)$. Then, $FH(\pi_0 \ldots \pi_n, r) = (\pi'_0, I, o)$.
Proof:

- I) $\pi'_0 = \pi_n$ because $(\pi', I, o) = FP(\pi, n, r)$ (IV).
- II) Because $(\pi', I, o) = FP(\pi, n, r)$ (V).
- III) Because $(\pi', I, o) = FP(\pi, n, r)$ (VI). ■

Lemma 5 Let $(\pi', I, o) = FP(\pi, n, r)$. Then, $FP(\pi, n+1, r) = (\pi'_{1\ldots}, U_T(I, \pi'_1, o), o)$.
Proof: First, $FP(\pi, n + 1, r)$ is defined because $r = r_{\leq n} = r_{\leq n+1}$.

- IV) Because $(\pi', I, o) = FP(\pi, n, r)$ (IV), $\pi' = \pi_{n\ldots}$ and thus $\pi'_{1\ldots} = \pi_{n+1\ldots}$.
- V) Because $(\pi', I, o) = FP(\pi, n, r)$ (V).
- VI) We have $f((\pi_0 \ldots \pi_{n+1}), r_{\leq n}) = U_T(f(\pi_0 \ldots \pi_n, r_{\leq n}), \pi_{n+1}, o)$. Because $(\pi', I, o) = FP(\pi, n, r)$ (VI), we have $f(\pi_0 \ldots \pi_n, r_{\leq n}) = I$. Thus, $f((\pi_0 \ldots \pi_{n+1}), r_{\leq n}) = U_T(I, \pi'_1, o)$. ■

Lemma 6 Let $(\pi', I, o) = FP(\pi, n, r)$. Then, $\forall k \geq 0, FP(\pi, n+k, r) = (\pi'_{k\ldots}, U^k_T(I, \pi', o), o)$.
Proof We proceed by induction. For $k = 0$, we have to prove $(\pi'_{0\ldots}, I, o) = FP(\pi, n + 0, r)$, which is our hypothesis. For the inductive case:

- IV) $\pi_{n+k\ldots} = \pi'_{k\ldots}$ because $(\pi', I, o) = FP(\pi, n, r)$ (IV).
- V) Because $(\pi', I, o) = FP(\pi, n, r)$ (V) and $r = r_{\leq n}$.
- VI) By induction hypothesis, $f(\pi_0 \ldots \pi_{n+k}, r) = U^k_T(I, \pi', o)$. Then, $f(\pi_0 \ldots \pi_{n+k+1}, r) = U_T(U^k_T(I, \pi', o), \pi_{n+k=1}, o) = U^{k+1}_T(I, \pi', o)$ because r_{n+k+1} is empty. ■

7.2 Reduction Theorem for multiple agents

From $k \in \mathbb{N}$, a history h and observation record $r_1, \ldots, r_g$, we can get a corresponding k-tree and current observations. Similarly, from k, an infinite sequence π, a time n and records, we can get corresponding sequence π' that starts at the current state, k-tree and current observations.

We can define two partial functions, FH and FP, similarly to what was done for the single agent setting, Section 3: $FH(k, h, r_1, \ldots, r_g) = (t, o_1, \ldots, o_g)$ and $FP(k, \pi, n, r_1, \ldots, r_g) = (\pi', t, o_1, \ldots, o_g)$.

Finally, the Reduction Theorem can be adapted as follows:
$\forall \phi$ formula of CTL*KΔ, $k = depth(\phi)$,
if $\phi = \varphi$ is a history formula, $\forall h, r_1, \ldots, r_g, t, o_1, \ldots, o_g$ such that $FH(k, h, r_1, \ldots, r_g) = (t, o_1, \ldots, o_g)$, $\quad M, k, h, r_1, \ldots, r_g \models \varphi$ *iff* $M, t, o_1, \ldots, o_g \models \varphi$,
if $\phi = \psi$ is a path formula, $\forall \pi, n, r_1, \ldots, r_g, \pi', t, o_1, \ldots, o_g$ such that $FP(k, \pi, n, r_1, \ldots, r_g) = (\pi', t, o_1, \ldots, o_g)$, $\quad M, \pi, n, r_1, \ldots, r_g \models \psi$ *iff* $M, \pi', t, o_1, \ldots, o_g \models \psi$.

7.3 Model-Checking a CTL*KΔ formula for multiple agents

The new extended model is the following: $\hat{M} = (S'_1 \cup \cdots \cup S'_{depth(\varphi)}, T', V')$.

- $\forall i, S'_i = \mathcal{T}_i \times \mathcal{O}^g$: states are an i-tree and an observation for each player.
- $(t, o_1, \ldots, o_g)\ T'\ (t', o_1, \ldots, o_g)$ *iff* $root(t)\ T\ root(t')$ and $t' = U_{Ti}(t, root(t'), o_1, \ldots, o_g)$
- $V'(t, o_1, \ldots, o_g) = V(root(t))$. As the algorithm is executed, new atomic propositions will appear. We will update V' accordingly.

The algorithm is a marking algorithm very similar to what has been described in Section 4. For each $K_i\varphi$ or $\Delta_i^o\varphi$ formula where φ is a CTL* formula, we first check φ on each state. Then, we mark $(t, o_1, \ldots, o_g)$ with $p_{K_i\varphi}$ if $t \in \mathcal{T}_k$ with $k \geq depth(K\varphi)$ and $\forall t'$ i-child of t, $(t', o_1, \ldots, o_g)$ has been marked by p_φ. We mark it with $p_{\Delta_i^{o'}\varphi}$ if $(t, o_1, \ldots, o_{i-1}, o', \ldots, o_g)$ has been marked with p_φ.

The number of k-trees has been studied in [14]. Our model-checking algorithm for multiple agents has non-elementary complexity.