Dynamic Logic of Common Knowledge in a Proof Assistant

JÉRÔME PUISSÉGUR AND PIERRE LESCANNE

ABSTRACT. Common knowledge logic is meant to describe situations of the real world where a group of agents is involved. These agents share knowledge and make strong hypotheses on the knowledge of the other agents (the so called *common knowledge*). But as we know, the real world changes and mostly information on what is known about the world changes as well. The changes are described by dynamic logic. To describe knowledge changes, dynamic logic should be combined with logic of common knowledge. In this paper we describe experiments which we have made about the integration in a unique framework of common knowledge logic and dynamic logic in the proof assistant COQ. This results in a set of fully checked proofs for readable statements. We describe the framework and how a proof can be conducted.

Keywords: common knowledge, dynamic logic, proof assistant.

1 Introduction

Common knowledge logic is about knowledge on the world, whereas *dynamic logic* is about changes of the world. Both are presented as *modal logic*. In this paper we propose to analyze reasoning in a combination of those logics through a mechanization by a proof assistant.

By experience, we know that the knowledge we have of the world is not perennial, but is meant to evolve. Therefore, any faithful and complete approach of reasoning of agents about their surrounding world requires to take that evolution into account and to combine a logic that describes the state of the knowledge at a given time and a logics that accounts the changes due to external events. This kind of work is known as *belief revision* (or knowledge revision in our case) and is advocated by Johan van Benthem [20]. In this paper, following the work of [2, 3, 19, 20, 21], we combine two logics: the second logic is common knowledge logic [1, 6, 10, 13] and the second one is dynamic logic [7, 8]. The combination of both is called *dynamic logic of common knowledge*, but the novelty is that we do that combination in a proof assistant.

As we are neither designers of modal logic, nor philosophers, but only proof assistant users, what is presented in this paper is not a general discussion on the interest or the advantage of combining logics or how this can be made more appropriately. What we present is a record of experiments done on a mechanization of dynamic logic of common knowledge in COQ, one of the many proof assistants available on the market. This activity sheds light on the reality of reasoning in dynamic logic of common knowledge and on how the two components, namely epistemic and dynamic fit together. This paper does not address any comparison of using one proof assistant or another in that kind of implementation. We feel that actually higher order proof assistants like ACL-2 [9], HOL [18], Isabelle [15], LEGO [16], PHoX [17] or PVS [4], are not so deeply different w.r.t. modal logic that such a comparison would be informative, for the reader. We prefer to focus on the experience itself. We have taken COQ, because we have practiced it and we have an expert environment around us. This paper is essentially a careful examination of what is necessary to make an actual proof of correctness. We have chosen the *muddy children puzzle* (again not a very original choice) and we introduce the reader to the COQ script.

Why experiences on a proof assistant? We noticed that most of the presentations about logic of common knowledge or dynamic logic or a combination of both were made either through a model approach where no specific care is given to actual deductions, with rules and axioms. When proofs are given they are done at an intermediary level of abstraction, whereas we advocate a deep level, where no detail is left over. We are typically at a proof theory level. With a proof theoretic background, we feel that proofs and deductions are of main importance as it has been shown with most of experience with proof assistants. To summarize, this paper is about the actual integration of common knowledge and dynamic logic in a unique framework in a proof assistant.

2 Dynamic logic of common knowledge

Common knowledge logic is a modal logic with two main modalities. One modality K_i, which is associated with each agent i, is the *knowledge modality*. It is meant to express the knowledge an agent has on statements, facts and propositions. For instance, $K_i(\varphi)$ reads as *i knows* φ. The modality C_G, which is associated with a group G of agents is the *common knowledge modality*. $C_G(\varphi)$ translates the fact that a knowledge is common to a group G of agents, not only each agent in the group G knows φ, but also he knows that the others know φ and he knows that the others know that the others know φ, and this recursively. $C_G(\varphi)$ reads as φ *is a common knowledge of the group G*. It is formalized as a fixed point by an axiom and a rule:

$$\frac{}{\vdash C_G\varphi \to \varphi \wedge E_G C_G\varphi}\,FixPoint_C \qquad\qquad \frac{\vdash \rho \to \varphi \wedge E_G\rho}{\vdash \rho \to C_G\varphi}\,LeastFixPoint_C$$

Dynamic logic makes actions modalities. There are as many modalities as there are actions. If α is an action, then $[\alpha]$ is a modality and one writes $[\alpha]\varphi$ the proposition modified by an action α. If an universe satisfies φ, after the action α has been performed, it satisfies $[\alpha]\varphi$.

Hilbert-style is what has been chosen in the COQ implementation. We are not going to discuss here the pro and cons of such an approach. Let us just say that it is convenient both from the point of view of its presentation and from the point of view of its mechanization in a proof assistant. Therefore the forthcoming rules and axioms will be presented in that framework.

The axioms of modal logic are those of *classical logic* plus two axioms and one rule for each modality M:

- *Normalization axiom K_M:* $\vdash M\varphi \to M(\varphi \to \psi) \to M\psi$
- *Necessitation axiom T_M:* $\vdash M\varphi \to \varphi$
- *Generalization rule Gen_M:* $\dfrac{\vdash \varphi}{\vdash M\varphi}\,Gen_M$

These axioms of modal logic have to be duplicated for dynamic logic and common knowledge logic.

2.1 Epistemic and dynamic modalities: purely epistemic propositions

The central issue of this paper is to show how to integrate common knowledge and dynamic logics in a unique framework for using in a proof assistant. First we define a logic that we call $\mathcal{T}_G^{C[\alpha]}$ (see Figure 1). An interesting feature of $\mathcal{T}_G^{C[\alpha]}$ is axiom **KT1**:

$$\forall\varphi : \textit{proposition}\ \forall\alpha : \textit{event}\ \forall i \in G, \quad \vdash K_i[\alpha]\varphi \to [\alpha]K_i\varphi$$

It is well known in epistemic-temporal logic [5] and is appropriate for dynamic logic of common knowledge. It reads *“if agent i knows that, after event* α, φ *holds, then one can*

$$\frac{\vdash_{\mathcal{K}} \varphi}{\vdash \varphi}\,Classical \qquad \frac{\vdash \varphi \quad \vdash \varphi \to \psi}{\vdash \psi}\,MP$$

$$\frac{}{\vdash K_i\varphi \to K_i(\varphi \to \psi) \to K_i\psi}\,K_K \qquad \frac{}{\vdash K_i\varphi \to \varphi}\,T_K \qquad \frac{\vdash \varphi}{\vdash K_i\varphi}\,Gen_K$$

$$\frac{}{\vdash E_G\varphi \leftrightarrow \bigwedge_{i\in G} K_i\varphi}\,Def_E$$

$$\frac{}{\vdash C_G\varphi \to \varphi \wedge E_G C_G\varphi}\,FixPoint_C \qquad \frac{\vdash \rho \to \varphi \wedge E_G\rho}{\vdash \rho \to C_G\varphi}\,LeastFixPoint_C$$

$$\frac{}{\vdash [\alpha]\varphi \to [\alpha](\varphi \to \psi) \to [\alpha]\psi}\,K_{[\alpha]} \qquad \frac{}{\vdash [\alpha]\varphi \to \varphi}\,T_{[\alpha]} \qquad \frac{\vdash \varphi}{\vdash [\alpha]\varphi}\,Gen_{[\alpha]}$$

$$\frac{}{\vdash K_i[\alpha]\varphi \to [\alpha]K_i\varphi}\,KT1$$

Figure 1. The dynamic logic of common knowledge $\mathcal{T}_G^{C[\alpha]}$

infer that, after event α*, agent i knows that* φ *holds"*. This axiom allows to commute epistemic and dynamic modalities in one direction. Note that the converse is quite dubious in natural language and would certainly be rejected by philosophers. Indeed if *after* α*, I know that* φ *holds*, because action α is precisely to let me know proposition φ, then there no reason to infer that *I know that* φ *has to hold after* α. But looking carefully at axiom **KT1**, one notices that event α is transforming not actually the world in its physical reality, but the knowledge the agent has of it. Therefore to avoid troubles and paradoxes, we consider only events α that are *"purely epistemic"*. This means that in our approach of dynamic logic of common knowledge , we consider only actions or events that change the perception of the world which agents have, not the world itself. We borrowed this concept of purely epistemic action from A. Baltag [3, 2].

2.2 The axiomatization of dynamic logic of common knowledge

For the common knowledge modality C_G we have chosen the axiomatization proposed and implemented in Coq by the first of us [11] . The whole dynamic logic of common knowledge is made of the following ingredients:

- the logic $\mathbb{T}$ for K and for $[\alpha]$,
- the definition of *shared knowledge* E_G,
- the definition of *common knowledge* C_G by a least fixpoint axiom and a rule,
- the axiom **KT1** that makes the connection between dynamic logic and common knowledge logic.

3 A running example: the *muddy children puzzle*

The *muddy children puzzle* will serve as an example to show how dynamic and knowledge logic have been integrated in COQ. This problem is presented by several authors [5, 1, 14] as an illustration of common knowledge logic. The problem considers amazing children who are be able to carry perfectly logical reasoning.

3.1 The statement

First, let us recall the puzzle. The reader who knows the puzzle can skip this part and jump to Section 4, collecting the axioms. We follow more or less the presentation of Meyer and van der Hoek [14].

$n+1$ children are standing in a circle around their father. There are $k+1$ ($k \in \{0,...,n\}$) children with mud on their face. The children can see each other, but they cannot see themselves. In particular, they do not know if they have mud on their face. Father says aloud: "There is at least one child with mud on its face." Then he asks: "Will all children who know they have mud on their face please step forward?" This procedure is repeated until, after the $k+1$-th time Father has asked the same question, all muddy children miraculously step forward.

The conclusion which happens eventually is the result of a logical reasoning made by the children, especially the muddy ones, about what they know initially and what they know about the changes on what they know. It is a perfect example of an common knowledge and dynamic reasoning which fits with our frameworks.

3.2 The formalization

In this section, we try to say what justified our statements. A reader interested only by the formal rules and the mechanized reasoning can jumps over the text and go directly to the formal statements. This discussion is interesting to understand why we have chosen this system of axioms.

Two events

In this puzzle, the action are not very elaborated, since after Father's first statement, he keeps repeating the same sentence. Therefore we consider two events, one that starts the scenario and that we write "$\cdot$", it is also called the *initial event*, and one that corresponds to the sentence Father repeats and that we will write "*", it is also called the *progression event*. In our dynamic logic of common knowledge, we will have two types of propositions: $[.]\varphi$ and $[*]\varphi$. We will also write $[*]^k\varphi$ for $[*]...[*]\varphi$ where $[*]$ is repeated k times. Clearly $[*]^0\varphi$ means φ. In COQ, we will use the identifiers `Point` (abbreviated in `[¤]` in COQ) and `Star` (abbreviated in `[*]` in COQ).

Definitions

To study this puzzle, we must describe formally the situation and so define basic properties with axioms.

Let $n \in \mathbb{N}$ and $k \in \{0,...,n\}$, so that $n+1$ is the number of children (there is at least one of them) and $k+1$ the number of muddy ones (there is also at least one of them). Let G be the group of all children, of cardinality $n+1$: we identify it with $\{1,...,n+1\}$.

Let m_i ($i \in \{1,...,n+1\}$) be the proposition "child i has mud on his face".

Let φ_j ($j \in \mathbb{N}$) be the proposition "at least j children have mud on their face".

Let ψ_j ($j \in \mathbb{N}$) be the proposition "exactly j children have mud on their face", which is defined as follows:

$$\mathbf{EQ}_{\varphi\psi}: \qquad \forall j \in \mathbb{N}, \quad \vdash \psi_j \leftrightarrow \varphi_j \wedge \neg\varphi_{j+1}$$

what one can read "there are exactly j muddy children if and only if there are at least j and at the most j ones". Two trivial properties can be proved from this axiom (the proof is made in the COQ file): first, "if there are at least but not exactly j muddy children, then there are at least $j+1$ ones", which is:

$$\mathbf{IMP}_{\varphi\psi} \qquad \forall j \in \mathbb{N}, \quad \vdash \varphi_j \wedge \neg\psi_j \to \varphi_{j+1}$$

secondly, a principle of exclusion, "there cannot be exactly j and at least $j+1$ muddy children", which is:

$$\mathbf{EXCLU}_{\varphi\psi} \qquad \forall j \in \mathbb{N}, \quad \vdash \neg(\varphi_{j+1} \wedge \psi_j)$$

These propositions describe the *"physical world"*, i.e., the physical state of the children, whether they are muddy or not. They form the type *physical proposition*. As we only take into account *epistemic events*, physical propositions are *"persistent"*, which means they are not modified by *epistemic events*. This property is axiomatized as follows:

$$\textbf{PERSIST} \qquad \forall p : \textit{physical proposition} \;\; \forall \alpha : \textit{epistemic event}, \quad \vdash p \to [\alpha] p$$

The description of the situation in natural language reveal some key events which are to be precisely described.

The initial event and its consequences

First, *Father says loudly that there is at least one muddy child*: therefore this proposition becomes common knowledge. If `TRUE` is the logical constant, we notice that it is the only "true" proposition available to the children initially. The effect of the first statement is as follows:

$$\textbf{MC1}_1 \qquad \vdash [\text{¤}]\texttt{TRUE} \to C_G \varphi_1$$

this is the first axiom of our formalization.

The children are not blind, they see each other and they get pieces of information from it. The *initial event* records what they get: *every child counts the number of muddy children in front of him/her*. In particular, the muddy ones see k muddy children, thus they get a knowledge about the total number of muddy children, namely k or $k+1$:

$$\textbf{MC1}_2 \qquad \forall i \in G, \quad \vdash [\text{¤}]\texttt{TRUE} \to m_i \to K_i(\psi_k \vee \psi_{k+1})$$

Defined that way, the *initial event* is an *epistemic event*: No further action will change the world, only the knowledge the agents own on it will evolve. Therefore the muddy children problem is a paradigmatical example.

We said that physical propositions are persistent, but they are not the only ones. Indeed, the muddy children are able to remember what they have seen initially, in other words, the part $m_i \to K_i(\psi_k \vee \psi_{k+1})$ of axiom $\textbf{MC1}_2$ is also persistent:

$$\textbf{PERS}_{MC1_2} \qquad \forall \alpha : \textit{event} \quad \forall i \in G, \vdash (m_i \to K_i(\psi_k \vee \psi_{k+1})) \to [\alpha](m_i \to K_i(\psi_k \vee \psi_{k+1}))$$

The final statement

The problem gets to its end when the muddy children step forward. This happens when *muddy children know they are muddy*. Formally this is

$$\forall i \in G, \quad m_i \to K_i m_i$$

Muddy children are able to infer this statement when they know there are exactly $k+1$ muddy children: as every muddy child sees k ones (a persistent property), he/she knows that he/she is muddy when he/she knows there are exactly $k+1$ muddy children, i.e. the k ones he/she sees plus him/herself. *If a child is muddy and if he knows there are exactly* $k+1$ *muddy children, then he/she knows he/she is muddy.* This leads to the following axiom.

$$\textbf{MC3} \qquad \forall i \in G, \quad \vdash m_i \to K_i \psi_{k+1} \to K_i m_i$$

The progression event and the increase of knowledge

The core of the work consists in clarifying formally what is produced by Father's injunction and how this makes the muddy children's knowledge to grow.

In this scenario, a tempo is given by Father: time is made discrete and is divided into time intervals which every agents (here the children) can distinguish by counting Father's statements. Therefore, these intervals can be numbered as follows:

- First interval starts at Father's declaration and ends at Father's first injunction
- $(i+1)^{th}$ interval goes from i^{th} to $(i+1)^{st}$ injunction.

After $k+1$ injunctions, every muddy child steps forward, as we will prove it in our epistemic-dynamic system. To do so, we need to better understand what happens from an interval to another with each Father's injunction. These injunctions do not carry much semantics, but they are dynamically important: indeed, each injunction gives a tempo and helps every child in his/her quest of knowledge as it ends the previous interval. Then every child can deduce that no child has stepped forward during the previous interval which means that none has been able to conclude about his/her state, these increases the amount of information the children have..

Indeed, let us consider the first injunction. In the first interval, $C_G\varphi_1$ holds and two cases occur:

If $k=0$, the only muddy child can say at once, that he is muddy because he is the only one to see no other muddy child and after Father's first injunction, he steps forward.

If $k>0$, every child sees at least another muddy child, and so, no one can conclude whether he/she is muddy or not. Worst, Father's initial statement of φ_1 did not tell them anything they do not know, but the fact that this statement became common knowledge and when no one steps forward at Father's first injunction, every child can infer that no one sees no muddy child, this means that every one sees at least one muddy child. This can only happen if there are at least two muddy children. By an easy reasoning they exclude the case $k=0$.

To be more formal, *every child knows that every child knows there is at least one muddy child*, which leads the children to the following: *there are at least two muddy children.*

Father's first injunction translates formally into

$$\vdash E_G E_G \varphi_1 \to [*]E_G\neg\psi_1$$

which generalizes for any injunction:

$$\textbf{MC2} \qquad \forall j \in \{1,...,k\}, \quad \vdash E_G E_G \varphi_j \to [*]E_G\neg\psi_j$$

which is *if every child knows that every child knows there are at least j muddy children, then after Father's injunction, every child knows there cannot be exactly j ones.*

4 A knowledge gain lemma

One can deduce a knowledge gain lemma which says that *if every child knows that every child knows there are at least j muddy children, then after Father's injunction, every child knows there are at least $j+1$ ones.* Formally

LEMMA 1. **GainConn** $\quad \forall j \in \{1,...,k\}, \quad \vdash E_G E_G \varphi_j \to [*]E_G\varphi_{j+1}$

Proof. Let $j \in \{1,...,k\}$.

$$\cfrac{\cfrac{\cfrac{\cfrac{\cfrac{}{\vdash E_GE_G\varphi_j \to [*]E_G\neg\psi_j}\,MC2 \qquad \cfrac{\cfrac{}{\vdash E_GE_G\varphi_j \to E_G\varphi_j}\,T_E \qquad \cfrac{\cfrac{}{\vdash \varphi_j \to [*]\varphi_j}\,PERS_\varphi}{\vdash E_G\varphi_j \to [*]E_G\varphi_j}\,EPers}{\vdash E_GE_G\varphi_j \to [*]E_G\varphi_j}\,Cut}{\vdash E_GE_G\varphi_j \to [*]E_G\varphi_j \wedge [*]E_G\neg\psi_j}\,\wedge Intro}{\vdash E_GE_G\varphi_j \to [*](E_G\varphi_j \wedge E_G\neg\psi_j)}\,*/\wedge Dist}{\vdash E_GE_G\varphi_j \to [*]E_G(\varphi_j \wedge \neg\psi_j)}\,E/\wedge Dist}{\vdash E_GE_G\varphi_j \to [*]E_G\varphi_{j+1}}\,IMP_{\varphi\psi}$$

■

Summary of the proof of the muddy children puzzle theorem

A common knowledge induce an nested shared knowledge at any level, the **GainConn** lemma deduced from **MC2** axiom allows us to get a picture of the proof of the *muddy children puzzle* theorem, which we called **Concl** and which states as:

$$\textbf{Concl} \qquad \vdash \forall k \in \mathbb{N}\ \ \forall i \in G, \quad [\text{¤}]\texttt{TRUE} \to [*]^k(m_i \to K_i m_i)$$

Indeed, initially, φ_1 is a common knowledge (**MC1**$_1$), so it is as an arbitrarily nested shared knowledge. With each Father's injunction, children are able to make precise their knowledge about the total number of muddy children by dropping one level of their shared knowledge. Therefore, after j injunctions, they know φ_{j+1} by dropping j depths of their shared knowledge. But since initially this knowledge is arbitrarily deeply nested in shared knowledge, after the first k Father's injunctions, every child effectively knows φ_{k+1}.

At this point, the muddy children know there are at least $k+1$ muddy children; so, as they see k ones, they deduce there are exactly $k+1$ muddy children (**MC1**$_2$) and they know they are muddy themselves (**MC3**). At the $(k+1)^{st}$ injunction, they will step forward not miraculously (as Meyer and van der Hoek pretend to say) but perfectly logically!

One can notice that **Concl** is holds also for $k = 0$. This theorem describes all the scene: "if Father makes its initial statement, then after the k^{th} injunction, the agents who satisfy property m know they do.".

5 The proof of the *muddy children puzzle* theorem

In this section, we describe the mechanized proof previously summed up in more detail.

Let $n \in \mathbb{N}$ and $k \in \{0,...,n+1\}$.

LEMMA 2 (MultGainConn). $\forall n \in \mathbb{N}^*\ \ \forall j \in \{0,...,k\}, \quad \vdash E_G^{n+1}\varphi_j \to [*]E_G^n\varphi_{j+1}$

Proof. By induction on $n \in \mathbb{N}^*$:

- Initialization : $n = 1$,

$$\dfrac{}{\vdash E_G E_G \varphi_j \to [*]E_G\varphi_{j+1}}\ \mathit{GainConn}$$

- Heredity : Let $n \in \mathbb{N}^*$,

$$\dfrac{\dfrac{\dfrac{}{\vdash E_G^{n+1}\varphi_j \to [*]E_G^n\varphi_{j+1}}\ \texttt{HYP-REC}}{\vdash E_G E_G^{n+1}\varphi_j \to E_G[*]E_G^n\varphi_{j+1}}\ \mathit{EDist} \qquad \dfrac{}{\vdash E_G[*]E_G^n\varphi_{j+1} \to [*]E_G^{n+1}\varphi_{j+1}}\ \mathit{KT}1}{\vdash E_G E_G^{n+1}\varphi_j \to [*]E_G^{n+1}\varphi_{j+1}}\ \mathit{Cut}$$

■

LEMMA 3 (ComImpPartIt). $\forall n \in \mathbb{N}, \quad \vdash C_G p \to E_G^n p$

Proof. By induction on $n \in \mathbb{N}$:

- Initialization : $n = 0$,

$$\dfrac{\dfrac{}{\vdash C_G p \to p \land E_G C_G p}\ \mathit{PointFixe}_C}{\vdash C_G p \to p}\ \land \mathit{Elim}$$

- Heredity : Let $n \in \mathbb{N}$,

$$\dfrac{\dfrac{\dfrac{}{\vdash C_G p \to p \land E_G C_G p}\ \mathit{PointFixe}_C}{\vdash C_G p \to E_G C_G p}\ \land \mathit{Elim} \qquad \dfrac{\dfrac{}{\vdash C_G p \to E_G^n p}\ \texttt{HYP-REC}}{\vdash E_G C_G p \to E_G^{n+1} p}\ \mathit{EDist}}{\vdash C_G p \to E_G^{n+1} p}\ \mathit{Cut}$$

■

LEMMA 4 (PointImpPartIt). $\forall n \in \mathbb{N}^*, \quad \vdash [\text{¤}]\mathtt{TRUE} \to E_G^n\varphi_1$

Proof. Let $n \in \mathbb{N}^*$.

$$\dfrac{\dfrac{}{\vdash [\text{¤}]\mathtt{TRUE} \to C\varphi_1}\,MC1_1 \qquad \dfrac{}{\vdash C_G\varphi_1 \to E_G^n\varphi_1}\,ComImpPartIt}{\vdash [\text{¤}]\mathtt{TRUE} \to E_G^n\varphi_1}\,Cut$$

■

LEMMA 5 (PointImpProgr). $\forall n \geq k \ \ \forall j \in \{1,...,k+1\}, \quad \vdash [\text{¤}]\mathtt{TRUE} \to [*]^{j-1}E_G^{n-j+1}\varphi_j$

Proof. Let $n \geq k$. By induction on $j \in \{1,...,k+1\}$:

- Initialization : $j = 1$,

$$\dfrac{}{\vdash [\text{¤}]\mathtt{TRUE} \to E_G^n\varphi_1}\,PointImpPartIt$$

- Heredity : Let $j \in \{1,...,k\}$,

$$\dfrac{\dfrac{\dfrac{}{\vdash [\text{¤}]\mathtt{TRUE} \to [*]^{j-1}E_G^{n-j+1}\varphi_j}\,\mathtt{HYP\text{-}REC}}{\vdash [\text{¤}]\mathtt{TRUE} \to [*]^{j-1}E_GE_G^{n-j}\varphi_j}\,Id \qquad \dfrac{\dfrac{}{\vdash E_GE_G^{n-j}\varphi_j \to [*]E_G^{n-j}\varphi_{j+1}}\,MultGainConn}{\vdash [*]^{j-1}E_GE_G^{n-j}\varphi_j \to [*]^jE_G^{n-j}\varphi_{j+1}}\,(j-1)*Dist}{\vdash [\text{¤}]\mathtt{TRUE} \to [*]^jE_G^{n-j}\varphi_{j+1}}\,Cut$$

■

From those lemma we get the following ones
<u>With $j = k+1$</u>

LEMMA 6 (ResInter$_1$). $\forall n \geq k, \quad \vdash [\text{¤}]\mathtt{TRUE} \to [*]^kE_G^{n-k}\varphi_{k+1}$

<u>With $n = k+1$</u>

LEMMA 7 (ResInter$_2$). $\vdash [\text{¤}]\mathtt{TRUE} \to [*]^kE_G\varphi_{k+1}$

And the *muddy children puzzle* theorem comes out (almost) easily.

THEOREM 8 (Concl). $\vdash \forall k \in \mathbb{N} \ \ \forall i \in G, \quad [\text{¤}]\mathtt{TRUE} \to [*]^k(m_i \to K_im_i)$

Proof.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\vdash [\text{¤}]\mathtt{TRUE} \to [*]^kE_G\varphi_{k+1} \wedge (m_i \to K_i(\psi_k \vee \psi_{k+1}))}\,ResInter_2\&MC1_2}{\vdash [\text{¤}]\mathtt{TRUE} \to [*]^kE_G\varphi_{k+1} \wedge [*]^k(m_i \to K_i(\psi_k \vee \psi_{k+1}))}\,PERS_{MC1_2}}{\vdash [\text{¤}]\mathtt{TRUE} \to [*]^k(E_G\varphi_{k+1} \wedge (m_i \to K_i(\psi_k \vee \psi_{k+1})))}\,*/\wedge Dist}{\vdash [\text{¤}]\mathtt{TRUE} \to [*]^k(K_i\varphi_{k+1} \wedge (m_i \to K_i(\psi_k \vee \psi_{k+1})))}\,(E_Gp \to K_ip)}{\vdash [\text{¤}]\mathtt{TRUE} \to [*]^k(m_i \to K_i\varphi_{k+1} \wedge K_i(\psi_k \vee \psi_{k+1}))}\,(a \wedge (b \to c) \to (b \to a \wedge c))}{\vdash [\text{¤}]\mathtt{TRUE} \to [*]^k(m_i \to K_i(\varphi_{k+1} \wedge (\psi_k \vee \psi_{k+1})))}\,K/\wedge Dist}{\vdash [\text{¤}]\mathtt{TRUE} \to [*]^k(m_i \to K_i((\varphi_{k+1} \wedge \psi_k) \vee (\varphi_{k+1} \wedge \psi_{k+1})))}\,\wedge/\vee Dist}{\vdash [\text{¤}]\mathtt{TRUE} \to [*]^k(m_i \to K_i((\varphi_{k+1} \wedge \psi_k) \vee \psi_{k+1}))}\,(\varphi_k \wedge \psi_k \to \psi_{k+1})}{\vdash [\text{¤}]\mathtt{TRUE} \to [*]^k(m_i \to K_i(\bot \vee \psi_{k+1}))}\,(EXCLU_{\varphi\psi})}{\vdash [\text{¤}]\mathtt{TRUE} \to [*]^k(m_i \to K_i\psi_{k+1})}\,(\bot \vee p \to p)}{\vdash [\text{¤}]\mathtt{TRUE} \to [*]^k(m_i \to K_im_i)}\,MC2$$

■

6 The dynamic logic of common knowledge in Coq

6.1 What is Coq?

Coq is a proof assistant, i.e., a program which verifies step by step the validity of a mathematical proof given by the user. In logic, it is generally not obvious to follow a hand-made proof and to determine whether it is right or wrong [22]. A proof assistant, such as Coq, becomes a necessary tool if one chooses to be absolutely sure of a result.

Moreover, Coq is a very good means to build proofs. Indeed, managing a proof step by step, as required by a proof assistant, allows us understanding in a very precise way what is done and what has to be done to complete a proof. Coq is also a way to reach a good formalism as it requires from the user to define exactly all what he manipulates. Into [11], we reports how we found a flaw in well accepted axiomatization.

6.2 Implementation of $\mathcal{T}_G^{C[\alpha]}$ in Coq

This implementation is based on another implementation, namely this of logic of common knowledge done by the second author who implemented all the epistemic multi-agent logic with common knowledge (system $\mathcal{T}_G^C$), of which a Coq file is available on the web:

`http://perso.ens-lyon.fr/pierre.lescanne/COQ/EpistemicLogic.v8`.

This paper comes out with its own Coq file:

`http://perso.ens-lyon.fr/pierre.lescanne/COQ/EpistemicAndDynamicLogic.v`

which implements the whole system $\mathcal{T}_G^{C[\alpha]}$ and a complete proof of the *muddy children puzzle* theorem **Concl**.

6.3 Why this implementation?

The first aim of this implementation was to ensure a reader that the proof is totally checkable. This lead to a proof of nearly 1100 lines of Coq code, where every lemma is the direct translation of the hand-made proof for a maximal legibility. We do not claim that proof are readable as they would be in an English paper, a certain technicality is required for giving all the detail of the proof; however we claim that the statements of the lemmas as easily readable.

As an added value, this implementation allows any future development by adding axioms or new modalities. This makes our work flexible and reusable.

7 Conclusion

The proof theoretic approach we have used in this paper combines easily epistemic and dynamic logics together, thanks to a general epistemic-dynamic axiom (**KT1**). **(KT1)** involves a commutativity between epistemic modality and a dynamic modality. In the current implementation of **(KT1)**, type is not used to check whether the axiom is only invoked on purely epistemic propositions. In a future implementation, we will create a new type *epistemic proposition* on which (KT1) can only be invoked.

After manipulating the logical system presented in this paper with the proof assistant Coq, we feel that it is quite simple and intuitive. It only uses axioms and rules from classical logic plus a few additional axioms and rules. Statements can be made in a language close to this of the hand proof.

In epistemic-dynamic logic, there are events. In a formal statement, an event becomes a dynamic modality which transforms a proposition that describes the world before the event into a proposition that describes the world after that event. Said otherwise a dynamic modality transforms properties into others. Here we have limited our work to epistemic events which only transform agent knowledge, but this is not a big restriction, as this is what happens most of the time.

We notice that we had to adapt the system for the specific situation generated by the *muddy children puzzle*. But this is no so different from situation where classical logic or another system is used. However, conceptual tools or practical tools (for instance implemented in Coq) could be built to ease the task of the person who mechanizes a proof.

Acknowledgment

We would like to thank Stéphane Le Roux, for discussion and advice about COQ.

BIBLIOGRAPHY

[1] R. Aumann and S. Hart, editors. *Handbook of Game Theory*, volume 2, chapter Common knowledge, pages 1437–1496. Elsevier, Amsterdam, 1994.

[2] A. Baltag. A logic of epistemic actions. In W. van der Hoek, J.-J. Meyer, and C. Witteveen, editors, *Proceedings of the ESSLLI 1999 workshop on Foundations and Applications of Collective Agent-Based Systems*, Utrecht University, 1999.

[3] A. Baltag, L. Moss, and S. Solecki. The logic of public announcements, common knowledge and private suspicion. In *Proc. of TARK*, pages 43–56. Morgan Kaufmann Publishers, 1998.

[4] Judy Crow, Sam Owre, John Rushby, Natarajan Shankar, , and Mandayam Srivas. A tutorial introduction to PVS, April 1995.

[5] R. Fagin, J. Y. Halpern, Y. Moses, and M. Y. Vardi. *Reasoning about Knowledge*. The MIT Press, 1995.

[6] Joseph Y. Halpern and Yoram Moses. Knowledge and common knowledge in a distributed environment. In *PODC '84: Proceedings of the third annual ACM symposium on Principles of distributed computing*, pages 50–61, New York, NY, USA, 1984. ACM Press.

[7] D. Harel. *First-Order Dynamic Logic*, volume 68 of *Lecture Notes in Computer Science*. Springer-Verlag, 1979.

[8] David Harel, Jerzy Tiuryn, and Dexter Kozen. *Dynamic Logic*. MIT Press, Cambridge, MA, USA, 2000.

[9] Matt Kaufmann, J. Strother Moore, and Panagiotis Manolios. *Computer-Aided Reasoning: An Approach*. Kluwer Academic Publishers, Norwell, MA, USA, 2000.

[10] D. Lehmann. Knowledge, common knowledge and related puzzles (extended summary). In *PODC '84: Proceedings of the third annual ACM symposium on Principles of distributed computing*, pages 62–67, New York, NY, USA, 1984. ACM Press.

[11] P. Lescanne. Mechanizing epistemic logic with Coq. Research Report RR2004-27, LIP-ENS de Lyon, 2004. An updated version is avalaible at
`http://perso.ens-lyon.fr/pierre.lescanne/PUBLICATIONS/CommonKnowledge.pdf`.

[12] Hector J. Levesque and Gerhard Lakemeyer. *The Logic of Knowledge Bases*. MIT Press, 2001.

[13] J. McCarthy, M. Sato, T. Hayashi, and S. Igarashi. On the model theory of knowledge. Technical Report AIM-312, Stanford University, 1977.

[14] J-J Ch. Meyer and W van der Hoek. *Epistemic Logic for Computer Science and Artificial Intelligence*, volume 41 of *Cambridge Tracts in Theoretical Computer Science*. Cambridge University Press, 1995.

[15] Tobias Nipkow, Lawrence C. Paulson, and Markus Wenzel. *Isabelle/HOL — A Proof Assistant for Higher-Order Logic*, volume 2283 of *LNCS*. Springer, 2002.

[16] A. Pollack *et.al.* The LEGO Proof Assistant, 2001.

[17] Christophe Raffalli. The PhoX Proof Assistant. `http://www.lama.univ-savoie.fr/~RAFFALLI/phox.html`, 2005.

[18] HOL Team. The HOL System DESCRIPTION, September 2005. Kananaskis release.

[19] Johan van Benthem. *Exploring Logical Dynamics*. CLSI Publications, 1996.

[20] Johan van Benthem. Games in dynamic epistemic logic. *Bulletin of Economic Research*, 53(4):219–248, 2001.

[21] Hans P. van Ditmarsch, Wiebe van der Hoek, and Barteld P. Kooi. Dynamic epistemic logic with assignment. In *AAMAS*, pages 141–148, 2005.

[22] Eric W. Weisstein. Kepler conjecture. From MathWorld–A Wolfram Web Resource,
`http://mathworld.wolfram.com/KeplerConjecture.html`.

Jérôme Puisségur
Ecole Normale Supérieure de Lyon
46, allée d'Italie, 69364 Lyon 07, FRANCE
jerome.puissegur@ens-lyon.fr

Pierre Lescanne
46, allée d'Italie, 69364 Lyon 07, FRANCE
pierre.lescanne@ens-lyon.fr

Excerpts of the Coq script

Here is the statement of the main lemmas and the proof in COQ of the last theorem **Concl**.

```
Lemma GainConn :
  forall  (G: list nat) (i j : nat),
    |- E (i::G) (E (i::G) (phi j)) ==> [*] (E (i::G) (phi (j+1))).


Lemma MultGainConn :
  forall (G: list nat) (m i j : nat),
    |- F ((m+1)+1) (i::G) (phi j) ==> [*] (F (m+1) (i::G) (phi (j+1))).


Lemma ComImpPartIt :
  forall (p:proposition) (n:nat) (G: list nat),
    |- C G p ==> F n G p.

Lemma PointImpPartIt :
  forall (G:list nat) (m:nat),
    |- Point TRUE ==> F m G (phi 1).


Lemma PointImpProgr :
  forall (G:list nat) (i j n:nat),
    |- Point TRUE ==> [*]<:j:> (F (n+1) (i::G) (phi (j+1) ) ).


Theorem Concl :
  forall (G:list nat) (i j k : nat),
    In i (j::G) ->
    |- Point TRUE ==> [*]<:k:> (muddy i ==> (K i (muddy i))).
Proof.
intros.
apply Transitivity_of_Imp with ([*]<:k:> (muddy i ==> (K i (psi (k+1))))).
apply Transitivity_of_Imp with ([*]<:k:> (muddy i ==> (K i (FALSE V psi (k+1))))).
apply Transitivity_of_Imp with ([*]<:k:> (muddy i ==> (K i ((phi (k+1) & psi (k)) V psi (k+1))))).
apply Transitivity_of_Imp with ([*]<:k:> (muddy i ==> (K i ((phi (k+1) & psi (k)) V (phi (k+1) & psi (k+1)))))
apply Transitivity_of_Imp with ([*]<:k:> (muddy i ==> (K i (phi (k+1) & (psi (k) V psi (k+1)))))).
apply Transitivity_of_Imp with ([*]<:k:> (muddy i ==> (K i (phi (k+1)) & K i (psi (k) V psi (k+1))))).
apply Transitivity_of_Imp with ([*]<:k:> (K i (phi (k+1)) & (muddy i ==> (K i (psi (k) V psi (k+1)))))).
apply Transitivity_of_Imp with ( ([*]<:k:> (E (j::G) (phi (k+1))) & (muddy i ==> (K i (psi (k) V psi (k+1)))))
apply Transitivity_of_Imp with ( ([*]<:k:> (E (j::G) (phi (k+1)))) & [*]<:k:> (muddy i ==> (K i (psi (k) V psi
apply Transitivity_of_Imp with ( ([*]<:k:> (E (j::G) (phi (k+1)))) & (muddy i ==> (K i (psi (k) V psi (k+1))))
apply Transitivity_of_Imp with ( ([*]<:k:> (E (j::G) (phi (k+1)))) & (muddy i ==> (K i (psi (k) V psi (k+1))))
apply Transitivity_of_Imp with ( Point TRUE & Point TRUE).
apply AND_idempotence.
apply rule_glueAND.
apply Transitivity_of_Imp with ([*]<:k:> (F 1 (j::G) (phi (k+1)))).
apply PointImpProgr.
apply rule_STAR_Imp.
unfold F.
apply I.
apply MC1_2.
apply I.
apply rule_glueAND.
apply I.
apply PERS_STAR_MC1_2.
apply ANDSTAR.
apply rule_STAR_Imp.
apply Transitivity_of_Imp with ( (E (j::G) (phi (k +1))) & (muddy i ==> K i (psi k V psi (k +1)))).
apply I.
apply rule_AND_right_regular.
apply E_Ki.
trivial.
apply rule_STAR_Imp.
apply TOOL2.
apply rule_STAR_Imp.
apply MP with (K i (phi (k +1)) & K i (psi k V psi (k +1)) ==> K i (phi (k +1) & (psi k V psi (k +1)))).
apply B.
apply AND_Ki.
apply rule_STAR_Imp.
apply MP with ( (K i (phi (k +1) & (psi k V psi (k +1)))) ==> (K i ((phi (k +1) & psi k) V (phi (k +1) & ps
apply B.
apply rule_K_K_rev.
apply K_rule.
apply dist_AND.
apply rule_STAR_Imp.
apply MP with (K i (phi (k +1) & psi k V (phi (k +1) & psi (k +1))) ==> K i (phi (k +1) & psi k V psi (k +
apply B.
apply rule_K_K_rev.
apply K_rule.
apply rule_OR_left_regular.
apply AND2.
apply rule_STAR_Imp.
apply MP with (K i (phi (k +1) & psi k V psi (k +1)) ==> K i (FALSE V psi (k +1))).
apply B.
apply rule_K_K_rev.
apply K_rule.
apply rule_OR_right_regular.
apply phi_psi_Exclu.
apply rule_STAR_Imp.
apply MP with (K i (FALSE V psi (k +1)) ==> K i (psi (k +1))).
apply B.
apply rule_K_K_rev.
apply K_rule.
apply Transitivity_of_Imp with (psi (k+1) V FALSE).
apply OR_comm.
apply OR_FALSE.
apply rule_STAR_Imp.
apply MP with (muddy i ==> K i (psi (k+1)) ==> K i (muddy i)).
apply Hilbert_S.
apply MC3.
Qed.
```