Expedition in the Update Universe

Guillaume Aucher

Expedition in the Update Universe

Guillaume Aucher

Univ Rennes, CNRS, IRISA
263, Avenue du Général Leclerc
35042 Rennes Cedex, France
guillaume.aucher@irisa.fr

Abstract. Dynamic epistemic logic (DEL) is a logic dealing with knowledge and belief change based on the concepts of event model and product update. The product update accounts for the way we update our knowledge and beliefs about situations when events occur. However, DEL does not account for the way we update our knowledge and beliefs about events when other events occur. Indeed, events are assumed to occur instantaneously in DEL and this idealization precludes to study this kind of update. We provide a logical analysis of updates without this assumption. It leads us to identify a graph structure for events based on their relative dependence of occurence and to introduce a generic product update. The DEL product update is a specific instance of this generic product update.

1 Introduction

It is commonly believed that only our knowledge and beliefs *about situations* can be updated, whereas our knowledge and beliefs *about events* can*not*. This common belief implies that what we represent has always a manichean nature: on the one hand we have situations and on the other hand we have events, and the occurrence of events update our knowledge and beliefs about situations. The most prominent logical formalisms of knowledge representation and reasoning are all based on this approach [13, 14].

As we shall see, this manichean distinction is not fine enough to account for the dynamics of knowledge and beliefs. In fact, our knowledge and beliefs *about events* can also be updated and this can be demonstrated by the following scenario. Assume that there are two barrels of wine: barrel 1 and barrel 2. Barrel 1 is being filled with wine but Ann and Bob do not know which of these barrels is being filled. Clearly, this filling of barrel 1 with wine is an event, perceived identically by Ann and Bob. Now, assume that the wine waiter privately announces to Bob that it is actually barrel 1 which is being filled. Again, clearly, this announcement is another event, perceived differently by Ann and Bob. Then, as a result of this second event, Bob knows that barrel 1 is being filled but Ann still does not know which barrel is being filled. So, Bob's knowledge and beliefs of the first event (the filling with wine) has been updated by his perception of the second event (the announcement).

This scenario cannot be directly represented in DEL because only situations, and not events, can be updated by events. This stems from the assumption that events are implicitly assumed to be instantaneous in DEL, thus leading to a new situation, and our perception of an event can be updated only if this event lasts long enough, obviously. Hence, this idealization precludes the study of important logical dynamics like the one of the barrel example. However, this assumption can be perfectly removed from the DEL framework. Once we remove it, we realize that the fact that events and not only situations can be updated by other events is only the 'tip of the iceberg' and many other logical dynamics start to appear. In particular, we realize that events have an internal and rich structure based on their relative dependence of occurrence. Moreover, this structure constrains and determines the updates which are possible and a generic kind of product updates can then be identified. A contribution of this article is to provide a formal account of these logical dynamics by eliciting a series of principles. These principles will guide us for defining our formal framework.

Organization of the article. In Section 2, we briefly recall DEL. In Section 3, we analyze the structure of events by means of various examples and we elicit a number of intuitive principles about events. We use them in Section 4 for motivating our formal definitions of event structures and generic product updates. We end Section 4 by providing an example of scenario which cannot (or hardly) be modeled in DEL. We conclude and discuss our approach in Section 5.

2 Dynamic Epistemic Logic

Dynamic epistemic logic (DEL) is a relatively recent non-classical logic introduced by [4]. It extends ordinary modal epistemic logic [11] by the inclusion of *event models* to describe actions/events, and a *product update* operator that defines how epistemic models are updated as the consequence of executing actions described through event models. For more details about DEL, see [3, 14].

2.1 Epistemic Models

In the rest of this article, $\mathbb{A} := \{1, \ldots, N\}$ is a finite set of indices called *agents* and $\mathbb{P}_0$ is a set of propositional letters called *atomic facts* which describe static situations.

Definition 1 (Language $\mathcal{L}(\mathbb{P})$). *Let $\mathbb{P}$ be a set of propositional letters. We define the* language $\mathcal{L}(\mathbb{P})$ *inductively by the following grammar in BNF:*

$$\mathcal{L}(\mathbb{P}): \quad \varphi \quad ::= \quad p \quad | \quad \neg\varphi \quad | \quad (\varphi \wedge \varphi) \quad | \quad \Box_j \varphi$$

where p ranges over $\mathbb{P}$ and j over $\mathbb{A}$. When $\mathbb{P} = \mathbb{P}_0$, $\mathcal{L}(\mathbb{P}_0)$ is called the epistemic language. *We will use the following abbreviations: $\varphi \vee \psi := \neg(\neg\varphi \wedge \neg\psi)$ and $\varphi \to \psi := \neg\varphi \vee \psi$. To save parenthesis, we use the following ranking of binding strength: $\Box_j, \neg, \wedge, \vee, \to$ (*i.e., *$\Box_j$ binds stronger than $\neg$, which binds stronger than $\wedge$,* etc.*). For example, $\Box_j \neg p \wedge q \to r \vee s$ means $((\Box_j(\neg p)) \wedge q) \to (r \vee s)$. If $E = \{\varphi_1, \ldots, \varphi_n\}$, we write $\bigwedge E := \varphi_1 \wedge \ldots \wedge \varphi_n$ and $\bigvee E := \varphi_1 \vee \ldots \vee \varphi_n$.*

A (pointed) epistemic model $(\mathcal{M}, w)$ represents how the actual world represented by w is perceived by the agents. Atomic facts are used to state properties of this actual world. Intuitively, wR_jv means that in world w agent j considers that world v might be the actual world w.

Definition 2 ($\mathbb{P}$–model and epistemic model). *Let $\mathbb{P}$ be a set of propositional letters. A* $\mathbb{P}$–model *is a tuple $\mathcal{M} = (W, R, V)$ where:*

- *W is a non-empty set;*
- *$R : \mathbb{A} \to 2^{W \times W}$ assigns an* accessibility relation *to each agent;*
- *$V : \mathbb{P} \to 2^W$ is a* valuation *which assigns a subset of W to each atomic event of $\mathbb{P}$.*

If $w, v \in W$, we write wR_jv for $(w, v) \in R(j)$, and $R_j(w)$ denotes $\{v \in W \mid wR_jv\}$. We write $w \in \mathcal{M}$ for $w \in W$ and $(\mathcal{M}, w)$ is called a pointed $\mathbb{P}$–model. *When $\mathbb{P} = \mathbb{P}_0$, $\mathcal{M}$ is called an* epistemic model *and $(\mathcal{M}, w)$ is called a* pointed epistemic model *(w often represents the actual world). The class of pointed $\mathbb{P}$–models is denoted $\mathcal{C}(\mathbb{P})$.*

As one can easily notice, a $\mathbb{P}$–model is an 'ordinary' Kripke model. Then, the epistemic language can be used to describe and state properties of epistemic models.

Definition 3 (Epistemic logic). *Let $\mathbb{P}$ be a set of propositional letters. We define the* satisfaction relation $\models \subseteq \mathcal{C}(\mathbb{P}) \times \mathcal{L}(\mathbb{P})$ *inductively as follows. In the truth conditions below, $(\mathcal{M}, w) \in \mathcal{C}(\mathbb{P})$ is any pointed $\mathbb{P}$–model and $\varphi, \psi \in \mathcal{L}(\mathbb{P})$.*

$$
\begin{array}{lll}
\mathcal{M}, w \models p & \textit{iff} & w \in V(p) \\
\mathcal{M}, w \models \neg\psi & \textit{iff} & \textit{it is not the case that } \mathcal{M}, w \models \psi \\
\mathcal{M}, w \models \varphi \wedge \psi & \textit{iff} & \mathcal{M}, w \models \varphi \textit{ and } \mathcal{M}, w \models \psi \\
\mathcal{M}, w \models \Box_j\varphi & \textit{iff} & \textit{for all } v \in R_j(w), \textit{ we have } \mathcal{M}, v \models \varphi
\end{array}
$$

We write $\mathcal{M} \models \varphi$ when $\mathcal{M}, w \models \varphi$ for all $w \in \mathcal{M}$. If $S \subseteq \mathcal{L}(\mathbb{P})$, we write $\mathcal{M}, w \models S$ ($\mathcal{M} \models S$) when for all $\varphi \in S$, $\mathcal{M}, w \models \varphi$ (resp. $\mathcal{M} \models \varphi$). The triple $(\mathcal{L}(\mathbb{P}), \mathcal{C}(\mathbb{P}), \models)$ is a logic called the epistemic logic based on $\mathbb{P}$.

The formula $\Box_j\varphi$ reads as "agent j believes φ". Its truth conditions are defined in such a way that agent j believes φ holds in a possible world when φ holds in all the worlds agent j considers possible in this possible world.

Example 1 ('Barrel' example). Assume that there are two agents Ann and Bob and that there are two barrels of wine: barrel 1 and barrel 2. So, we have $\mathbb{A} := \{A, B\}$ with A standing for Ann and B standing for Bob, and $\mathbb{P}_0 := \{p_0, q_0\}$ with p_0 standing for 'barrel 1 is full' and q_0 for 'barrel 2 is full'. The situation is such that barrel 1 is not full and barrel 2 is full, but Ann and Bob do not know which one is full. This situation is depicted in the pointed epistemic model $(\mathcal{M}_0, w_0) = (W, R, V, w_0)$ of Fig. 1 (left). We have $W = \{w_0, v_0\}$ and the circled world w represents the actual world. Possible worlds are labeled by the propositional letters of $\mathbb{P}_0$ that are true at these worlds. The accessibility relations are

represented by arrows indexed by A or B: an arrow indexed by A (or B) from a world u to a world u' means that $(u, u') \in R_A$ (resp. $(u, u') \in R_B$). So, we have $\mathcal{M}_0, w_0 \models \Box_A(p_0 \vee q_0) \wedge \Box_B(p_0 \vee q_0)$: 'Ann and Bob both know that one of the two barrels is full'. The situation where both barrels are full and both Ann and Bob know it is represented in the pointed epistemic model $(\mathcal{N}_0, u_0)$ of Fig. 1 (right).

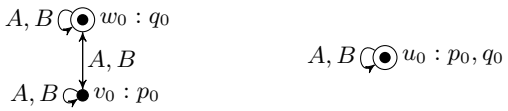

Fig. 1: Pointed epistemic models $(\mathcal{M}_0, w_0)$ and $(\mathcal{N}_0, u_0)$

2.2 Event Models

A pointed event model $(\mathcal{E}, e)$ represents how the actual event represented by e is perceived by the agents. Intuitively, $eR_j^\alpha f$ means that while the possible event represented by e is occurring, agent j considers possible that the possible event represented by f is actually occurring.

Definition 4 (Event model). *An* event model *is a tuple* $\mathcal{E} = (W^\alpha, R^\alpha, \textsc{Pre}, \textsc{Post})$ *where:*

- W^α *is a finite and non-empty set of* possible events*;*
- $R^\alpha : \mathbb{A} \to 2^{W^\alpha \times W^\alpha}$ *assigns an* accessibility relation *to each agent;*
- $\textsc{Pre} : W^\alpha \to \mathcal{L}(\mathbb{P}_0)$ *is a* precondition function *which assigns to each possible event a formula of* $\mathcal{L}(\mathbb{P}_0)$*;*
- $\textsc{Post} : W^\alpha \to (\mathbb{P}_0 \to \mathcal{L}(\mathbb{P}_0))$ *is a* postcondition function *which assigns to each possible event a function from* $\mathbb{P}_0$ *to* $\mathcal{L}(\mathbb{P}_0)$*.*

If $e, f \in W^\alpha$*, we write* $eR_j^\alpha f$ *for* $(e, f) \in R^\alpha(j)$*, and* $R_j^\alpha(e)$ *denotes* $\{f \in W^\alpha \mid eR_j^\alpha f\}$*. We write* $e \in \mathcal{E}$ *for* $e \in W^\alpha$*, and* $(\mathcal{E}, e)$ *is called a* pointed event model *(*e *often represents the actual event).*

Our definition of event models corresponds to the definition of [16]. It embeds the definition of [15] based on the notion of substitutions.

Example 2 ('barrel' example). Assume that barrel 1 is being filled with wine. Ann and Bob do not know whether it is barrel 1 or barrel 2 which is being filled. This event and its perception by the agents Ann and Bob is represented in Fig. 2 (left). We use the same notations for the possible events and the accessibility relations as in Fig. 1. The preconditions are such that $\textsc{Pre}(e) = \neg p_0$ (and $\textsc{Pre}(f) = \neg q_0$): barrel 1 (resp. barrel 2) is not full when barrel 1 (resp. barrel

2) is being filled with wine. The postconditions are such that $\textsc{Post}(e)(p_0) = \top$ and $\textsc{Post}(e)(q_0) = q_0$ (and $\textsc{Post}(f)(p_0) = p_0$ and $\textsc{Post}(f)(q_0) = \top$): when the filling of barrel 1 (resp. barrel 2) terminates, barrel 1 (resp. barrel 2) is full, the other barrel remaining in the same state.

Fig. 2: Pointed event models $(\mathcal{E}, e)$ and $(\mathcal{F}, g)$

2.3 Product Update

The DEL product update of [4] is defined as follows. This update yields a new epistemic model $\mathcal{M} \otimes \mathcal{E}$ representing how the new situation which was previously represented by $\mathcal{M}$ is perceived by the agents after the occurrence of the event represented by $\mathcal{E}$.

Definition 5 (Product update). *Let $\mathcal{M} = (W, R, V)$ be an epistemic model and let $\mathcal{E} = (W^\alpha, R_1^\alpha, \ldots, R_N^\alpha, \textsc{Pre}, \textsc{Post})$ be an event model. We define the epistemic model $\mathcal{M} \otimes \mathcal{E} = (W^\otimes, R^\otimes, V^\otimes)$ as follows (with the proviso that $W^\otimes$ is not empty): for all $p \in \mathbb{P}_0$ and all $j \in \mathbb{A}$,*

- $W^\otimes := \{(w, e) \in W \times W^\alpha \mid \mathcal{M}, w \models \textsc{Pre}(e)\}$;
- $(v, f) \in R_j^\otimes(w, e)$ *iff* $v \in R_j(w)$ *and* $f \in R_j^\alpha(e)$;
- $(w, e) \in V^\otimes(p)$ *iff* $\mathcal{M}, w \models \textsc{Post}(e)(p)$.

If $(\mathcal{M}, w)$ and $(\mathcal{E}, e)$ are pointed epistemic and event models. If $\mathcal{M}, w \models \textsc{Pre}(e)$, we define the pointed epistemic model $(\mathcal{M}, w) \otimes (\mathcal{E}, e) = (\mathcal{M} \otimes \mathcal{E}, (w, e))$.

Example 3. The product update of $(\mathcal{M}_0, w_0)$ by $(\mathcal{E}, e)$ results in the epistemic model represented on the extreme right of Fig. 3. This epistemic model is in fact bisimilar to the epistemic model $(\mathcal{N}_0, u_0)$ of Fig. 1. In this epistemic model, we have that $\mathcal{N}_0, u_0 \models (p_0 \wedge q_0) \wedge \Box_A(p_0 \wedge q_0) \wedge \Box_B(p_0 \wedge q_0)$: 'both barrels are full and Ann and Bob both know it'.

2.4 DEL

Definition 6 ([4]). *We define the* language $\mathcal{L}_{\text{DEL}}$ *inductively by the following grammar in BNF:*

$$\mathcal{L}_{\text{DEL}}: \quad \varphi \quad ::= \quad p \quad | \quad \neg\varphi \quad | \quad (\varphi \wedge \varphi) \quad | \quad \Box_j\varphi \quad | \quad [\mathcal{E}, e]\varphi$$

where p ranges over $\mathbb{P}_0$, j over $\mathbb{A}$ and $(\mathcal{E}, e)$ over $\mathcal{C}^\alpha$.

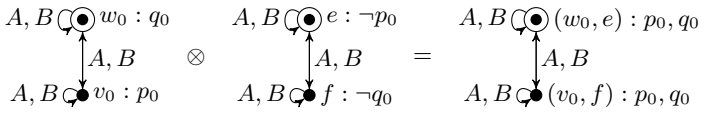

Fig. 3: Product update of $(\mathcal{M}_0, w_0)$ by $(\mathcal{E}, e)$

Intuitively, $[\mathcal{E}, e]\varphi$ reads as 'φ will hold after the occurence of the event represented by $(\mathcal{E}, e)$' and $\langle \mathcal{E}, e\rangle\varphi$ reads as 'the event represented by $(\mathcal{E}, e)$ is executable in the current situation and φ will hold after its execution'.

Definition 7 (Dynamic epistemic logic). *We define the* satisfaction relation $\models \subseteq \mathcal{C}(\mathbb{P}_0) \times \mathcal{L}_{\text{DEL}}$ *inductively as follows. In the truth conditions below,* $(\mathcal{M}, w) \in \mathcal{C}(\mathbb{P})$ *is any pointed* $\mathbb{P}$*–model and* $\varphi \in \mathcal{L}(\mathbb{P})$.

$$\mathcal{M}, w \models [\mathcal{E}, e]\varphi \text{ iff if } \mathcal{M}, w \models \textsc{Pre}(e) \text{ then } (\mathcal{M}, w) \otimes (\mathcal{E}, e) \models \varphi$$

The other truth conditions for the Boolean and modal cases are identical to those of Definition 3. The triple $\left(\mathcal{L}_{\text{DEL}}, \mathcal{C}(\mathbb{P}_0), \models\right)$ *is a logic called* dynamic epistemic logic *(DEL).*

Proposition 1 ([4]). *DEL is as expressive as the epistemic logic based on* $\mathbb{P}_0$.

3 Analyzing the Structure of Events

In this section, we discuss and analyze two examples from which we elicit a series of principles about events. These principles, whose some of them are obvious, are introduced to motivate the formal definitions of Section 4. They will allow us to show that the dependence graph of Definition 8 can be any directed graph, and not necessarily a tree or a bipartite graph for example.

In philosophy, the exact definition of an event is a moot topic [17] and we do not intend to take any stance in this debate. Here, we are not so much interested in the nature of events but rather in their logical and internal structure. Our examples will always be chosen so that events are indisputable events.

3.1 The 'Barrel' Example

Assume at time t_1 that there are two barrels of wine: barrel 1 and barrel 2. Barrel 2 is full and barrel 1 is being filled with wine, but Ann and Bob do not know which of these barrels is being filled. However, they know that one of them is full (and therefore cannot be filled) but they do not know which one. Clearly, this filling of barrel 1 with wine is an event, perceived identically by Ann and Bob. We consider two Scenarios from which we are going to elicit a series of principles.

Scenario 1. Assume that during barrel 1 is filled the wine waiter privately announces to Bob that it is actually barrel 1 which is being filled. Again, clearly, this announcement is another event, perceived differently by Ann and Bob. Then, as a result of this second event, Bob knows that barrel 1 is being filled but Ann still does not know which barrel is being filled. So, Bob's knowledge and beliefs of the first event (the filling with wine) has been updated by his perception of the second event (the announcement).

It is important to modify the event model while it is being executed, as opposed to modifying the Kripke model that is obtained by applying the event model because if we sever the relation between (w, e_1) and (w, e_2) after the update, this means that the event modeled by e_1 has already ended and therefore the update is on the resulting situation and not on the perception of the event. What we want to model is an update about the perception of the event itself while this event is occuring, not after. This example leads us to state the following principle:

PRINCIPLE 1: Our knowledge about ongoing events can be updated by the perception of other events.

Moreover, this announcement depends on the fact that barrel 1 is being filled and not on the precondition of this event, *i.e.* the fact that barrel 1 is not full. In this scenario, the nature of the event is "barrel 1 is being filled with wine", its precondition is "barrel 1 is not full" and its postcondition is "barrel 1 is full". This example entails that the very nature of events has to be taken into account when knowledge of events is updated by the perception of other events. This is captured by the following principle:

PRINCIPLE 2: The occurrence of events sometimes depends on the nature of other events and not on their preconditions.

Scenario 2. Assume at time t_2 that the wine waiter publicly announces that barrel 1 is not full. As a result of this announcement, at time t_3, Ann and Bob both know that barrel 2 *is* full. From this new piece of information, they can infer at time t_5 that it is barrel 1 which is being filled and not barrel 2 (since the latter is full). Hence, from this example, we can state the following principle:

PRINCIPLE 3: Our knowledge about a *situation* or an event can update our knowledge of another event while this other event is occuring.

Moreover, Ann and Bob may not make immediately the inference that it is barrel 1 which is being filled, but only as an afterthought at time t_4. Hence,

PRINCIPLE 4: After learning a new piece of information, we do not always update immediately our knowledge to take it into account.

Note that PRINCIPLE 4 is very much related to well-known problems in epistemic logic dealing with bounded rationality and logical omniscience (for more details on these problems, see [7, Chap. 9], [8, p. 157–168] or [9]). In fact, for some time, Ann and Bob may entertain the inconsistent possibility that barrel 2 is full and that at the same time it is being filled. So,

PRINCIPLE 5: We may consider possible at the same time that some event is occurring and that its precondition does not hold.

Formalizing the 'barrel' example. We can formalize the example by introducing the following sets of propositional letters:

- $\mathbb{P}_0 := \{p_0, q_0\}$, where p_0 stands for "Barrel 1 is full" and q_0 for "Barrel 2 is full".
- $\mathbb{P}_1 := \{p_1\}$, where p_1 stands for "The wine waiter truthfully announces that barrel 1 is not full".
- $\mathbb{P}_2 := \{p_2, q_2\}$, where p_2 stands for "Barrel 1 is being filled" and q_2 for "Barrel 2 is being filled".
- $\mathbb{P}_3 := \{p_3\}$, where p_3 stands for "The wine waiter truthfully announces that barrel 1 is being filled".

Then, we can represent the dependence between these sets of propositional letters by the graph $(\mathcal{P}, \mathcal{S})$ of Fig. 4 (where $\mathcal{P} := \{\mathbb{P}_0, \mathbb{P}_1, \mathbb{P}_2, \mathbb{P}_3\}$ and $\mathcal{S} \subseteq \mathcal{P} \times \mathcal{P}$ is defined in Fig. 4). An edge $(\mathbb{P}_i, \mathbb{P}_j) \in \mathcal{S}$ means that the events described by $\mathbb{P}_i$ depend on the events/situations described by $\mathbb{P}_j$. More precisely, an edge is set from $\mathbb{P}_i$ to $\mathbb{P}_j$ when the preconditions for the occurrence of any atomic event of $\mathbb{P}_i$ depends on the truth value of formula(s) of $\mathcal{L}(\mathbb{P}_j)$ or that the occurrence of the atomic events of $\mathbb{P}_i$ will affect in some way or another the occurrence of the atomic events of $\mathbb{P}_j$ or their knowledge and beliefs (to be more concrete, see all subsequent examples, and in particular the 'traffic lights' example).

Note that we have an arrow from $\mathbb{P}_0$ to $\mathbb{P}_2$. This arrow is motivated by the example that we used to introduce PRINCIPLE 3: our knowledge about a situation can also update our perception/knowledge about an ongoing event.

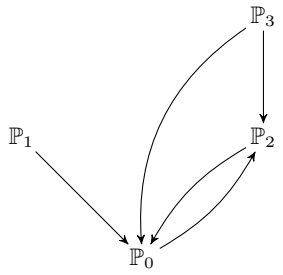

Fig. 4: Dependence graph for the 'barrel' example

For each edge $(\mathbb{P}', \mathbb{P}) \in \mathcal{S}$, we can define $(\mathbb{P}', \mathbb{P})$–preconditions functions $\text{PRE}_{\mathbb{P}',\mathbb{P}} : \mathbb{P}' \to 2^{\mathcal{L}(\mathbb{P})}$ and $(\mathbb{P}', \mathbb{P})$–postconditions functions $\text{POST}_{\mathbb{P}',\mathbb{P}} : \mathbb{P}' \to \left(\mathbb{P} -> 2^{\mathcal{L}(\mathbb{P})}\right)$. The reading of $\text{PRE}_{\mathbb{P}',\mathbb{P}}(p) := \varphi$ is 'the precondition of the atomic event p is φ'; the reading of $\text{POST}_{\mathbb{P}',\mathbb{P}}(q)(p) := \varphi$ is 'p holds after the occurrence of the atomic event q if, and only if, φ held before this occurrence'.

- $\textsc{Pre}_{\mathbb{P}_1,\mathbb{P}_0}(p_1) := \neg p_0$: the wine waiter can truthfully announce that barrel 1 is not full only if it is indeed not full.
  $\textsc{Post}_{\mathbb{P}_1,\mathbb{P}_0}(p_1)(p) := p$ for all $p \in \mathbb{P}_0$: the announcement of the wine waiter does not change the actual state of the world.
- $\textsc{Pre}_{\mathbb{P}_2,\mathbb{P}_0}(p_2) := \neg p_0$ and $\textsc{Pre}_{\mathbb{P}_2,\mathbb{P}_0}(q_2) := \neg q_0$: barrel 1 and barrel 2 can be filled only if they are not full.
  $\textsc{Post}_{\mathbb{P}_2,\mathbb{P}_0}(p_2)(p_0) := \top$ and $\textsc{Post}_{\mathbb{P}_2,\mathbb{P}_0}(p_2)(q_0) := q_0$: after the filling of barrel 1, it is full, the status of barrel 2 remains unchanged.
  $\textsc{Post}_{\mathbb{P}_2,\mathbb{P}_0}(q_2)(p_0) := p_0$ and $\textsc{Post}_{\mathbb{P}_2,\mathbb{P}_0}(q_2)(q_0) := \top$: after the filling of barrel 2, it is full, the status of barrel 1 remains unchanged.
- $\textsc{Pre}_{\mathbb{P}_3,\mathbb{P}_2}(p_3) := p_2$: the wine waiter can truthfully announce that barrel 1 is being filled only if it is indeed being filled.
  $\textsc{Post}_{\mathbb{P}_3,\mathbb{P}_2}(p_3)(p) := p$ for all $p \in \mathbb{P}_2$: the announcement of the wine waiter does not change the actual state of the world.
- $\textsc{Pre}_{\mathbb{P}_3,\mathbb{P}_0}(p_3) := \neg p_0$: the wine waiter can truthfully announce that barrel 1 is being filled only if it is not full (so that it can indeed be filled).
  $\textsc{Post}_{\mathbb{P}_3,\mathbb{P}_0}(p_3)(p) := p$ for all $p \in \mathbb{P}_0$: the announcement of the wine waiter does not change the actual state of the world.
- $\textsc{Pre}_{\mathbb{P}_0,\mathbb{P}_2}(p_0) := \neg p_2$ and $\textsc{Pre}_{\mathbb{P}_0,\mathbb{P}_2}(q_0) := \neg q_2$: if one of the barrels is full, it is not possible that it is being filled.
  $\textsc{Post}_{\mathbb{P}_0,\mathbb{P}_2}(p')(p) := p$ for all $p \in \mathbb{P}_2$ and $p' \in \mathbb{P}_0$.

Note that there is no arrows towards $\mathbb{P}_1$ nor $\mathbb{P}_3$. This is because these announcements are instantaneous, and therefore it is not possible that the agents' beliefs about them change *while* they are occurring, unlike the filling event of $\mathbb{P}_2$. This said, we could add arrows from, say, $\mathbb{P}_0$ to $\mathbb{P}_1$ if we considered that the agents can assess the truthfulness of the announcement, which can be a lie, before applying the product update on the event model, and may then revise it beforehand, based on their beliefs about the barrels.

3.2 The 'Traffic Lights' Example

We consider a naïve representation of a traffic lights system on a road. This example would be classically modeled by means of timed–automata [2], but we follow here the modeling approach of DEL to investigate what this example implies in term of representational requirements (formalized by our principles) for a DEL style modeling approach based on event models and product updates.

Assume that there are n traffic lights on a road. Each traffic light can be either 'green', 'yellow' or 'red' and only one of them at the same time. The color changes and goes from green via yellow to red and then back to green. Between any two of these states, a timer counts the time that elapses and eventually changes the traffic light from one state to the next after a certain amount of time. Then, each time the state of a light changes (from 'green' to 'yellow', from 'yellow' to 'red', or from 'red' to 'green'), the corresponding timer starts (timer 'yellow', timer 'red' or timer 'green'). Multiple timers run at the same time and they can be arbitrarily many. So,

PRINCIPLE 6: Arbitrary many events can occur at the same time and in parallel.

Moreover, we assume that there is a synchronization between the different traffic lights: when the 'red' timer of light k starts, the 'green' timer of traffic light $k+1$ ends and the traffic light $k+1$ goes to state 'yellow' (and then the yellow timer of traffic light $k+1$ starts). This synchronization is set in order to ease the flow of cars on the road so that cars do not stop at each traffic light.

When a pedestrian comes at traffic light k and presses the 'crossing button', the 'green' timer changes its timer mode and goes to another mode in order to shorten the amount of time that the pedestrian will have to wait. As a result, and in order to synchronize the other traffic lights on the road, the timer mode of traffic light $k+1$ also changes mode if its timer is currently in its usual 'green' mode, so as to keep the synchronization between the different traffic lights. (To be really precise, this change of timer mode of traffic light k should also affect the timer mode of traffic light $k-1$ in order to keep the system fully synchronized.) Hence,

PRINCIPLE 7: There can be an arbitrarily long chain of events, each event depending on the occurrence of the previous one.

Moreover, if the pedestrian presses the 'crossing button' of traffic light k when it is green, the new green timer mode will affect not only the green timer mode of traffic light $k+1$ but also the color of traffic light k (when the new green timer of traffic light k ends). Therefore,

PRINCIPLE 8: The occurrence of an event can have effects on multiple situations or types of events.

Formalizing the 'Traffic lights' example. We can formalize the example by introducing the following sets of propositional letters: for all $k \in \{1, \ldots, n\}$,

- $\mathbb{P}_k := \{\text{GTIMER}_k, \text{YTIMER}_k, \text{RTIMER}_k, \text{GTIMER}'_k\}$,
  where GTIMER_k (resp. YTIMER_k, RTIMER_k, GTIMER'_k) stands for "the green (resp. yellow, red, modified green) timer of traffic light k is running".
- $\mathbb{P}_{n+k} := \{\text{PRESS}_k\}$,
  where PRESS_k stands for "a pedestrian is pressing the crossing button of traffic light k while it is green".
- $\mathbb{P}_0 := \{\text{GREEN}_k, \text{YELLOW}_k, \text{RED}_k \mid k \in \{1, \ldots, n\}\}$,
  where GREEN_k (resp. YELLOW_k, RED_k) stands for "traffic light k is green (resp. yellow, red)".

Then, we can represent the dependence between these sets of propositional letters by the graph $(\mathcal{P}, \mathcal{S})$ of Fig. 5 (where $\mathcal{P} := \{\mathbb{P}_i \mid i \in \{0, \ldots, 2n\}\}$ and $\mathcal{S} \subseteq \mathcal{P} \times \mathcal{P}$ is defined in Fig. 5). An edge $(\mathbb{P}', \mathbb{P}) \in \mathcal{S}$ means that the events described by $\mathbb{P}'$ depend on the events/situations described by $\mathbb{P}$. We spell out the precondition and postcondition functions. We only do it for the edges of the form $(\mathbb{P}_k, \mathbb{P}_0)$ and $(\mathbb{P}_0, \mathbb{P}_k)$, where $k \in \{1, \ldots, n\}$. We define $(\mathbb{P}', \mathbb{P})$–preconditions functions $\text{PRE}_{\mathbb{P}',\mathbb{P}} : \mathbb{P}' \to 2^{\mathcal{L}(\mathbb{P})}$ and $(\mathbb{P}', \mathbb{P})$–postconditions functions $\text{POST}_{\mathbb{P}',\mathbb{P}} : \mathbb{P}' \to \left(\mathbb{P} -> 2^{\mathcal{L}(\mathbb{P})}\right)$ as follows: for all $k \in \{1, \ldots, n\}$,

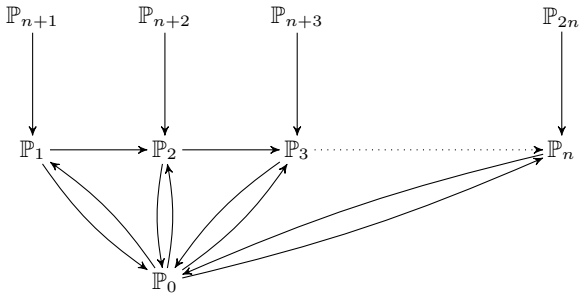

Fig. 5: Dependence graph for the 'Traffic lights' example

- $\textsc{Pre}_{\mathbb{P}_k,\mathbb{P}_0}(\textsc{gTimer}_k) := \textsc{green}_k$,
  $\textsc{Pre}_{\mathbb{P}_k,\mathbb{P}_0}(\textsc{yTimer}_k) := \textsc{yellow}_k$,
  $\textsc{Pre}_{\mathbb{P}_k,\mathbb{P}_0}(\textsc{rTimer}_k) := \textsc{red}_k$,
  $\textsc{Pre}_{\mathbb{P}_k,\mathbb{P}_0}(\textsc{gTimer}'_k) := \textsc{green}_k$,
  $\textsc{Post}_{\mathbb{P}_k,\mathbb{P}_0}(x)(p_0) := \begin{cases} \top & \text{if } p_0 = y \\ \bot & \text{otherwise} \end{cases}$
  for all $(x, y) \in \{(\textsc{gTimer}_k, \textsc{yellow}_k), (\textsc{yTimer}_k, \textsc{red}_k), (\textsc{rTimer}_k, \textsc{green}_k)\}$ and for all $p_0 \in \mathbb{P}_0$.
- $\textsc{Pre}_{\mathbb{P}_0,\mathbb{P}_k}(\textsc{green}_k) := \textsc{gTimer}_k \vee \textsc{gTimer}'_k$,
  $\textsc{Pre}_{\mathbb{P}_0,\mathbb{P}_k}(\textsc{yellow}_k) := \textsc{yTimer}_k$,
  $\textsc{Pre}_{\mathbb{P}_0,\mathbb{P}_k}(\textsc{red}_k) := \textsc{rTimer}_k$,
  $\textsc{Post}_{\mathbb{P}_0,\mathbb{P}_k}(x)(p_k) := \begin{cases} \top & \text{if } p_k = y \\ \bot & \text{otherwise} \end{cases}$
  for all $(x, y) \in \{(\textsc{green}_k, \textsc{gTimer}_k), (\textsc{yellow}_k, \textsc{yTimer}_k), (\textsc{red}_k, \textsc{rTimer}_k)\}$ and for all $p_k \in \mathbb{P}_k$.

4 The Update Universe

The principles that we have elicited in Section 3 lead us to define what we call an *event structure* and a *generic product update*. An event structure captures the dependence relation between different types of events, based on their relative pre/postconditions, while the generic product update deals with the dynamics of knowledge and beliefs within the frame of a given event structure.

4.1 Event Structure

Because of Principle 2, the very nature of events plays a role in the dynamics of knowledge and beliefs. Our idea is to define formally an 'event' model completely identically to the way we define an epistemic model. The propositional letters for 'event' models will determine the factual nature of events, just as

they determine the factual nature of situations in epistemic models. Also, we 'externalize' the precondition and postcondition functions that were fused with the event model in DEL. So, on the one hand, we have event types represented by nodes in the dependence graph and on the other hand we have the different pre/postconditions between these events. These relative pre/postconditions determine in general the different edges of the dependence graph: an edge is set from a node $\mathbb{P}'$ to a node $\mathbb{P}$ when the preconditions for the occurrence of any atomic event of $\mathbb{P}'$ depends on the truth value of formulas of $\mathcal{L}(\mathbb{P})$ or when the occurrence of the atomic events of $\mathbb{P}'$ will affect in some way or another the occurrence of the atomic events of $\mathbb{P}$. This leads us to the following definition.

Definition 8 (Dependence graph, event structure). *A* dependence graph *is an irreflexive directed graph $(\mathcal{P}, \mathcal{S})$ such that $\mathcal{P}$ is a family of disjoint sets of propositional letters. These sets are called* event types *and their elements* atomic events or facts*. Let $(\mathcal{P}, \mathcal{S})$ be a dependence graph. If $(\mathbb{P}', \mathbb{P}) \in \mathcal{S}$,*

- *a* $(\mathbb{P}', \mathbb{P})$–precondition function *is a mapping* $\textsc{Pre}_{\mathbb{P}',\mathbb{P}} : \mathbb{P}' \to 2^{\mathcal{L}(\mathbb{P})}$*. We denote by* $\mathrm{PRE}_{\mathbb{P}',\mathbb{P}}$ *the set of all* $(\mathbb{P}', \mathbb{P})$*–precondition functions;*
- *a* $(\mathbb{P}', \mathbb{P})$–postcondition function *is a mapping* $\textsc{Post}_{\mathbb{P}',\mathbb{P}} : \mathbb{P}' \to \big(\mathbb{P} \to 2^{\mathcal{L}(\mathbb{P})}\big)$*. We denote by* $\mathrm{POST}_{\mathbb{P}',\mathbb{P}}$ *the set of all* $(\mathbb{P}', \mathbb{P})$*–postcondition functions.*

An event structure $(\mathcal{P}, \mathcal{S}, \mathrm{PRE}, \mathrm{POST})$ *is a dependence graph $(\mathcal{P}, \mathcal{S})$ together with two sets of precondition and postcondition functions* $\mathrm{PRE} := \{\textsc{Pre}_{\mathbb{P}',\mathbb{P}} \in \mathrm{PRE}_{\mathbb{P}',\mathbb{P}} \mid (\mathbb{P}', \mathbb{P}) \in \mathcal{S}\}$ *and* $\mathrm{POST} := \{\textsc{Post}_{\mathbb{P}',\mathbb{P}} \in \mathrm{POST}_{\mathbb{P}',\mathbb{P}} \mid (\mathbb{P}', \mathbb{P}) \in \mathcal{S}\}$.

A dependence graph is a directed graph without specific constraint except its irreflexivity. It seems natural to wonder whether it is in fact a specific kind of graph: a tree, a chain, a clique,... The other principles can help us answering this question. Indeed, we learn from Principle 1 that there can be more than three event types. In fact, Principle 7 even indicates us that the number of nodes in the dependence graph can be arbitrary. Moreover, from Principle 3, we infer that there can be cycles in the dependence graph and it turns out that our two examples illustrate this phenomenon. Hence, the dependence graph is in general not a tree. Finally, we learn from Principle 8 that there can be multiple outgoing edges from a given node of the dependence graph. Therefore, it is not a chain either in general. So, from this analysis, we conclude that we cannot impose any particular constraint on the definition of this dependence graph and we state that it can be any kind of directed graph.

4.2 A Generic Product Update

In this section, $(\mathcal{P}, \mathcal{S}, \mathrm{PRE}, \mathrm{POST})$ is an event structure and $(\mathbb{P}', \mathbb{P}) \in \mathcal{S}$. Each edge of an event structure induces a product update. To define it, we first need to recover the pre/postconditions for each world of a $\mathbb{P}'$–model from the $(\mathbb{P}', \mathbb{P})$–pre/postcondition functions associated to an event structure.

Definition 9 ($(\mathbb{P}', \mathbb{P})$–precondition and postcondition functions of a $\mathbb{P}'$–model). *Let $\mathcal{M}' := (W', R', V')$ be a $\mathbb{P}'$–model.*

– *The* $(\mathbb{P}', \mathbb{P})$–precondition function of $\mathcal{M}'$, $\textsc{Pre}^{\mathcal{M}'}_{\mathbb{P}',\mathbb{P}} : W' \to 2^{\mathcal{L}(\mathbb{P})}$, *is such that for all* $w' \in W'$,

$$\textsc{Pre}^{\mathcal{M}'}_{\mathbb{P}',\mathbb{P}}(w') := \bigcup_{p' \in \mathbb{P}'} \{\textsc{Pre}_{\mathbb{P}',\mathbb{P}}(p') \mid w' \in V'(p')\}$$

– *The* $(\mathbb{P}', \mathbb{P})$–postcondition function of $\mathcal{M}'$, $\textsc{Post}^{\mathcal{M}'}_{\mathbb{P}',\mathbb{P}} : W' \to \big(\mathbb{P} \to 2^{\mathcal{L}(\mathbb{P})}\big)$, *is such that for all* $w' \in W'$, *all* $p \in \mathbb{P}$,

$$\textsc{Post}^{\mathcal{M}'}_{\mathbb{P}',\mathbb{P}}(w')(p) := \bigcup_{p' \in \mathbb{P}'} \{\textsc{Post}_{\mathbb{P}',\mathbb{P}}(p')(p) \mid w' \in V'(p')\} \tag{1}$$

Note that the range of our precondition and postcondition functions are *sets* of formulas, and not single formulas like for event models (see Definition 4). This generalization of the DEL framework is meaningful. Indeed, there is no particular reason that the occurrence of an event depends on a property definable in modal logic by a single formula. The precondition of an event is implicitly determined by the class of pointed epistemic models in which this event can occur.[1] This class of epistemic models is often infinite and there is no reason that it should be definable by a single formula. In general, and especially in an infinite setting, it is quite possible that an event occurs in a class of epistemic models which is only definable by an *infinite set* of formulas [6, Sect. 2.6-3.3].

Definition 10 (Generic product update). *Let* $\mathcal{M} = (W, R, V)$ *be a* $\mathbb{P}$*–model and let* $\mathcal{M}' = (W', R', V')$ *be a* $\mathbb{P}'$*–model. The* $(\mathbb{P}', \mathbb{P})$–product update of $\mathcal{M}$ and $\mathcal{M}'$ *is the* $\mathbb{P}$*–model* $\mathcal{M} \odot \mathcal{M}' = (W^{\odot}, R^{\odot}, V^{\odot})$ *defined as follows (with the proviso that* $W^{\odot}$ *is not empty): for all* $p \in \mathbb{P}_0$ *and all* $j \in \mathbb{A}$,

– $W^{\odot} := \Big\{(w, w') \in W \times W' \mid \mathcal{M}, w \models \textsc{Pre}^{\mathcal{M}'}_{\mathbb{P}',\mathbb{P}}(w')\Big\}$;
– $(v, v') \in R^{\odot}_j(w, w')$ *iff* $v \in R_j(w)$ *and* $v' \in R_j(w')$;
– $(w, w') \in V^{\odot}(p)$ *iff* $\mathcal{M}, w \models \varphi$ *for some* $\varphi \in \textsc{Post}^{\mathcal{M}'}_{\mathbb{P}',\mathbb{P}}(w')(p)$.

The following example is a concrete example that the standard event update cannot account for: if everything was put on the same level, we could not account for the update of events by other events and then subsequently the update of events by the situation. This example will be discussed once again in the next section.

Example 4 ('Barrel' example). In Fig. 6 we represent the generic product update that occurred in Scenario 1 of Section 3.1, whereby Ann and Bob's perception of the ongoing event was updated by their perception of another event (namely the fully private announcement to Bob that barrel 1 is being filled). In Fig. 7, we represent the Scenario 2 of Section 3.1. At each line, we represent the situations and the events that occur at the corresponding time stamp as they are defined in

[1] [5] defined independently from the DEL community a variant of the DEL framework where preconditions are replaced by classes of pointed epistemic models.

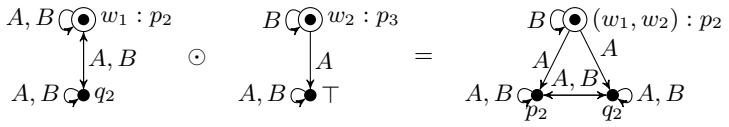

Fig. 6: Barrel example: Scenario 1

Scenario 2. To simplify notations, edges are represented without arrows, so the reader must assume that all arrows are bidirectional and that there are reflexive arrows indexed by all agents at each node. When there are no arrows, this means that edges are bidirectional and reflexive. We start at time t_1 with two models, one showing knowledge and beliefs concerning q_0, p_0, the other one concerning p_2 and q_2 (which actually represent events). Then an announcement is added (represented by $w_1 : p_1$) such that in the system state at time t_2 we have three models. Then two of the models "amalgamate" by a product update leading to the next system state at time t_3 with two models. At time t_4, the situation updates in a backward fashion the perception of the event, leading to the final situation at time t_5.

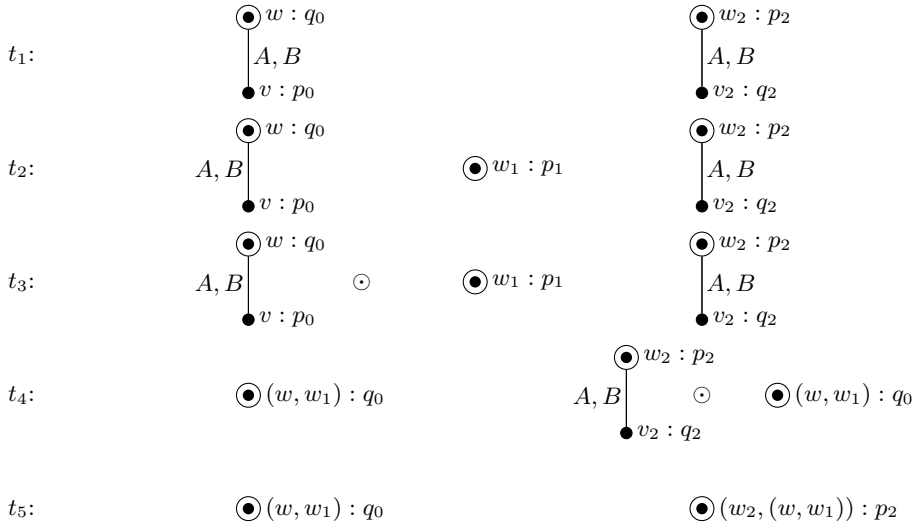

Fig. 7: Barrel example: Scenario 2

5 Discussion

One may still argue that DEL can already handle our examples. Because we deal with lasting events, what seems to be needed are propositions stating the status of events in the epistemic models, such as "(the filling of barrel 1) has ended", "(the filling of barrel 2) is still happening" together with constraints such as "(the filling of barrel 2) is still happening"→"barrel 2 is not full". Then, the standard DEL setting can also handle the updates of knowledge of events by the perception of other events. In our approach, the uncertainty of the occurrences of the events and the uncertainty of basic facts are captured in separate models initially and the updates are on each model separately. However, they can be put in the same model if the propositions about the events can be expressed in the language. In a sense, our whole framework and the event structure that we have elicited could simply be 'flattened' by adding some sort of predicates about events that would be formalized by specific propositional letters. Even if that would work out from a formal point of view, this *ad hoc* solution is very far to be satisfactory from a conceptual and modeling point of view. Indeed, the intuitive insights that we have elicited by means of our principles would then be disguised under the form of (meta-)predicates and constraints between propositional letters. These predicates in disguise and constraints would actually encode our dependence graphs and event structures.

One may then argue that our examples could be dealt with by existing extensions of DEL such as temporal DEL with past. In particular, Scenario 2 of the 'barrel' example could be reformulated in terms of uncertainty about which actual event *history* the agents are in. This kind of modeling is however subject to problems which are inherent to any state–based models such as all dynamic and process logics [10, 12]. It is hardly possible to express in these logics that "barrel 1 is being filled" and to model Scenario 1. One could express it in an *ad hoc* way by adding the propositional letter "barrel 1 is being filled" in the language, but we would need to also update its truth value when the filling ends and we would need another specific update to formalize this ending of the filling event. Likewise, Scenario 1 would be possible in temporal DEL with past only if we had that propositional letter in our model and language. In fact, we would need again to somehow encode our event structure. Moreover, the kind of modular reasoning with bounded rationality which occurs in Scenario 2 at time t_4 would not be really captured with this type of state–based and history–based logic.

One may then argue that it is not clear exactly how specific scenarios are supposed to be modeled with dependence graphs and event structures. The answer is that this problem is not inherent to our approach but applies to any modeling approach of epistemic scenarios and in particular already with epistemic and event models. There is no procedure or algorithm for constructing neither epistemic or event models nor dependence graphs or event structure. So it is a general problem of epistemic modeling. We have striven to give some guidelines that would help modelers to build their models and dependency graphs, but the

general problem of how to model epistemic situations is still more at the stage of an art than a science for the moment.

We have demonstrated that the current modeling approach of DEL is not adequate enough to account for certain information dynamics. This defect should not be ignored and dismissed, even if the examples that we have chosen to illustrate it were, intentionally, extreme, borderline and different from the usual examples encountered in DEL.

Extending [1], we identified various principles that events fulfill, by means of examples. They led us to motivate the formal definitions of dependence graph and event structure. These should be the main ingredients for a genuine logical framework. Yet, before defining this general framework, the preliminary logical analysis presented so far was necessary to be carried out in order to identify the key features that needed to be formalized and included as well as highlight the weaknesses of the current application of the DEL modeling approach, based on event models and product updates.

Acknowledgments. I thank Johan van Benthem and two anonymous referees for helpful comments. I thank Sabine Frittella for a discussion which enabled to find out a defect.

References

1. Guillaume Aucher. BMS revisited. In Aviad Heifetz, editor, *TARK*, pages 24–33, 2009.
2. C. Baier and J.P. Katoen. *Principles of model checking*. MIT press, 2008.
3. Alexandru Baltag and Lawrence S. Moss. Logic for epistemic programs. *Synthese*, 139(2):165–224, 2004.
4. Alexandru Baltag, Lawrence S. Moss, and Slawomir Solecki. The logic of public announcements and common knowledge and private suspicions. In Itzhak Gilboa, editor, *TARK*, pages 43–56. Morgan Kaufmann, 1998.
5. Antoine Billot, Jean-Christophe Vergnaud, and Bernard Walliser. Multiagent belief revision. *Journal of Mathematical Economics*, 59:47 – 57, 2015.
6. Patrick Blackburn, Maarten de Rijke, and Yde Venema. *Modal Logic*, volume 53 of *Cambridge Tracts in Computer Science*. Cambridge University Press, 2001.
7. Ronald Fagin, Joseph Halpern, Yoram Moses, and Moshe Vardi. *Reasoning about knowledge*. MIT Press, 1995.
8. Paul Gochet and Pascal Gribomont. Epistemic logic. In Dov Gabbay and John Woods, editors, *Handbook of the History of Logic*, volume 7, Twentieth Century Modalities, pages 99–195. Elsevier, Amsterdam, 2006.
9. Joseph Y. Halpern and Riccardo Pucella. Dealing with logical omniscience: Expressiveness and pragmatics. *Artificial intelligence*, 175(1):220–235, 2011.
10. David Harel, Dexter Kozen, and Jerzy Tiuryn. *Dynamic Logic*. MIT Press, 2000.
11. Jaakko Hintikka. *Knowledge and Belief, An Introduction to the Logic of the Two Notions*. Cornell University Press, Ithaca and London, 1962.
12. V. R. Pratt. Process logic. In *Proceedings of the 6th ACM symposium on Principles of Programming Languages*, San Antonio, 1979.
13. Raymond Reiter. *Knowledge in Action: Logical Foundations for Specifying and Implementing Dynamical Systems*. MIT Press, 2001.

14. Johan van Benthem. *Logical Dynamics of Information and Interaction*. Cambridge University Press, 2011.
15. Johan van Benthem, Jan van Eijck, and Barteld Kooi. Logics of communication and change. *Information and Computation*, 204(11):1620–1662, 2006.
16. Hans van Ditmarsch and Barteld Kooi. Semantic results for ontic and epistemic change. In G. Bonanno, W. van der Hoek, and M. Wooldridge, editors, *Logic and the Foundations of Game and Decision Theory (LOFT 7)*, Texts in Logic and Games 3, pages 87–117. Amsterdam University Press, 2008.
17. George Wilson and Samuel Shpall. Action. In Edward N. Zalta, editor, *The Stanford Encyclopedia of Philosophy*. Summer 2012 edition, 2012.