# A GENERAL MODAL FRAMEWORK FOR THE EVENT CALCULUS AND ITS SKEPTICAL AND CREDULOUS VARIANTS

ILIANO CERVESATO AND ANGELO MONTANARI

▷ We propose a general and uniform modal framework for the Event Calculus (EC) and its skeptical and credulous variants. The resulting temporal formalism, called the Generalized Modal Event Calculus (GMEC), extends considerably the expressive power of EC when information about the ordering of events is incomplete. It provides means of inquiring about the evolution of the maximal validity intervals of properties relative to all possible refinements of the ordering data by allowing a free mix of propositional connectives and modal operators. We first give a semantic definition of GMEC and relate it to known systems of modal logic; then, we propose a declarative encoding of GMEC in the language of hereditary Harrop formulas and prove the soundness and completeness of the resulting logic programs. ◁

*This web of times — the strands of which approach to one another, bifurcate, intersect or ignore each other through the centuries — embraces every possibility. We do not exist in most of them. In some you exist and not I, while in others I do, and you do not, and in yet others both of us exist. In this one, in which chance has favored me, you have come to my gate. In another, you, crossing the garden, have found me dead. In yet another, I say these very same words, but am an error, a phantom.*

— *The Garden of the Forking Paths*, Jorge Luis Borges

*Address correspondence to* Iliano Cervesato, Department of Computer Science, Stanford University, Stanford, CA 94305-9045, USA, Email: iliano@cs.stanford.edu.

*Address correspondence to* Angelo Montanari, Dipartimento di Matematica e Informatica, Università di Udine, 33100 Udine, Italy, Email: montana@dimi.uniud.it.

This work was done while the first author was at the Department of Computer Science, Carnegie Mellon University, Pittsburgh, PA 15213-3891, USA, Email: iliano@cs.cmu.edu.





655 Avenue of the Americas, New York, NY 10010 0743-1066/94/$7.00

## 1. INTRODUCTION

This paper proposes a general and uniform modal framework for Kowalski and Sergot's Event Calculus (EC) [20] and its skeptical and credulous variants [2, 5, 10]. Given a set of event occurrences, EC allows one to derive *maximal validity intervals* (*MVI*s hereafter) over which properties initiated or terminated by those events hold. As new events or additional ordering information about known events are recorded, EC updates accordingly the set of MVIs. Most approaches based on EC assume the occurrence time of each event to be known; here, we explore the case of partially ordered events devoid of an explicit occurrence time. In such a situation, EC is neither able to determine the set of MVIs that can be derived in at least one refinement of the given partial ordering (possible MVIs) nor to establish which of the currently derivable MVIs are also derivable in all refinements of the given ordering (necessary MVIs) and which of them are derivable in some, but not all, refinements (defeasible MVIs).

The problem of computing which facts must be or may possibly be true over certain time intervals in presence of partially ordered events has been already addressed in the literature, e.g. [2, 5, 10, 11, 12, 25, 30]. In particular, complexity issues have been addressed in [11], while case studies in the domains of diagnosis and planning have been analyzed in [10] and [25], respectively.

With regard to the problem of reasoning about partially ordered events in EC, two variants of the basic calculus, called Skeptical EC (SKEC) and Credulous EC (CREC), have been proposed in [5] and [10]. These variants respectively compute the necessarily true MVIs and the possibly true MVIs in the restricted setting where the occurrence of events is not subject to preconditions. SKEC and CREC can be given a polynomial implementation, that can be further enhanced by exploiting transitive reduction graph processing techniques, as shown in [8]. In [2], Cervesato et al. defined a uniform modal interpretation for EC, SKEC and CREC, called the Modal Event Calculus (MEC). MEC deals with atomic formulas (MVIs computed by EC) as well as simply moded atomic formulas, i.e. atomic formulas prefixed by only one modality (MVIs computed by SKEC and CREC). It is provided with a sound and complete axiomatic formulation in a logic programming framework.

In this paper, we define a Generalized Modal Event Calculus (GMEC) that extends MEC by allowing a free mix of propositional connectives and modal operators. Such a capability is useful in order to deal with real-world applications, as pointed out in [10]. Perhaps more important than the resulting calculus itself is the method we adopt to achieve it. We initially capture the intuitions underlying GMEC by giving a *semantic* formulation of EC and extending it to a modal interpretation that takes into account all possible refinements of the ordering data. Then, we provide a sound and complete *axiomatization* of GMEC in the language of hereditary Harrop formulas and rely on a *proof-theoretic* approach for proving the faithfulness of our implementations with respect to the behavior of GMEC, as expressed by the semantics.

We believe that our approach contributes to the conceptual understanding of EC, an important but not yet fully understood formalism for reasoning about events and their effects. Moreover, the proposed method can be exploited to increase the confidence in alternative axiomatizations of EC by proving them sound and complete with respect to the corresponding semantics via syntactic (proof-theoretic) methods. Finally, it seems suited to act as a general framework for studying sig-

nificant extensions of EC (e.g., GMEC). We expect this approach to be applicable to related formalisms as well (e.g., McCarthy and Hayes' Situation Calculus [22]).

The paper is organized as follows. In Section 2, we first introduce the basic concepts underlying the Event Calculus; then we recall some basic definitions about orderings and tailor them to the needs of the subsequent discussion; finally, we formally define GMEC and present its fundamental properties. In Section 3, we summarize the definition and operational semantics of hereditary Harrop formulas and use this language to give two sound and complete encodings of GMEC. The conclusions provide an assessment of the work done and discuss future developments. For the sake of readability, we have collected the proofs of the results presented in Section 3 in Appendix A.

## 2. THE GENERALIZED MODAL EVENT CALCULUS

In this section, we formally define the Generalized Modal Event Calculus (GMEC). We consider the case in which the set of event occurrences has been fixed once and for all, and only partial information about their relative ordering is given. In such a situation, the update process may only consist in the addition of further information about the relative ordering of event pairs. Furthermore, we assume that events do not happen simultaneously and that the available ordering information is always consistent.

The section is organized as follows. We first give an intuitive account of the basic concepts underlying EC and recall some notions about ordering relations. Then, we provide EC with a semantic interpretation that, given the current partial ordering of events, validates precisely the MVIs computed by EC. By considering all possible refinements of the current ordering, with the associated reachability relation, this model is naturally lifted to a modal interpretation. The corresponding extension of EC with propositional connectives and modalities substantially augments the expressive power of the calculus. Next, we formally state a number of properties of the proposed formalization that will be later exploited to increase the efficiency of a first naive implementation of GMEC.

### *2.1. An Informal Account of the Event Calculus*

This section is devoted to providing a description of the basic features of the Event Calculus at an introductory level. In order to make the presentation more intuitive, we rely on an example describing the operations of a simple beverage dispenser. We will use this example again in Section 2.4 to illustrate the benefits of adding modal operators to the basic EC.

The structure of the beverage dispenser is depicted in Figure 2.1. It can output either apple juice or orange juice (but not both simultaneously). The choice is made by means of a selector with three positions (*apple*, *orange* and *stop*): by setting the selector to the *apple* or to the *orange* position, apple juice or orange juice is obtained, respectively; choosing the *stop* position terminates the production of juice.

EC proposes a general approach to representing and reasoning about events and their effects in a logic programming framework. It defines a model of change in which *event* occurrences initiate and/or terminate *time-intervals* over which some

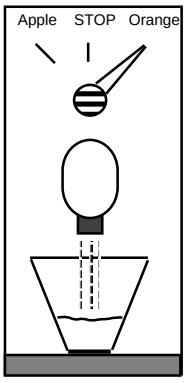


**FIGURE 2.1.** A Beverage Dispenser

*property* holds. In our example, we distinguish three types of events corresponding to the various settings of the selector and two relevant properties, *supplyApple* and *supplyOrange*, indicating that apple juice or orange juice is being dispensed, respectively. The event of setting the selector to the *apple* (*orange*) position *initiates* the property *supplyApple* (*supplyOrange*), while setting it to the *stop* position *terminates* both properties. The properties *supplyApple* and *supplyOrange* are *exclusive* since apple juice and orange juice cannot be output simultaneously. This intuitive description will be formalized in Section 2.4.

Given a domain description in terms of events, properties and initiate, terminate, or exclusive relations, EC computes the maximal validity intervals (MVIs) over which properties hold uninterruptedly. To this end, it relies on a notion of default persistence according to which properties are assumed to persist until an event that interrupts them occurs. For the sake of simplicity, we will restrict ourselves to finite MVIs. The generalization to MVIs whose validity extends infinitely in either direction is, however, straightforward. Mechanisms for dealing with both persistence in the future (properties that hold forever from the occurrence time of a given initiating event) and persistence in the past (properties that hold from the beginning of time up to the occurrence time of a given terminating event) are described in [7].

We illustrate the basic computational mechanism of EC by means of four situations relative to the beverage dispenser example.

- Consider a scenario consisting of a pair of events $e_a$ and $e_s''$ that respectively set the selector to the *apple* position and reset it to the *stop* position. Assuming that $e_a$ precedes $e_s''$, EC computes the interval $(e_a, e_s'')$ as an MVI for the property *supplyApple*.

- Enrich the previous situation by adding a *stop* event $e_s'$ occurring between $e_a$ and $e_s''$. The interval $(e_a, e_s'')$ is not an MVI for *supplyApple* anymore, since it is interrupted by the occurrence of $e_s'$. The two *stop* events, indeed, must necessarily be interleaved by (at least) one event that sets the selector

either to the apple or to the orange position. Nevertheless, it may happen that incomplete knowledge about the set of event occurrences or about their temporal ordering makes it impossible to detect such an event. In this scenario, EC derives $(e_a, e'_s)$ as an MVI for *supplyApple*, while the dangling event $e''_s$ does not terminate any MVI.

- Alternatively, modify the first scenario by inserting an event $e_o$ between $e_a$ and $e''_s$ that sets the selector on the *orange* position. This invalidates the MVI $(e_a, e''_s)$ since *supplyApple* and *supplyOrange* cannot consistently hold over the same subinterval. Indeed to move from a state in which the selector is on the *apple* position to a state in which it is on *orange*, we must go through the *stop* position (that has not been recorded as an event — EC naturally supports incomplete specifications). In this situation, EC derives $(e_o, e''_s)$ as an MVI for *supplyOrange*, while the dangling event $e_a$ does not initiate any MVI.

- Finally, suppose to add both $e'_s$ and $e_o$ between $e_a$ and $e''_s$, with $e'_s$ preceding $e_o$. As above, we cannot keep $(e_a, e''_s)$ as an MVI for *supplyApple* since it is interrupted by both $e'_s$ and $e_o$. EC instead computes two MVIs: $(e_a, e'_s)$ for the property *supplyApple* and $(e_o, e''_s)$ for *supplyOrange*.

As a general rule, an event $e$ interrupts the validity of a property $p$ if it initiates or terminates $p$ itself or a property $q$ which is incompatible with $p$. This rule adopts the so-called *strong interpretation* of the initiate and terminate relations, which has been discussed in detail in [29, 7, 9]: given a pair of events $e_i$ and $e_t$, with $e_i$ occurring before $e_t$, that respectively initiate and terminate a property $p$, we conclude that $p$ does not hold over $(e_i, e_t)$ if an event $e$ which initiates or terminates $p$, or a property incompatible with $p$, occurs during this interval, that is, $(e_i, e_t)$ is a candidate MVI for $p$, but $e$ forces us to reject it. The strong interpretation is needed when dealing with incomplete sequences of events or, as in our case, incomplete information about their ordering. For example, consider a switch that can take two different positions (*on* and *off*). Its behavior can be described by means of two types of event: one that changes the position from *off* to *on* (*turn-on*), the other from *on* to *off* (*turn-off*). While two *turn-on* events cannot occur consecutively in the real world, it may happen that an incomplete sequence consisting of two consecutive *turn-on* events, followed by a *turn-off* event, is recorded in the database. The strong interpretation of the initiate relation allows EC to recognize that a missing *turn-off* event must have occurred between the two *turn-on* events. However, since it is not possible to temporally locate such an event, EC only concludes that the switch is *on* between the second *turn-on* event and the *turn-off* event, and it considers the first *turn-on* event as a *pending* initiating event.

An alternative interpretation of the initiate and terminate relations, called *weak interpretation* [7, 9], is also possible. According to such an interpretation, a property $p$ is initiated by an initiating event unless it has been already initiated and not yet terminated (and dually for terminating events). The weak interpretation is needed to aggregate homogeneous states[1]. Consider, for instance, the problem of

[1]The operation of aggregation of homogeneous states is very similar to the operation of *coalesce* exploited in temporal databases to replace two or more value-equivalent tuples with consecutive or overlapping time-stamps by a single, value-equivalent tuple with an interval-valued time-stamp

monitoring patients who receive a partial mechanical respiratory assistance. It often happens that data acquired with two consecutive examinations do not cause any transition in the classification of the patient ventilatory state. Adopting the weak interpretation, the second data acquisition does not clip the MVI of patient state initiated by the first one. A detailed report on the application of EC to the management of mechanical ventilation, that describes in detail the effects of adopting such a weak interpretation of relations, can be found in [6].

The distinction between strong and weak interpretation of the initiate and terminate relations can be precisely stated as follows: we derive an MVI for a property $p$ whenever there exist a sequence of one or more events $e_{i,1}, \ldots, e_{i,h}$ that initiate $p$, followed by a sequence of one or more events $e_{t,1}, \ldots, e_{t,k}$ that terminate $p$, and there exists no event that initiate or terminate a property $q$, incompatible with $p$, in between (that is, that occurs between $e_{i,1}$ and $e_{t,k}$). If we adopt a strong interpretation of both initiate and terminate relations, the MVI for $p$ is $(e_{i,h}, e_{t,1})$; if we adopt a weak interpretation of initiate and a strong interpretation of terminate (this is the case in most medical applications), the MVI for $p$ is $(e_{i,1}, e_{t,1})$; finally, if we adopt a weak interpretation of both initiate and terminate relations (rare, but not impossible), the MVI for $p$ is $(e_{i,1}, e_{t,k})$.

In the remainder of the paper, we will adopt the strong interpretation of the initiate and terminate relations, since it is more suited to modeling incompletely specified situations. However, we expect our results to apply also relative to the weak interpretation.

## *2.2. Ordering Relations*

In the following, we will rely upon different notions of ordering and ordered set. For reason of efficiency, ordering information is usually *represented* as a binary acyclic relation on the set of events, that is, as an ordering relation possibly missing some transitive links. However, this information is *used* in EC as a (strict) partial order which can be recovered as the transitive closure of the given binary acyclic relation. Furthermore, the structure representing the effects of various possible updates to the information about event ordering constitutes a reflexive partial order.

*Definition 2.1.* (DAGs, strictly ordered sets, non-strictly ordered sets)

Let $E$ be a set and $R$ a binary relation on $E$. $R$ is called a (strict) *partial order* if it is irreflexive and transitive (and, thus, asymmetric) and a *reflexive partial order* if it is reflexive, antisymmetric, and transitive. The pair $(E, R)$ is called a *directed acyclic graph* ($DAG$) if $R$ is a binary acyclic relation; a *strictly ordered set* if $R$ is a partial order; a *non-strictly ordered set* if $R$ is a reflexive partial order. □

We denote the sets of all binary acyclic relations and of all partial orders on $E$ as $O_E$ and $W_E$, respectively. It is easy to show that, for any set $E$, $W_E \subseteq O_E$ (actually, $W_E \subset O_E$ if $E$ has at least three elements). We will use the letters $o$ and $w$ possibly subscripted to denote binary acyclic relations and partial orders, respectively.

[19].

We indicate the transitive closure of a relation $R$ as $R^+$. Clearly, if $(E, o)$ is a directed acyclic graph, then $(E, o^+)$, is a strictly ordered set. Two binary acyclic relations $o_1, o_2 \in O_E$ are *equally informative* if $o_1^+ = o_2^+$. This induces an equivalence relation $\sim$ on $O_E$. It is easy to prove that, for any set $E$, $O_{E/\sim}$ and $W_E$ are isomorphic. In the following, we will often identify a binary acyclic relation $o$ with the corresponding element $o^+$ of $W_E$.

The set $\mathbf{2}^{E\times E}$ of all binary relations on $E$ naturally becomes a non-strictly ordered set when considered together with the usual subset relation $\subseteq$. Moreover, $(\mathbf{2}^{E\times E}, \cup, \cap, \bar{}, E \times E, \emptyset)$ is a boolean lattice. Since $W_E$ is a subset of $\mathbf{2}^{E\times E}$, the restriction of $\subseteq$ to this set still forms a reflexive partial order. Indeed, we have that, for any set $E$, $(W_E, \subseteq)$ is a non-strictly ordered set. It can be easily proved that $(W_E, \cap, \emptyset)$ forms a lower semi-lattice. Moreover, for any $w_1, w_2 \in W_E$, the relation $w_1 \uparrow w_2 = (w_1 \cup w_2)^+$ is the least upper bound (*lub*) of $w_1$ and $w_2$ whenever this element belongs to $W_E$. Note that $w_1 \uparrow w_2 \notin W_E$ if $w_1$ and $w_2$ contain symmetric pairs.

Given $w$ in $W_E$, any $w' \in W_E$ such that $w \subseteq w'$ is called an *extension* of $w$. We denote the set of all extensions of $w$ as $Ext(w)$. We have that for any $w \in W_E$, if $(e_1, e_2) \in w$, then for all $w' \in Ext(w)$, $(e_1, e_2) \in w'$. For any $w \in W_E$, $Ext(w)$ enjoys the same properties of $W_E$. More precisely, $(Ext(w), \subseteq)$ is a non-strictly ordered set, $(Ext(w), \cap, w)$ is a lower semi-lattice, and $\uparrow$ characterizes the partial operation of *lub* over this semi-lattice. Notice in particular that $Ext(\emptyset) = W_E$.

Whenever $E$ is a finite set, also $W_E$ is finite since it is a subset of $\mathbf{2}^{E\times E}$. Moreover, $Ext(w)$ for $w \in W_E$ is finite as well. This property allows us to prove statements by induction on the cardinality of $Ext(w)$ for $w \in W_E$ and $E$ finite. We will need this fact in the proofs of the results of Section 3.

We conclude the treatment of orderings by giving some definitions related to the notion of interval. Let $E$ be a set and $w \in W_E$. A pair $(e_1, e_2) \in w$ is called an *interval* of $w$. Given two *distinct* intervals $(e_1, e_2)$ and $(e_1', e_2')$ over $w$, we say that $(e_1, e_2)$ is a *subinterval* of $(e_1', e_2')$ (or $(e_1', e_2')$ is a *superinterval* of $(e_1, e_2)$) with respect to $w$ if either $e_1 = e_1'$ or $(e_1', e_1) \in w$ and dually $e_2 = e_2'$ or $(e_2, e_2') \in w$. We write in this case $(e_1, e_2) \sqsubset_w (e_1', e_2')$. We have that, for any ordering $w \in W_E$, $(w, \sqsubset_w)$ is a strictly ordered set.

## *2.3. Formalization of GMEC*

In order to formalize the Event Calculus and its modal variants, we first define the notion of EC-structure that records the time-independent (factual) parameters of an EC problem, i.e. the sets of relevant events and properties, the relations that associate events to the properties they initiate and to the properties they terminate, and the pairs of mutually incompatible properties.

*Definition 2.2.* (EC-structure)

A *structure* for the *Event Calculus (*or *EC-structure)* is a quintuple $\mathcal{H} = (E,\ P,\ [\cdot\rangle,\ \langle\cdot],\ ]\cdot,\cdot[)$ such that:

- $E = \{e_1, \ldots, e_n\}$ and $P = \{p_1, \ldots, p_m\}$ are finite sets of *events* and *properties*, respectively.
- $[\cdot\rangle : P \to \mathbf{2}^E$ and $\langle\cdot] : P \to \mathbf{2}^E$ are respectively the *initiating* and *terminating map* of $\mathcal{H}$. For every property $p \in P$, $[p\rangle$ and $\langle p]$ represent the set of events

that initiate and terminate $p$, respectively.

- $]\cdot,\cdot[ \subseteq P \times P$ is an irreflexive and symmetric relation, called the *exclusivity relation*, that models exclusivity among properties. □

Any *EC-structure* is also a structure for the *Generalized Modal Event Calculus* (hereafter *GMEC-structure*).

Notice that the above definition does not prevent that $[p\rangle \cap \langle p] \neq \emptyset$, for some property $p$. We never needed to exploit this rather odd feature in any practical application. Nonetheless, we keep the definition of EC in its most general form since it does not hinder the development of this work.

Since we consider situations where events are ordered relative to one another, we will represent an MVI for a property $p$ as $p(e_i, e_t)$, where $e_i$ and $e_t$ are the events that initiate and terminate $p$, respectively. MVIs are thus intervals labeled by properties. We will adopt the set of all property-labeled intervals as the language of EC. The task performed by EC thus reduces to deciding which formulas are MVIs and which are not. GMEC extends this language by allowing combinations of property-labeled intervals by means of propositional connectives and modal operators. The language for GMEC is defined as follows.

*Definition 2.3.* (GMEC-language)

Let $\mathcal{H} = (E,\ P,\ [\cdot\rangle,\ \langle\cdot],\ ]\cdot,\cdot[)$ be a GMEC-structure. The *base language* of $\mathcal{H}$ *(EC-language)* is the set of propositional letters $\mathcal{A}_\mathcal{H} = \{p(e_1, e_2) : p \in P \text{ and } e_1, e_2 \in E\}$. The *GMEC-language* of $\mathcal{H}$, denoted by $\mathcal{L}_\mathcal{H}$, is the modal language with propositional letters in $\mathcal{A}_\mathcal{H}$ and logical operators in $\{\neg, \wedge, \vee, \Box, \Diamond\}$. We refer to the elements of $\mathcal{A}_\mathcal{H}$ and $\mathcal{L}_\mathcal{H}$ as *atomic formulas* and *GMEC-formulas*, respectively. □

Notice that, in spite of the structured notation we use for atomic formulas, $\mathcal{L}_\mathcal{H}$ is a propositional language.

We call *knowledge state* a partial (consistent) specification of the events ordering. Standard implementations of EC represent knowledge states as binary acyclic relations, and take their transitive closure in order to make inferences concerning MVIs. Therefore, given a GMEC-structure $\mathcal{H} = (E,\ P,\ [\cdot\rangle,\ \langle\cdot],\ ]\cdot,\cdot[)$, we interpret atomic formulas relative to the set $W_E$ (denoted $W_\mathcal{H}$ in this context) of partial orders among events in $E$. Given a *current state of knowledge* $w$, the semantics of EC is defined by the (propositional) valuation $v^w_\mathcal{H}$, which discriminates MVIs from other intervals in $w$.

In order for $p(e_1, e_2)$ to be an MVI relative to the knowledge state $w$, $(e_1, e_2)$ must be an interval in $w$, i.e. $(e_1, e_2) \in w$. Moreover, $e_1$ and $e_2$ must witness the validity of the property $p$ at the ends of this interval by initiating and terminating $p$, respectively. These requirements are enforced by conditions $(iii)$, $(i)$ and $(ii)$, respectively, in the definition of valuation given below. The maximality requirement is caught by the meta-predicate $nb(p, e_1, e_2, w)$ in condition $(iv)$, which expresses the fact that the validity of an MVI must not be *broken* by any interrupting event. Any event $e$ which is known to have happened between $e_1$ and $e_2$ in $w$ and that initiates or terminates a property that is either $p$ itself or a property exclusive with $p$ interrupts the validity of $p(e_1, e_2)$.

EC has been traditionally defined by means of a set of axioms [20]. In its logic programming implementation, the valuation $\upsilon^w_{\mathcal{H}}$ is represented by the predicate `holds`, which relies on the predicate `broken` for testing for interrupting events (i.e. the negation of the meta-predicate $nb$). The original definition of these predicates will be recovered in our implementation in Section 3.

GMEC expands the scope of EC by shifting the focus from the current knowledge state — say $w$ — to all knowledge states that are reachable from $w$, i.e. $Ext(w)$, and more generally to $W_{\mathcal{H}}$. By definition, $w'$ is an extension of $w$ if $w \subseteq w'$. Since $\subseteq$ is a reflexive partial order, $(W_{\mathcal{H}}, \subseteq)$ can be naturally viewed as a finite, reflexive, transitive and antisymmetric modal frame. If we consider this frame together with the straightforward modal extension of the valuation $\upsilon^w_{\mathcal{H}}$ to an arbitrary knowledge state, we obtain a modal model for GMEC.

Let $\mathcal{H} = (E,\ P,\ [\cdot\rangle,\ \langle\cdot],\ ]\cdot,\cdot[)$ be a GMEC-structure. We denote as $O_{\mathcal{H}}$ and $W_{\mathcal{H}}$ the set $O_E$ of binary acyclic relations and the set $W_E$ of partial orders over $E$, respectively. We call the elements of $O_{\mathcal{H}}$ (and consequently of $W_{\mathcal{H}}$) *knowledge states*. The *GMEC-frame* $\mathcal{F}_{\mathcal{H}}$ of $\mathcal{H}$ is the frame $(W_{\mathcal{H}}, \subseteq)$. The intended GMEC-model is defined as follows.

*Definition 2.4.* (GMEC-model)

The *intended GMEC-model* of a GMEC-structure $\mathcal{H}$ is the modal model $\mathcal{I}_{\mathcal{H}} = (W_{\mathcal{H}}, \subseteq, \upsilon_{\mathcal{H}})$, where the valuation $\upsilon_{\mathcal{H}} : W_{\mathcal{H}} \to \mathbf{2}^{\mathcal{A}_{\mathcal{H}}}$ is defined in such a way that $p(e_1, e_2) \in \upsilon_{\mathcal{H}}(w)$ if and only if conditions (*i*–*iv*) below hold.

*i.* $e_1 \in [p\rangle$;

*ii.* $e_2 \in \langle p]$;

*iii.* $(e_1, e_2) \in w$;

*iv.* $nb(p, e_1, e_2, w)$, where

$$nb(p, e_1, e_2, w) \text{ iff } \neg\exists e \in E.\ (e_1, e) \in w \wedge (e, e_2) \in w \wedge \exists q \in P.\ ((e \in [q\rangle \vee e \in \langle q]) \wedge (]p, q[ \vee p = q)).$$

Given $w \in W_{\mathcal{H}}$ and $\varphi \in \mathcal{L}_{\mathcal{H}}$, the satisfiability relation $\mathcal{I}_{\mathcal{H}}; w \models \varphi$ is defined as follows:

$$\begin{array}{lll}
\mathcal{I}_{\mathcal{H}}; w \models p(e_1, e_2) & \text{iff} & p(e_1, e_2) \in \upsilon_{\mathcal{H}}(w); \\
\mathcal{I}_{\mathcal{H}}; w \models \neg\varphi & \text{iff} & \mathcal{I}_{\mathcal{H}}; w \not\models \varphi; \\
\mathcal{I}_{\mathcal{H}}; w \models \varphi_1 \wedge \varphi_2 & \text{iff} & \mathcal{I}_{\mathcal{H}}; w \models \varphi_1 \text{ and } \mathcal{I}_{\mathcal{H}}; w \models \varphi_2; \\
\mathcal{I}_{\mathcal{H}}; w \models \varphi_1 \vee \varphi_2 & \text{iff} & \mathcal{I}_{\mathcal{H}}; w \models \varphi_1 \text{ or } \mathcal{I}_{\mathcal{H}}; w \models \varphi_2; \\
\mathcal{I}_{\mathcal{H}}; w \models \Box\varphi & \text{iff} & \forall w' \in W_{\mathcal{H}} \text{ such that } w \subseteq w', \mathcal{I}_{\mathcal{H}}; w' \models \varphi; \\
\mathcal{I}_{\mathcal{H}}; w \models \Diamond\varphi & \text{iff} & \exists w' \in W_{\mathcal{H}} \text{ such that } w \subseteq w' \text{ and } \mathcal{I}_{\mathcal{H}}; w' \models \varphi.
\end{array}$$

A GMEC-formula $\varphi$ is *valid* in $\mathcal{I}_{\mathcal{H}}$, written $\mathcal{I}_{\mathcal{H}} \models \varphi$, if $\mathcal{I}_{\mathcal{H}}; w \models \varphi$ for all $w \in W_{\mathcal{H}}$. □

We will drop the subscripts $_{\mathcal{H}}$ whenever this does not lead to ambiguities. Moreover, given a knowledge state $w$ in $W_{\mathcal{H}}$ and a GMEC-formula $\varphi$ over $\mathcal{H}$, we write $w \models \varphi$ for $\mathcal{I}_{\mathcal{H}}; w \models \varphi$. Similarly, we abbreviate $\mathcal{I}_{\mathcal{H}} \models \varphi$ as $\models \varphi$.

This definition formalizes the strong interpretation of the initiate and terminate relations, as discussed in Section 2.1.

Notice that the definition of satisfiability given in the previous inductive definition is always *consistent*, i.e. for every knowledge state $w \in W_{\mathcal{H}}$ and formula $\varphi$ it is not possible to have both $w \models \varphi$ and $w \models \neg\varphi$. In the sequel, we will take advantage of a slightly different formulation of consistency. We have the following property, easily proved by induction on the structure of the formula $\varphi$.

*Property 2.1.* (*Completeness of the satisfiability relation*)

*Let $\mathcal{H} = (E,\ P,\ [\cdot\rangle,\ \langle\cdot],\ ]\cdot,\cdot[)$ be a GMEC-structure. For all $w \in W$ and GMEC-formula $\varphi$, if $w \not\models \neg\varphi$, then $w \models \varphi$.* □

The attempt to characterize GMEC within the rich taxonomy of modal logics [18] reveals *Sobocinski logic*, also known as *system K1.1* [28], as its closest relative. Syntactically, this logic extends S4 with the formula $\Box(\Box(\varphi \to \Box\varphi) \to \varphi) \to \varphi$, added as a further axiom to the traditional formulation of that system. Semantically, it is characterized by the class of the finite, reflexive, transitive and antisymmetric frames, i.e. by the class of all finite reflexive partial orderings. The relationship between GMEC and K1.1 is captured by the following theorem, where derivability in K.1.1 has been indicated as $\vdash_{K1.1}$.

*Theorem 2.2.* (*GMEC and K1.1*)

*If a GMEC-formula $\varphi$ is a thesis of K1.1, then it is a valid formula of GMEC, i.e., for each GMEC-formula $\varphi$, if $\vdash_{K1.1} \varphi$, then $\models \varphi$.* □

Since the intended GMEC-model $\mathcal{I}_{\mathcal{H}}$ is based on a finite, reflexive, transitive and antisymmetric frame, Theorem 2.2 immediately follows from the soundness of K1.1 with respect to the class of all finite reflexive partial orderings [28].

From the above syntactic characterization of Sobocinski logic, every formula valid in S4 is valid in K1.1. Therefore, Theorem 2.2 permits lifting to GMEC the following well-known equivalences of S4 [18].

*Corollary 2.1.* (*Some equivalent GMEC-formulas*)

*Let $\varphi$, $\varphi_1$ and $\varphi_2$ be GMEC-formulas. Then, for every knowledge state $w \in W$,*

- $w \models \Box\neg\varphi$ *iff* $w \models \neg\Diamond\varphi$
- $w \models \Diamond\neg\varphi$ *iff* $w \models \neg\Box\varphi$
- $w \models \Box(\varphi_1 \wedge \varphi_2)$ *iff* $w \models \Box\varphi_1 \wedge \Box\varphi_2$
- $w \models \Diamond(\varphi_1 \vee \varphi_2)$ *iff* $w \models \Diamond\varphi_1 \vee \Diamond\varphi_2$
- $w \models \Box\Box\varphi$ *iff* $w \models \Box\varphi$
- $w \models \Diamond\Diamond\varphi$ *iff* $w \models \Diamond\varphi$
- $w \models \Box\Diamond\Box\Diamond\varphi$ *iff* $w \models \Box\Diamond\varphi$
- $w \models \Diamond\Box\Diamond\Box\varphi$ *iff* $w \models \Diamond\Box\varphi$ □

These equivalences are often presented in the literature using the equivalence connective $\leftrightarrow$ (e.g., the first case would be expressed as $w \models \Box\neg\varphi \leftrightarrow \neg\Diamond\varphi$ for every state of knowledge $w$). Since we did not include this connective in the language of GMEC, we cannot exploit this somewhat simpler option.

Also specific properties of K1.1 will turn out useful in order to implement GMEC. The following equivalences can be obtained by exploiting the *McKinsey formula*,

$\Box\Diamond\varphi \rightarrow \Diamond\Box\varphi$, valid in K1.1 (but not in S4).

*Corollary 2.2.* (*Further equivalent GMEC-formulas*)

*Let $\varphi$ be a GMEC-formula. Then, for every knowledge state $w \in W$,*

- $w \models \Box\Diamond\Box\varphi$ *iff* $w \models \Box\Diamond\varphi$
- $w \models \Diamond\Box\Diamond\varphi$ *iff* $w \models \Diamond\Box\varphi$ □

An interesting consequence of Corollaries 2.1 and 2.2 is that each GMEC-formula $\varphi$ is logically equivalent to a formula of one of the following forms: $\psi$, $\Box\psi$, $\Diamond\psi$, $\Box\Diamond\psi$, $\Diamond\Box\psi$, where the main connective of $\psi$ is non-modal. Such reductions will result particularly useful in Section 3. Observe also that, unfortunately, there is no way of reducing formulas of the form $\Box(\varphi_1 \vee \varphi_2)$ and $\Diamond(\varphi_1 \wedge \varphi_2)$.

## *2.4. Properties of the Formalization*

We will now give a number of results concerning the adequacy of the definition of GMEC-structure with respect to the informal concept of MVI introduced in [20], and the modal extensions defined in [2, 5, 10]. We have already shown that a satisfiable atomic formula $p(e_1, e_2)$ identifies an interval during which the property $p$ holds. These intervals are maximal and uninterrupted, i.e. $p$ does not hold on any superinterval or subinterval of $(e_1, e_2)$:

*Lemma 2.1.* (*Satisfiable atomic formulas are MVIs*)

*Let $\mathcal{H} = (E,\ P,\ [\cdot\rangle,\ \langle\cdot],\ ]\cdot,\cdot[)$ be a GMEC-structure and $w \in W$ such that $w \models p(e_1, e_2)$. Then $\forall e_1', e_2' \in E$,*

*a. if $(e_1', e_2') \sqsubset_w (e_1, e_2)$, then $w \not\models p(e_1', e_2')$;*

*b. if $(e_1, e_2) \sqsubset_w (e_1', e_2')$, then $w \not\models p(e_1', e_2')$.*

Proof.

*a.* Assume *ab absurdum* that $(e_1', e_2') \sqsubset_w (e_1, e_2)$ and $w \models p(e_1', e_2')$. If $(e_1, e_1') \in w$, then, $e_1'$ would violate $nb(p, e_1, e_2, w)$, and therefore $w \not\models p(e_1, e_2)$. The situation is similar if $(e_2', e_2) \in w$.

*b.* By assuming $(e_1, e_2) \sqsubset_w (e_1', e_2')$ and $w \models p(e_1', e_2')$, we obtain a situation that is dual to the previous case. □

In this paper, we use GMEC to investigate how the MVIs derivable within the current set of ordered pairs of events is updated due to the arrival of new ordering information. We have shown in [5] that the set of MVIs computed by EC can change non-monotonically in response to the acquisition of ordering data. We wish to find the laws that rule this behavior. GMEC entitles us to identify on the one hand the set of MVIs that cannot be invalidated no matter how the ordering information is updated (as far as it remains consistent), and on the other hand those intervals that will possibly become MVIs depending on which ordering data are acquired. Notice that this statement must be relativized to the current set of events: we do not (and in general cannot) predict the behavior of the system as new event happenings are

recorded, but we are able to draw conclusions about how the current system can evolve as the ordering information is refined.

The sets of MVIs that are necessarily and possibly valid in the current state of knowledge $w$ correspond respectively to the $\Box$- and $\Diamond$-moded atomic formulas which are valid in $w$. We define the sets $MVI(w)$, $\Box MVI(w)$ and $\Diamond MVI(w)$ of respectively MVIs, necessary MVIs and possible MVIs with respect to $w$ as follows:

$$\begin{aligned} MVI(w) &= \{p(e_1, e_2) : w \models p(e_1, e_2)\}; \\ \Box MVI(w) &= \{p(e_1, e_2) : w \models \Box p(e_1, e_2)\}; \\ \Diamond MVI(w) &= \{p(e_1, e_2) : w \models \Diamond p(e_1, e_2)\}. \end{aligned}$$

In the following, it will be useful to view these sets as functions $MVI(\cdot)$, $\Box MVI(\cdot)$ and $\Diamond MVI(\cdot)$ of the knowledge state $w$.

We show now that the set of necessary MVIs with respect to $w$ persists whatever the evolution of the ordering information will be. Similarly, each element in the set of possible MVIs of $w$ is valid in at least one extension of $w$.

*Lemma 2.2.* (*Behavior of* $\Box MVI(\cdot)$ *and* $\Diamond MVI(\cdot)$ *with respect to* $MVI(\cdot)$)
*Let* $\mathcal{H} = (E,\ P,\ [\cdot\rangle,\ \langle\cdot],\ ]\cdot,\cdot[)$ *be a GMEC-structure and* $w \in W$, *then*

*a.* *if* $p(e_1, e_2) \in \Box MVI(w)$, *then* $\forall w' \in Ext(w)$, $p(e_1, e_2) \in MVI(w')$;
*b.* *if* $p(e_1, e_2) \in \Diamond MVI(w)$, *then* $\exists w' \in Ext(w)$, $p(e_1, e_2) \in MVI(w')$.

Proof.

*a.* $p(e_1, e_2) \in \Box MVI(w)$ *iff* $w \models \Box p(e_1, e_2)$,
*iff* $\forall w'$ such that $w \subseteq w', w' \models p(e_1, e_2)$
*iff* $\forall w' \in Ext(w), w' \models p(e_1, e_2)$
*iff* $\forall w' \in Ext(w), p(e_1, e_2) \in MVI(w')$.

*b.* Similar. □

The sets of necessary MVIs, MVIs and possible MVIs in the current state of knowledge form an inclusion chain as formally stated by Lemma 2.3.

*Lemma 2.3.* (*Necessary MVIs and possible MVIs enclose MVIs*)
*Let* $\mathcal{H} = (E,\ P,\ [\cdot\rangle,\ \langle\cdot],\ ]\cdot,\cdot[)$ *be a GMEC-structure and* $w \in W$, *then*

$$\Box MVI(w) \subseteq MVI(w) \subseteq \Diamond MVI(w).$$

Proof. By the definition of the involved sets, these relations can be rewritten as follows:

*a.* if $w \models \Box p(e_1, e_2)$, then $w \models p(e_1, e_2)$;
*b.* if $w \models p(e_1, e_2)$, then $w \models \Diamond p(e_1, e_2)$.

The validity of these expressions is a direct consequence of the reflexivity of the accessibility relation of GMEC-frames. Indeed, $w \models \Box p(e_1, e_2)$ *iff* $p(e_1, e_2)$ is valid in every extension of $w$, in particular in $w$ itself. Analogously, if $w \models p(e_1, e_2)$, then $w \models \Diamond p(e_1, e_2)$. □

When the arrival of a new piece of ordering information causes a transition into a more refined state of knowledge, the current set of MVIs can be subject to two

transformations. On the one hand, the update may create a new MVI by connecting an event $e_1$, initiating a property $p$, and an event $e_2$ terminating $p$. On the other hand, a new link can transform a previously innocuous event $e$ into an interrupting event for some MVI $p(e_1, e_2)$. Therefore, the function $MVI(\cdot)$ is non-monotonic with respect to the evolution of the ordering information.

On the other hand, $\Box MVI(\cdot)$ and $\Diamond MVI(\cdot)$ possess a monotonic behavior: the set of necessary MVIs can only grow as the current ordering information is refined, while the set of possible MVIs shrinks monotonically as we acquire new ordering information and a smaller number of future states is viable.

*Lemma 2.4.* (*Monotonicity of* $\Box$- *and* $\Diamond$-*moded atomic formulas*)

*Let* $\mathcal{H} = (E,\ P,\ [\cdot\rangle,\ \langle\cdot],\ ]\cdot,\cdot[)$ *be a GMEC-structure and* $w$ *and* $w'$ *two states of knowledge, then*

*a. if* $w \subseteq w'$ *then* $\Box MVI(w) \subseteq \Box MVI(w')$*;*

*b. if* $w \subseteq w'$ *then* $\Diamond MVI(w') \subseteq \Diamond MVI(w)$.

Proof. By the definition of $MVI(\cdot)$, $\Box MVI(\cdot)$ and $\Diamond MVI(\cdot)$, these relations can be rewritten as follows:

*a.* if $w \models \Box p(e_1, e_2)$, then $w' \models \Box p(e_1, e_2)$;

*b.* if $w' \models \Diamond p(e_1, e_2)$, then $w \models \Diamond p(e_1, e_2)$.

By the definition of GMEC-frame, where $\subseteq$ plays the role of accessibility relation, these relations hold trivially: if $w \models \Box p(e_1, e_2)$, then $p(e_1, e_2)$ is valid in every extension of $w$, but these comprise all extensions of $w'$, thus $w' \models \Box p(e_1, e_2)$; similarly, if $w' \models \Diamond p(e_1, e_2)$ then $p(e_1, e_2)$ holds in an extension $w^*$ of $w'$, but since $w \subseteq w'$ and $\subseteq$ is transitive, $w^*$ is an extension of $w$ as well, and thus $w \models \Diamond p(e_1, e_2)$. □

By combining the interpretations of Lemmas 2.3 and 2.4, we have that $\Box MVI(\cdot)$ and $\Diamond MVI(\cdot)$ constrain the variability of the set of MVIs derivable using EC. The state of minimum information corresponds to the absence of any ordering data: $\Box MVI(\cdot)$ and $MVI(\cdot)$ derive no formula, while $\Diamond MVI(\cdot)$ derives all consistent property-labeled intervals. As new ordering information arrives, $\Box MVI(\cdot)$ increases, $\Diamond MVI(\cdot)$ decreases, but $MVI(\cdot)$ always sits somewhere between them. When enough ordering information has been entered (at worst when the set of events has been completely ordered) $\Box MVI(\cdot)$ and $\Diamond MVI(\cdot)$ meet at a common value constraining $MVI(\cdot)$ to assume that same value.

The following example shows that the GMEC fragment including only atomic formulas and simply moded atomic formulas is expressive enough to model the operations of the beverage dispenser.

*Example 2.1.* (Beverage dispenser)

*We consider again the operations of the simple beverage dispenser introduced in Section 2.1 and depicted in Figure 2.2 (left). We recall that by setting the selector to the* apple *or to the* orange *position, apple juice or orange juice is obtained, respectively. On the other hand, choosing the* stop *position terminates the production of juice.*

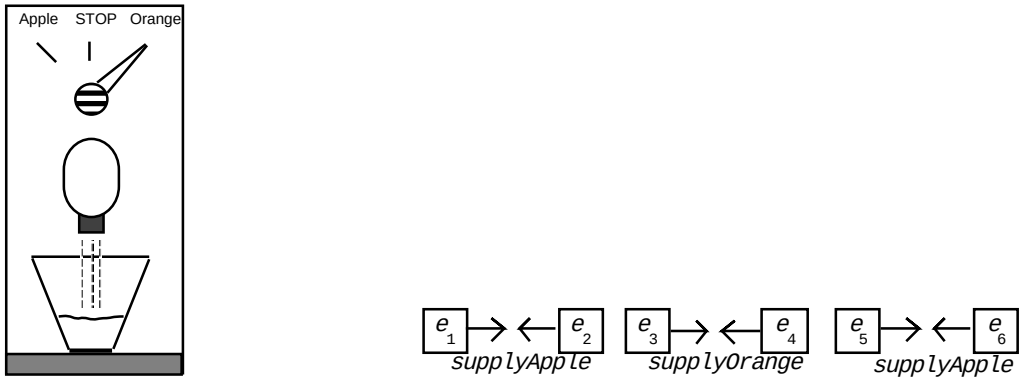


**FIGURE 2.2.** The Beverage Dispenser Revisited

*We consider a scenario consisting of two events ($e_1$ and $e_5$) that initiate the property* supplyApple*, an event ($e_3$) that initiates the property* supplyOrange*, and three stop events ($e_2$, $e_4$ and $e_6$) that terminate both properties. In GMEC, this knowledge is modeled as follows:*

$$E = \{e_1, e_2, e_3, e_4, e_5, e_6\};$$
$$P = \{supplyApple,\ supplyOrange\};$$
$$[supplyApple\rangle = \{e_1, e_5\};$$
$$[supplyOrange\rangle = \{e_3\};$$
$$\langle supplyApple] = \langle supplyOrange] = \{e_2, e_4, e_6\};$$
$$]supplyApple, supplyOrange[.$$

*Suppose that, in the intended final ordering, events are ordered according to their indices (such a situation is described in Figure 2.2, right). Let us consider the following sequence of ordered pairs, which are assumed to be entered in the database one at a time: $(e_1, e_4)$; $(e_1, e_6)$; $(e_2, e_4)$; $(e_1, e_2)$; $(e_3, e_4)$; $(e_4, e_5)$; $(e_2, e_3)$; $(e_2, e_6)$; $(e_5, e_6)$. This sequence has been devised so that the complete situation shown in Figure 2.2 can be fully derived only after the last update. These* 9 *ordered pairs are entered into the database in sequence by means of the predicate* UpdOrd*. Figure 2.3 shows the evolution of the computation: each row corresponds to the addition of one of these ordered pairs to the database.*

*The first column shows which update is being performed. The second column contains the list of the MVIs derived by EC. Here, we write $a(e_i, e_t)$ for the MVI concerning the property* supplyApple*, initiated by event $e_i$ and terminated by $e_t$, and $o(e_i, e_t)$ for the similar situation involving* supplyOrange*. The third and fourth columns contain the list of necessary and possible MVIs, respectively.*

*This trace clearly shows the non-monotonic behavior of $MVI(\cdot)$: as new ordering information is entered, the set of MVIs grows bigger and bigger till the pair $(e_2, e_3)$ is asserted; then the MVI $a(e_1, e_6)$ is dropped. Instead, $\Box MVI(\cdot)$ and $\Diamond MVI(\cdot)$ evolve monotonically and anti-monotonically, respectively. Notice that non-trivial necessary MVIs are generated only when almost all the ordering information has been entered; therefore, it provides useful data only when the state of knowledge is*

| | MVIs derived by EC | Necessary MVIs | Possible MVIs |
|---|---|---|---|
| | Ø | Ø | $a(e_1,e_2), a(e_1,e_4)$ $a(e_1,e_6), o(e_3,e_2)$ $o(e_3,e_4), o(e_3,e_6)$ $a(e_5,e_2), a(e_5,e_4)$ $a(e_5,e_6)$ |
| ?- updOrd$(e_1,e_4)$. | $a(e_1,e_4)$ | Ø | $a(e_1,e_2), a(e_1,e_4)$ $a(e_1,e_6), o(e_3,e_2)$ $o(e_3,e_4), o(e_3,e_6)$ $a(e_5,e_2), a(e_5,e_4)$ $a(e_5,e_6)$ |
| ?- updOrd$(e_1,e_6)$. | $a(e_1,e_4), a(e_1,e_6)$ | Ø | $a(e_1,e_2), a(e_1,e_4)$ $a(e_1,e_6), o(e_3,e_2)$ $o(e_3,e_4), o(e_3,e_6)$ $a(e_5,e_2), a(e_5,e_4)$ $a(e_5,e_6)$ |
| ... | ... | ... | ... |
| ?- updOrd$(e_2,e_3)$. | $a(e_1,e_2), a(e_1,e_6)$ $o(e_3,e_4)$ | Ø | $a(e_1,e_2), a(e_1,e_6)$ $o(e_3,e_4), o(e_3,e_6)$ $a(e_5,e_6)$ |
| ?- updOrd$(e_2,e_6)$. | $a(e_1,e_2), o(e_3,e_4)$ | $a(e_1,e_2)$ | $a(e_1,e_2), o(e_3,e_4)$ $o(e_3,e_6), a(e_5,e_6)$ |
| ?- updOrd$(e_5,e_6)$. | $a(e_1,e_2), o(e_3,e_4)$ $a(e_5,e_6)$ | $a(e_1,e_2), o(e_3,e_4)$ $a(e_5,e_6)$ | $a(e_1,e_2), o(e_3,e_4)$ $a(e_5,e_6)$ |

**FIGURE 2.3.** Computation in the The Beverage Dispenser Example

*nearly complete. On the other hand, possible MVIs are significatively pruned at much earlier stages of the insertion process.* □

We now move to the general case of arbitrary GMEC-formulas. The following lemma stands as the basis for the treatment of the modal operators in Section 3. It shows how the satisfiability test for an arbitrary GMEC-formula having a modality as its main connective can be reduced to first testing the satisfiability of its immediate subformula in the current world and then checking the satisfiability of the original formula in the 'one-step' extensions of the current knowledge state.

*Lemma 2.5.* (*Unfolding modalities*)

*Let* $\mathcal{H} = (E,\ P,\ [\cdot\rangle,\ \langle\cdot],\ ]\cdot,\cdot[)$ *be a GMEC-structure,* $\varphi \in \mathcal{L}_{\mathcal{H}}$ *a GMEC-formula over* $\mathcal{H}$*, and* $w \in W$*. Then*

*a.* $w \models \Box\varphi$ *iff* $w \models \varphi$ *and* $\forall(e_1,e_2)$ *such that* $(e_1,e_2),(e_2,e_1) \notin w$*,* $w \uparrow \{(e_1,e_2)\} \models \Box\varphi$*;*

*b.* $w \models \Diamond\varphi$ *iff* $w \models \varphi$ *or* $\exists(e_1,e_2)$ *such that* $(e_1,e_2),(e_2,e_1) \notin w$*,* $w \uparrow \{(e_1,e_2)\} \models \Diamond\varphi$*.*

Proof. First notice that if $(e_1,e_2) \notin w$ and $(e_2,e_1) \notin w$, then $w \uparrow \{(e_1,e_2)\} \in W$ since, in this case, upgrading $w$ with $(e_1,e_2)$ cannot violate asymmetry in any way.

Moreover, for every $w \in W$,

$$Ext(w) = \{w\} \cup \bigcup_{\substack{(e_1,e_2)\notin w \\ (e_2,e_1)\notin w}} Ext(w \uparrow \{(e_1, e_2)\}).$$

Indeed, let $w' \in Ext(w)$. Then, by definition, $w \subseteq w'$. Therefore, either $w' = w$ or there exists a pair $(e_1, e_2) \in w' \setminus w$. In the latter case, $w' \in Ext(w \uparrow \{(e_1, e_2)\})$. The opposite inclusion is straightforward.

We have now the needed tools to prove the statement of the lemma.

*a.* $w \models \Box\varphi$ *iff* $\forall w' \in Ext(w), w' \models \varphi$
*iff* $\forall w' \in \{w\} \cup \bigcup_{\substack{(e_1,e_2)\notin w \\ (e_2,e_1)\notin w}} Ext(w \uparrow \{(e_1, e_2)\}), w' \models \varphi$
*iff* $w \models \varphi$ and for each $e_1, e_2 \in E$ such that $(e_1, e_2) \notin w$ and $(e_2, e_1) \notin w$, it holds that for each $w' \in Ext(w \uparrow \{(e_1, e_2)\})$, $w' \models \varphi$
*iff* $w \models \varphi$ and for each $e_1, e_2 \in E$ such that $(e_1, e_2), (e_2, e_1) \notin w, w \uparrow \{(e_1, e_2)\} \models \Box\varphi$.

*b.* The proof is similar to *a.* □

In the sequel, we will use a different but clearly equivalent form of $(a)$:

$w \models \Box\varphi$ iff $w \models \varphi$ and
it is not the case that
$\exists (e_1, e_2)$ such that $(e_1, e_2), (e_2, e_1) \notin w$. $w \uparrow \{(e_1, e_2)\} \not\models \Box\varphi$.

Next, we seek for a manner of computing necessary and possible MVIs (simply moded atomic formulas) that does not require to explore future states of knowledge. In both cases, we will be able to devise necessary and sufficient local conditions. These properties stand as the basis for the implementation of SKEC and CREC [2, 10], and will allow us to improve the naive implementation of GMEC in Section 3 on the basis of the results of Lemma 2.5.

An MVI $p(e_1, e_2)$ is undefeasible whatever ordering information is acquired if no event can interrupt it. An event $e$ can possibly interrupt the validity of $p(e_1, e_2)$ if it initiates or terminates $p$ or a property that is exclusive with $p$, and it could be consistently located between $e_1$ and $e_2$ with respect to $w$. This intuition is formalized in the following lemma: the first three conditions express the validity of $p(e_1, e_2)$ as a $p$-labeled interval of $w$; the meta-predicate $nsb(p, e_1, e_2, w)$ in the fourth condition states that no event $e$ can possibly interrupt the validity of $p(e_1, e_2)$ in the sense just explained.

*Lemma 2.6.* (*Local condition for atomic necessity*)

*Let $\mathcal{H} = (E,\ P,\ [\cdot\rangle,\ \langle\cdot],\ ]\cdot,\cdot[)$ be a GMEC-structure. Then for any $e_1, e_2 \in E$, $p \in P$ and $w \in W$, $p(e_1, e_2) \in \Box MVI(w)$ iff the following conditions are satisfied:*

- $e_1 \in [p\rangle$,
- $e_2 \in \langle p]$,

- $(e_1, e_2) \in w$,
- $nsb(p, e_1, e_2, w)$

*where* $nsb(p, e_1, e_2, w)$ *stands for the expression*

$$\begin{aligned}\forall e \in E.\, \forall q \in P.\ & e = e_1 \\ & \vee\ e = e_2 \\ & \vee\ (e, e_1) \in w \\ & \vee\ (e_2, e) \in w \\ & \vee\ (e \in [q\rangle\ \vee\ e \in \langle q]\ \rightarrow\ \neg]p, q[\wedge p \neq q).\end{aligned}$$

Proof. By definition, the first member of the equivalence, $p(e_1, e_2) \in \Box MVI(w)$, reduces to $w \models \Box p(e_1, e_2)$. We will take advantage of this formulation in the proof.

($\Leftarrow$) Let us proceed by contradiction. So, assume that $e_1 \in [p\rangle$, $e_2 \in \langle p]$, $(e_1, e_2) \in w$ and $nsb(p, e_1, e_2, w)$, but there exist an extension $w'$ of $w$ such that $w' \models p(e_1, e_2)$ does not hold, i.e. such that $nb(e_1, e_2, w')$ is false. After some logical manipulations, the latter statement rewrites to

$$\exists e \in E.\, \exists q \in P.\, ((e_1, e) \in w' \wedge (e, e_2) \in w' \wedge (e \in [q\rangle\ \vee\ e \in \langle q]) \wedge (]p, q[\vee p = q).$$

Let $e'$ and $q'$ witness the validity of this formula. By instantiation, we obtain:

$$(e_1, e') \in w' \wedge (e', e_2) \in w' \wedge (e' \in [q'\rangle\ \vee\ e' \in \langle q']) \wedge (]p, q'[\vee p = q') \quad (2.1)$$

We can instantiate the expression for $nsb(p, e_1, e_2, w)$ with these values too. The resulting formula is:

$$\begin{aligned}& e' = e_1\ \vee\ e' = e_2\ \vee\ (e', e_1) \in w\ \vee \\ & (e_2, e') \in w\ \vee\ (e' \in [q'\rangle \wedge e' \in \langle q']\ \rightarrow\ \neg]p, q'[\wedge p \neq q')\end{aligned} \quad (2.2)$$

We must show that none of the alternatives in formula (2.2) applies. Since $w'$ is a (strict) partial order, the validity of (2.1) implies that $e'$ can be neither $e_1$ nor $e_2$. Analogously, by Lemma 2.4, either $(e', e_1) \in w$ or $(e_2, e') \in w$ would violate the asymmetry of $w'$. Finally, the choice of $q'$ contradicts the last alternative, i.e. that $(e' \in [q'\rangle \vee e' \in \langle q'] \rightarrow \neg]p, q'[\wedge p \neq q')$. This concludes this direction of the proof.

($\Rightarrow$) We will again proceed by contradiction. Clearly, if $e_1 \notin [p\rangle$ or $e_2 \notin \langle p]$, then we cannot obtain $w' \models p(e_1, e_2)$ in any state of knowledge $w'$. If $(e_1, e_2) \notin w$, then there exist extensions of $w$ containing $(e_2, e_1)$. Because of asymmetry, these extensions cannot contain $(e_1, e_2)$, thus $p(e_1, e_2)$ cannot be valid in them.

Assume now that $e_1 \in [p\rangle$, $e_2 \in \langle p]$ and $(e_1, e_2) \in w$, but that $nsb(p, e_1, e_2, w)$ does not hold. Therefore, there are an event $e'$ and a property $q'$ such that:

$$e' \neq e_1 \wedge e' \neq e_2 \wedge (e', e_1) \notin w \wedge (e_2, e') \notin w \wedge (e' \in [q'\rangle \vee e' \in \langle q']) \wedge (]p, q'[\vee p = q').$$

Since the pair $(e_1, e_2) \in w$, there exists at least one extension $w'$ of $w$ such that $(e_1, e') \in w'$ and $(e', e_2) \in w'$. Therefore,

$$(e_1, e') \in w' \wedge (e', e_2) \in w' \wedge (e' \in [q'\rangle\ \vee\ e' \in \langle q']) \wedge (]p, q'[\vee p = q')$$

hence $nb(p, e_1, e_2, w')$ does not hold. This contradicts the hypothesis that $w \models \Box p(e_1, e_2)$. □

It is worth noting that the definition of $nsb(p, e_1, e_2, w)$ is more restrictive than that of $nb(p, e_1, e_2, w)$. We call *critical* for a given property $p$ an event $e$ initiating or

terminating a property $q$ such that either $p = q$ or $]p, q[$. Condition $nb(p, e_1, e_2, w)$ states that there are no critical events $e \in E$ such that both $(e_1, e) \in w$ and $(e, e_2) \in w$, while condition $nsb(p, e_1, e_2, w)$ states that there are no critical events $e \in E$, with $e \neq e_1$ and $e \neq e_2$, such that both $(e, e_1) \notin w$ and $(e_2, e) \notin w$.

A labeled interval $p(e_1, e_2)$ might become an MVI for $p$ in an extension of the current knowledge state $w$ if $e_1$ initiates $p$, $e_2$ terminates $p$, the interval $(e_1, e_2)$ is consistent with $w$ (i.e. $(e_2, e_1) \notin w$), and there are no already known interrupting events between $e_1$ and $e_2$. More formally, we have that:

*Lemma 2.7.* (*Local condition for atomic possibility*)

*Let $\mathcal{H} = (E,\ P,\ [\cdot\rangle,\ \langle\cdot],\ ]\cdot,\cdot[)$ be a GMEC-structure. Then for any $e_1, e_2 \in E$, $p \in P$ and $w \in W$, $p(e_1, e_2) \in \Diamond MVI(w)$ iff the following conditions are satisfied:*

- $e_1 \in [p\rangle$,
- $e_2 \in \langle p]$,
- $(e_2, e_1) \notin w$,
- $nb(p, e_1, e_2, w)$.

Proof. As in the previous proof, we reduce the relation $p(e_1, e_2) \in \Diamond MVI(w)$ to $w \models \Diamond p(e_1, e_2)$. We operate on this equivalent formulation.

($\Leftarrow$) Let us construct an extension $w'$ of $w$ such that $w' \models p(e_1, e_2)$. The state of knowledge $w'$ is defined as $w' = (w \cup \{(e_1, e_2)\})^+$. First notice that $w'$ is consistent (i.e. it does not violate asymmetry) since $w$ is consistent and $(e_2, e_1) \notin w$. Then observe that $nb(p, e_1, e_2, w')$ holds by the definition of $w'$. Otherwise, we should be able to conclude that there is an event $e \in E$ such that $(e_1, e) \in w'$, $(e, e_2) \in w'$ and either $e \in [q\rangle$ or $e \in \langle q]$ for some property $q \in P$, with $]p, q[$ or $p = q$, but in that case, $(e_1, e) \in w$ and $(e, e_2) \in w$ contradicting the assumption that $nb(p, e_1, e_2, w)$ holds. Therefore, conditions (*i–iv*) of Definition 4 are satisfied w.r.t. $w'$; hence $w' \models p(e_1, e_2)$, and thus $w \models \Diamond p(e_1, e_2)$.

($\Rightarrow$) We proceed by contradiction. Clearly, if $e_1 \notin [p\rangle$ or $e_2 \notin \langle p]$, then we cannot obtain $w' \models p(e_1, e_2)$ in any state of knowledge $w'$. Analogously, if $(e_2, e_1) \in w$, then $(e_2, e_1)$ belongs to every extension of $w$, forbidding in this way condition (*iii*) of Definition 4 to be satisfied. Finally, if $nb(p, e_1, e_2, w)$ does not hold, (i.e. there is an event $e \in E$ such that $(e_1, e) \in w$, $(e, e_2) \in w$ and $e \in [q\rangle$ or $e \in \langle q]$ for some property $q \in P$ with $]p, q[ \vee p = q)$, then, by Lemma 2.4, the same condition would apply to every extension $w'$ as well, thus $nb(p, e_1, e_2, w')$ would not hold in any extension $w'$ of $w$ and $p(e_1, e_2) \notin \Diamond MVI(w)$. □

Notice that the four conditions in the statement of this lemma differ from conditions (*i–iv*) in Definition 4 only by the replacement of $(e_1, e_2) \in w$ with $(e_2, e_1) \notin w$. In EC, we need to know that $(e_1, e_2)$ is indeed an interval of $w$, while, in the present case, we only need to know that this interval is compatible with $w$ (i.e. that this ordering does not contain the dual interval).

## 3. A LOGIC PROGRAMMING IMPLEMENTATION OF GMEC

In this section, we present an abstract implementation of GMEC in the language of hereditary Harrop formulas and prove its soundness and completeness with respect

to the GMEC semantics presented in Section 2. In Section 3.1, we recall the definition of hereditary Harrop formulas (HH-formulas for short) and their operational semantics as a logic programming language. In Section 3.2, we define an encoding of GMEC-structures, orderings and GMEC-formulas as HH-formulas. We also give a first naive program modeling the validity relation for GMEC-formulas. Section 3.3 proves the soundness and completeness of this program with respect to the notion of GMEC-model. Finally, in Section 3.4, we present an improved (semi-naive) implementation of GMEC and prove its soundness and completeness.

### *3.1. Hereditary Harrop Formulas*

So far, the implementation language for EC has almost always been the language of Horn clauses augmented with negation-as-failure [21], which constitutes the core of the logic programming language Prolog. This traditional Prolog implementation can be easily extended to cover the propositional connectives. Moreover, we showed in [2] that a restriction of the purely modal extension of EC can be conveniently encoded in this language by taking advantage of Lemmas 2.6 and 2.7. However, when mixing arbitrarily propositional connectives and modalities, as in GMEC, a direct encoding in Prolog appears unsatisfactory. The resulting program is in fact either highly non-declarative (for the necessary presence of a large number of `assert` and `retract` statements), or extremely complex (as we experienced in [10]). In conclusion, Prolog is not adequate for a declarative description of GMEC. In particular, it makes quite difficult to prove the fundamental soundness and completeness properties.

For the implementation of GMEC, we chose the language of first-order hereditary Harrop formulas [24] augmented with negation-as-failure. Extensions of this language, with or without negation-as-failure, have been used as the underlying logic of many logic programming languages successfully proposed in the last ten years, including Miller's $\lambda$Prolog [23], Gabbay's N-Prolog [13], Bonner's language for hypothetical reasoning in deductive databases [1], Pfenning's Elf [27] and Hodas and Miller's Lolli [17]. In this section, we extend the usual proof-theoretic semantics of HH-formulas [24] in order to encompass negation-as-failure. This presentation is new, although there are some similarities with the work of Harland [14, 15, 16].

Hereditary Harrop formulas extend Horn clauses by allowing the presence of implication and universal quantification in goal formulas. The former feature will give us declarative means of temporarily augmenting the program with new facts and performing in this manner a form of hypothetical reasoning. Universal quantification in goals provides a powerful tool for data and program abstraction: it allows, for instance, a purely declarative definition of abstract data types and modules. We will not take advantage of this last feature.

The language of hereditary Harrop formulas, defined in [24], is a subset of first-order intuitionistic logic. Formulas in this language are functionally subdivided in program formulas and goal formulas depending on whether they can appear as program clauses or they can only be used in queries. We use the syntactic variables $D$ and $G$ respectively to refer to these formulas. Program and goal formulas are mutually defined according to the following formal grammar, where $A$ ranges over atomic formulas:

$$
\begin{array}{lll}
D ::= A \mid \top \mid D_1 \wedge D_2 \mid G \to A \mid \forall x.D & & (\textit{Program formulas}) \\
G ::= A \mid \top \mid G_1 \wedge G_2 \mid D \to G \mid \forall x.G & & \\
\quad \mid \bot \mid G_1 \vee G_2 \quad\quad\quad\quad \mid \exists x.G & & (\textit{Goal formulas})
\end{array}
$$

Syntactically, hereditary Harrop formulas differ from Horn clauses only for the admissibility of implication and universal quantification in goal formulas (items 4 and 5 in the definition of $G$): as soon as we get rid of these productions, we obtain a language that is equivalent to Horn clauses. In order to represent negation-as-failure, we augment the definition of goal formulas with expressions of the form *not* $G$. A *hereditary Harrop clause* is a closed program formula of the form $\forall\vec{x}.(G \to A)$, where $\forall\vec{x}$ represents a possibly empty sequence of universal quantifications; $A$ and $G$ are called the *head* and the *body* of the clause, respectively. A closed formula of the form $\forall\vec{x}.A$ is called a *fact* and is considered as a clause with an empty body (i.e. $\forall\vec{x}.(\top \to A)$). Any program formula can be transformed into a set of clauses. In the following, we will use the terms $D$-formula and $G$-formula as synonyms of program formula and goal formula, respectively.

We will describe the semantics of hereditary Harrop formulas with negation-as-failure by means of two judgments called *positive* and *negative sequents* and denoted $\mathcal{P} \Longrightarrow G$ and $\mathcal{P} \not\Longrightarrow G$, respectively, where $\mathcal{P}$ is a set of $D$-formulas and $G$ a $G$-formula. Negative sequents are needed for defining negation-as-failure. In both cases, $\mathcal{P}$ and $G$ are called the *program* and the *goal* of the sequent respectively. If $c$ is a clause and $\mathcal{P}$ is a program, we abbreviate $\mathcal{P} \cup \{c\}$ as $(\mathcal{P}, c)$. As we said, any program is equivalent, modulo elementary logical manipulations, to a program consisting uniquely of clauses. We write $\mathcal{P}^c$ for the *clausal form* of the program $\mathcal{P}$.

Hereditary Harrop formulas constitute the biggest sublanguage of first-order logic that is complete with respect to *uniform proofs* [24]. Uniform provability views logical connectives in goal formulas as search directives for the construction of derivations and clauses as partial definitions of atomic formulas. The goal part of a sequent is decomposed up to the level of atomic formulas, and only then the program part is accessed in order to retrieve a clause defining this atom. The computation fails when trying to solve undefined instances of atomic formulas.

The non-deterministic search for a proof of the goal $G$ from the program $\mathcal{P}$ corresponds to the construction of a derivation for the positive sequent $\mathcal{P} \Longrightarrow G$ according to the rules to be defined below. Every derivation tree for $\mathcal{P} \Longrightarrow G$ built in this manner constitutes a *proof* of $G$ from $\mathcal{P}$. Therefore, $G$ is *provable* from $\mathcal{P}$ if there exists a proof-tree for the positive sequent $\mathcal{P} \Longrightarrow G$.

Conversely, $G$ is not provable from $\mathcal{P}$ if there is no proof-tree for $\mathcal{P} \Longrightarrow G$. In terms of (uniform) proof search, non-provability can come in two flavors: either every attempt at building a derivation for $\mathcal{P} \Longrightarrow G$ generates a sequent $\mathcal{P}' \Longrightarrow G'$ to which no rule is applicable, or an infinite tree is obtained by the application of the derivation rules, and in this case the search does not terminate. In the first case, we say that this sequent is *finitely non-provable*. In the second case, we say that the sequent is *divergent*. We define a (*finitely*) *failed derivation* as a derivation containing at least one leaf which sequent cannot be reduced by any of the rules discussed below. We call such a sequent *initially failed*.

When trying to find a proof for a negated goal *not* $G$ from the program $\mathcal{P}$, we want to show that there is no proof of $G$ from $\mathcal{P}$, i.e. that $\mathcal{P} \Longrightarrow G$ is not provable. It will become evident from the examples below that, in general, diverging sequents cannot be finitely recognized. Therefore, we are reduced to showing that $\mathcal{P} \Longrightarrow G$

is finitely non-provable. The (failed) proof-trees constructed during a search for a proof of $\mathcal{P} \Longrightarrow G$ are not accessible for this purpose. Therefore, we internalize them and model finite non-provability by means of negative sequents, $\mathcal{P} \not\Longrightarrow G$ in this case. Again, a derivation tree for $\mathcal{P} \not\Longrightarrow G$ constitutes a proof of this sequent. Notice that a positive sequent $\mathcal{P} \Longrightarrow G$ is initially failed if and only if $G = \bot$ or $G = A$ and there is no clause $\forall \vec{x}.(G \rightarrow A')$ and no substitution $\sigma$ such that $A'^{\sigma} = A$. Instead $\mathcal{P} \not\Longrightarrow G$ can be initially failed only if $G = \top$.

With an abuse of notation, we will sometimes informally use $\mathcal{P} \Longrightarrow G$ as an abbreviation for the sentence "the sequent $\mathcal{P} \Longrightarrow G$ is derivable", and similarly for negative sequents.

The derivability rules for positive and negative sequents have dual definitions. Moreover, the proof-search semantics of negated goals makes them mutually recursive. The complete definition is given in Figure 3.1. The rules that do not apply to Horn clauses are outlined. Notice that the rules **exist**– and **atom**– are non standard since some of the involved parameters (the term $t$ and the clause $\forall \vec{x}.(G \rightarrow A')$ respectively) are subject to extensional universal quantification. Therefore, **exist**– can be viewed as a rule with an infinite number of premisses (Harland has shown in [14, 16] that it is sufficient to consider a finite set of *representations*). Similarly **atom**– is better seen as a rule with a variable number of premisses depending on the number of matching clauses. We will discuss in depth the rule for universal quantification at the end of this section.

Let us now give some examples that better illustrate the distinction among provable, finitely non-provable and diverging sequents. These notions apply also to negative sequents and are defined similarly to the positive case.

- Let $\mathcal{P}_1 = \{a, a \rightarrow b\}$. The clausal form of $\mathcal{P}_1$ is $\mathcal{P}_1^c = \{\top \rightarrow a, a \rightarrow b\}$. The sequent $\mathcal{P}_1 \Longrightarrow b$ is provable by applying in sequence the rules **atom+**, **atom+** and **true+**. However, $\mathcal{P}_1 \not\Longrightarrow b$ fails after two applications of **atom**– (therefore it is finitely non-provable). On the other hand, $\mathcal{P}_1 \Longrightarrow c$ fails immediately while $\mathcal{P}_1 \not\Longrightarrow c$ succeeds by rule **atom**–.

- Let $\mathcal{P}_2 = \{a \rightarrow a\}$. Then both $\mathcal{P}_2 \Longrightarrow a$ and $\mathcal{P}_2 \not\Longrightarrow a$ diverge by infinite applications of the rules **atom+** and **atom**–, respectively. It is easy to notice that these sequents do not have any derivations since each step reproduces the original sequent. However, a simple loop-detection mechanism is not sufficient in most cases. Consider for instance a first-order variant of this example: $\mathcal{P}_2' = \{\forall x.(a(f(x)) \rightarrow a(x)\}$. Then, the sequents $\mathcal{P}_2' \Longrightarrow a(v)$ and $\mathcal{P}_2' \not\Longrightarrow a(v)$ diverge in the same manner, but at each stage the sequent to be proved is different. As a less trivial example, consider a non-terminating program that computes the decimal expansion of $\pi$.

- Finally, let $\mathcal{P}_3 = \{a, a \rightarrow a\}$. Clearly $\mathcal{P}_3 \Longrightarrow a$ is derivable. Notice that this sequent has infinitely many proofs, as well as a diverging derivation. On the other hand, $\mathcal{P}_3 \not\Longrightarrow a$ is not derivable since, after applying rule **atom**–, there is no way to proceed with the branch corresponding to $a$. Notice however that the resulting (failed) proof-tree is infinite.

We will now state the duality between positive and negative sequents. First, since we defined negative sequents with the aim of formalizing finite non-provability, it should not be possible that both the positive and the negative sequents involving the same program and goal are provable.

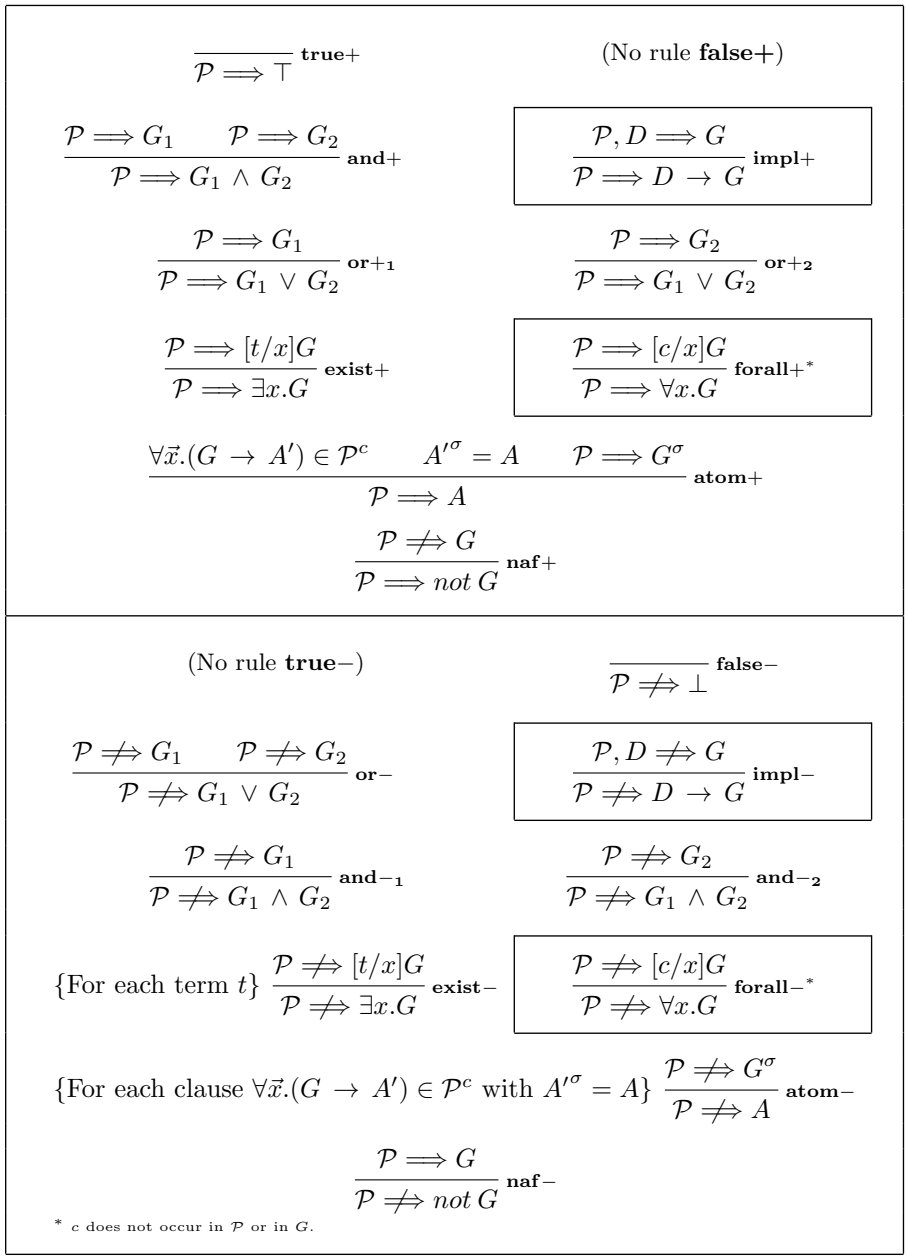


**FIGURE 3.1.** Sequent Rules for Hereditary Harrop Formulas with Negation-as-Failure

*Property 3.1.* (*Consistency of positive and negative sequents*)

*For given program $\mathcal{P}$ and goal $G$, either $\mathcal{P} \Longrightarrow G$ or $\mathcal{P} \not\Longrightarrow G$ is not derivable.*

PROOF. The proof proceeds by mutual induction on the structure of derivations for a positive and a negative sequent. More precisely, we show that if we assume given a derivation $\mathcal{D}^+$ of $\mathcal{P} \Longrightarrow G$, then there cannot be any derivation $\mathcal{D}^-$ of

$\mathcal{P} \not\Longrightarrow G$, and vice versa. We will analyze three representative situation. The remaining cases are similar or simpler.

**and+:** Assume that the given derivation $\mathcal{D}^+$ ends in the application of rule **and+**. The endsequent is therefore $\mathcal{P} \Longrightarrow G_1 \wedge G_2$. Let $\mathcal{D}_1^+$ and $\mathcal{D}_2^+$ be the immediate subderivations of $\mathcal{D}^+$, with endsequents $\mathcal{P} \Longrightarrow G_1$ and $\mathcal{P} \Longrightarrow G_2$, respectively. By induction hypothesis, there is no derivation of the negative sequents $\mathcal{P} \not\Longrightarrow G_1$ and $\mathcal{P} \not\Longrightarrow G_2$.

By inspection of the rules in Figure 3.1, the only ways to construct a derivation for the sequent $\mathcal{P} \not\Longrightarrow G_1 \wedge G_2$ are either to apply rule **and−1** to a derivation of $\mathcal{P} \not\Longrightarrow G_1$ or to apply rule **and−2** to a derivation of $\mathcal{P} \not\Longrightarrow G_2$. However we know that neither derivation can exist.

**atom−:** Assume that $\mathcal{D}^-$ ends with an application of rule **atom−**. Therefore $G$ is some atomic goal $A$, and for each clause $c_i = \forall \vec{x}.(G_i \rightarrow A_i') \in \mathcal{P}^c$ such that $A_i'^{\sigma_i} = A$ for substitutions $\sigma_i$, there is a derivation $\mathcal{D}_i^-$ of $\mathcal{P} \not\Longrightarrow G_i^{\sigma_i}$. By induction hypothesis, there is no derivation of any of the positive sequents $\mathcal{P} \Longrightarrow G_i^{\sigma_i}$. However, a derivation of $\mathcal{P} \Longrightarrow A$ can be produced only if one such derivation is achievable.

**naf+:** Assume that $\mathcal{D}^+$ ends with an application of rule **naf+** to a derivation of the sequent $\mathcal{P} \not\Longrightarrow G$. Then, by induction hypothesis, there is no derivation of $\mathcal{P} \Longrightarrow G$ and therefore of $\mathcal{P} \not\Longrightarrow \mathit{not}\, G$ since this goal can only be achieved if a derivation of that sequent is given. □

Harland has proved a similar result for a closely related rule system [14, 15, 16].

This property can be sharpened by considering finite non-provability. Indeed a positive sequent is finitely non-provable if and only if the corresponding negative sequent is derivable and has only finite derivations. The dual property obtained by flipping the adjectives positive and negative holds as well. For convenience, we prove the two direction of this property separately.

*Property 3.2.* (*Duality of positive/negative sequents for finite derivations—Part I*)
*Let $\mathcal{P}$ and $G$ be a program and a goal respectively. Then:*

- *If $\mathcal{P} \Longrightarrow G$ is finitely non-provable, then $\mathcal{P} \not\Longrightarrow G$ is provable;*
- *If $\mathcal{P} \not\Longrightarrow G$ is finitely non-provable, then $\mathcal{P} \Longrightarrow G$ is provable.*

Proof. Let $\mathcal{F}^+$ ($\mathcal{F}^-$) the set of all finitely failed derivations of $\mathcal{P} \Longrightarrow G$ ($\mathcal{P} \not\Longrightarrow G$, respectively). We proceed by induction on the height of the longest (finitely) failed derivation in $\mathcal{F}^+$ and $\mathcal{F}^-$, and by cases on the structure of $G$. We analyze two representative cases.

$G = G_1 \wedge G_2$ : The sequent $\mathcal{P} \Longrightarrow G_1 \wedge G_2$ is not initially failed since it can be reduced by means of rule **and+**. Moreover, every finitely failed derivation of this sequent (i.e. every element of $\mathcal{F}^+$) must end in this rule. Therefore either $\mathcal{P} \Longrightarrow G_1$ or $\mathcal{P} \Longrightarrow G_2$ (or both) must be finitely non-provable. Assume the first of the two is finitely non-provable, then by induction hypothesis there is a derivation of the negative sequent $\mathcal{P} \not\Longrightarrow G_1$. Therefore, we can apply

rule **and**$-_1$ in order to obtain a derivation of $\mathcal{P} \not\Longrightarrow G_1 \wedge G_2$. We proceed similarly in the other possible case.

In the negative case, the sequent $\mathcal{P} \not\Longrightarrow G_1 \wedge G_2$ is not initial since both rules **and**$-_1$ and **and**$-_2$ could have been applied. A finitely failed derivation in $\mathcal{F}^-$ then belongs to one of two groups: those ending in rule **and**$-_1$ and those ending in **and**$-_2$. Therefore, both $\mathcal{P} \not\Longrightarrow G_1$ and $\mathcal{P} \not\Longrightarrow G_2$ are finitely non-provable. By induction hypothesis, there are derivations of the positive sequents $\mathcal{P} \Longrightarrow G_1$ and $\mathcal{P} \Longrightarrow G_2$, to which it suffices to apply rule **and**+

$G = A$ : If $\mathcal{P} \Longrightarrow A$ is initially failed, then there is no clause $c = \forall \vec{x}.(G \rightarrow A') \in \mathcal{P}^c$ and substitution $\sigma$ such that $A'^{\sigma} = A$. Therefore rule **atom**$-$ is applicable without premisses in order to obtain a derivation of $\mathcal{P} \not\Longrightarrow A$.

If $\mathcal{P} \Longrightarrow A$ is not initially failed, then there are clauses $c_i = \forall \vec{x}.(G_i \rightarrow A_i') \in \mathcal{P}^c$ and substitutions $\sigma_i$ such that $A_i'^{\sigma_i} = A$. We can then partition $\mathcal{F}^+$ in classes $\mathcal{F}_i^+$ on the basis of the clauses $c_i$ (and substitutions $\sigma_i$) that have been used. Thus, each of the sequents $\mathcal{P} \Longrightarrow G_i^{\sigma_i}$ is finitely non-provable, and therefore by induction hypothesis, $\mathcal{P} \not\Longrightarrow G_i^{\sigma_i}$ is derivable. Then, simply apply rule **atom**$-$ to obtain the desired derivation of $\mathcal{P} \not\Longrightarrow A$.

Finally, the sequent $\mathcal{P} \not\Longrightarrow A$ is finitely non-provable if there is at least one clause $c = \forall \vec{x}.(G \rightarrow A') \in \mathcal{P}^c$ such that $A = A'^{\sigma}$ for some substitution $\sigma$ and $\mathcal{P} \not\Longrightarrow G^{\sigma}$ is finitely non-provable. Then, by induction hypothesis, $\mathcal{P} \Longrightarrow G^{\sigma}$ is derivable, and therefore, by rule **atom**+, so is $\mathcal{P} \Longrightarrow A$. □

We now prove the second part of the above property.

*Property 3.3.* (*Duality of positive/negative sequents for finite derivations—Part II*)
*Let $\mathcal{P}$ and $G$ be a program and a goal respectively. Then:*

- *If $\mathcal{P} \Longrightarrow G$ is provable and has only finite derivations, then $\mathcal{P} \not\Longrightarrow G$ is finitely non-provable;*
- *if $\mathcal{P} \not\Longrightarrow G$ is provable and has only finite derivations, then $\mathcal{P} \Longrightarrow G$ is finitely non-provable.*

Proof. Let $\mathcal{S}^+$ ($\mathcal{S}^-$) the set of all proofs of $\mathcal{P} \Longrightarrow G$ ($\mathcal{P} \not\Longrightarrow G$, respectively). We proceed by induction on the height of the longest derivation in $\mathcal{S}^+$ and $\mathcal{S}^-$, and by cases on the structure of $G$. The details of the proof are handled similarly to the previous property. □

We will take advantage of these results as follows. Let $p(\mathcal{P}, G)$ be a property of a given program $\mathcal{P}$ and a goal $G$. Assume that we are able to prove that $p(\mathcal{P}, G)$ *iff* $\mathcal{P} \Longrightarrow G$ is derivable. Then, if we know that $\mathcal{P} \Longrightarrow G$ has finite derivations only, we obtain as an immediate consequence that $\neg p(\mathcal{P}, G)$ *iff* $\mathcal{P} \Longrightarrow not\, G$ is derivable.

Negation-as-failure is distinct from classical negation: for example, the monotonicity property (*if $\mathcal{P} \Longrightarrow G$ is derivable, then so is $\mathcal{P}, c \Longrightarrow G$ for every $c$*) does not hold in general in languages embedding negation-as-failure (e.g. *not a* is derivable in the empty program, but not in the program $\{a\}$). However, as noted in [14] in a slightly different setting, negation-as-failure is a close approximation of the usual concept of negation in mathematical logic for programs characterized by finite derivations: indeed, in this specific setting, *not* $G$ is derivable in $\mathcal{P}$ precisely

when $G$ does not hold with respect to some notion of *completion* of $\mathcal{P}$ [14]. The representations of GMEC we will propose possess this property.

The fact that *not* can emulate to some extent classical negation does not turn the logic of hereditary Harrop formulas into a classical formalism, not even when dealing only with finite derivations. In particular, $\rightarrow$ is truly intuitionistic implication and it cannot be defined in terms of *not* and $\wedge$ (or $\vee$): the goal formula $D \rightarrow G$ is provable if and only if $G$ is provable assuming $D$ as a further program clause. This gives us the means of *changing* the program at hand by temporarily asserting new clauses. On the other hand, the operational semantics of *not* permits a non-monotonic behavior, as exemplified above.

According to the traditional semantics of hereditary Harrop formulas [24], when a universal goal of the form $\forall x.G(x)$ is encountered, it is reduced to $G(c)$, where $c$ is a *new* constant (rule **forall+**). Solving this goal requires to work abstractly with the generic individual $c$ only. Therefore, if this goal succeeds, $G(t)$ holds for every term $t$. This form of universal quantification is called *intensional*.

Below, we will need a different interpretation of this operator: $\forall x.G(x)$ is valid if $G(t)$ holds for every concrete term $t$ in a given collection, rather than for a generic individual. We may model this situation by means of the formula $\forall x \in S.\, G(x)$, where $S$ is some (recursive) set. This form of universal quantification is called *extensional*. Harland has investigated it in detail in conjunction with embedded implication [14, 16]. This quantifier cannot be represented within plain hereditary Harrop formulas. However, the presence of classical negation (modeled to some extent by negation-as-failure) allows to recover it as soon as we manage to represent the relation $x \in S$ by a predicate. Then, we rewrite the previous formula as $\neg\exists x.\, (x \in S \wedge \neg G(x))$. This formula is in turn equivalent to the goal

$$(\forall x.\, ((x \in S \wedge \neg G(x)) \rightarrow p')) \rightarrow \neg p',$$

where $p'$ is a new atomic formula. Notice that the quantifier is now in a program position; therefore, it will not be solved intensionally. As soon as we substitute logical negation ($\neg$) with negation-as-failure (*not*), we obtain a formula that is acceptable in our framework. We will take advantage of this implementation technique in order to model the semantics of the modalities of GMEC in Section 3.2.

We conclude this section by defining a concrete syntax for the language of hereditary Harrop formulas. We use identifiers beginning with lower case letters (e.g., `must`, `before`, ...) for constants and symbols beginning with uppercase letters for implicitly quantified variables (e.g., `Ei`, `P`, ...). We write terms and atoms in curried form (e.g., `(before Ei Et)` for the binary predicate `before` applied to the variables `Ei` and `Et`). The unary operator `not` is reserved to represent negation-as-failure when used in a goal formula (it will be convenient to overload it in Section 3.2 to model object level negation in a term position). The constants `true` and `fail` are reserved for the logical symbols $\top$ and $\bot$ respectively. We represent the logical operators $\wedge$, $\vee$ and $\rightarrow$ as the infix symbols `,` (comma), `;` (semicolon) and `=>` respectively, and the quantifiers $\forall x.$ and $\exists x.$ as `forall[X]` and `exist[X]` respectively. In a program position, we represent $\rightarrow$ as `:-` with the antecedent and the consequent reversed. We follow the usually accepted convention to drop the leading universal quantifiers when representing a clause in the concrete syntax.

### *3.2. Encoding of GMEC as Hereditary Harrop Formulas*

The aim of this section is twofold. We will first give a precise encoding of GMEC into the language of hereditary Harrop formulas. Then we will show a naive implementation of GMEC and give an informal overview of its features. The soundness and completeness of this encoding will be proved in the next section. Section 3.4 analyzes a more refined version of this implementation.

We define a family of representation functions $\ulcorner \cdot \urcorner^{\cdot}$ that relate the mathematical entities we have been using in Section 2 to the terms of the logic programming language we have chosen for the implementation. Specifically, we will need to encode GMEC-structures, the associated orderings, and the GMEC-language. In the remainder of this section, we will refer to the GMEC-structure $\mathcal{H} = (E,\ P,\ [\cdot\rangle,\ \langle\cdot],\ ]\cdot,\cdot[)$.

In order to represent $\mathcal{H}$, we need to give an encoding of the entities that constitute it. For this purpose, we first specify the functions $\ulcorner \cdot \urcorner^{E}$ and $\ulcorner \cdot \urcorner^{P}$ that give the concrete syntax of individual events and properties, respectively. We explicitly assume that these functions are injective, i.e. that every event $e$ in $E$ (property $p$ in $P$) has a representation that is different from that of all other events (resp. properties). Moreover, we want $\ulcorner \cdot \urcorner^{E}$ and $\ulcorner \cdot \urcorner^{P}$ to give distinct representations to events and properties. The exact definition of these functions is problem-specific.

The injectivity of the representation functions on events and properties enables us to utilize the respective inverse functions, $\llcorner \cdot \lrcorner_{E}$ and $\llcorner \cdot \lrcorner_{P}$, whenever they are defined. Notice indeed that $\llcorner t \lrcorner_{E}$ and $\llcorner t \lrcorner_{P}$ cannot be defined for all terms $t$. As a matter of convenience, we take the liberty of writing ill-formed expressions of the form $\llcorner t \lrcorner_{E} \in E$ for a generic $t$, assigning these expressions the truth value *false* whenever $t$ is not in the range of $\ulcorner \cdot \urcorner^{E}$.

The next step consists in defining the translation maps for $[\cdot\rangle$, $\langle\cdot]$ and $]\cdot,\cdot[$. We represent these relations by means of the binary predicates `initiates`, `terminates` and `exclusive`, respectively. The traditional formulations of EC give an explicit representation to the occurrences of events. We utilize the unary predicate `happens` for this purpose. The corresponding representation functions are defined as follows:

- $\ulcorner [\cdot\rangle \urcorner^{I} = \{\texttt{initiates}\ \ulcorner e \urcorner^{E}\ \ulcorner p \urcorner^{P} : e \in E, p \in P, \text{ and } e \in [p\rangle\}$;
- $\ulcorner \langle\cdot] \urcorner^{T} = \{\texttt{terminates}\ \ulcorner e \urcorner^{E}\ \ulcorner p \urcorner^{P} : e \in E, p \in P, \text{ and } e \in \langle p]\}$;
- $\ulcorner ]\cdot,\cdot[ \urcorner^{X} = \{\texttt{exclusive}\ \ulcorner p \urcorner^{P}\ \ulcorner q \urcorner^{P} : p, q \in P \text{ and } ]p, q[\}$;
- $\ulcorner E \urcorner^{H} = \{\texttt{happens}\ \ulcorner e \urcorner^{E} : e \in E\}$.

At this point, we define the representation of the GMEC-structure $\mathcal{H}$ by taking the union of the representations of its constituent entities:

$$\ulcorner \mathcal{H} \urcorner^{S} = \ulcorner E \urcorner^{H} \cup \ulcorner [\cdot\rangle \urcorner^{I} \cup \ulcorner \langle\cdot] \urcorner^{T} \cup \ulcorner ]\cdot,\cdot[ \urcorner^{X}.$$

In Section 2, we assumed that the ordering information of a GMEC problem was specified by means of partial orders in $W$. When integrating GMEC into practical applications, e.g. [10], this assumption turns out to be inadequate since, in general, the host system will simply pass the raw ordering data to the GMEC module as they are recorded. Therefore, we choose to represent this kind of information as our knowledge states and to reconstruct the corresponding strict ordering as needed. We assume the information source to be reliable, and thus the raw ordering information constitutes a binary acyclic relation in $O$. We use the binary predicate `beforeFact` to represent the atomic ordered pairs contained in a binary acyclic relation $o \in O$.

The function $\ulcorner \cdot \urcorner^O$ relates a knowledge state to its concrete syntax. It is defined as follows:

$$\ulcorner o \urcorner^O = \{\texttt{beforeFact } \ulcorner e_1 \urcorner^E \ \ulcorner e_2 \urcorner^E : (e_1, e_2) \in o\}.$$

The last entity we need to represent is the GMEC-language of $\mathcal{H}$. We encode the formulas in $\mathcal{L}_\mathcal{H}$ as terms in the language of hereditary Harrop formulas. Specifically, we use the ternary function symbol `period` to represent atomic formulas and the constants `not`, `and`, `or`, `must` and `may`, with the obvious arities, as the concrete syntax of the logical symbols of GMEC: $\neg$, $\wedge$, $\vee$, $\Box$ and $\Diamond$, respectively. The representation function $\ulcorner \cdot \urcorner^L$ for GMEC-formulas is specified by the following recursive definition, based on the structure of the formula in $\mathcal{L}_\mathcal{H}$ being represented:

- $\ulcorner p(e_1, e_2) \urcorner^L = \texttt{period } \ulcorner e_1 \urcorner^E \ \ulcorner p \urcorner^P \ \ulcorner e_2 \urcorner^E$
- $\ulcorner \neg\varphi \urcorner^L = \texttt{not } \ulcorner \varphi \urcorner^L$
- $\ulcorner \varphi_1 \wedge \varphi_2 \urcorner^L = \texttt{and } \ulcorner \varphi_1 \urcorner^L \ \ulcorner \varphi_2 \urcorner^L$
- $\ulcorner \varphi_1 \vee \varphi_2 \urcorner^L = \texttt{or } \ulcorner \varphi_1 \urcorner^L \ \ulcorner \varphi_2 \urcorner^L$
- $\ulcorner \Box\varphi \urcorner^L = \texttt{must } \ulcorner \varphi \urcorner^L$
- $\ulcorner \Diamond\varphi \urcorner^L = \texttt{may } \ulcorner \varphi \urcorner^L$

Notice that we have overloaded the symbol `not`. However, its position dictates its use: within a term, it represents the negation of $\mathcal{L}_\mathcal{H}$, and at the predicate level it stands as the negation-as-failure operator. In order to simplify the notation, we will write the previously defined translation maps as $\ulcorner \cdot \urcorner$, whenever the omitted subscript is easily deducible from the context.

Figure 3.2 shows an implementation of GMEC in the language of HH-formulas. We call this program the *naive* implementation of GMEC, and refer to it as `GMEC`. Clause (1) models object level equality. Clauses (2) and (3) define the predicate `before` that reconstructs the transitive closure of the ordering information currently stored in the program. The remaining clauses show the actual implementation of GMEC. We use the unary predicate `holds` to represent the validity of a GMEC-formula with respect to the GMEC-structure and the knowledge state represented in the program. Said in a different way, we aim at representing the judgment $\mathcal{I}_\mathcal{H}; o^+ \models \varphi$ by means of the relation $\texttt{GMEC}, \ulcorner \mathcal{H} \urcorner, \ulcorner o \urcorner \Longrightarrow \texttt{holds } \ulcorner \varphi \urcorner$.

Clauses (4) and (5) implement the definition of modal valuation of the standard GMEC-model given in Definition 4; the latter corresponds to the negation of the meta-predicate *nb* (recall that `;` is the concrete syntax for disjunction in the language of HH-formulas). These clauses coincide with the standard Prolog axiomatization of EC [20]. Clauses (6-8) map the object-level propositional connectives to the corresponding meta-level operators.

Clauses (9-10) define `holds` for $\Box$-moded GMEC-formulas. They implement directly the statement of the remark following Lemma 2.5. In order to check that the formula $\Box\varphi$ holds in the current state of knowledge, first we check $\varphi$ locally and then we ascertain that there is no future knowledge state where $\Box\varphi$ does not hold. Clause (10) attempts to find a counterexample to this requirement, i.e. a proper extension of the current world (i.e. a state of knowledge that orders two currently unrelated events $e_1$ and $e_2$) where $\Box\varphi$ fails to hold. No such knowledge state must exist for the body of clause (9) to hold. Notice the essential use of implication in the goal position in these cases.

```
% -------- Equality
X = X.                                   (1)

% -------- Transitive closure of knowledge states
before E1 E2 :-                          (2)
     beforeFact E1 E2.

before E1 E2 :-                          (3)
     beforeFact E1 E, before E E2.

% -------- Propositional formulas
holds (period Ei P Et) :-                (4)
     happens Ei, initiates Ei P,
     happens Et, terminates Et P,
     before Ei Et,
     not (broken Ei P Et).

broken Ei P Et :-                        (5)
     happens E,
     before Ei E, before E Et,
     (initiates E Q ; terminates E Q),
     (exclusive P Q; P = Q).

holds (not X) :-                         (6)
     not (holds X).

holds (and X Y) :-                       (7)
     holds X, holds Y.

holds (or X Y) :-                        (8)
     holds X; holds Y.

% -------- Modal formulas
holds (must X) :-                        (9)
     holds X,
     not (fails_must X).

fails_must X :-                          (10)
     happens E1, happens E2,
     not (E1 = E2),
     not (before E1 E2),
     not (before E2 E1),
     beforeFact E1 E2 =>
          not (holds (must X)).

holds (may X) :-                         (11)
     holds X.

holds (may X) :-                         (12)
     happens E1, happens E2,
     not (E1 = E2),
     not (before E1 E2),
     not (before E2 E1),
     beforeFact E1 E2 =>
          holds (may X).
```

**FIGURE 3.2.** `GMEC`, a Naive Implementation of GMEC.

The remaining clauses deal with GMEC-formulas having $\Diamond$ as their main connective in a similar manner. Note that the implementation of `holds` for GMEC-formulas involving modalities requires the exhaustive exploration of all extensions of the current knowledge state. This approach is clearly expensive, and for this reason we qualify `GMEC` as the *naive* implementation of GMEC. An enhanced program for GMEC that takes the specific properties of GMEC into account is analyzed in detail in section 3.4.

In section 3.1, we described hereditary Harrop formulas as an extension to Horn clauses permitting the use of implication and universal quantification in goal positions. Since the latter connective is available, it is tempting to replace clauses (9-10) by

```
holds (must X) :-                                                    (*)
     holds X,
     forall[E1, E2]
         (happens E1, happens E2,
          not (before E1 E2),
          not (before E2 E1),
          beforeFact E1 E2 => holds (must X)).
```

implementing in this way the statement of Lemma 2.5 directly, instead of taking the complicated detours dictated by the subsequent remark. Unfortunately, this clause is not a faithful transcription of the lemma. The bug originates from the confusion between the two forms of universal quantification discussed at the end of Section 3.1.

Recall that universal quantification in the language of hereditary Harrop formulas is interpreted intensionally. Therefore, solving the body of clause $(*)$ requires generating two new events, say $e_1^*$ and $e_2^*$, and using them to solve the embedded goal. However, $e_1^*, e_2^* \notin E$, therefore the subgoals `happens` $\ulcorner e_1^* \urcorner$ and `happens` $\ulcorner e_2^* \urcorner$ will never succeed.

This is obviously not the behavior that we have in mind. We would rather want the variables `E1` and `E2` to be instantiated to *all* events in $E$ in turn, i.e. have the quantifier interpreted extensionally. We showed in Section 3.1 how this effect can be achieved by taking advantage of negation-as-failure and embedded implication: recall that an extensional goal $\forall x \in S.\, G(x)$ can be expressed in the language of hereditary Harrop formulas as the goal $(\forall x.\, ((x \in S \wedge \neg G(x)) \rightarrow p')) \rightarrow \neg p'$ for some new atomic formula $p'$. Indeed the quantifier has been moved to a program position and is therefore solved extensionally by unification. These are precisely the steps that led to the displayed formulation of clauses (9-10), where `fails_must` is used as the accessory atomic formula.

### *3.3. Soundness and Completeness Results*

In this section, we show that `GMEC` is a faithful implementation of the semantics given in Section 2.3 for GMEC. This statement is formalized in the soundness and completeness theorem (Theorem 3.9) that concludes the section. This result is accomplished in a number of steps: we present here only the most important ones; their proofs, together with auxiliary lemmas, can be found in Appendix A. First we need to prove that `before` is a sound and complete implementation of the transitive closure over knowledge states, then we show that the implementation of atomic formulas is sound and complete, and finally we will be able to freely mix boolean connectives and modal operators.

We begin with a lemma about the properties of `before`. When only ordering information is concerned, we do not need to refer to the representation of the underlying GMEC-structure, but only implicitly to the representation of events. First, we show that the HH-formula `before` $\ulcorner e_1 \urcorner$ $\ulcorner e_2 \urcorner$ is provable precisely when $(e_1, e_2)$ is in the transitive closure of the current knowledge state. Moreover, the goal `before` $\ulcorner e_1 \urcorner$ $\ulcorner e_2 \urcorner$ finitely fails exactly when $(e_1, e_2)$ is not in the transitive closure of the current knowledge state.

The second part of this lemma will be of extreme importance when dealing with negative sequents since `before` is the only predicate, besides `holds`, that has a

recursive definition, and therefore that could diverge.

*Lemma 3.1.* (*Soundness and completeness of* `before` *w.r.t. transitive closure*)
*Let* $\mathcal{H} = (E,\ P,\ [\cdot\rangle,\ \langle\cdot],\ ]\cdot,\cdot[)$ *be a GMEC-structure and* $o$ *a state of knowledge, then for any* $e_1, e_1 \in E$

*a.* `GMEC,`$\ulcorner o \urcorner \Longrightarrow$ `before` $\ulcorner e_1 \urcorner\ \ulcorner e_2 \urcorner$ *iff* $(e_1, e_2) \in o^+$;
*b.* `GMEC,`$\ulcorner o \urcorner \Longrightarrow$ `not (before` $\ulcorner e_1 \urcorner\ \ulcorner e_2 \urcorner$`)` *iff* $(e_1, e_2) \notin o^+$.

On the basis of this result, we address the problem of proving that the clauses for atomic GMEC-formulas implement the semantics of MVIs. We start by proving a lemma that states that the predicate `broken` behaves like the negation of the meta-predicates *nb*.

*Lemma 3.2.* (*Correspondence between* `broken` *and nb*)
*Let* $\mathcal{H} = (E,\ P,\ [\cdot\rangle,\ \langle\cdot],\ ]\cdot,\cdot[)$ *be a GMEC-structure and* $o$ *a state of knowledge, then*

*a.* `GMEC,`$\ulcorner \mathcal{H} \urcorner, \ulcorner o \urcorner \Longrightarrow$ `broken` $\ulcorner e_1 \urcorner\ \ulcorner p \urcorner\ \ulcorner e_2 \urcorner$ *iff* $\neg nb(p, e_1, e_2, o^+)$ *holds in* $\mathcal{H}$;
*b.* `GMEC,`$\ulcorner \mathcal{H} \urcorner, \ulcorner o \urcorner \Longrightarrow$ `not (broken` $\ulcorner e_1 \urcorner\ \ulcorner p \urcorner\ \ulcorner e_2 \urcorner$`)` *iff* $nb(p, e_1, e_2, o^+)$ *holds in* $\mathcal{H}$.

At this point, we have all the tools we need to prove that the implementation of `holds` on bare atomic formulas behaves isomorphically to the satisfiability relation on these formulas. Therefore, GMEC provides an effective implementation of MVIs.

*Theorem 3.6.* (`GMEC` *computes MVIs*)
*Let* $\mathcal{H} = (E,\ P,\ [\cdot\rangle,\ \langle\cdot],\ ]\cdot,\cdot[)$ *be a GMEC-structure and* $o$ *a state of knowledge, then*

*a.* `GMEC,`$\ulcorner \mathcal{H} \urcorner, \ulcorner o \urcorner \Longrightarrow$ `holds (period` $\ulcorner e_1 \urcorner\ \ulcorner p \urcorner\ \ulcorner e_2 \urcorner$`)` *iff* $p(e_1, e_2) \in MVI(o^+)$;
*b.* `GMEC,`$\ulcorner \mathcal{H} \urcorner, \ulcorner o \urcorner \Longrightarrow$ `not (holds (period` $\ulcorner e_1 \urcorner\ \ulcorner p \urcorner\ \ulcorner e_2 \urcorner$`))` *iff* $p(e_1, e_2) \notin MVI(o^+)$.

We conclude this section by stating its main result, namely, soundness and completeness of `GMEC` with respect to the GMEC-frame semantics.

*Theorem 3.9.* (*Soundness and completeness of* `GMEC` *w.r.t. GMEC-frames*)
*Let* $\mathcal{H} = (E,\ P,\ [\cdot\rangle,\ \langle\cdot],\ ]\cdot,\cdot[)$ *be a GMEC-structure,* $o$ *a state of knowledge and* $\varphi$ *and GMEC-formula, then*

*a.* `GMEC,`$\ulcorner \mathcal{H} \urcorner, \ulcorner o \urcorner \Longrightarrow$ `holds` $\ulcorner \varphi \urcorner$ *iff* $o^+ \models \varphi$;
*b.* `GMEC,`$\ulcorner \mathcal{H} \urcorner, \ulcorner o \urcorner \Longrightarrow$ `not (holds` $\ulcorner \varphi \urcorner$`)` *iff* $o^+ \not\models \varphi$.

## *3.4. A Semi-Naive Implementation of GMEC*

Theorem 3.9 establishes a strong connection between the GMEC semantics and the hereditary Harrop program `GMEC` displayed in Figure 3.2, providing in this way a computational flavor to the Generalized Modal Event Calculus presented in Section 2. Although this is a valuable theoretical property, it loses most of its practical appeal as soon as we give a close look at the treatment of the modal operators in `GMEC`.

Indeed, checking the validity of a goal having $\Box$ as its main connective (clauses (9) and (10)) triggers the exploration of all the states of knowledge reachable from the current ordering information (unless failure occurs). The situation is not better in the case of $\Diamond$-moded formulas (clauses (11-12)): "only" an arbitrarily large subset of the extension of the current state of knowledge must be examined. It is easy to figure out that the cardinality of the set of extensions of a given state of knowledge is in general exponential in the number of events. This contrasts with the polynomial complexity of EC [8] and of its simply moded extensions CREC and SKEC.

In this section, we solve these problems, up to a certain extent, by providing an alternative implementation for the GMEC semantics. We will not be able to completely avoid the exhaustive exploration of the set of possible future knowledge states. However, the resulting decision procedure will operate solely on the local state in a number of cases that are likely to occur in real applications (this is the case, for instance, of the beverage dispenser example from Section 2).

The key idea behind our enhanced implementation of GMEC is to take into account the meta-properties of our framework for the modal event calculus. First, we exploit the intrinsic properties of GMEC. In particular, Lemmas 2.6 and 2.7 suggest a local method for checking the validity of atomic formulas preceded by a single occurrence of a modal operator. Remember that the definition of the functions $\Box MVI(\cdot)$ and $\Diamond MVI(\cdot)$ relies on formulas of this form. Being able to compute the value of these functions locally is clearly of crucial importance for practical applications. Second, we can take advantage of the equivalences that hold in GMEC (Corollaries 2.1 and 2.2). Although they occasionally permit eliminating occurrences of a modal operator, we will mainly use these equivalences as rewriting rules to push the modalities as close to the atomic formulas as possible, with the goal of using Lemmas 2.6 and 2.7 whenever possible. Alternatively, we could have used the equivalences of Corollaries 2.1 and 2.2 to precompile a GMEC-formula into a form on which these lemmas can be applied directly.

This technique cannot be applied systematically. In particular, we know from Section 2.3 that formulas of the form $\Box(\varphi' \vee \varphi'')$, and dually $\Diamond(\varphi' \wedge \varphi'')$, cannot be reduced. Moreover, the formulas $\Box\Diamond\varphi$ and $\Diamond\Box\varphi$ are reducible only for particular $\varphi$s. In these cases, and only in these cases, the actual exploration of the extensions of the current knowledge state cannot be avoided.

On the basis of these considerations, we will now describe a second (semi-naive) implementation of GMEC in the language of hereditary Harrop formulas. The enhanced program, that we call `GMEC+`, shares with `GMEC` the encoding presented in Section 3.2 for the various entities at hand. Moreover, for the sake of simplicity, we use the same names as in Figure 3.2 for predicates performing the same functionalities. This program is presented in Figures 3.3–3.4. Clauses (1'-8') in Figure 3.3 do not undergo any change.

The upper part of Figure 3.4 illustrates the definition of `holds` for $\Box$-moded GMEC-formulas $\Box\varphi$. In order to apply the previous observations, we need to look at the main connective of $\varphi$. Clauses (9′) and (10′) deal with the case where $\varphi$ is atomic by implementing the statement of Lemma 2.6, with (10′) corresponding to the negation of the meta-predicate *nsb*. Clauses (11′-12′, 15′-16′) implement some of the reductions described by Corollaries 2.1 and 2.2. The other clauses deal with the remaining patterns for $\varphi$ by means of the brute-force approach derived from the remark following Lemma 2.5. They are instances of clauses (9-10) of `GMEC`. Notice

```
% -------- Equality
X = X.                                  (1′)

% -------- Transitive closure of knowledge states
before E1 E2 :-                         (2′)
     beforefact E1 E2.

before E1 E2 :-                         (3′)
     beforefact E1 E, before E E2.

% -------- Propositional formulas
holds (period Ei P Et) :-               (4′)
     happens Ei, initiates Ei P,
     happens Et, terminates Et P,
     before Ei Et,
     not (broken Ei P Et).

broken Ei P Et :-                       (5′)
     happens E,
     before Ei E, before E Et,
     (initiates E Q; terminates E Q),
     (exclusive P Q; P = Q).

holds (not X) :-                        (6′)
     not (holds X).

holds (and X Y) :-                      (7′)
     holds X, holds Y.

holds (or X Y) :-                       (8′)
     holds X; holds Y.
```

**FIGURE 3.3.** `GMEC+`, a Semi-Naive Implementation of GMEC (*part I*).

that clause (17′) subsumes clause (16′). Therefore, the latter ought to be given precedence over the former.

The lower part of Figure 3.4 shows the treatment of GMEC-formulas having $\Diamond$ as their main connective. The underlying idea is similar to the previous case. Notice that clause (26′) subsumes clause (25′).

We have extensively investigated in [2, 10] two axiomatic variants of the Event Calculus based on clauses (9′-10′) and (19′, 5′) respectively. These calculi, called respectively the *Skeptical Event Calculus* (*SKEC*) and the *Credulous Event Calculus* (*CREC*), now emerge as a by-product of the broader notion of Generalized Modal Event Calculus.

We want now to prove that `GMEC+` is a faithful implementation of the GMEC semantics presented in Section 2.3. In order to achieve this goal, we need to process `GMEC+` through the same steps applied to `GMEC` in Section 3.3. Fortunately we can borrow from that section Lemmas 3.4, 3.5, 3.1, 3.2 and Theorem 3.6 since on the one hand the clauses of `GMEC` involved in these statements are present also in `GMEC+`, and on the other hand, they are not subject to interferences from the new clauses. This claim, which validity can be easily checked, will save us a lot of work.

Our first endeavor will be to prove that the predicate `skeBroken` behaves like the negation of the meta-predicate *nsb*. The statement and the proof of this result recall Lemma 3.2, according to which `broken` is a sound and complete implementation of *nb*.

*Lemma 3.3.* (*Correspondence between* `skeBroken` *and nsb*)

*Let* $\mathcal{H} = (E,\ P,\ [\cdot\rangle,\ \langle\cdot],\ ]\cdot,\cdot[)$ *be a GMEC-structure and* $o$ *a state of knowledge, then*

```
% -------- Must-formulas
holds (must (period Ei P Et)) :-                    (9′)
     happens Ei, initiates Ei P,
     happens Et, terminates Et P,
     before Ei Et,
     not (skeBroken Ei P Et).

skeBroken Ei P Et :-                                (10′)
     happens E,
     not (E = Ei), not (E = Et),
     not (before E Ei),
     not (before Et E),
     (initiates E Q; terminates E Q),
     (exclusive P Q; P = Q).

holds (must (not X)) :-                             (11′)
     holds (not (may X)).

holds (must (and X Y)) :-                           (12′)
     holds (and (must X) (must Y)).

holds (must (or X Y)) :-                            (13′)
     holds (or X Y),
     not (fails_must_or X Y).

fails_must_or X Y :-                                (14′)
     happens E1, happens E2,
     not (E1 = E2),
     not (before E1 E2),
     not (before E2 E1),
     beforeFact E1 E2 =>
          not (hold (must (or X Y))).

holds (must (must X)) :-                            (15′)
     holds (must X).

holds (must (may (must X))) :-                      (16′)
     holds (must (may X)).

holds (must (may X)) :-                             (17′)
     holds (may X),
     not (fails_must_may X).

fails_must_may X :-                                 (18′)
     happens E1, happens E2,
     not (E1 = E2),
     not (before E1 E2),
     not (before E2 E1),
     beforeFact E1 E2 =>
          not (hold (must (may X))).

%-------- May-formulas
holds (may (period Ei P Et)) :-                     (19′)
     happens Ei, initiates Ei P,
     happens Et, terminates Et P,
     not (before Et Ei),
     not (broken Ei P Et).

holds (may (not X)) :-                              (20′)
     holds (not (must X)).

holds (may (and X Y)) :-                            (21′)
     holds (and X Y).

holds (may (and X Y)) :-                            (22′)
     happens E1, happens E2,
     not (E1 = E2),
     not (before E1 E2),
     not (before E2 E1),
     beforeFact E1 E2 =>
          holds (may (and X Y)).

holds (may (or X Y)) :-                             (23′)
     holds (or (may X) (may Y)).

holds (may (may X)) :-                              (24′)
     holds (may X).

holds (may (must (may X))) :-                       (25′)
     holds (may (must X)).

holds (may (must X)) :-                             (26′)
     holds (must X).

holds (may (must X)) :-                             (27′)
     happens E1, happens E2,
     not (E1 = E2),
     not (before E1 E2),
     not (before E2 E1),
     beforeFact E1 E2 =>
          holds (may (must X)).
```

**FIGURE 3.4.** `GMEC+`, a Semi-Naive Implementation of GMEC (*part II*).

*a.* `GMEC+`, $\ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow$ `skeBroken` $\ulcorner e_1\urcorner\ \ulcorner p\urcorner\ \ulcorner e_2\urcorner$ *iff* $\neg nsb(p, e_1, e_2, o^+)$ *holds in* $\mathcal{H}$;

*b.* `GMEC+`, $\ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow$ `not (skeBroken` $\ulcorner e_1\urcorner\ \ulcorner p\urcorner\ \ulcorner e_2\urcorner$`)` *iff* $nsb(p, e_1, e_2, o^+)$ *holds in* $\mathcal{H}$.

We will now prove that `holds` applied to the encoding of atomic formulas pre-

ceded by one occurrence of a modal operator behaves isomorphically to the satisfiability relation for these formulas. Therefore, `GMEC+` provides an effective implementation of MVIs (by Theorem 3.6), necessary MVIs and possible MVIs.

We first consider $\Box$-moded atomic formulas and make explicit their relation to necessary MVIs. The proof of this statement relies on the previous lemma.

*Theorem 3.10.* (`GMEC+` *computes necessary MVIs*)

*Let* $\mathcal{H} = (E,\ P,\ [\cdot\rangle,\ \langle\cdot],\ ]\cdot,\cdot[)$ *be a GMEC-structure and* $o$ *a state of knowledge, then*

*a.* $\texttt{GMEC+}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{holds (must (period } \ulcorner e_1\urcorner\ \ulcorner p\urcorner\ \ulcorner e_2\urcorner\texttt{))}$ *iff* $p(e_1, e_2) \in \Box MVI(o^+)$;

*b.* $\texttt{GMEC+}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{not (holds (must (period } \ulcorner e_1\urcorner\ \ulcorner p\urcorner\ \ulcorner e_2\urcorner\texttt{)))}$ *iff* $p(e_1, e_2) \notin \Box MVI(o^+)$.

A similar result holds for possible MVIs, formalized as the function $\Diamond MVI(\cdot)$. Indeed, `holds` constitutes a decision procedure for the validity relation for $\Diamond$-moded atomic formulas.

*Theorem 3.11.* (`GMEC+` *computes possible MVIs*)

*Let* $\mathcal{H} = (E,\ P,\ [\cdot\rangle,\ \langle\cdot],\ ]\cdot,\cdot[)$ *be a GMEC-structure and* $o$ *a state of knowledge, then*

*a.* $\texttt{GMEC+}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{holds (may (period } \ulcorner e_1\urcorner\ \ulcorner p\urcorner\ \ulcorner e_2\urcorner\texttt{))}$ *iff* $p(e_1, e_2) \in \Diamond MVI(o^+)$;

*b.* $\texttt{GMEC+}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{not (holds (may (period } \ulcorner e_1\urcorner\ \ulcorner p\urcorner\ \ulcorner e_2\urcorner\texttt{)))}$ *iff* $p(e_1, e_2) \notin \Diamond MVI(o^+)$.

Finally, we can prove that a formula is valid in the GMEC semantics if and only if the goal obtained by encoding it and using it as the argument of `holds` is derivable in `GMEC+`. Moreover, a goal of this form has only finite derivations since each step in the computation either simplifies the encoded formula itself within the current ordering, or leads to a more complete state of knowledge but keeps the goal unchanged. Therefore `holds` captures also the unsatisfiability of a GMEC-formula.

*Theorem 3.12.* (*Soundness and completeness of* `GMEC+` *w.r.t. GMEC-frames*)

*Let* $\mathcal{H} = (E,\ P,\ [\cdot\rangle,\ \langle\cdot],\ ]\cdot,\cdot[)$ *be a GMEC-structure,* $o$ *a state of knowledge and* $\varphi$ *and GMEC-formula, then*

*a.* $\texttt{GMEC+}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{holds } \ulcorner\varphi\urcorner$ *iff* $o^+ \models \varphi$;

*b.* $\texttt{GMEC+}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{not (holds } \ulcorner\varphi\urcorner\texttt{)}$ *iff* $o^+ \not\models \varphi$.

## 4. CONCLUSIONS AND FURTHER DEVELOPMENTS

This paper proposed and formally analyzed GMEC, a modal extension of EC to compute current, necessary and possible MVIs in a context where the ordering of events is relative, partial and incremental. Unlike previous modal extensions of EC (e.g., MEC [2]), GMEC supports a free mix of boolean connectives and modal operators. The paper presented two sound and complete implementations of GMEC as logic programs in the language of hereditary Harrop formulas. Maybe more

important than the results themselves is the method we adopted to achieve them. First, we provided a precise semantic formalization in order to capture the intuitions underlying EC and its modal extensions. In this way, we could prove properties of EC (and subsequentially of GMEC) rather than claim them. Second, we used a proof-theoretic approach for proving the faithfulness of our implementations with respect to the behavior of GMEC, as expressed by the semantics.

We are developing this work in several directions. First, we are investigating intermediate modal event calculi featuring the polynomial complexity of MEC without sacrificing too much of the expressive power of GMEC. Preliminary results can be found in [4], where we developed a new modal event calculus which retains enough of the expressive power of GMEC while admitting an efficient polynomial implementation in the style of MEC. The practical usefulness of such a calculus is showed by applying it to a case study taken from the domain of fault diagnosis. We are also exploring the possibility of dealing with preconditions, boolean connectives and modal operators in a uniform framework. As proved in [11], an indiscriminated use of preconditions immediately makes the problem of MVIs computation NP-hard. Nevertheless, we believe that a formal study of various modal event calculi with preconditions can shed some light on the dynamics of preconditions, and possibly lead to polynomial approximations of the computation of MVIs. Preliminary results in this direction can be found in [3]. Finally, we are considering more complex specifications of the ordering information such as non-committed data (e.g. disjunctive orderings) and possibly inconsistent orderings.

## *Acknowledgments*

The first author was partially supported by NFS grant CCR-9303383. The work of the second author was partially supported by the CNR project *Ambienti e strumenti per la gestione di informazioni temporali.* We would like to thank Luca Chittaro for many useful observations and inspiring comments, and James Harland for valuable discussions concerning the syntactic formulation of negation-as-failure presented in Section 3.1. Thanks also to Kristof Van Belleghem for the interesting remarks about the expressive power of MEC (the GMEC fragment including atomic formulas and simply moded atomic formulas only). The comments of Massimo Franceschet on early drafts of this paper proved particularly useful. Finally, the comments of the anonymous referees permitted improving considerably this paper.

## REFERENCES

1. A.J. Bonner. *Hypothetical Reasoning in Deductive Databases.* PhD thesis, Rutgers University, New Brunswick, NJ, 1991.
2. I. Cervesato, L. Chittaro, A. Montanari. A Modal Calculus of Partially Ordered Events in a Logic Programming Framework. *Proc. of ICLP'95: 12th International Conference on Logic Programming*, L. Sterling (ed.), pages 299–313, MIT Press, 1995.
3. I. Cervesato, M. Franceschet, A. Montanari. Modal Event Calculi with Preconditions. *Proc. of TIME'97: 4th International Workshop on Temporal Representation and Reasoning*, R. Morris, L. Khatib (eds.), pages 38–45, IEEE Computer Society Press, 1997.

4. I. Cervesato, M. Franceschet, A. Montanari. A Hierarchy of Modal Event Calculi: Expressiveness and Complexity. *Proc. of ICTL'97: 2nd International Conference on Temporal Logic*, H. Barringer, M. Fisher, D. Gabbay, G. Gough (eds.), pages 1–17, 1997.
5. I. Cervesato, A. Montanari, A. Provetti. On the Non-Monotonic Behavior of Event Calculus for Deriving Maximal Time Intervals. *Interval Computations*, 3(2):83–119, 1993.
6. L. Chittaro, M. Del Rosso, M. Dojat. Modeling medical reasoning with the Event Calculus: an application to the management of mechanical ventilation. In *Proc. of AIME'95: 5th European Conference on Artificial Intelligence in Medicine*, P. Barahona, J. Wyatt, M. Stefanelli (eds.), pages 79–90, LNAI 934, Springer Verlag, 1995.
7. L. Chittaro, A. Montanari. Efficient Temporal Reasoning in the Cached Event Calculus. *Computational Intelligence*, 12(3):359–382, 1996.
8. L. Chittaro, A. Montanari, I. Cervesato. Speeding up temporal reasoning by exploiting the notion of kernel of an ordering relation. *Proc. of TIME'95: 2nd International Workshop on Temporal Representation and Reasoning*, S. Goodwin, H. Hamilton (eds.), pages 73–80, University of Regina, Canada, 1995.
9. L. Chittaro, A. Montanari, E. Peressi. An integrated framework for temporal aggregation and omission in the Event Calculus. *Proc. of Applications of Artificial Intelligence in Engineering X*, R.A. Adey, G. Rzevski, C. Tasso (eds.), pages 47–54, Computational Mechanics Publications, Boston, 1995.
10. L. Chittaro, A. Montanari, A. Provetti. Skeptical and Credulous Event Calculi for Supporting Modal Queries. *Proc. of ECAI'94: 11th European Conference on Artificial Intelligence*, A. Cohn (ed.), pages 361–365, John Wiley & Sons, 1994.
11. T. Dean, M. Boddy. Reasoning about partially ordered events. *Artificial Intelligence*, 36:375–399, 1988.
12. M. Denecker, L. Missiaen, M. Bruynooghe. Temporal reasoning with abductive Event Calculus. *Proc. of ECAI'92: 10th European Conference on Artificial Intelligence*, B. Neumann (ed.), pages 384–388, John Wiley & Sons, 1992.
13. D.M. Gabbay, U. Reyle. N-Prolog: an Extension of Prolog with Hypothetical Implication - I. *Journal of Logic Programming*, 1(4):319–355, 1984.
14. J. Harland. *On Hereditary Harrop Formulae as a basis for Logic Programming.* PhD Thesis, University of Edinburgh, UK, 1991.
15. J. Harland. *Toward a static proof system for negation as failure.* Technical Report CITRI/TR-92-49, Department of Computer Science, University of Melbourne, Australia, 1992.
16. J. Harland. Success and Failure for Hereditary Harrop Formulae. *Journal of Logic Programming*, 17:1–29, 1993.
17. J. Hodas and D. Miller. Logic programming in a fragment of intuitionistic linear logic. *Journal of Information and Computation*, 110(2):327–365, 1994.
18. G. Hughes, M. Cresswell. *An Introduction to Modal Logic*, Methuen, London, 1968.
19. C.S. Jensen et al. A Consensus Glossary of Temporal Database Concepts. *SIGMOD RECORD*, 23(1):52–64, 1994.
20. R. Kowalski, M. Sergot. A Logic-based Calculus of Events. *New Generation Computing*, 4(1):67–95, 1986.
21. J.W. Lloyd. *Foundations of Logic Programming (2nd ed.)*, Springer-Verlag, 1987.

22. J. McCarthy, P. Hayes. Some philosophical problems from the standpoint of artificial intelligence. *Machine Intelligence 4*, B. Meltzer, D. Michie (eds.), pages 463-502, Edinburgh University Press, 1969.

23. D. Miller, G. Nadathur. An overview of λProlog. In *Proc. of ICSLP'88: 5th International Conference and Symposium on Logic Programming*, K.A. Bowen, R. Kowalski (eds.), pages 810–827, MIT Press, 1988.

24. D. Miller, G. Nadathur, F. Pfenning, A. Scedrov. Uniform proofs as a foundation for logic programming. *Annals of Pure and Applied Logic*, 51:125–157, 1991.

25. D.C. Moffat, G.D. Ritchie. Modal Queries about Partially-ordered Plans. *Journal of Expt. Theor. Artificial Intelligence*, 2:341–368, 1990.

26. A. Montanari, L. Chittaro, I. Cervesato. A general Modal Framework for the Event Calculus and its Skeptical and Credulous Variants. *Proc. of ECAI'96: 12th European Conference on Artificial Intelligence*, W. Wahlster (ed.), pages 33–37, John Wiley & Sons, 1996.

27. F. Pfenning. Elf: A language for logic definition and verified meta-programming. In *Proc. of LICS'89: 7th Symposium on Logic in Computer Science*, pages 702–712, IEEE Computer Society Press, 1989.

28. K. Segerberg. *An Essay in Classical Modal Logic.* Uppsala Filosofiska Studier, 1971.

29. M. Sergot. *An introduction to the Event Calculus.* Lecture Notes of the GULP Advanced School on Foundations of Logic Programming, Alghero, Sardinia, 1990.

30. M.P. Shanahan. Prediction is Deduction but Explanation is Abduction. *Proc. of IJCAI'89: 11th International Joint Conference on Artificial Intelligence*, pages 1055–1060, Morgan Kauffman, 1989.

## A. PROOFS AND AUXILIARY LEMMAS FROM SECTION 3

**Lemma 3.4** (*Soundness and completeness of* `=` *w.r.t. equality for events*)

*Let* $\mathcal{H} = (E,\ P,\ [\cdot\rangle,\ \langle\cdot],\ ]\cdot,\cdot[)$ *be a GMEC-structure,* $e_1, e_2 \in E$ *and* $o$ *a state of knowledge. Then*

*a.* $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \ulcorner e_1\urcorner \texttt{ = } \ulcorner e_2\urcorner$ *is derivable* *iff* $e_1 = e_2$;

*b.* $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{not (}\ulcorner e_1\urcorner \texttt{ = } \ulcorner e_2\urcorner\texttt{)}$ *is derivable* *iff* $e_1 \neq e_2$.

Proof.
($a.\ \Rightarrow$) Being the goal $\ulcorner e_1\urcorner \texttt{ = } \ulcorner e_2\urcorner$ atomic, the last rule applied must have been **atom+** with clause (1) and substitution $\sigma = \{X \mapsto \ulcorner e_1\urcorner, X \mapsto \ulcorner e_2\urcorner\}$. This substitution is well-formed *iff* $\ulcorner e_1\urcorner = \ulcorner e_2\urcorner$. Therefore, we have that $e_1 = e_2$ by the injectivity of the representation function $\ulcorner\cdot\urcorner^E$.

($a.\ \Leftarrow$) If $e_1 = e_2$, a derivation of $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \ulcorner e_1\urcorner \texttt{ = } \ulcorner e_2\urcorner$ is obtained by application of rules **atom+** and **true+**.

($b.\ \Rightarrow$) By the uniform provability property, the last inference rule applied is **naf+**. Therefore, the sequent $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \not\Longrightarrow \ulcorner e_1\urcorner \texttt{ = } \ulcorner e_2\urcorner$ is provable. By property 3.1, the sequent $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \ulcorner e_1\urcorner \texttt{ = } \ulcorner e_2\urcorner$ has no derivation. Finally, by ($a$), $e_1 \neq e_2$.

(b. $\Leftarrow$) If $e_1 \neq e_2$, we have that $\ulcorner e_1\urcorner \neq \ulcorner e_2\urcorner$ since $\ulcorner\cdot\urcorner^E$ is injective. Therefore, rule **atom–** succeeds with no premisses for the sequent $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \not\Longrightarrow \ulcorner e_1\urcorner \texttt{ = } \ulcorner e_2\urcorner$. Therefore, by rule **naf+**, $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{not (}\ulcorner e_1\urcorner \texttt{ = } \ulcorner e_2\urcorner\texttt{)}$ is derivable. □

**Lemma 3.5** (*Soundness and completeness of* `=` *w.r.t. equality for properties*)

*Let* $\mathcal{H} = (E,\ P,\ [\cdot\rangle,\ \langle\cdot],\ ]\cdot,\cdot[)$ *be a GMEC-structure,* $p_1, p_2 \in P$ *and* $o$ *a state of knowledge. Then*

*a.* $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \ulcorner p_1\urcorner$ `=` $\ulcorner p_2\urcorner$ *is derivable* *iff* $p_1 = p_2$;

*b.* $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow$ `not (`$\ulcorner p_1\urcorner$ `=` $\ulcorner p_2\urcorner$`)` *is derivable* *iff* $p_1 \neq p_2$.

Proof. Similar to the proof of previous lemma. □

**Lemma 3.1** (*Soundness and completeness of* `before` *w.r.t. transitive closure*)

*Let* $\mathcal{H} = (E,\ P,\ [\cdot\rangle,\ \langle\cdot],\ ]\cdot,\cdot[)$ *be a GMEC-structure and* $o$ *a state of knowledge, then for any* $e_1, e_2 \in E$

*a.* $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow$ `before` $\ulcorner e_1\urcorner\ \ulcorner e_2\urcorner$ *is derivable* *iff* $(e_1, e_2) \in o^+$;

*b.* $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow$ `not (before` $\ulcorner e_1\urcorner\ \ulcorner e_2\urcorner$`)` *is derivable* *iff* $(e_1, e_2) \notin o^+$.

Proof. We will provide a rigorous proof of this simple statement. The proofs given in the rest of this appendix will be more sketchy. However, it should be clear to the reader how to rewrite them in a similar style. Indeed, in order to limit the length of these proofs, we will mainly focus on the critical steps, that correspond to the applications of rules **atom+** and **atom–**. The application of the remaining rules will often be maintained implicit in the informal arguments used to chain critical rules.

Throughout this and many of the subsequent inductive proofs, we will rely on the following strict schema. The cases of the induction are treated in dedicated paragraphs headed with an identifying label. Within each paragraph, the proof is organized in a series of lines consisting of three zones. On the left, we have a counter used for referencing. The central field contains a formal relation that is claimed to hold. The right part of each line provides a justification of this claim. Each step is in general justified with respect to the previous line (to the statement of the theorem in the case of the first line). Occasionally, the justification will refer to one or more non-immediate predecessors of the current line. In these cases, we take advantage of the counter. In certain occasions, we will have to follow alternative courses in the proof, and each should be proved in order for the overall proof to be correct. We use bullets (•) to identify the first line of each alternative, and indent the subsequent lines.

($a. \Rightarrow$) We proceed by induction on the structure of a derivation tree for the positive sequent $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow$ `before` $\ulcorner e_1\urcorner\ \ulcorner e_2\urcorner$. Since `before` $\ulcorner e_1\urcorner\ \ulcorner e_2\urcorner$ is atomic, the last rule applied must have been **atom+**. The only program formulas that match this atom are clauses (2) and (3). Therefore, the proof can proceed in two ways:

| | | |
|---|---|---|
| [1] | $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow$ `before` $\ulcorner e_1\urcorner\ \ulcorner e_2\urcorner$ | assumption |
| [2] | • $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow$ `beforeFact` $\ulcorner e_1\urcorner\ \ulcorner e_2\urcorner$ | by rule **atom+** on [1] and clause (2), |
| [3] | `(beforeFact` $\ulcorner e_1\urcorner\ \ulcorner e_2\urcorner$`)` $\in \ulcorner o\urcorner$ | by rule **atom+** on [2] and since no rule for `beforeFact` is defined in `GMEC` or $\ulcorner\mathcal{H}\urcorner$, |
| [4] | $(e_1, e_2) \in o$ | by definition of $\ulcorner\cdot\urcorner^O$, |
| [5] | $(e_1, e_2) \in o^+$ | by definition of transitive closure; |
| [6] | • $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow$ `beforeFact` $\ulcorner e_1\urcorner\ \ulcorner e\urcorner$`,` `before` $\ulcorner e\urcorner\ \ulcorner e_2\urcorner$ | by rule **atom+** on [1] and clause (3), for some event $e$, |
| [7] | $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow$ `beforeFact` $\ulcorner e_1\urcorner\ \ulcorner e\urcorner$ | by rule **and+** on [6] (left branch), |
| [8] | `(beforeFact` $\ulcorner e_1\urcorner\ \ulcorner e\urcorner$`)` $\in \ulcorner o\urcorner$ | by rule **atom+** on [7] and since no rule for `beforeFact` is defined in `GMEC` or $\ulcorner\mathcal{H}\urcorner$, |
| [9] | $(e_1, e) \in o$ | by definition of $\ulcorner\cdot\urcorner^O$, |

| | | |
|---|---|---|
| [10] | $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{before}\ \ulcorner e\urcorner\ \ulcorner e_2\urcorner$ | by rule **and+** on [6] (right branch), |
| [11] | $(e, e_2) \in o^+$ | by induction hypothesis on [10], |
| [12] | $(e_1, e_2) \in o^+$ | by definition of transitive closure on [9, 11]. |

($a. \Leftarrow$) Let $\sigma = e'_1 \dots e'_l$, with $e'_1 = e_1$ and $e'_l = e_2$ be a sequence of events such that, for $i = 1 \dots l-1$, $(e'_i, e'_{i+1}) \in o$, proving in this way that $(e_1, e_2) \in o^+$. We conduct the proof by induction on the length $l$ of this sequence.

Case $l = 1$:

| | | |
|---|---|---|
| [1] | $(e_1, e_2) \in o$ | assumption |
| [2] | $(\texttt{beforeFact}\ \ulcorner e_1\urcorner\ \ulcorner e_2\urcorner) \in \ulcorner o\urcorner$ | by definition of $\ulcorner\cdot\urcorner^{O}$, |
| [3] | $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{beforeFact}\ \ulcorner e_1\urcorner\ \ulcorner e_2\urcorner$ | by rules **true+** and **atom+**, |
| [4] | $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{before}\ \ulcorner e_1\urcorner\ \ulcorner e_2\urcorner$ | by rules **atom+** on [3] and clause (2). |

Case $l > 1$: Then $\sigma = e_1, e \dots e_2$ with $(e_1, e) \in o$ and $(e, e_2) \in o^+$. Thus

| | | |
|---|---|---|
| [1] | $(e_1, e) \in o$ *and* $(e, e_2) \in o^+$ | assumption |
| [2] | $(e_1, e) \in o$ | conjunct from [1], |
| [3] | $(\texttt{beforeFact}\ \ulcorner e_1\urcorner\ \ulcorner e\urcorner) \in \ulcorner o\urcorner$ | by definition of $\ulcorner\cdot\urcorner^{O}$, |
| [4] | $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{beforeFact}\ \ulcorner e_1\urcorner\ \ulcorner e\urcorner$ | by rules **true+** and **atom+**, |
| [5] | $(e, e_2) \in o^+$ | conjunct from [1], |
| [6] | $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{before}\ \ulcorner e\urcorner\ \ulcorner e_2\urcorner$ | by induction hypothesis, |
| [7] | $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{beforeFact}\ \ulcorner e_1\urcorner\ \ulcorner e\urcorner\texttt{,}\ \texttt{before}\ \ulcorner e\urcorner\ \ulcorner e_2\urcorner$ | by rule **and+** on [4, 6], |
| [8] | $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{before}\ \ulcorner e_1\urcorner\ \ulcorner e_2\urcorner$ | by rule **atom+** on [7] and clause (2). |

(b. $\Rightarrow$) The last rule applied in a derivation of $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{not (before}\ \ulcorner e_1\urcorner\ \ulcorner e_2\urcorner\texttt{)}$ must have been **naf+**. Therefore, the negative sequent $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \not\Longrightarrow \texttt{before}\ \ulcorner e_1\urcorner\ \ulcorner e_2\urcorner$ has a derivation. Now, by property 3.1, the sequent $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{before}\ \ulcorner e_1\urcorner\ \ulcorner e_2\urcorner$ is not derivable. Thus, by ($a$), $(e_1, e_2) \notin o^+$.

(b. $\Leftarrow$) By property 3.2, it is enough to show that, whenever $(e_1, e_2) \notin o^+$, the sequent $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{before}\ \ulcorner e_1\urcorner\ \ulcorner e_2\urcorner$ is finitely non-provable. We show a stronger property, i.e. that the search for a proof of a sequent of this form must terminate (either with success, as in ($a$), or with failure).

Assume *ab absurdum* that the sequent $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{before}\ \ulcorner e_1\urcorner\ \ulcorner e_2\urcorner$ has an infinite derivation. Being the goal atomic, this sequent must result from the application of rule **atom+** to either clause (2) or clause (3), which define `before`. As the former is a fact in program `GMEC`, we must discard this alternative: the derivation would otherwise terminate after one application of rule **true+**. Therefore, rule **atom+** has been used on clause (3) and the sequent $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{beforeFact}\ \ulcorner e_1\urcorner\ \ulcorner \hat{e}_1\urcorner\texttt{,}\ \texttt{before}\ \ulcorner \hat{e}_1\urcorner\ \ulcorner e_2\urcorner$ for some event $\hat{e}_1 \in E$. By an application of rule **and+**, we reduce this sequent to $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{beforeFact}\ \ulcorner e_1\urcorner\ \ulcorner \hat{e}_1\urcorner$ and $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{before}\ \ulcorner \hat{e}_1\urcorner\ \ulcorner e_2\urcorner$. By definition of $\ulcorner\cdot\urcorner^{O}$, the former corresponds to $(e_1, \hat{e}_1) \in o$. The latter is a reinstantiation of our original problem.

By iterating this reasoning pattern *ad infinitum*, we conclude that the recursive clause (3) must have been applied infinitely many time for the original sequent to have an infinite derivation. In particular, the sequents $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{beforeFact}\ \ulcorner \hat{e}_i\urcorner\ \ulcorner \hat{e}_{i+1}\urcorner$ are derivable for an infinite sequence of events $\{\hat{e}_i\}_{i\in\omega}$ (with $\hat{e}_0 = e_1$). Thus $(\hat{e}_i, \hat{e}_{i+1}) \in o$ for all $i \in \omega$. At this point, we must remember that $E$ is finite. Therefore, there are two distinct indices $i, j$ with $i < j$ such that $\hat{e}_i = \hat{e}_j$. Then, by definition of transitive closure, we have that $(\hat{e}_i, \hat{e}_j) \in o^+$, but this violates the irreflexivity of $o^+$. □

**Lemma 3.2** (*Correspondence between* `broken` *and nb*)

*Let* $\mathcal{H} = (E,\ P,\ [\cdot\rangle,\ \langle\cdot],\ ]\cdot,\cdot[)$ *be a GMEC-structure and* $o$ *a state of knowledge, then*

*a.* $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{broken}\ \ulcorner e_1\urcorner\ \ulcorner p\urcorner\ \ulcorner e_2\urcorner$ *iff* $\neg nb(p, e_1, e_2, o^+)$ *holds in* $\mathcal{H}$;

*b.* $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{not (broken}\ \ulcorner e_1\urcorner\ \ulcorner p\urcorner\ \ulcorner e_2\urcorner\texttt{)}$ *iff* $nb(p, e_1, e_2, o^+)$ *holds in* $\mathcal{H}$.

PROOF.
(*a.* $\Rightarrow$) Assume that the sequent $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{broken}\ \ulcorner e_1\urcorner\ \ulcorner p\urcorner\ \ulcorner e_2\urcorner$ is derivable. By rule **atom+** on clause (5) and a number of applications of rule **and+**, we are left with the sequents below. For the sake of conciseness, we display the proof in a tabular form: the left column displays the derived sequents, the corresponding meta-mathematical property is shown in the central column, and the right column contains a justification of this correspondence.

| | | |
|---|---|---|
| $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{happens}\ \ulcorner e\urcorner$ | $e \in E$ | by definition of $\ulcorner E\urcorner$, |
| $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{before}\ \ulcorner e_1\urcorner\ \ulcorner e\urcorner$ | $(e_1, e) \in o^+$ | by lemma 3.1, |
| $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{before}\ \ulcorner e\urcorner\ \ulcorner e_2\urcorner$ | $(e, e_2) \in o^+$ | by lemma 3.1, |
| $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{initiates}\ \ulcorner e\urcorner\ \ulcorner q\urcorner\texttt{;}$ | $(e \in [q\rangle$ | by definition of $\ulcorner[\cdot\rangle\urcorner$ |
| $\texttt{terminates}\ \ulcorner e\urcorner\ \ulcorner q\urcorner$ | $\vee\, e \in \langle q])$ | and of $\ulcorner\langle\cdot]\urcorner$, |
| $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{exclusive}\ \ulcorner p\urcorner\ \ulcorner q\urcorner\texttt{;}$ | $(e \in ]p, q[$ | by definition of $\ulcorner]\cdot,\cdot[\urcorner$ |
| $\ulcorner p\urcorner\texttt{=}\ulcorner q\urcorner$ | $\vee\, p = q)$ | and lemma 3.5. |

We need to take the conjunction of the items in the central column in order to obtain a statement equivalent to $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{broken}\ \ulcorner e_1\urcorner\ \ulcorner p\urcorner\ \ulcorner e_2\urcorner$:

$$(e_1, e) \in o^+ \wedge (e, e_2) \in o^+ \wedge (e \in [q\rangle \vee e \in \langle q]) \wedge (]p, q[ \vee p = q).$$

We now abstract over the event $e$ and the property $q$ and obtain the formula

$$\exists e \in E.\, \exists q \in P.\, ((e_1, e) \in o^+ \wedge (e, e_2) \in o^+ \wedge (e \in [q\rangle \vee e \in \langle q]) \wedge (]p, q[ \vee p = q))$$

that is equivalent, after some logical manipulations, to $\neg nb(p, e_1, e_2, o^+)$.

(*a.* $\Leftarrow$) Assume now that $\neg nb(p, e_1, e_2, o^+)$ is valid in $\mathcal{H}$, i.e. that

$$\exists e \in E.\, ((e_1, e) \in o^+ \wedge (e, e_2) \in o^+ \wedge \exists q \in P.\, ((e \in [q\rangle \vee e \in \langle q]) \wedge (]p, q[ \vee p = q))).$$

Let $e'$ and $q'$ be such $e$ and $q$ respectively. Then, by instantiation, we obtain:

$$(e_1, e') \in o^+ \wedge (e', e_2) \in o^+ \wedge (e' \in [q'\rangle \vee e' \in \langle q']) \wedge (]p, q'[ \vee p = q').$$

Each conjunct, plus the fact that $e' \in E$, can be immediately rewritten as a valid sequent. We use conventions similar to the ones adopted in the first part of this proof.

| | | |
|---|---|---|
| $e' \in E$ | $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{happens}\ \ulcorner e'\urcorner$ | by definition of $\ulcorner E\urcorner$, |
| $(e_1, e') \in o^+$ | $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{before}\ \ulcorner e_1\urcorner\ \ulcorner e'\urcorner$ | by lemma 3.1, |
| $(e', e_2) \in o^+$ | $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{before}\ \ulcorner e'\urcorner\ \ulcorner e_2\urcorner$ | by lemma 3.1, |
| $(e' \in [q\rangle$ | $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{initiates}\ \ulcorner e'\urcorner\ \ulcorner q\urcorner\texttt{;}$ | by definition of $\ulcorner[\cdot\rangle\urcorner$ |
| $\vee\, e' \in \langle q])$ | $\texttt{terminates}\ \ulcorner e'\urcorner\ \ulcorner q\urcorner$ | and of $\ulcorner\langle\cdot]\urcorner$, |
| $(e \in ]p, q[$ | $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{exclusive}\ \ulcorner p\urcorner\ \ulcorner q\urcorner\texttt{;}$ | by definition of $\ulcorner]\cdot,\cdot[\urcorner$ |
| $\vee\, p = q)$ | $\ulcorner p\urcorner\texttt{=}\ulcorner q\urcorner$ | and lemma 3.5. |

We have proved in this way every goal in the body of clause (5). Thus, by a number of applications of rule **and+** and an application of rule **atom+**, the head of this clause is valid, i.e.

$$\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{broken}\ \ulcorner e_1\urcorner\ \ulcorner p\urcorner\ \ulcorner e_2\urcorner.$$

(b. $\Rightarrow$) By property 3.1 and (*a*).

(b. $\Leftarrow$) By rule **naf+** and properties 3.2–3.3, we are reduced to proving that $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{broken}\ \ulcorner e_1\urcorner\ \ulcorner p\urcorner\ \ulcorner e_2\urcorner$ has only (failed) finite derivations. Assume *ab absurdum* that

there is an infinite derivation of this sequent. The last inference rules applied in this derivation must be **atom+** and **and+**. Therefore, one of the atomic formulas in the body of rule (5) must have an infinite derivation. Clearly, only predicates having a recursive definition are candidate. The only predicate having this property is `before`, but by lemma 3.1 this sequent has finite derivations only. □

**Theorem 3.6** (`GMEC` *computes MVIs*)

*Let* $\mathcal{H} = (E,\ P,\ [\cdot\rangle,\ \langle\cdot],\ ]\cdot,\cdot[)$ *be a GMEC-structure and* $o$ *a state of knowledge, then*

*a.* $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{holds (period } \ulcorner e_1\urcorner\ \ulcorner p\urcorner\ \ulcorner e_2\urcorner\texttt{)}$ *iff* $p(e_1, e_2) \in MVI(o^+)$;

*b.* $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{not (holds (period } \ulcorner e_1\urcorner\ \ulcorner p\urcorner\ \ulcorner e_2\urcorner\texttt{))}$ *iff* $p(e_1, e_2) \notin MVI(o^+)$.

Proof.

($a. \Rightarrow$) Assume that $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{holds (period } \ulcorner e_1\urcorner\ \ulcorner p\urcorner\ \ulcorner e_2\urcorner\texttt{)}$. We prove that, under this hypothesis, $(p(e_1, e_2), o^+) \in \upsilon_{\mathcal{H}}$; the thesis will follow by the definitions of validity and of the function $MVI(\cdot)$.

By applying rule **atom+** on clause (4), and then rule **and+**, we get reduced to proving the following relations, where, as in the proof of lemma 3.2, the left and central columns stand in an *if-and-only-if* relation justified by the right column.

| | | |
|---|---|---|
| $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{happens } \ulcorner e_1\urcorner$ | $e_1 \in E$ | by definition of $\ulcorner E\urcorner$, |
| $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{initiates } \ulcorner e_1\urcorner\ \ulcorner p\urcorner$ | $e_1 \in [p\rangle$ | by definition of $\ulcorner[\cdot\rangle\urcorner$, |
| $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{happens } \ulcorner e_2\urcorner$ | $e_2 \in E$ | by definition of $\ulcorner E\urcorner$, |
| $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{terminates } \ulcorner e_2\urcorner\ \ulcorner p\urcorner$ | $e_2 \in \langle p]$ | by definition of $\ulcorner\langle\cdot]\urcorner$, |
| $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{before } \ulcorner e_1\urcorner\ \ulcorner e_2\urcorner$ | $(e_1, e_2) \in o^+$ | by lemma 3.1, |
| $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{not (broken } \ulcorner e_1\urcorner\ \ulcorner p\urcorner\ \ulcorner e_2\urcorner\texttt{)}$ | $nb(p, e_1, e_2, o^+)$ | by lemma 3.2. |

Now, it suffices to notice that the second, fourth, fifth and sixth relation on the right-hand side correspond respectively to the conditions $(i)$, $(ii)$, $(iii)$ and $(iv)$ of the definition of evaluation. Therefore $(p(e_1, e_2), o^+) \in \upsilon_{\mathcal{H}}$, thus $\mathcal{I}_{\mathcal{H}}; o^+ \models p(e_1, e_2)$ and finally $p(e_1, e_2) \in MVI(o^+)$.

($a. \Leftarrow$) Assume that $p(e_1, e_2) \in MVI(o^+)$. Therefore, by definition, $(p(e_1, e_2), o^+) \in \upsilon_{\mathcal{H}}$, i.e.

$$e_1 \in [p\rangle \wedge e_2 \in \langle p] \wedge (e_1, e_2) \in o^+ \wedge nb(p, e_1, e_2, o^+).$$

Each conjunct and the fact that $e_1, e_2 \in E$ can be related to sequent derivations by reversing the previous construction:

| | | |
|---|---|---|
| $e_1 \in E$ | $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{happens } \ulcorner e_1\urcorner$ | by definition of $\ulcorner E\urcorner$, |
| $e_1 \in [p\rangle$ | $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{initiates } \ulcorner e_1\urcorner\ \ulcorner p\urcorner$ | by definition of $\ulcorner[\cdot\rangle\urcorner$, |
| $e_2 \in E$ | $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{happens } \ulcorner e_2\urcorner$ | by definition of $\ulcorner E\urcorner$, |
| $e_2 \in \langle p]$ | $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{terminates } \ulcorner e_2\urcorner\ \ulcorner p\urcorner$ | by definition of $\ulcorner\langle\cdot]\urcorner$, |
| $(e_1, e_2) \in o^+$ | $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{before } \ulcorner e_1\urcorner\ \ulcorner e_2\urcorner$ | by lemma 3.1, |
| $nb(p, e_1, e_2, o^+)$ | $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{not (broken } \ulcorner e_1\urcorner\ \ulcorner p\urcorner\ \ulcorner e_2\urcorner\texttt{)}$ | by lemma 3.2. |

Therefore, we have derivations for all the atomic formulas in the body of clause 4. By some applications of rule **and+** and then of rule **atom+**, we produce a derivation for the sequent

$$\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{holds (period } \ulcorner e_1\urcorner\ \ulcorner p\urcorner\ \ulcorner e_2\urcorner\texttt{)}$$

(b. $\Rightarrow$) By property 3.1 and $(a)$.

(b. $\Leftarrow$) As in the proof of lemma 3.2, it suffices to prove that the sequent:

$$\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{holds (period } \ulcorner e_1\urcorner\ \ulcorner p\urcorner\ \ulcorner e_2\urcorner\texttt{)}$$

has only (possibly failed) finite derivations. The last inference rule applied during the search for a proof of this sequent must be **atom+** on clause (4). Therefore, it has an infinite derivation if and only if an atomic subgoal in the body of this clause has an infinite derivation. However, by lemmas 3.1 and 3.2, and the definition of $\ulcorner\mathcal{H}\urcorner$, every such subgoal is finitely provable or unprovable. □

**Lemma 3.7** (*Soundness of* GMEC *w.r.t. the GMEC-frame semantics*)

*Let* $\mathcal{H} = (E,\ P,\ [\cdot\rangle,\ \langle\cdot],\ ]\cdot,\cdot[)$ *be a GMEC-structure,* $o$ *a state of knowledge and* $\varphi$ *and GMEC-formula, then*

*a. if* $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{holds}\ \ulcorner\varphi\urcorner$, *then* $\mathcal{I}_{\mathcal{H}}; o^+ \models \varphi$;

*b. if* $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \not\Longrightarrow \texttt{holds}\ \ulcorner\varphi\urcorner$, *then* $\mathcal{I}_{\mathcal{H}}; o^+ \not\models \varphi$.

PROOF. Since the definition of the predicate `holds` contains recursive calls in the context of negation-as-failure (clauses (6), (9) and (10)), the statements ($a$) and ($b$) depend on each other. Therefore, we need to use a proof technique somewhat more elaborated than in the case of the previous results.

Indeed we will prove the two statements simultaneously by mutual induction. The inductive argument is on the ordered pair consisting of the number of connectives in the formula $\varphi$ and height of the derivation trees for the sequents

*a.* $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{holds}\ \ulcorner\varphi\urcorner$ *and*

*b.* $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \not\Longrightarrow \texttt{holds}\ \ulcorner\varphi\urcorner$.

Technically, this corresponds to a nested induction over the structure of $\varphi$ and on the structure of the two sequent derivations.

For the sake of readability, we use singly framed labels to denote the proof cases for ($a$) and double frames for the proof cases for ($b$).

$\boxed{\varphi = p(e_1, e_2)}$ and $\boxed{\boxed{\varphi = p(e_1, e_2)}}$ The result follows by theorem 3.6.

$\boxed{\varphi = \neg\varphi'}$

| | | |
|---|---|---|
| [1] | $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{not (holds}\ \ulcorner\varphi'\urcorner\texttt{)}$ | by rule **atom+** on clause (6), |
| [2] | $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \not\Longrightarrow \texttt{holds}\ \ulcorner\varphi'\urcorner$ | by rule **naf+**, |
| [3] | $\mathcal{I}_{\mathcal{H}}; o^+ \not\models \varphi'$ | by induction hypothesis ($b$), |
| [4] | $\mathcal{I}_{\mathcal{H}}; o^+ \models \neg\varphi'$ | by definition of $\models$. |

$\boxed{\boxed{\varphi = \neg\varphi'}}$

| | | |
|---|---|---|
| [1] | $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \not\Longrightarrow \texttt{not (holds}\ \ulcorner\varphi'\urcorner\texttt{)}$ | by rule **atom–** on clause (6), |
| [2] | $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{holds}\ \ulcorner\varphi'\urcorner$ | by rule **naf–**, |
| [3] | $\mathcal{I}_{\mathcal{H}}; o^+ \models \varphi'$ | by induction hypothesis ($a$), |
| [4] | $\mathcal{I}_{\mathcal{H}}; o^+ \not\models \neg\varphi'$ | by the consistency of $\models$. |

$\boxed{\varphi = \varphi' \wedge \varphi''}$

| | | |
|---|---|---|
| [1] | $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{holds}\ \ulcorner\varphi'\urcorner\texttt{, holds}\ \ulcorner\varphi''\urcorner$ | by rules **atom+** on clause (7), |
| [2] | $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{holds}\ \ulcorner\varphi'\urcorner$ | by rule **and+** on [1], |
| [3] | $\mathcal{I}_{\mathcal{H}}; o^+ \models \varphi'$ | by induction hypothesis ($a$), |

[4] $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{holds}\ \ulcorner\varphi''\urcorner$ by rule **and+** on [1],
[5] $\mathcal{I}_\mathcal{H}; o^+ \models \varphi''$ by induction hypothesis ($a$),
[6] $\mathcal{I}_\mathcal{H}; o^+ \models \varphi' \wedge \varphi''$ by definition of $\models$ on [3, 5].

$\boxed{\varphi = \varphi' \wedge \varphi''}$

[1] $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \not\Longrightarrow \texttt{holds}\ \ulcorner\varphi'\urcorner\texttt{, holds}\ \ulcorner\varphi''\urcorner$ by rules **atom–** on clause (7),
[2] • $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \not\Longrightarrow \texttt{holds}\ \ulcorner\varphi'\urcorner$ by rule **and**$-_1$ on [1],
[3] $\mathcal{I}_\mathcal{H}; o^+ \not\models \varphi'$ by induction hypothesis ($b$),
[4] • $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \not\Longrightarrow \texttt{holds}\ \ulcorner\varphi''\urcorner$ by rule **and**$-_2$ on [1],
[5] $\mathcal{I}_\mathcal{H}; o^+ \not\models \varphi''$ by induction hypothesis ($b$),
[6] $\mathcal{I}_\mathcal{H}; o^+ \not\models \varphi' \wedge \varphi''$ by the consistency of $\models$ on [3, 5].

$\boxed{\varphi = \varphi' \vee \varphi''}$

[1] $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{holds}\ \ulcorner\varphi'\urcorner\texttt{; holds}\ \ulcorner\varphi''\urcorner$ by rules **atom+** on clause (8),
[2] • $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{holds}\ \ulcorner\varphi'\urcorner$ by rule **or**$+_1$ on [1],
[3] $\mathcal{I}_\mathcal{H}; o^+ \models \varphi'$ by induction hypothesis ($a$),
[4] • $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{holds}\ \ulcorner\varphi''\urcorner$ by rule **or**$+_2$ on [1],
[5] $\mathcal{I}_\mathcal{H}; o^+ \models \varphi''$ by induction hypothesis ($a$),
[6] $\mathcal{I}_\mathcal{H}; o^+ \models \varphi' \vee \varphi''$ by definition of $\models$ on [3, 5].

$\boxed{\varphi = \varphi' \vee \varphi''}$

[1] $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \not\Longrightarrow \texttt{holds}\ \ulcorner\varphi'\urcorner\texttt{; holds}\ \ulcorner\varphi''\urcorner$ by rules **atom–** on clause (8),
[2] $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \not\Longrightarrow \texttt{holds}\ \ulcorner\varphi'\urcorner$ by rule **or–** on [1],
[3] $\mathcal{I}_\mathcal{H}; o^+ \not\models \varphi'$ by induction hypothesis ($b$),
[4] $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \not\Longrightarrow \texttt{holds}\ \ulcorner\varphi''\urcorner$ by rule **or–** on [1],
[5] $\mathcal{I}_\mathcal{H}; o^+ \not\models \varphi''$ by induction hypothesis ($b$),
[6] $\mathcal{I}_\mathcal{H}; o^+ \not\models \varphi' \vee \varphi''$ by the consistency of $\models$ on [3, 5].

$\boxed{\varphi = \Box\varphi'}$

[1] $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{holds}\ \ulcorner\varphi'\urcorner\texttt{,}$ $\texttt{not (fails\_must}\ \ulcorner\varphi'\urcorner\texttt{)}$ by rule **atom+** on clause (9),
[2] $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{holds}\ \ulcorner\varphi'\urcorner$ by rule **and+** on [1],
[3] $\mathcal{I}_\mathcal{H}; o^+ \models \varphi'$ by induction hypothesis ($a$),
[4] $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{not (fails\_must}\ \ulcorner\varphi'\urcorner\texttt{)}$ by rule **and+** on [1],
[5] $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \not\Longrightarrow \texttt{fails\_must}\ \ulcorner\varphi'\urcorner$ by rules **naf+**,
[6] $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \not\Longrightarrow \texttt{happens}\ t_1\texttt{,}$ $\texttt{happens}\ t_2\texttt{,}$ $\texttt{not (before}\ t_1\ t_2\texttt{),}$ $\texttt{not (before}\ t_2\ t_1\texttt{),}$ $\texttt{beforeFact}\ t_1\ t_2$ $\texttt{=> not (holds (must}\ \ulcorner\varphi'\urcorner\texttt{))}$ by rule **atom–** on clause (10); since the variables `E1` and `E2` are implicitly quantified in front of the clause, this relation should hold for all terms $t_1$ and $t_2$,
[7] • $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \not\Longrightarrow \texttt{happens}\ t_1$ by rule **and**$-_1$ on [6],
[8] $\llcorner t_1 \lrcorner \notin E$ by rule **atom–** and definition of $\ulcorner E\urcorner$,
[9] • $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \not\Longrightarrow \texttt{happens}\ t_2$ by rules **and**$-_2$ and **and**$-_1$ on [6],
[10] $\llcorner t_2 \lrcorner \notin E$ by rule **atom–** and definition of $\ulcorner E\urcorner$,
[11] • $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \not\Longrightarrow \texttt{not (before}\ t_1\ t_2\texttt{)}$ by rules **and**$-_2$ and **and**$-_1$ on [6],
[12] $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{before}\ t_1\ t_2$ by rules **naf–**,
[13] $(\llcorner t_1 \lrcorner, \llcorner t_2 \lrcorner) \in o^+$ by lemma 3.1,
[14] • $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \not\Longrightarrow \texttt{not (before}\ t_2\ t_1\texttt{)}$ by rules **and**$-_2$ and **and**$-_1$ on [6],
[15] $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{before}\ t_2\ t_1$ by rules **naf–**,
[16] $(\llcorner t_2 \lrcorner, \llcorner t_1 \lrcorner) \in o^+$ by lemma 3.1,

[17] • $\texttt{GMEC},\ulcorner\mathcal{H}\urcorner,\ulcorner o\urcorner \Longrightarrow \texttt{beforeFact}\ t_1\ t_2\ \texttt{=> not (holds (must}\ \ulcorner\varphi'\urcorner\texttt{))}$ — by rule **and**–$_2$ on [6],

[18] $\texttt{GMEC},\ulcorner\mathcal{H}\urcorner,\ulcorner o \uparrow (\llcorner t_1 \lrcorner, \llcorner t_2 \lrcorner)\urcorner \Longrightarrow \texttt{not (holds (must}\ \ulcorner\varphi'\urcorner\texttt{))}$ — by rule **impl**– and the definition of $\ulcorner\cdot\urcorner^O$,

[19] $\texttt{GMEC},\ulcorner\mathcal{H}\urcorner,\ulcorner o \uparrow (\llcorner t_1 \lrcorner, \llcorner t_2 \lrcorner)\urcorner \Longrightarrow \texttt{holds (must}\ \ulcorner\varphi'\urcorner\texttt{)}$ — by rule **naf**–,

[20] $\mathcal{I}_\mathcal{H}; \{o \uparrow (\llcorner t_1 \lrcorner, \llcorner t_2 \lrcorner)\}^+ \models \Box\varphi'$ — by induction hypothesis ($a$), since $\{o \uparrow (\llcorner t_1 \lrcorner, \llcorner t_2 \lrcorner)\}^+$ is a proper extension of $o$,

[21] $\forall t_1, t_2.\ (\llcorner t_1 \lrcorner \notin E \vee \llcorner t_2 \lrcorner \notin E \vee (\llcorner t_1 \lrcorner, \llcorner t_2 \lrcorner) \in o^+ \vee (\llcorner t_2 \lrcorner, \llcorner t_1 \lrcorner) \in o^+ \vee \mathcal{I}_\mathcal{H}; \{o \uparrow (\llcorner t_1 \lrcorner, \llcorner t_2 \lrcorner)\}^+ \models \Box\varphi')$ — by taking the disjunction of [8, 10, 13, 16, 20],

[22] $\neg\exists t_1, t_2.\ (\llcorner t_1 \lrcorner \in E \wedge \llcorner t_2 \lrcorner \in E \wedge (\llcorner t_1 \lrcorner, \llcorner t_2 \lrcorner) \notin o^+ \wedge (\llcorner t_2 \lrcorner, \llcorner t_1 \lrcorner) \notin o^+ \wedge \mathcal{I}_\mathcal{H}; \{o \uparrow (\llcorner t_1 \lrcorner, \llcorner t_2 \lrcorner)\}^+ \not\models \Box\varphi')$ — by logical equivalences,

[23] $\neg\exists e_1, e_2 \in E.\ ((e_1, e_2) \notin o^+ \wedge (e_2, e_1) \notin o^+ \wedge \mathcal{I}_\mathcal{H}; \{o \uparrow (e_1, e_2)\}^+ \not\models \Box\varphi')$ — by definition of $\ulcorner\cdot\urcorner^E$,

[24] $\mathcal{I}_\mathcal{H}; o^+ \models \Box\varphi'$ — by combining [3] and [23] and lemma 2.5.

$\boxed{\varphi = \Box\varphi'}$

[1] $\texttt{GMEC},\ulcorner\mathcal{H}\urcorner,\ulcorner o\urcorner \Longrightarrow \texttt{holds}\ \ulcorner\varphi'\urcorner\texttt{, not (fails\_must}\ \ulcorner\varphi'\urcorner\texttt{)}$ — by rule **atom**– on clause (9),

[2] • $\texttt{GMEC},\ulcorner\mathcal{H}\urcorner,\ulcorner o\urcorner \Longrightarrow \texttt{holds}\ \ulcorner\varphi'\urcorner$ — by rule **and**–$_1$ on [1]

[3] $\mathcal{I}_\mathcal{H}; o^+ \not\models \varphi'$ — by induction hypothesis ($b$),

[4] • $\texttt{GMEC},\ulcorner\mathcal{H}\urcorner,\ulcorner o\urcorner \Longrightarrow \texttt{not (fails\_must}\ \ulcorner\varphi'\urcorner\texttt{)}$ — by rule **and**–$_1$ on [1]

[5] $\texttt{GMEC},\ulcorner\mathcal{H}\urcorner,\ulcorner o\urcorner \Longrightarrow \texttt{fails\_must}\ \ulcorner\varphi'\urcorner$ — by rule **naf**–,

[6] $\texttt{GMEC},\ulcorner\mathcal{H}\urcorner,\ulcorner o\urcorner \Longrightarrow \texttt{happens}\ t_1\texttt{, happens}\ t_2\texttt{, not (before}\ t_1\ t_2\texttt{), not (before}\ t_2\ t_1\texttt{), beforeFact}\ t_1\ t_2\ \texttt{=> not (holds (must}\ \ulcorner\varphi'\urcorner\texttt{))}$ — by rule **atom+** on clause (10), for some term $t_1$ and $t_2$,

[7] $\texttt{GMEC},\ulcorner\mathcal{H}\urcorner,\ulcorner o\urcorner \Longrightarrow \texttt{happens}\ t_1$ — by rule **and+** on [6],

[8] $t_1 = \ulcorner e_1\urcorner$ with $e_1 \in E$ — by definition of $\ulcorner E\urcorner$,

[9] $\texttt{GMEC},\ulcorner\mathcal{H}\urcorner,\ulcorner o\urcorner \Longrightarrow \texttt{happens}\ t_2$ — by rule **and+** on [6],

[10] $t_2 = \ulcorner e_2\urcorner$ with $e_2 \in E$ — by definition of $\ulcorner E\urcorner$,

[11] $\texttt{GMEC},\ulcorner\mathcal{H}\urcorner,\ulcorner o\urcorner \Longrightarrow \texttt{not (before}\ \ulcorner e_1\urcorner\ \ulcorner e_2\urcorner\texttt{)}$ — by rule **and+** on [6],

[12] $(e_1, e_2) \notin o^+$ — by lemma 3.1,

[13] $\texttt{GMEC},\ulcorner\mathcal{H}\urcorner,\ulcorner o\urcorner \Longrightarrow \texttt{not (before}\ \ulcorner e_2\urcorner\ \ulcorner e_1\urcorner\texttt{)}$ — by rule **and+** on [6],

[14] $(e_2, e_1) \notin o^+$ — by lemma 3.1,

[15] $\texttt{GMEC},\ulcorner\mathcal{H}\urcorner,\ulcorner o\urcorner \Longrightarrow \texttt{beforeFact}\ \ulcorner e_1\urcorner\ \ulcorner e_2\urcorner\ \texttt{=> not (holds (must}\ \ulcorner\varphi'\urcorner\texttt{))}$ — by rule **and+** on [6],

[16] $\texttt{GMEC},\ulcorner\mathcal{H}\urcorner,\ulcorner o \uparrow (e_1, e_2)\urcorner \Longrightarrow \texttt{not (holds (must}\ \ulcorner\varphi'\urcorner\texttt{))}$ — by rule **impl+** and the definition of $\ulcorner\cdot\urcorner^O$,

[17] $\texttt{GMEC},\ulcorner\mathcal{H}\urcorner,\ulcorner o \uparrow (e_1, e_2)\urcorner \not\Longrightarrow \texttt{holds (must}\ \ulcorner\varphi'\urcorner\texttt{)}$ — by rule **naf+**,

[18] $\mathcal{I}_\mathcal{H}; \{o \uparrow (e_1, e_2)\}^+ \not\models \Box\varphi'$ — by induction hypothesis ($b$), since $\{o \uparrow (\llcorner t_1 \lrcorner, \llcorner t_2 \lrcorner)\}^+$ is a proper extension of $o$,

[19] $\exists e_1, e_2 \in E.\ ((e_1, e_2) \notin o^+ \wedge (e_2, e_1) \notin o^+ \wedge \mathcal{I}_\mathcal{H}; \{o \uparrow (e_1, e_2)\}^+ \not\models \Box\varphi')$ — by taking the conjunction of [8, 10, 12, 14, 18],

[20] $\mathcal{I}_\mathcal{H}; o^+ \not\models \Box\varphi'$ — by Lemma 2.5 on [3, 19].

$\boxed{\varphi = \Diamond\varphi'}$

[1] • $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{holds}\ \ulcorner\varphi'\urcorner$ — by rules **atom+** on clause (11),

[2] $\mathcal{I}_\mathcal{H}; o^+ \models \varphi'$ — by induction hypothesis ($a$),

[3] • $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{happens}\ t_1, \texttt{happens}\ t_2, \texttt{not (before}\ t_1\ t_2\texttt{)}, \texttt{not (before}\ t_2\ t_1\texttt{)}, \texttt{beforeFact}\ t_1\ t_2\ \texttt{=> holds (may}\ \ulcorner\varphi'\urcorner\texttt{)}$ — by rule **atom+** on clause (12), for some term $t_1$ and $t_2$,

[4] $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{happens}\ t_1$ — by rule **and+** on [3],

[5] $t_1 = \ulcorner e_1\urcorner$ with $e_1 \in E$ — by definition of $\ulcorner E\urcorner$,

[6] $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{happens}\ t_2$ — by rule **and+** on [3],

[7] $t_2 = \ulcorner e_2\urcorner$ with $e_2 \in E$ — by definition of $\ulcorner E\urcorner$,

[8] $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{not (before}\ \ulcorner e_1\urcorner\ \ulcorner e_2\urcorner\texttt{)}$ — by rule **and+** on [3],

[9] $(e_1, e_2) \notin o^+$ — by lemma 3.1,

[10] $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{not (before}\ \ulcorner e_2\urcorner\ \ulcorner e_1\urcorner\texttt{)}$ — by rule **and+** on [3],

[11] $(e_2, e_1) \notin o^+$ — by lemma 3.1,

[12] $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{beforeFact}\ \ulcorner e_1\urcorner\ \ulcorner e_2\urcorner\ \texttt{=> holds (may}\ \ulcorner\varphi'\urcorner\texttt{)}$ — by rule **and+** on [3],

[13] $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o \uparrow (e_1, e_2)\urcorner \Longrightarrow \texttt{holds (may}\ \ulcorner\varphi'\urcorner\texttt{)}$ — by rule **impl+** and the definition of $\ulcorner \cdot \urcorner_O$,

[14] $\mathcal{I}_\mathcal{H}; \{o \uparrow (e_1, e_2)\}^+ \models \Diamond\varphi'$ — by induction hypothesis ($b$), since $\{o \uparrow (\llcorner t_1 \lrcorner, \llcorner t_2 \lrcorner)\}^+$ is a proper extension of $o$,

[15] $\exists e_1, e_2 \in E.\ ((e_1, e_2) \notin o^+ \wedge (e_2, e_1) \notin o^+ \wedge \mathcal{I}_\mathcal{H}; \{o \uparrow (e_1, e_2)\}^+ \models \Diamond\varphi')$ — by taking the conjunction of [5, 7, 9, 11, 14],

[16] $\mathcal{I}_\mathcal{H}; o^+ \models \Diamond\varphi'$ — by lemma 2.5 on [2, 15].

$\boxed{\boxed{\varphi = \Diamond\varphi'}}$

[1] $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \not\Longrightarrow \texttt{holds}\ \ulcorner\varphi'\urcorner$ — by rules **atom**– on clause (11),

[2] $\mathcal{I}_\mathcal{H}; o^+ \not\models \varphi'$ — by induction hypothesis ($b$),

[3] $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \not\Longrightarrow \texttt{happens}\ t_1, \texttt{happens}\ t_2, \texttt{not (before}\ t_1\ t_2\texttt{)}, \texttt{not (before}\ t_2\ t_1\texttt{)}, \texttt{beforeFact}\ t_1\ t_2\ \texttt{=> holds (may}\ \ulcorner\varphi'\urcorner\texttt{)}$ — by rule **atom**– on clause (12); since the variables `E1` and `E2` are implicitly quantified in front of the clause, this relation should hold for all terms $t_1$ and $t_2$,

[4] • $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \not\Longrightarrow \texttt{happens}\ t_1$ — by rule **and**–$_1$ on [3],

[5] $\llcorner t_1 \lrcorner \notin E$ — by rule **atom**– and definition of $\ulcorner E\urcorner$,

[6] • $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \not\Longrightarrow \texttt{happens}\ t_2$ — by rules **and**–$_2$ and **and**–$_1$ on [3],

[7] $\llcorner t_2 \lrcorner \notin E$ — by rule **atom**– and definition of $\ulcorner E\urcorner$,

[8] • $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \not\Longrightarrow \texttt{not (before}\ t_1\ t_2\texttt{)}$ — by rules **and**–$_2$ and **and**–$_1$ on [3],

[9] $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{before}\ t_1\ t_2$ — by rules **naf**–,

[10] $(\llcorner t_1 \lrcorner, \llcorner t_2 \lrcorner) \in o^+$ — by lemma 3.1,

[11] • $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \not\Longrightarrow \texttt{not (before}\ t_2\ t_1\texttt{)}$ — by rules **and**–$_2$ and **and**–$_1$ on [3],

[12] $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{before}\ t_2\ t_1$ — by rules **naf**–,

[13] $(\llcorner t_2 \lrcorner, \llcorner t_1 \lrcorner) \in o^+$ — by lemma 3.1,

[14] • $\texttt{GMEC},\ulcorner\mathcal{H}\urcorner,\ulcorner o\urcorner \not\Longrightarrow \texttt{beforeFact}\ t_1\ t_2$ $\texttt{=> holds (may}\ \ulcorner\varphi'\urcorner\texttt{)}$ — by rule **and**$-_2$ on [3],

[15] $\texttt{GMEC},\ulcorner\mathcal{H}\urcorner,\ulcorner o\uparrow(\llcorner t_1\lrcorner,\llcorner t_2\lrcorner)\urcorner \not\Longrightarrow \texttt{holds (must}\ \ulcorner\varphi'\urcorner\texttt{)}$ — by rule **impl**$-$ and the definition of $\ulcorner\cdot\urcorner^O$,

[16] $\mathcal{I}_\mathcal{H};\{o\uparrow(\llcorner t_1\lrcorner,\llcorner t_2\lrcorner)\}^+ \not\models \Diamond\varphi'$ — by induction hypothesis ($b$), since $\{o\uparrow(\llcorner t_1\lrcorner,\llcorner t_2\lrcorner)\}^+$ is a proper extension of $o$,

[17] $\forall t_1,t_2.\ (\llcorner t_1\lrcorner \notin E \vee \llcorner t_2\lrcorner \notin E \vee (\llcorner t_1\lrcorner,\llcorner t_2\lrcorner)\in o^+ \vee (\llcorner t_2\lrcorner,\llcorner t_1\lrcorner)\in o^+ \vee \mathcal{I}_\mathcal{H};\{o\uparrow(\llcorner t_1\lrcorner,\llcorner t_2\lrcorner)\}^+ \not\models \Diamond\varphi')$ — by taking the disjunction of [5, 7, 10, 13, 16],

[18] $\neg\exists t_1,t_2.\ (\llcorner t_1\lrcorner \in E \wedge \llcorner t_2\lrcorner \in E \wedge (\llcorner t_1\lrcorner,\llcorner t_2\lrcorner)\notin o^+ \wedge (\llcorner t_2\lrcorner,\llcorner t_1\lrcorner)\notin o^+ \wedge \mathcal{I}_\mathcal{H};\{o\uparrow(\llcorner t_1\lrcorner,\llcorner t_2\lrcorner)\}^+ \models \Diamond\varphi')$ — by logical equivalences,

[19] $\neg\exists e_1,e_2\in E.\ ((e_1,e_2)\notin o^+ \wedge (e_2,e_1)\notin o^+ \wedge \mathcal{I}_\mathcal{H};\{o\uparrow(e_1,e_2)\}^+ \models \Diamond\varphi')$ — by definition of $\ulcorner\cdot\urcorner^E$,

[20] $\mathcal{I}_\mathcal{H};o^+ \not\models \Diamond\varphi'$ — by combining [2] and [19] and lemma 2.5. □

**Lemma 3.8** (*Completeness of* GMEC *w.r.t. the GMEC-frame semantics*)

*Let* $\mathcal{H}=(E,\ P,\ [\cdot\rangle,\ \langle\cdot],\ ]\cdot,\cdot[)$ *be a GMEC-structure,* $o$ *a state of knowledge and* $\varphi$ *and GMEC-formula, then*

*a. if* $\mathcal{I}_\mathcal{H};o^+ \models \varphi$, *then* $\texttt{GMEC},\ulcorner\mathcal{H}\urcorner,\ulcorner o\urcorner \Longrightarrow \texttt{holds}\ \ulcorner\varphi\urcorner$;

*b. if* $\mathcal{I}_\mathcal{H};o^+ \not\models \varphi$, *then* $\texttt{GMEC},\ulcorner\mathcal{H}\urcorner,\ulcorner o\urcorner \not\Longrightarrow \texttt{holds}\ \ulcorner\varphi\urcorner$.

PROOF. As in the previous lemma, we need to cope with the two statements simultaneously. Therefore, we proceed by a nested mutual induction on the structure of the formula $\varphi$ and the cardinality of $Ext(o^+)$.

We rely on essentially the same conventions as in the proof of lemma 3.7. The two proofs are essentially dual.

$\boxed{\varphi = p(e_1,e_2)}$ and $\boxed{\boxed{\varphi = p(e_1,e_2)}}$ The desired result follows by theorem 3.6.

$\boxed{\varphi=\neg\varphi'}$

[1] $\mathcal{I}_\mathcal{H};o^+ \not\models \varphi'$ — by definition of $\models$,
[2] $\texttt{GMEC},\ulcorner\mathcal{H}\urcorner,\ulcorner o\urcorner \not\Longrightarrow \texttt{holds}\ \ulcorner\varphi'\urcorner$ — by induction hypothesis ($b$),
[3] $\texttt{GMEC},\ulcorner\mathcal{H}\urcorner,\ulcorner o\urcorner \Longrightarrow \texttt{not (holds}\ \ulcorner\varphi'\urcorner\texttt{)}$ — by rule **naf+**,
[4] $\texttt{GMEC},\ulcorner\mathcal{H}\urcorner,\ulcorner o\urcorner \Longrightarrow \texttt{holds}\ \ulcorner\neg\varphi'\urcorner$ — by rule **atom+** on clause (6).

$\boxed{\boxed{\varphi=\neg\varphi'}}$

[1] $\mathcal{I}_\mathcal{H};o^+ \models \varphi'$ — by the consistency of $\models$,
[2] $\texttt{GMEC},\ulcorner\mathcal{H}\urcorner,\ulcorner o\urcorner \Longrightarrow \texttt{holds}\ \ulcorner\varphi'\urcorner$ — by induction hypothesis ($a$),
[3] $\texttt{GMEC},\ulcorner\mathcal{H}\urcorner,\ulcorner o\urcorner \not\Longrightarrow \texttt{not (holds}\ \ulcorner\varphi'\urcorner\texttt{)}$ — by rule **naf**$-$,
[4] $\texttt{GMEC},\ulcorner\mathcal{H}\urcorner,\ulcorner o\urcorner \not\Longrightarrow \texttt{holds}\ \ulcorner\neg\varphi'\urcorner$ — by rule **atom**$-$ on clause (6).

$\boxed{\varphi=\varphi'\wedge\varphi''}$

| | | |
|---|---|---|
| [1] | $\mathcal{I}_{\mathcal{H}}; o^+ \models \varphi'$ *and* $\mathcal{I}_{\mathcal{H}}; o^+ \models \varphi''$ | by definition of $\models$, |
| [2] | $\mathcal{I}_{\mathcal{H}}; o^+ \models \varphi'$ | conjunct in [1] |
| [3] | $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{holds}\ \ulcorner\varphi'\urcorner$ | by induction hypothesis ($a$), |
| [4] | $\mathcal{I}_{\mathcal{H}}; o^+ \models \varphi''$ | conjunct in [1] |
| [5] | $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{holds}\ \ulcorner\varphi''\urcorner$ | by induction hypothesis ($a$), |
| [6] | $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{holds}\ \ulcorner\varphi'\urcorner, \texttt{holds}\ \ulcorner\varphi''\urcorner$ | by rule **and+** on [3, 5], |
| [7] | $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{holds}\ \ulcorner\varphi' \wedge \varphi''\urcorner$ | by rule **atom+** on clause (7). |

$\boxed{\boxed{\varphi = \varphi' \wedge \varphi''}}$

| | | |
|---|---|---|
| [1] | $\mathcal{I}_{\mathcal{H}}; o^+ \not\models \varphi'$ *or* $\mathcal{I}_{\mathcal{H}}; o^+ \not\models \varphi''$ | by the consistency of $\models$, |
| [2] | • $\mathcal{I}_{\mathcal{H}}; o^+ \not\models \varphi'$ | subcase of [1] |
| [3] | $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \not\Longrightarrow \texttt{holds}\ \ulcorner\varphi'\urcorner$ | by induction hypothesis ($b$), |
| [4] | $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \not\Longrightarrow \texttt{holds}\ \ulcorner\varphi'\urcorner, \texttt{holds}\ \ulcorner\varphi''\urcorner$ | by rule **and**$-_1$, |
| [5] | • $\mathcal{I}_{\mathcal{H}}; o^+ \not\models \varphi''$ | subcase of [1] |
| [6] | $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \not\Longrightarrow \texttt{holds}\ \ulcorner\varphi''\urcorner$ | by induction hypothesis ($b$), |
| [7] | $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \not\Longrightarrow \texttt{holds}\ \ulcorner\varphi'\urcorner, \texttt{holds}\ \ulcorner\varphi''\urcorner$ | by rule **and**$-_2$, |
| [8] | $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \not\Longrightarrow \texttt{holds}\ \ulcorner\varphi' \wedge \varphi''\urcorner$ | by rules **atom**$-$ on [4, 7] and clause (7). |

$\boxed{\varphi = \varphi' \vee \varphi''}$

| | | |
|---|---|---|
| [1] | $\mathcal{I}_{\mathcal{H}}; o^+ \models \varphi'$ *or* $\mathcal{I}_{\mathcal{H}}; o^+ \models \varphi''$ | by definition of $\models$, |
| [2] | • $\mathcal{I}_{\mathcal{H}}; o^+ \models \varphi'$ | subcase of [1] |
| [3] | $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{holds}\ \ulcorner\varphi'\urcorner$ | by induction hypothesis ($a$), |
| [4] | $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{holds}\ \ulcorner\varphi'\urcorner; \texttt{holds}\ \ulcorner\varphi''\urcorner$ | by rule **or+**$_1$, |
| [5] | • $\mathcal{I}_{\mathcal{H}}; o^+ \models \varphi''$ | subcase of [1] |
| [6] | $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{holds}\ \ulcorner\varphi''\urcorner$ | by induction hypothesis ($a$), |
| [7] | $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{holds}\ \ulcorner\varphi'\urcorner; \texttt{holds}\ \ulcorner\varphi''\urcorner$ | by rule **or+**$_2$, |
| [8] | $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{holds}\ \ulcorner\varphi' \vee \varphi''\urcorner$ | by rules **atom+** on [4, 7] and clause (8). |

$\boxed{\boxed{\varphi = \varphi' \vee \varphi''}}$

| | | |
|---|---|---|
| [1] | $\mathcal{I}_{\mathcal{H}}; o^+ \not\models \varphi'$ *and* $\mathcal{I}_{\mathcal{H}}; o^+ \not\models \varphi''$ | by the consistency of $\models$, |
| [2] | $\mathcal{I}_{\mathcal{H}}; o^+ \not\models \varphi'$ | conjunct in [1] |
| [3] | $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \not\Longrightarrow \texttt{holds}\ \ulcorner\varphi'\urcorner$ | by induction hypothesis ($b$), |
| [4] | $\mathcal{I}_{\mathcal{H}}; o^+ \not\models \varphi''$ | conjunct in [1] |
| [5] | $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \not\Longrightarrow \texttt{holds}\ \ulcorner\varphi''\urcorner$ | by induction hypothesis ($b$), |
| [6] | $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \not\Longrightarrow \texttt{holds}\ \ulcorner\varphi'\urcorner; \texttt{holds}\ \ulcorner\varphi''\urcorner$ | by rule **or**$-$ on [3, 5], |
| [7] | $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \not\Longrightarrow \texttt{holds}\ \ulcorner\varphi' \vee \varphi''\urcorner$ | by rule **atom**$-$ on clause (8). |

$\boxed{\varphi = \Box\varphi'}$

| | | |
|---|---|---|
| [1] | $\mathcal{I}_{\mathcal{H}}; o^+ \models \Box\varphi'$ | assumption |
| [2] | $\mathcal{I}_{\mathcal{H}}; o^+ \models \varphi'$ | by Lemma 2.5 on [1], |
| [3] | $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{holds}\ \ulcorner\varphi'\urcorner$ | by induction hypothesis ($a$), |
| [4] | $\neg\exists e_1, e_2 \in E.\ ((e_1, e_2) \notin o^+ \wedge (e_2, e_1) \notin o^+ \wedge \mathcal{I}_{\mathcal{H}}; \{o \uparrow (e_1, e_2)\}^+ \not\models \Box\varphi')$ | by Lemma 2.5 on [1], |
| [5] | $\neg\exists t_1, t_2.\ (\llcorner t_1 \lrcorner \in E \wedge \llcorner t_2 \lrcorner \in E \wedge (\llcorner t_1 \lrcorner, \llcorner t_2 \lrcorner) \notin o^+ \wedge (\llcorner t_2 \lrcorner, \llcorner t_1 \lrcorner) \notin o^+ \wedge \mathcal{I}_{\mathcal{H}}; \{o \uparrow (\llcorner t_1 \lrcorner, \llcorner t_2 \lrcorner)\}^+ \not\models \Box\varphi')$ | by definition of $\ulcorner\cdot\urcorner^E$ |

| | | |
|---|---|---|
| [6] | $\forall t_1, t_2.\ (\llcorner t_1 \lrcorner \notin E$<br>$\quad \vee\ \llcorner t_2 \lrcorner \notin E$<br>$\quad \vee\ (\llcorner t_1 \lrcorner, \llcorner t_2 \lrcorner) \in o^+$<br>$\quad \vee\ (\llcorner t_2 \lrcorner, \llcorner t_1 \lrcorner) \in o^+$<br>$\quad \vee\ \mathcal{I}_{\mathcal{H}}; \{o \uparrow (\llcorner t_1 \lrcorner, \llcorner t_2 \lrcorner)\}^+ \models \Box\varphi')$ | by logical equivalences |
| [7] | $\bullet\ \llcorner t_1 \lrcorner \notin E$ | subcase of [6] |
| [8] | $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \not\Longrightarrow \texttt{happens}\ t_1$ | by rule **atom**– and definition of $\ulcorner E\urcorner$, |
| [9] | $\bullet\ \llcorner t_2 \lrcorner \notin E$ | subcase of [6] |
| [10] | $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \not\Longrightarrow \texttt{happens}\ t_2$ | by rule **atom**– and definition of $\ulcorner E\urcorner$, |
| [11] | $\bullet\ (\llcorner t_1 \lrcorner, \llcorner t_2 \lrcorner) \in o^+$ | subcase of [6] |
| [12] | $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{before}\ t_1\ t_2$ | by lemma 3.1, |
| [13] | $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \not\Longrightarrow \texttt{not (before}\ t_1\ t_2\texttt{)}$ | by rules **naf**–, |
| [14] | $\bullet\ (\llcorner t_2 \lrcorner, \llcorner t_1 \lrcorner) \in o^+$ | subcase of [6] |
| [15] | $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{before}\ t_2\ t_1$ | by lemma 3.1, |
| [16] | $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \not\Longrightarrow \texttt{not (before}\ t_2\ t_1\texttt{)}$ | by rules **naf**–, |
| [17] | $\bullet\ \mathcal{I}_{\mathcal{H}}; \{o \uparrow (\llcorner t_1 \lrcorner, \llcorner t_2 \lrcorner)\}^+ \models \Box\varphi'$ | subcase of [6] |
| [18] | $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o \uparrow (\llcorner t_1 \lrcorner, \llcorner t_2 \lrcorner)\urcorner$<br>$\quad \Longrightarrow \texttt{holds (must}\ \ulcorner\varphi'\urcorner\texttt{)}$ | by induction hypothesis ($a$), since $\{o \uparrow (\llcorner t_1 \lrcorner, \llcorner t_2 \lrcorner)\}^+$ is a proper extension of $o$, |
| [19] | $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o \uparrow (\llcorner t_1 \lrcorner, \llcorner t_2 \lrcorner)\urcorner$<br>$\quad \not\Longrightarrow \texttt{not (holds (must}\ \ulcorner\varphi'\urcorner\texttt{))}$ | by rule **naf**–, |
| [20] | $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \not\Longrightarrow \texttt{beforeFact}\ t_1\ t_2$<br>$\quad \texttt{=> not (holds (must}\ \ulcorner\varphi'\urcorner\texttt{))}$ | by rule **impl**– and the definition of $\ulcorner\cdot\urcorner^O$, |
| [21] | $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \not\Longrightarrow \texttt{happens}\ t_1\texttt{,}$<br>$\texttt{happens}\ t_2\texttt{,}$<br>$\texttt{not (before}\ t_1\ t_2\texttt{),}$<br>$\texttt{not (before}\ t_2\ t_1\texttt{),}$<br>$\texttt{beforeFact}\ t_1\ t_2$<br>$\quad \texttt{=> not (holds (must}\ \ulcorner\varphi'\urcorner\texttt{))}$ | by rules **and**–$_1$ and **and**–$_2$ on [8, 10, 13, 16, 20], |
| [22] | $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \not\Longrightarrow \texttt{fails\_must}\ \ulcorner\varphi'\urcorner$ | by rule **atom**– on clause (10); this relation should hold for all terms $t_1$ and $t_2$ since the variables `E1` and `E2` are implicitly quantified in front of the clause, |
| [23] | $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{not (fails\_must}\ \ulcorner\varphi'\urcorner\texttt{)}$ | by rules **naf+**, |
| [24] | $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{holds}\ \ulcorner\Box\varphi'\urcorner$ | by rules **and+** on [3, 23] and **atom+** on clause (9). |

$\boxed{\varphi = \Box\varphi'}$

| | | |
|---|---|---|
| [1] | $\mathcal{I}_{\mathcal{H}}; o^+ \not\models \varphi'$ *or*<br>$\exists e_1, e_2 \in E.\ ((e_1, e_2) \notin o^+$<br>$\quad \wedge\ (e_2, e_1) \notin o^+$<br>$\quad \wedge\ \mathcal{I}_{\mathcal{H}}; \{o \uparrow (e_1, e_2)\}^+ \not\models \Box\varphi')$ | by Lemma 2.5, |
| [2] | $\bullet\ \mathcal{I}_{\mathcal{H}}; o^+ \not\models \varphi'$ | subcase of [1] |
| [3] | $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \not\Longrightarrow \texttt{holds}\ \ulcorner\varphi'\urcorner$ | by induction hypothesis ($b$), |
| [4] | $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \not\Longrightarrow \texttt{holds}\ \ulcorner\varphi'\urcorner\texttt{,}$<br>$\quad \texttt{not (fails\_must}\ \ulcorner\varphi'\urcorner\texttt{)}$ | by rule **and**–$_1$, |
| [5] | $\bullet\ \exists e_1, e_2 \in E.\ ((e_1, e_2) \notin o^+$<br>$\quad \wedge\ (e_2, e_1) \notin o^+$<br>$\quad \wedge\ \mathcal{I}_{\mathcal{H}}; \{o \uparrow (e_1, e_2)\}^+ \not\models \Box\varphi')$ | subcase of [1] |
| [6] | $e_1 \in E$ | conjunct in [5] |
| [7] | $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{happens}\ \ulcorner e_1\urcorner$ | by definition of $\ulcorner E\urcorner$, |
| [8] | $e_2 \in E$ | conjunct in [5] |
| [9] | $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{happens}\ \ulcorner e_2\urcorner$ | by definition of $\ulcorner E\urcorner$, |

| | | |
|---|---|---|
| [10] | $(e_1, e_2) \notin o^+$ | conjunct in [5] |
| [11] | $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{not (before } \ulcorner e_1\urcorner\ \ulcorner e_2\urcorner\texttt{)}$ | by lemma 3.1, |
| [12] | $(e_2, e_1) \notin o^+$ | conjunct in [5] |
| [13] | $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{not (before } \ulcorner e_2\urcorner\ \ulcorner e_1\urcorner\texttt{)}$ | by lemma 3.1, |
| [14] | $\mathcal{I}_\mathcal{H}; \{o \uparrow (e_1, e_2)\}^+ \not\models \Box\varphi'$ | conjunct in [5] |
| [15] | $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o \uparrow (e_1, e_2)\urcorner \not\Longrightarrow \texttt{holds (must } \ulcorner\varphi'\urcorner\texttt{)}$ | by induction hypothesis ($b$), since $\{o \uparrow (\lfloor t_1 \rfloor, \lfloor t_2 \rfloor)\}^+$ is a proper extension of $o$ |
| [16] | $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o \uparrow (e_1, e_2)\urcorner \Longrightarrow \texttt{not (holds (must } \ulcorner\varphi'\urcorner\texttt{))}$ | by rule **naf+**, |
| [17] | $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{beforeFact } \ulcorner e_1\urcorner\ \ulcorner e_2\urcorner\ \texttt{=> not (holds (must } \ulcorner\varphi'\urcorner\texttt{))}$ | by rule **impl+** and the definition of $\ulcorner\cdot\urcorner^O$, |
| [18] | $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{happens } \ulcorner e_1\urcorner\texttt{, happens } \ulcorner e_2\urcorner\texttt{, not (before } \ulcorner e_1\urcorner\ \ulcorner e_2\urcorner\texttt{), not (before } \ulcorner e_2\urcorner\ \ulcorner e_1\urcorner\texttt{), beforeFact } \ulcorner e_1\urcorner\ \ulcorner e_2\urcorner\ \texttt{=> not (holds (must } \ulcorner\varphi'\urcorner\texttt{))}$ | by rule **and+** on [7, 9, 11, 13, 17], |
| [19] | $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{fails\_must } \ulcorner\varphi'\urcorner$ | by rule **atom+** on clause (10), with `E1` and `E2` instantiated to $\ulcorner e_1\urcorner$ and $\ulcorner e_2\urcorner$ respectively, |
| [20] | $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \not\Longrightarrow \texttt{not (fails\_must } \ulcorner\varphi'\urcorner\texttt{)}$ | by rule **naf–**, |
| [21] | $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \not\Longrightarrow \texttt{holds } \ulcorner\varphi'\urcorner\texttt{, not (fails\_must } \ulcorner\varphi'\urcorner\texttt{)}$ | by rule **and–**$_2$, |
| [22] | $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \not\Longrightarrow \texttt{holds } \ulcorner\Box\varphi'\urcorner$ | by rules **atom–** on [4, 21] and clause (9). |

$\boxed{\varphi = \Diamond\varphi'}$

| | | |
|---|---|---|
| [1] | $\mathcal{I}_\mathcal{H}; o^+ \models \varphi'$ *or* $\exists e_1, e_2 \in E.\ ((e_1, e_2) \notin o^+ \wedge (e_2, e_1) \notin o^+ \wedge \mathcal{I}_\mathcal{H}; \{o \uparrow (e_1, e_2)\}^+ \models \Diamond\varphi')$ | by lemma 2.5, |
| [2] | • $\mathcal{I}_\mathcal{H}; o^+ \models \varphi'$ | subcase of [1] |
| [3] | $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{holds } \ulcorner\varphi'\urcorner$ | by induction hypothesis ($a$), |
| [4] | $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{holds } \ulcorner\Diamond\varphi'\urcorner$ | by rule **atom+** on clause (11), |
| [5] | • $\exists e_1, e_2 \in E.\ ((e_1, e_2) \notin o^+ \wedge (e_2, e_1) \notin o^+ \wedge \mathcal{I}_\mathcal{H}; \{o \uparrow (e_1, e_2)\}^+ \models \Diamond\varphi')$ | subcase of [1] |
| [6] | $e_1 \in E$ | conjunct in [5] |
| [7] | $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{happens } \ulcorner e_1\urcorner$ | by definition of $\ulcorner E\urcorner$, |
| [8] | $e_2 \in E$ | conjunct in [5] |
| [9] | $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{happens } \ulcorner e_2\urcorner$ | by definition of $\ulcorner E\urcorner$, |
| [10] | $(e_1, e_2) \notin o^+$ | conjunct in [5] |
| [11] | $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{not (before } \ulcorner e_1\urcorner\ \ulcorner e_2\urcorner\texttt{)}$ | by lemma 3.1, |
| [12] | $(e_2, e_1) \notin o^+$ | conjunct in [5] |
| [13] | $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{not (before } \ulcorner e_2\urcorner\ \ulcorner e_1\urcorner\texttt{)}$ | by lemma 3.1, |
| [14] | $\mathcal{I}_\mathcal{H}; \{o \uparrow (e_1, e_2)\}^+ \models \Diamond\varphi'$ | conjunct in [5] |
| [15] | $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o \uparrow (e_1, e_2)\urcorner \Longrightarrow \texttt{holds (may } \ulcorner\varphi'\urcorner\texttt{)}$ | by induction hypothesis ($a$), since $\{o \uparrow (\lfloor t_1 \rfloor, \lfloor t_2 \rfloor)\}^+$ is a proper extension of $o$, |
| [16] | $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{beforeFact } \ulcorner e_1\urcorner\ \ulcorner e_2\urcorner\ \texttt{=> holds (may } \ulcorner\varphi'\urcorner\texttt{)}$ | by rule **impl+** and the definition of $\ulcorner\cdot\urcorner^O$, |

| | | |
|---|---|---|
| [17] | $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow$ `happens` $\ulcorner e_1\urcorner$, `happens` $\ulcorner e_2\urcorner$, `not (before` $\ulcorner e_1\urcorner$ $\ulcorner e_2\urcorner$`),` `not (before` $\ulcorner e_2\urcorner$ $\ulcorner e_1\urcorner$`),` `beforeFact` $\ulcorner e_1\urcorner$ $\ulcorner e_2\urcorner$ `=> holds (may` $\ulcorner\varphi'\urcorner$`)` | by rule **atom+** on [7, 9, 11, 13, 16] and clause (12), with $\ulcorner e_1\urcorner$ and $\ulcorner e_2\urcorner$ substituted for the variables `E1` and `E2` respectively, |
| [18] | $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow$ `holds` $\ulcorner\Diamond\varphi'\urcorner$ | by rules **atom+** on clause (12). |

$\boxed{\varphi = \Diamond\varphi'}$

| | | |
|---|---|---|
| [1] | $\mathcal{I}_\mathcal{H}; o^+ \not\models \Diamond\varphi'$ | assumption |
| [2] | $\mathcal{I}_\mathcal{H}; o^+ \not\models \varphi'$ | by lemma 2.5 on [1], |
| [3] | $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \not\Longrightarrow$ `holds` $\ulcorner\varphi'\urcorner$ | by induction hypothesis ($b$), |
| [4] | $\neg\exists e_1, e_2 \in E.\ ((e_1, e_2) \notin o^+ \wedge (e_2, e_1) \notin o^+ \wedge \mathcal{I}_\mathcal{H}; \{o \uparrow (e_1, e_2)\}^+ \models \Diamond\varphi')$ | by lemma 2.5 on [1], |
| [5] | $\neg\exists t_1, t_2.\ (\llcorner t_1 \lrcorner \in E \wedge \llcorner t_2 \lrcorner \in E \wedge (\llcorner t_1 \lrcorner, \llcorner t_2 \lrcorner) \notin o^+ \wedge (\llcorner t_2 \lrcorner, \llcorner t_1 \lrcorner) \notin o^+ \wedge \mathcal{I}_\mathcal{H}; \{o \uparrow (\llcorner t_1 \lrcorner, \llcorner t_2 \lrcorner)\}^+ \models \Diamond\varphi')$ | by definition of $\ulcorner\cdot\urcorner^E$, |
| [6] | $\forall t_1, t_2.\ (\llcorner t_1 \lrcorner \notin E \vee \llcorner t_2 \lrcorner \notin E \vee (\llcorner t_1 \lrcorner, \llcorner t_2 \lrcorner) \in o^+ \vee (\llcorner t_2 \lrcorner, \llcorner t_1 \lrcorner) \in o^+ \vee \mathcal{I}_\mathcal{H}; \{o \uparrow (\llcorner t_1 \lrcorner, \llcorner t_2 \lrcorner)\}^+ \not\models \Diamond\varphi')$ | by logical equivalences, |
| [7] | • $\llcorner t_1 \lrcorner \notin E$ | subcase of [6] |
| [8] | $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \not\Longrightarrow$ `happens` $t_1$ | by rule **atom–** and definition of $\ulcorner E\urcorner$, |
| [9] | • $\llcorner t_2 \lrcorner \notin E$ | subcase of [6] |
| [10] | $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \not\Longrightarrow$ `happens` $t_2$ | by rule **atom–** and definition of $\ulcorner E\urcorner$, |
| [11] | • $(\llcorner t_1 \lrcorner, \llcorner t_2 \lrcorner) \in o^+$ | subcase of [6] |
| [12] | $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow$ `before` $t_1$ $t_2$ | by lemma 3.1, |
| [13] | $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \not\Longrightarrow$ `not (before` $t_1$ $t_2$`)` | by rules **naf–**, |
| [14] | • $(\llcorner t_2 \lrcorner, \llcorner t_1 \lrcorner) \in o^+$ | subcase of [6] |
| [15] | $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow$ `before` $t_2$ $t_1$ | by lemma 3.1, |
| [16] | $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \not\Longrightarrow$ `not (before` $t_2$ $t_1$`)` | by rules **naf–**, |
| [17] | • $\mathcal{I}_\mathcal{H}; \{o \uparrow (\llcorner t_1 \lrcorner, \llcorner t_2 \lrcorner)\}^+ \not\models \Diamond\varphi'$ | subcase of [6] |
| [18] | $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o \uparrow (\llcorner t_1 \lrcorner, \llcorner t_2 \lrcorner)\urcorner \not\Longrightarrow$ `holds (must` $\ulcorner\varphi'\urcorner$`)` | by induction hypothesis ($b$), since $\{o \uparrow (\llcorner t_1 \lrcorner, \llcorner t_2 \lrcorner)\}^+$ is a proper extension of $o$, |
| [19] | $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \not\Longrightarrow$ `beforeFact` $t_1$ $t_2$ `=> holds (may` $\ulcorner\varphi'\urcorner$`)` | by rule **impl–** and the definition of $\ulcorner\cdot\urcorner^O$, |
| [20] | $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \not\Longrightarrow$ `happens` $t_1$`,` `happens` $t_2$`,` `not (before` $t_1$ $t_2$`),` `not (before` $t_2$ $t_1$`),` `beforeFact` $t_1$ $t_2$ `=> holds (may` $\ulcorner\varphi'\urcorner$`)` | by rules **and–**$_1$ and **and–**$_2$ on [8, 10, 13, 16, 19]; this is provable for all terms $t_1$ and $t_2$, |
| [21] | $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \not\Longrightarrow$ `holds` $\ulcorner\Diamond\varphi'\urcorner$ | by rule **atom–** on clauses (11) and (12) for [3] and [20] respectively. □ |

**Theorem 3.9** (*Soundness and completeness of* `GMEC` *w.r.t. GMEC-frames*)
*Let* $\mathcal{H} = (E,\ P,\ [\cdot\rangle,\ \langle\cdot],\ ]\cdot,\cdot[)$ *be a GMEC-structure,* $o$ *a state of knowledge and*

*$\varphi$ and GMEC-formula, then*

*a.* $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{holds } \ulcorner\varphi\urcorner$ *iff* $\mathcal{I}_{\mathcal{H}}; o^+ \models \varphi$;

*b.* $\texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{not (holds } \ulcorner\varphi\urcorner\texttt{)}$ *iff* $\mathcal{I}_{\mathcal{H}}; o^+ \not\models \varphi$.

PROOF. By rules **naf+** and **naf–**, the second statement can be rewritten as

$$\text{b}'.\ \texttt{GMEC}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \not\Longrightarrow \texttt{holds } \ulcorner\varphi\urcorner \ \textit{iff}\ \ \mathcal{I}_{\mathcal{H}}; o^+ \not\models \varphi.$$

It suffices now to apply lemmas 3.7 and 3.8 to the two directions of $(a)$ and $(b')$ to prove the theorem. □

**Lemma 3.3** (*Correspondence between* `skeBroken` *and nsb*)

*Let $\mathcal{H} = (E,\ P,\ [\cdot\rangle,\ \langle\cdot],\ ]\cdot,\cdot[)$ be a GMEC-structure and $o$ a state of knowledge, then*

*a.* $\texttt{GMEC+}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{skeBroken } \ulcorner e_1\urcorner\ \ulcorner p\urcorner\ \ulcorner e_2\urcorner$ *iff* $\neg nsb(p, e_1, e_2, o^+)$ *holds in* $\mathcal{H}$;

*b.* $\texttt{GMEC+}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{not (skeBroken } \ulcorner e_1\urcorner\ \ulcorner p\urcorner\ \ulcorner e_2\urcorner\texttt{)}$ *iff* $nsb(p, e_1, e_2, o^+)$ *holds in* $\mathcal{H}$.

PROOF. We proceed as in the proof of lemma 3.2.

($a.\ \Rightarrow$) By unfolding clause $(5')$, we obtain the following relations.

| | | |
|---|---|---|
| $\texttt{GMEC+}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{happens } \ulcorner e\urcorner$ | $e \in E$ | by definition of $\ulcorner E\urcorner$, |
| $\texttt{GMEC+}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{not (}\ulcorner e\urcorner = \ulcorner e_1\urcorner\texttt{)}$ | $e \neq e_1$ | by lemma 3.4, |
| $\texttt{GMEC+}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{not (}\ulcorner e\urcorner = \ulcorner e_2\urcorner\texttt{)}$ | $e \neq e_2$ | by lemma 3.4, |
| $\texttt{GMEC+}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{not (before } \ulcorner e\urcorner\ \ulcorner e_1\urcorner\texttt{)}$ | $(e, e_1) \notin o^+$ | by lemma 3.1, |
| $\texttt{GMEC+}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{not (before } \ulcorner e_2\urcorner\ \ulcorner e\urcorner\texttt{)}$ | $(e_2, e) \notin o^+$ | by lemma 3.1, |
| $\texttt{GMEC+}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{initiates } \ulcorner e\urcorner\ \ulcorner q\urcorner\texttt{;}$ | $(e \in [q\rangle$ | by definition of $\ulcorner[\cdot\rangle\urcorner$ |
| $\texttt{terminates } \ulcorner e\urcorner\ \ulcorner q\urcorner$ | $\vee\ e \in \langle q])$ | and of $\ulcorner\langle\cdot]\urcorner$, |
| $\texttt{GMEC+}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{exclusive } \ulcorner p\urcorner\ \ulcorner q\urcorner\texttt{;}$ | $(e \in ]p, q[$ | by definition of $\ulcorner]\cdot,\cdot[\urcorner$ |
| $\ulcorner p\urcorner\texttt{=}\ulcorner q\urcorner$ | $\vee\ p = q)$ | and lemma 3.5. |

By taking the conjunction of the formulas displayed in the central column, we have:

$$e \neq e_1 \wedge e \neq e_2 \wedge (e, e_1) \notin o^+ \wedge (e_2, e) \notin o^+ \wedge ((e \in [q\rangle \vee e \in \langle q]) \wedge (]p, q[ \vee p = q))$$

By abstracting over $e$ and $q$, we obtain

$$\begin{aligned} \exists e \in E.\, \exists q \in P.\ & e \neq e_1 \\ & \wedge\ e \neq e_2 \\ & \wedge\ (e, e_1) \notin o^+ \\ & \wedge\ (e_2, e) \notin o^+ \\ & \wedge\ ((e \in [q\rangle \vee e \in \langle q]) \wedge (]p, q[ \vee p = q)) \end{aligned}$$

that is equivalent, after some logical manipulations, to $nsb(p, e_1, e_2, o^+)$.

($a.\ \Leftarrow$) Similarly to the situation encountered in the proof of lemma 3.2, this direction of the proof follows by simply reversing the reasoning pattern just used. We omit it.

(b. $\Leftarrow$) By property 3.1 and $(a)$.

(b. $\Rightarrow$) This direction follows by property 3.2 since the only calls in clause $(10')$ that invoke recursive definitions involve the predicate `before`, that has only finite derivations, by lemma 3.1. □

**Theorem 3.10** (`GMEC+` *computes necessary MVIs*)

*Let* $\mathcal{H} = (E,\ P,\ [\cdot\rangle,\ \langle\cdot],\ ]\cdot,\cdot[)$ *be a GMEC-structure and* $o$ *a state of knowledge, then*

*a.* `GMEC+,`$\ulcorner\mathcal{H}\urcorner,\ulcorner o\urcorner \Longrightarrow$ `holds (must (period` $\ulcorner e_1\urcorner\ \ulcorner p\urcorner\ \ulcorner e_2\urcorner$`))` *iff* $p(e_1, e_2) \in \Box MVI(o^+)$;

*b.* `GMEC+,`$\ulcorner\mathcal{H}\urcorner,\ulcorner o\urcorner \Longrightarrow$ `not (holds (must (period` $\ulcorner e_1\urcorner\ \ulcorner p\urcorner\ \ulcorner e_2\urcorner$`)))` *iff* $p(e_1, e_2) \notin \Box MVI(o^+)$.

PROOF. We proceed as in the proof of theorem 3.6.
($a.\ \Rightarrow$) Assume that `GMEC+,`$\ulcorner\mathcal{H}\urcorner,\ulcorner o\urcorner \Longrightarrow$ `holds (must (period` $\ulcorner e_1\urcorner\ \ulcorner p\urcorner\ \ulcorner e_2\urcorner$`))` is derivable. We will prove that $e_1 \in [p\rangle$, $e_2 \in \langle p]$, $(e_1, e_2) \in o^+$ and $nsb(p, e_1, e_2, o^+)$ are entailed by this hypothesis. The thesis will follow by lemma 2.6.

By unfolding clause $(9')$ we obtain the following relations:

| | | |
|---|---|---|
| `GMEC+,`$\ulcorner\mathcal{H}\urcorner,\ulcorner o\urcorner \Longrightarrow$ `happens` $\ulcorner e_1\urcorner$ | $e_1 \in E$ | by definition of $\ulcorner E\urcorner$, |
| `GMEC+,`$\ulcorner\mathcal{H}\urcorner,\ulcorner o\urcorner \Longrightarrow$ `initiates` $\ulcorner e_1\urcorner\ \ulcorner p\urcorner$ | $e_1 \in [p\rangle$ | by definition of $\ulcorner[\cdot\rangle\urcorner$, |
| `GMEC+,`$\ulcorner\mathcal{H}\urcorner,\ulcorner o\urcorner \Longrightarrow$ `happens` $\ulcorner e_2\urcorner$ | $e_2 \in E$ | by definition of $\ulcorner E\urcorner$, |
| `GMEC+,`$\ulcorner\mathcal{H}\urcorner,\ulcorner o\urcorner \Longrightarrow$ `terminates` $\ulcorner e_2\urcorner\ \ulcorner p\urcorner$ | $e_2 \in \langle p]$ | by definition of $\ulcorner\langle\cdot]\urcorner$, |
| `GMEC+,`$\ulcorner\mathcal{H}\urcorner,\ulcorner o\urcorner \Longrightarrow$ `before` $\ulcorner e_1\urcorner\ \ulcorner e_2\urcorner$ | $(e_1, e_2) \in o^+$ | by lemma 3.1, |
| `GMEC+,`$\ulcorner\mathcal{H}\urcorner,\ulcorner o\urcorner \Longrightarrow$ `not (skeBroken` $\ulcorner e_1\urcorner\ \ulcorner p\urcorner\ \ulcorner e_2\urcorner$`)` | $nb(p, e_1, e_2, o^+)$ | by lemma 3.3. |

The central column contains all the hypotheses needed for the application of lemma 2.6.

($a.\ \Leftarrow$) As in the proof of theorem 3.6, this direction follows by simply reversing the reasoning pattern just used. We omit it.

(b. $\Leftarrow$) By property 3.1 and $(a)$.

(b. $\Rightarrow$) By the definition of $\ulcorner\mathcal{H}\urcorner$ and lemmas 3.1 and 3.3, clause $(9')$ cannot start a diverging derivation. The desired result follows from property 3.2. □

**Theorem 3.11** (`GMEC+` *computes possible MVIs*)

*Let* $\mathcal{H} = (E,\ P,\ [\cdot\rangle,\ \langle\cdot],\ ]\cdot,\cdot[)$ *be a GMEC-structure and* $o$ *a state of knowledge, then*

*a.* `GMEC+,`$\ulcorner\mathcal{H}\urcorner,\ulcorner o\urcorner \Longrightarrow$ `holds (may (period` $\ulcorner e_1\urcorner\ \ulcorner p\urcorner\ \ulcorner e_2\urcorner$`))` *iff* $p(e_1, e_2) \in \Diamond MVI(o^+)$;

*b.* `GMEC+,`$\ulcorner\mathcal{H}\urcorner,\ulcorner o\urcorner \Longrightarrow$ `not (holds (may (period` $\ulcorner e_1\urcorner\ \ulcorner p\urcorner\ \ulcorner e_2\urcorner$`)))` *iff* $p(e_1, e_2) \notin \Diamond MVI(o^+)$.

PROOF. Similar to the proofs of theorems 3.6 and 3.10. □

**Theorem 3.12** (*Soundness and completeness of* `GMEC+` *w.r.t. GMEC-frames*)

*Let* $\mathcal{H} = (E,\ P,\ [\cdot\rangle,\ \langle\cdot],\ ]\cdot,\cdot[)$ *be a GMEC-structure,* $o$ *a state of knowledge and* $\varphi$ *and GMEC-formula, then*

*a.* `GMEC+,`$\ulcorner\mathcal{H}\urcorner,\ulcorner o\urcorner \Longrightarrow$ `holds` $\ulcorner\varphi\urcorner$ *iff* $\mathcal{I}_{\mathcal{H}}; o^+ \models \varphi$;

*b.* `GMEC+,`$\ulcorner\mathcal{H}\urcorner,\ulcorner o\urcorner \Longrightarrow$ `not (holds` $\ulcorner\varphi\urcorner$`)` *iff* $\mathcal{I}_{\mathcal{H}}; o^+ \not\models \varphi$.

PROOF. As in theorem 3.9, $(a)$ and $(b)$ must be proved simultaneously in each direction. We will only sketch the proof for the forward direction ($\Rightarrow$). The present discussion together with the detailed proof of the analogous case treated as lemma 3.8 should suffice to the intrepid reader to reconstruct this long proof in its entirety.

The forward direction of the proof requires the techniques exploited in the proof of lemma 3.7, with the only difference that we need to distinguish finer proof cases for the modal formulas. More precisely, whenever the main connective of a formula is modal, we must consider the main connective of its immediate subformula.

For the sake of conciseness, we will perform the proof only for cases where the main connective is $\Box$. Again, we leave the rest of the proof to the valiant reader (the cases for $\Diamond$ are similar, and whenever the main connective is non-modal, the analogous cases in the proof of lemma 3.7 apply unchanged).

We are performing a mutual nested induction on the structure of the formula $\varphi$ and of the derivations for the sequents

$a$. $\texttt{GMEC+}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{holds } \ulcorner\varphi\urcorner$ *and*

$b'$. $\texttt{GMEC+}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \not\Longrightarrow \texttt{holds } \ulcorner\varphi\urcorner$.

Again, we use single frames to label proof cases for $(a)$, and double frames for proof cases for $(b')$.

$\boxed{\varphi = \Box p(e_1, e_2)}$ and $\boxed{\boxed{\varphi = \Box p(e_1, e_2)}}$

The result follows by theorem 3.10.

$\boxed{\varphi = \Box\neg\varphi'}$

| | | |
|---|---|---|
| [1] | $\texttt{GMEC+}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{holds (not (may } \ulcorner\varphi'\urcorner\texttt{))}$ | by rule **atom+** on clause $(11')$, |
| [2] | $\texttt{GMEC+}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{not (holds } \ulcorner\Diamond\varphi'\urcorner\texttt{)}$ | by rule **atom+** on clause $(6')$, |
| [3] | $\texttt{GMEC+}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \not\Longrightarrow \texttt{holds } \ulcorner\Diamond\varphi'\urcorner$ | by rule **naf+**, |
| [4] | $\mathcal{I}_{\mathcal{H}}; o^+ \not\models \Diamond\varphi'$ | by induction hypothesis $(b)$, |
| [5] | $\mathcal{I}_{\mathcal{H}}; o^+ \models \neg\Diamond\varphi'$ | by definition of $\models$, |
| [6] | $\mathcal{I}_{\mathcal{H}}; o^+ \models \Box\neg\varphi'$ | by property 2.1. |

$\boxed{\boxed{\varphi = \Box\neg\varphi'}}$

| | | |
|---|---|---|
| [1] | $\texttt{GMEC+}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \not\Longrightarrow \texttt{holds (not (may } \ulcorner\varphi'\urcorner\texttt{))}$ | by rule **atom–** on clause $(11')$, |
| [2] | $\texttt{GMEC+}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \not\Longrightarrow \texttt{not (holds } \ulcorner\Diamond\varphi'\urcorner\texttt{)}$ | by rule **atom–** on clause $(6')$, |
| [3] | $\texttt{GMEC+}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{holds } \ulcorner\Diamond\varphi'\urcorner$ | by rule **naf–**, |
| [4] | $\mathcal{I}_{\mathcal{H}}; o^+ \models \Diamond\varphi'$ | by induction hypothesis $(a)$, |
| [5] | $\mathcal{I}_{\mathcal{H}}; o^+ \not\models \neg\Diamond\varphi'$ | by the consistency of $\models$, |
| [6] | $\mathcal{I}_{\mathcal{H}}; o^+ \not\models \Box\neg\varphi'$ | by property 2.1. |

$\boxed{\varphi = \Box(\varphi' \wedge \varphi'')}$ and $\boxed{\boxed{\varphi = \Box(\varphi' \wedge \varphi'')}}$

Similarly to the previous case, clause $(12')$ is used to push the modality inside the formula. Then the technique seen in the proof of lemma 3.7 for the cases concerning conjunction is applied. Finally, we appeal to property 2.1 to restore $\varphi$ by pushing $\Box$ out as its main connective.

$\boxed{\varphi = \Box(\varphi' \vee \varphi'')}$ and $\boxed{\boxed{\varphi = \Box(\varphi' \vee \varphi'')}}$

Take verbatim the proof cases for $\Box$ from the proof of lemma 3.7 changing simply the reference to clause (9) and (10) to references to clauses $(13')$ and $(14')$ respectively. Clearly the structure of the subformula $\varphi' \vee \varphi''$ needs not to be expanded.

$\boxed{\varphi = \Box\Box\varphi'}$

[1] $\texttt{GMEC+}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{holds (must } \ulcorner\varphi'\urcorner\texttt{)}$ by rule **atom+** on clause $(15')$,
[2] $\mathcal{I}_{\mathcal{H}}; o^+ \models \Box\varphi'$ by induction hypothesis $(a)$,
[3] $\mathcal{I}_{\mathcal{H}}; o^+ \models \Box\Box\varphi'$ by property 2.1.

$\boxed{\boxed{\varphi = \Box\Box\varphi'}}$

[1] $\texttt{GMEC+}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \not\Longrightarrow \texttt{holds (must } \ulcorner\varphi'\urcorner\texttt{)}$ by rule **atom–** on clause $(15')$,
[2] $\mathcal{I}_{\mathcal{H}}; o^+ \not\models \Box\varphi'$ by induction hypothesis $(b)$,
[3] $\mathcal{I}_{\mathcal{H}}; o^+ \not\models \Box\Box\varphi'$ by property 2.1.

$\boxed{\varphi = \Box\Diamond\varphi'}$

Both clauses $(17')$ and $(16')$ can have been used by rule **atom+** as the last derivation step. In the first case, we simply need to transpose the corresponding proof case for $\Box$ from the proof of lemma 3.7. The second case applies only if $\varphi = \Box\Diamond\Box\varphi''$. We have the following derivation:

[1] $\texttt{GMEC+}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \Longrightarrow \texttt{holds (must (may } \ulcorner\varphi'\urcorner\texttt{))}$ by rule **atom+** on clause $(16')$,
[2] $\mathcal{I}_{\mathcal{H}}; o^+ \models \Box\Diamond\varphi''$ by induction hypothesis $(a)$,
[3] $\mathcal{I}_{\mathcal{H}}; o^+ \models \Box\Diamond\Box\varphi''$ by property 2.2.

$\boxed{\boxed{\varphi = \Box\Diamond\varphi'}}$

We must again distinguish two cases, based on the structure of $\varphi'$. If this formula is not of the form $\Box\Diamond\varphi''$, we behave as in the corresponding proof case for $\Box$-moded formulas in the proof of lemma 3.7.

Otherwise, the last rule applied must be **atom–** on clauses $(17')$ and $(16')$. The branch concerning the first clause is handled again as the second proof case for $\Box$ from the proof of lemma 3.7. The branch referring to the second clause is instead handled as follows:

[1] $\texttt{GMEC+}, \ulcorner\mathcal{H}\urcorner, \ulcorner o\urcorner \not\Longrightarrow \texttt{holds (must (may } \ulcorner\varphi'\urcorner\texttt{))}$ subcase generated by clause $(16')$
[2] $\mathcal{I}_{\mathcal{H}}; o^+ \not\models \Box\Diamond\varphi''$ by induction hypothesis $(b)$,
[3] $\mathcal{I}_{\mathcal{H}}; o^+ \not\models \Box\Diamond\Box\varphi''$ by property 2.2. □

**Throw this page away**

*1.*
*2.*
*3.*
*4.*
*5.*
*6.*
*7.*
*8.*
*9.*
*10.*
*11.*
*12.*
*1′.*
*2′.*
*3′.*
*4′.*
*5′.*
*6′.*
*7′.*
*8′.*
*9′.*
*10′.*
*11′.*
*12′.*
*13′.*
*14′.*
*15′.*
*16′.*
*17′.*
*18′.*
*19′.*
*20′.*
*21′.*
*22′.*
*23′.*
*24′.*
*25′.*
*26′.*
*27′.*