# Probabilistic Event Calculus based on Markov Logic Networks

Anastasios Skarlatidis[1,2], Georgios Paliouras[1], George A. Vouros[2], and Alexander Artikis[1]

[1] Institute of Informatics and Telecommunications
NCSR "Demokritos", Athens 15310, Greece
{anskarl,paliourg,a.artikis}@iit.demokritos.gr
[2] Department of Information and Communication Systems Engineering,
University of the Aegean, Samos, Greece
georgev@aegean.gr

**Abstract.** In this paper, we address the issue of uncertainty in event recognition by extending the Event Calculus with probabilistic reasoning. Markov Logic Networks are a natural candidate for our logic-based formalism. However, the temporal semantics of Event Calculus introduce a number of challenges for the proposed model. We show how and under what assumptions we can overcome these problems. Additionally, we demonstrate the advantages of the probabilistic Event Calculus through examples and experiments in the domain of activity recognition, using a publicly available dataset of video surveillance.

## 1 Introduction

Symbolic event recognition has received attention in a variety of application domains, such as health care monitoring, public transport management, activity recognition etc [2]. The aim of a symbolic event recognition system is to recognise high-level events (HLE) of interest, based on an input stream of time-stamped symbols, that is low-level events (LLE).

HLE are defined as relational structures over other subevents, either HLE or LLE. Logic-based methods, such as the Event Calculus [12], can naturally and compactly represent relational HLE definitions [4]. These methods, however, cannot handle uncertainty, which naturally exists in real-world applications.

In this paper, we present a probabilistic extension to Event Calculus [12], using Markov Logic Networks [8]. Event Calculus (EC) is a formalism for representing events and their effects, with formal and declarative semantics. Markov Logic Networks (MLN) is a statistical relational framework, which combines the expressivity of first-order logic with the formal probabilistic semantics of graphical models. Thus, MLN are a natural candidate for a probabilistic EC – see [18] for a survey on first-order logic probabilistic models. However, the temporal semantics of EC introduce a number of challenges. We show how and under what assumptions, the Event Calculus axioms can be efficiently represented in MLN. Moreover, we show the effect of probabilistic modelling on some of the most interesting properties of EC, such as the persistence of HLE.

To demonstrate the proposed approach, we apply it to activity recognition, a subfield of event recognition. The definitions of HLE are domain-dependent rules that are naturally defined by humans. Each rule is expressed in first-order logic using the EC language and is associated with a degree of confidence. The knowledge base of the activity recognition system, consists of these domain-dependent rules, as well as the domain-independent axioms of the EC. The input of the system is a sequence of LLE. Probabilistic inference is performed, to recognise HLE.

The remainder of the paper is organised as follows. First, we present a succinct description of the EC in first-order logic. Then in section 3, we show how we can efficiently represent the EC axioms in MLN. In section 4, we explain how the probabilistic nature of MLN affects the EC semantics. In section 5, we demonstrate the benefits of probabilistic modelling, through examples and experiments in the domain of activity recognition. In section 6, we present related work. Finally, we outline directions for further research.

## 2 Event Calculus: A succinct presentation

The Event Calculus (EC), originally introduced by Kowalski and Sergot [12], is a many-sorted first-order predicate calculus for reasoning about events and their effects. A number of different dialects have been proposed and implemented using either logic programming or classical logic — see [20, 14] for a survey. Most EC dialects, however, share the same ontology and core domain-independent axioms. The ontology consists of timepoints, events and fluents. A timepoint represents an instant of time. The underlying time model is often linear and it may represent timepoints as real or integer numbers. A fluent is a property, whose value may change over time. When an event occurs, it may change the value of a fluent. The core domain-independent axioms define whether a fluent holds or not at a specific timepoint. Moreover, the axioms incorporate the common sense *law of inertia*, according to which fluents persist over time, unless they are affected by the occurrence of some event.

In this work we model uncertainty in EC with the use of Markov Logic Networks (MLN), which employ first-order logic as a representation language. As a result, we base our model on an axiomisation of EC in classical first-order logic. As a starting point, we use a subset of the Full Event Calculus, proposed by Shanahan [20]. For simplicity and without loss of generality the predicate *releases* is excluded. This predicate, is domain-dependent and defines under which conditions the *law of inertia* for a fluent is disabled. All fluents, therefore, are subject to inertia at all times. Table 1 summarizes the elements of the EC that we use. Variables (starting with an upper-case letter) are assumed to be universally quantified unless otherwise indicated. Predicates, function symbols and constants start with a lower-case letter. Fluents and events are represented by functions and involve domain objects as variables. In the context of activity recognition, for instance, the event that a person $id_1$ is walking, is represented by the function $walking(id_1)$. The domain of all variables and functions, that is the domain of fluents $\mathcal{F}$, events $\mathcal{E}$ and timepoints $\mathcal{T}$, is finite.

**Table 1.** Event Calculus predicates in classical logic.

| Predicate | Meaning |
|---|---|
| $happens(E, T)$ | Event $E$ occurs at time $T$ |
| $initially_P(F)$ | Fluent $F$ holds from time 0 |
| $initially_N(F)$ | Fluent $F$ does not hold from time 0 |
| $holdsAt(F, T)$ | Fluent $F$ holds at time $T$ |
| $initiates(E, F, T)$ | Event $E$ initiates fluent $F$ at time $T$ |
| $terminates(E, F, T)$ | Event $E$ terminates fluent $F$ at time $T$ |
| $clipped(F, T_0, T_1)$ | Fluent $F$ is terminated some time in the interval $[T_0, T_1]$ |
| $declipped(F, T_0, T_1)$ | Fluent $F$ is initiated some time in the interval $[T_0, T_1]$ |

The axioms that determine when a fluent holds, are defined below:

$$\begin{aligned} holdsAt(F,\ T) \leftarrow\ & \\ & initially_P(F) \wedge \\ & \neg clipped(F, 0, T) \end{aligned} \tag{1}$$

$$\begin{aligned} holdsAt(F,\ T) \leftarrow\ & \\ & happens(E,\ T_0) \wedge \\ & initiates(E,\ F,\ T_0) \wedge \\ & T_0 < T \wedge \\ & \neg clipped(F,\ T_0,\ T) \end{aligned} \tag{2}$$

According to axiom (1), a fluent holds at time $T$ if it held initially and has not been terminated in the interval 0 to $T$. Alternatively, in axiom (2), a fluent holds at time $T$ if it was initiated at some earlier time $T_0$ and has not been terminated between $T_0$ and $T$.

The axioms that determine when a fluent does not hold, are defined below:

$$\begin{aligned} \neg holdsAt(F,\ T) \leftarrow\ & \\ & initially_N(F) \wedge \\ & \neg declipped(F,\ 0,\ T) \end{aligned} \tag{3}$$

$$\begin{aligned} \neg holdsAt(F,\ T) \leftarrow\ & \\ & happens(E,\ T_0) \wedge \\ & terminates(E,\ F,\ T_0) \wedge \\ & T_0 < T \wedge \\ & \neg declipped(F,\ T_0,\ T) \end{aligned} \tag{4}$$

Axiom (3) defines that a fluent does not hold at time $T$ if it did not held initially and has not been initiated in the interval 0 to $T$. Axiom (4) defines that a fluent does not hold at time $T$ if it was terminated earlier at $T_0$ and has not been initiated between $T_0$ and $T$.

The auxiliary domain-independent predicates *clipped* and *declipped*, are defined as follows:

$$\begin{aligned} clipped(F, T_0, T_1) \leftrightarrow\ & \\ \exists\ E,\ T\ \ & happens(E, T) \wedge \\ & T_0 \leq T \wedge T < T_1 \wedge \\ & terminates(E, F, T) \end{aligned} \tag{5}$$

$$
\begin{aligned}
&\mathit{declipped}(F, T_0, T_1) \leftrightarrow \\
&\quad \exists\ E,\ T\ \mathit{happens}(E, T) \wedge \\
&\qquad\qquad T_0 \leq T \wedge T < T_1 \wedge \\
&\qquad\qquad \mathit{initiates}(E, F, T)
\end{aligned} \tag{6}
$$

According to axiom (5), a fluent is clipped when the occurrence of an event terminates the fluent in the interval $T_0$ to $T_1$. In the same manner, axiom (6) defines that a fluent is declipped when the occurrence of an event initiates the fluent in the interval $T_0$ to $T_1$.

## 3 Event Calculus in Markov Logic Networks

Event Calculus (EC) can compactly represent complex event relations, but as a logic-based formalism cannot handle uncertainty. A knowledge base of EC axioms and high-level event (HLE) definitions is defined by a set of first-order logic formulas. Each formula imposes a (hard) constraint over the set of possible worlds, that is, Herbrand interpretations. A missed or an erroneous low-level event (LLE) detection can have a significant effect on the event recognition results.

Markov Logic Networks (MLN) [8] soften these constraints by associating a weight value $w_i$ to each formula $F_i$ in the knowledge base. The higher the value of $w_i$, the stronger the constraint represented by $F_i$. In contrast to classical logic, all worlds in MLN are possible with a certain probability. The main concept behind MLN, is that the probability of a world increases as the number of formulas it violates decreases. A knowledge base in MLN, however, may contain both hard and soft-constrained formulas. Hard-constrained formulas are associated with an infinite weight value and capture the knowledge which is assumed to be certain. Therefore, an acceptable world must at least satisfy these hard constraints. Soft constraints capture imperfect knowledge in the domain, allowing for the existence of worlds where this knowledge is violated. The domain-independent axioms of EC need to be specified as hard constraints, in order to ensure that all acceptable worlds in the set of possible worlds satisfy them.

A knowledge base $L$ of weighted formulas together with a finite domain of constants $C$ can be transformed into a ground Markov network $M_{L,C}$, which defines a probability distribution over possible worlds. All formulas are converted into *clausal form* and each clause is grounded according to the domain of its distinct variables. The nodes in $M_{L,C}$ are Boolean random variables, each one corresponding to a possible grounding of a predicate that appears in $L$. The predicates of a ground clause, form a clique in the $M_{L,C}$. Each clique is associated with the corresponding weight $w_i$ and a Boolean *feature*, taking the value 1 when the ground clause is true and 0 otherwise. More formally, the probability distribution over possible worlds is represented as follows:

$$
P(X = x) = \tfrac{1}{Z} exp\left(\textstyle\sum_i^{|F_c|} w_i n_i(x)\right) \tag{7}
$$

where $x \in \mathcal{X}$ represents a possible world, $F_c$ is the set of clauses, $w_i$ is the weight of the $i$-th clause and $n_i(x)$ is the number of true groundings of the

$i$-th clause in $x$. $Z$ is the partition function used for normalisation, that is, $Z = \sum_{x \in \mathcal{X}} exp(\sum_i^{|F_c|} w_i n_i(x))$, where $\mathcal{X}$ is the set of all possible worlds.

For example, the EC axiom (1) produces one clause and has two distinct variables $F$ and $T$. Therefore, the number of its groundings is defined by the Cartesian product of the corresponding variable-binding constants, that is $\mathcal{F} \times \mathcal{T}$. Assuming that the domain of variable $F$ is relatively small compared to the domain of $T$, the number of groundings of axiom (1) grows linearly to the number of timepoints. Axioms (5) and (6), however, are triply quantified over timepoint variables ($T_0$, $T_1$ and $T$) and therefore, the number of their groundings has a cubic relation to the number of timepoints. In addition, the variables $E$ and $T$ are existentially quantified. During MLN grounding, existentially quantified formulas are replaced by the disjunction of their groundings [8]. This leads to clauses with a large number of disjunctions and a combinatorial explosion of the number of clauses that are generated from axioms (5) and (6). Therefore, representing the presented EC directly in MLN is not practical for real-world event recognition, as its axioms lead to an unmanageably large Markov network.

To eliminate the triply quantified axioms that lead to an explosion of the number of groundings, a discrete version of EC [16] can be used instead. The Discrete Event Calculus (DEC) has been proven to be logically equivalent with EC, when the domain of timepoints is limited to integers [16]. In a similar manner to the EC presented in section 2, we focus on the corresponding domain-independent axioms of DEC. The axioms of DEC utilize a subset of the EC elements (Table 1), that is *happens*, *holdsAt*, *initiates* and *terminates*.

The axioms that determine when a fluent holds, are defined as follows:

$$\begin{aligned} holdsAt(F, T+1) \leftarrow \\ happens(E, T) \wedge \\ initiates(E, F, T) \end{aligned} \tag{8}$$

$$\begin{aligned} holdsAt(F, T+1) \leftarrow \\ holdsAt(F, T) \wedge \\ \neg\exists\ E\ happens(E, T) \wedge \\ teminates(E, F, T) \end{aligned} \tag{9}$$

According to axiom (8), when an event $E$ that initiates a fluent $F$ occurs at time $T$, the fluent holds at the next timepoint. Axiom (9) implements the inertia of fluents, dictating that a fluent continues to hold unless an event terminates it.

The axioms that determine when a fluent does not hold, are defined similarly:

$$\begin{aligned} \neg holdsAt(F, T+1) \leftarrow \\ happens(E, T) \wedge \\ terminates(E, F, T) \end{aligned} \tag{10}$$

$$\begin{aligned} \neg holdsAt(F, T+1) \leftarrow \\ \neg holdsAt(F, T) \wedge \\ \neg\exists\ E\ happens(E, T) \wedge \\ initiates(E, F, T) \end{aligned} \tag{11}$$

Axiom (10) defines that when an event $E$ that terminates a fluent $F$ occurs at time $T$, then the fluent does not hold at the next timepoint. Axiom (11) specifies that a fluent continues not to hold unless an event initiates it.

Compared to EC, DEC axioms are defined over successive timepoints. Additionally, the DEC axioms are quantified over a single timepoint variable. Therefore the number of ground clauses is substantially smaller than EC. Axioms (9) and (11), however, contain the existentially quantified variable $E$. Each of these axioms will be transformed into $2^{|\mathcal{E}|}$ clauses, each producing $\mathcal{F} \times \mathcal{T}$ groundings. Moreover, each ground clause will contain a large number of disjunctions, causing large cliques in the ground Markov network. To overcome the creation of $2^{|\mathcal{E}|}$ clauses, we can employ the technique of subformula renaming [17], as it is used in [16]. According to this technique, the subformula $happens(E,T) \wedge initiates(E,F,T)$ in (11), is replaced by a utility predicate that applies over the same variables, e.g. $startAt(E,F,T)$. A corresponding utility formula, i.e $startAt(E,F,T) \leftrightarrow happens(E,T) \wedge initiates(E,F,T)$, is then added to the knowledge base. With this replacement, the axiom produces a single clause and the utility formula produces three clauses. However, the existential quantification remains in the axiom, causing large cliques in the ground network.

In order to eliminate the existential quantification and reduce further the number of variables, we adopt a similar representation as in [3], where the arguments of initiation and termination predicates are only defined in terms of fluents and timepoints — represented by the predicates *initiatedAt* and *terminatedAt* respectively. As a result, the domain-independent axioms of DEC presented above are universally quantified over fluents and timepoints.

The axioms that determine when a fluent holds are thus defined as follows:

$$holdsAt(F,T+1) \leftarrow initiatedAt(F,T) \tag{12}$$

$$holdsAt(F,T+1) \leftarrow holdsAt(F,T) \wedge \neg teminatedAt(F,T) \tag{13}$$

Axiom (12) defines that when a fluent $F$ is initiated at time $T$, then it holds at the next timepoint. Axiom (13) specifies that a fluent continues to hold unless it is terminated.

The axioms that determine when a fluent does not hold, are defined similarly:

$$\neg holdsAt(F,T+1) \leftarrow terminatedAt(F,T) \tag{14}$$

$$\neg holdsAt(F,T+1) \leftarrow \neg holdsAt(F,T) \wedge \neg initiatedAt(F,T) \tag{15}$$

Axiom (14) defines that when a fluent $F$ is terminated at time $T$ then it does not hold at the next timepoint. According to axiom (15), a fluent continues not to hold unless it is initiated.

The predicates *happens*, *initiatedAt* and *terminatedAt* are defined only in a domain-dependent manner. Specifically, the predicate *happens* provides the input evidence, determining the occurrence of an event at a specific timepoint. The predicates *initiatedAt* and *terminatedAt*, specify under which circumstances a fluent is to be initiated or terminated at a specific timepoint. According to the representation proposed by [3], a domain-dependent rule, e.g. the initiation of a fluent $fluent_1$ over objects $X$ and $Y$, has the following general form:

$$\begin{aligned} &initiatedAt(fluent_1(X,Y),T) \leftarrow \\ &\qquad happens(event_1(X),T) \ \wedge \ ... \ \wedge \\ &\qquad Conditions[X,Y,T] \end{aligned} \tag{16}$$

where $Conditions[X,Y,T]$ is a set of predicates that introduce further constraints in the definition, referring to time $T$ and the domain-dependent objects $X$ and $Y$. The initiation and termination of a fluent can be defined by more than one rule, each capturing a different initiation and termination case.

As an example, consider the following definition of the meeting activity between two persons. The rules represent the conditions under which the HLE *meet* is initiated or terminated.

$$\begin{aligned} &initiatedAt(meet(ID_1,ID_2),T) \leftarrow \\ &\qquad happens(active(ID_1),T) \wedge \\ &\qquad \neg happens(running(ID_2),T) \wedge \\ &\qquad close(ID_1,ID_2,25,T) \end{aligned} \tag{17}$$

$$\begin{aligned} &initiatedAt(meet(ID_1,ID_2),T) \leftarrow \\ &\qquad happens(inactive(ID_1),T) \wedge \\ &\qquad \neg happens(running(ID_2),T) \wedge \\ &\qquad \neg happens(active(ID_2),T) \wedge \\ &\qquad close(ID_1,ID_2,25,T) \end{aligned} \tag{18}$$

$$\begin{aligned} &terminatedAt(meet(ID_1,ID_2),T) \leftarrow \\ &\qquad happens(walking(ID_1),T) \wedge \\ &\qquad \neg close(ID_1,ID_2,34,T) \end{aligned} \tag{19}$$

$$\begin{aligned} &terminatedAt(meet(ID_1,ID_2),T) \leftarrow \\ &\qquad happens(running(ID_1),T) \end{aligned} \tag{20}$$

$$\begin{aligned} &terminatedAt(meet(ID_1,ID_2),T) \leftarrow \\ &\qquad happens(exit(ID_1),T) \end{aligned} \tag{21}$$

Predicate *close* is a preprocessed spatial constraint, stating that the distance between the persons $ID_1$ and $ID_2$ at time $T$ must be below a specified threshold in pixels, e.g. in (17) the threshold is 25 pixels.

## 4 The Law of Inertia in Probabilistic Event Calculus

A knowledge base with domain-dependent rules in the form of (16), describes explicitly the conditions under which fluents are initiated or terminated. It is usually impractical to define also when a fluent is *not* initiated and *not* terminated.

However, the open-world semantics of first-order logic result to an inherent uncertainty about the value of the fluent for many timepoints. In other words, if at a specific timepoint no event that terminates or initiates a fluent happens, we cannot rule out the possibility that the fluent has been initiated or terminated. As a result, it cannot be determined whether a fluent holds or not, causing the loss of the inertia.

This is also known as the frame problem and one solution for EC and DEC in classical logic is the use of circumscription [13, 19, 7]. The aim of circumscription, is to automatically rule out all those conditions which are not explicitly entailed by the given formulas. Hence, circumscription introduces a closed world assumption to first-order logic. We perform circumscription by predicate completion, as in [19, 16]. Technically, predicate completion is a syntactic transformation, in which formulas are translated into logically stronger ones.

As an example, consider a knowledge base of domain-dependent rules in the form of (16), containing, among others, the definition of HLE *meet* (17) - (21). The application of circumscription over the predicates *initiatedAt* and *terminatedAt* will transform all domain-dependent rules into the following form:

$$
\begin{aligned}
&initiatedAt(F,T) \leftrightarrow \\
&\qquad \exists\ ID_1,\ ID_2\ (F = meet(ID_1, ID_2) \wedge \\
&\qquad happens(active(ID_1), T) \wedge \\
&\qquad \neg happens(running(ID_2), T) \wedge \\
&\qquad close(ID_1, ID_2, 25, T)\ ) \vee \\
&\qquad \exists\ ID_1,\ ID_2\ (F = meet(ID_1, ID_2) \wedge \\
&\qquad happens(inactive(ID_1), T) \wedge \\
&\qquad \neg happens(running(ID_2), T) \wedge \\
&\qquad \neg happens(active(ID_2), T) \wedge \\
&\qquad close(ID_1, ID_2, 25, T)\ ) \vee \\
&\qquad ... \qquad (22) \\
&terminatedAt(F,T) \leftrightarrow \\
&\qquad \exists\ ID_1,\ ID_2\ (F = meet(ID_1, ID_2) \wedge \\
&\qquad happens(walking(ID_1), T) \wedge \\
&\qquad \neg close(ID_1, ID_2, 25, T)\ ) \vee \\
&\qquad \exists\ ID_1,\ ID_2\ (F = meet(ID_1, ID_2) \wedge \\
&\qquad happens(running(ID_1), T)\ ) \vee \\
&\qquad \exists\ ID_1,\ ID_2\ (F = meet(ID_1, ID_2) \wedge \\
&\qquad happens(exit(ID_1), T)\ ) \vee \\
&\qquad ...
\end{aligned}
$$

The resulting rules (22), define explicitly the unique condition, under which each fluent is initiated or terminated. Any other event occurrence cannot affect any fluent, as it is impossible to cause any initiation or termination. However, the presence of existentially quantified variables causes combinatorial explosion to the number of grounded clauses, as explained above.

To address this problem we make the assumption that every fluent of interest is defined in terms of at least one initiation and one termination rule. Addition-

ally, we assume that the variables that appear in the head of the *initiatedAt* and *terminatedAt* rules are the only variables in these rules. Therefore, each domain-dependent rule is implicitly universally quantified over these variables. These assumptions are reasonable in event recognition applications. The assumptions allow to compute the circumscription for each fluent separately, rather than computing the circumscription of the entire knowledge base over the predicates *initiatedAt* and *terminatedAt*. Furthermore, the knowledge base is enriched with additional formulas.

For example, the domain-dependent rules about the initiation of *meet* (rules (17) and (18)) are translated into the following form:

$$\Sigma = \begin{cases} initiatedAt(meet(ID_1, ID_2), T) \leftarrow \\ \qquad happens(active(ID_1), T) \wedge \\ \qquad \neg happens(running(ID_2), T) \wedge \\ \qquad close(ID_1, ID_2, 25, T) \\ initiatedAt(meet(ID_1, ID_2), T) \leftarrow \\ \qquad happens(inactive(ID_1), T) \wedge \\ \qquad \neg happens(running(ID_2), T) \wedge \\ \qquad \neg happens(active(ID_2), T) \wedge \\ \qquad close(ID_1, ID_2, 25, T) \end{cases} \tag{23}$$

$$\Sigma' = \begin{cases} initiatedAt(meet(ID_1, ID_2), T) \rightarrow \\ \qquad (happens(active(ID_1), T) \wedge \\ \qquad \neg happens(running(ID_2), T) \wedge \\ \qquad close(ID_1, ID_2, 25, T)\ ) \vee \\ \qquad (happens(inactive(ID_1), T) \wedge \\ \qquad \neg happens(running(ID_2), T) \wedge \\ \qquad \neg happens(active(ID_2), T) \wedge \\ \qquad close(ID_1, ID_2, 25, T)\ ) \end{cases}$$

Compared to the rules in (22), the rules in (23) are simpler, as they do not involve any existentially quantified variable. Compared to (17) - (21), the axioms in (23) introduce additional formulas, indicated by $\Sigma'$, which eliminate the possibility that worlds not described by the original knowledge base can satisfy the theory.

By assigning a weight to a formula in MLN it automatically becomes a soft constraint, allowing some worlds that do not satisfy this formula to become likely. This is desirable in event recognition, in order to allow for imperfect HLE definitions. In the presence of soft constraints, however, the behaviour of circumscribed formulas changes. To illustrate this, consider that a knowledge base of domain-dependent rules, in the form of (23), is separated into a set $\Sigma$ of the original rules and a set of additional formulas $\Sigma'$ that result from circumscription. By treating the formulas in these sets as either hard or soft constraints, we may distinguish the following four general cases:

1. The formulas in both sets are hard-constrained. This will produce the same results as crisp logic and there are no differences or benefits to be gained.

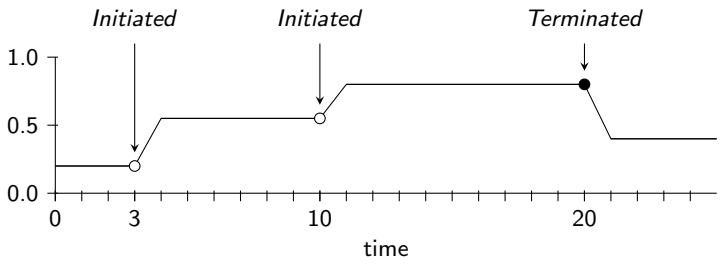


**Fig. 1.** The probability of *meet*

2. Only the formulas in $\Sigma$ are soft-constrained and thus worlds that violate these constraints become probable. This situation reduces the certainty of a fluent being initiated or terminated, when all the required conditions are met. As a result, fluents hold with some probability, instead of absolute certainty. Moreover, given a fluent that holds with some probability, when the initiating conditions are met the probability increases. Similarly, when the terminating conditions are satisfied, the probability that a fluent holds decreases. At the same time, the worlds that violate the formulas in $\Sigma'$ are rejected and therefore the inertia is retained. Consider, for example, that the fluent *meet* holds at time 0 with some probability. At time 3, the fluent is initiated by (18). Thereafter at time 10 it is initiated again by (17) and finally at time 20 it is terminated by (19). The probability of *meet* will increase at time 4. Since the inertia is retained, the probability of *meet* will persist in the interval 4 to 10. Similarly, at time 11, the probability of *meet* will increase again and persist until time 20. Thereafter, the probability that *meet* holds will decrease (see figure 1).
3. Only formulas in $\Sigma'$ are soft-constrained. A fluent is initiated or terminated with certainty when the corresponding conditions of the rules in $\Sigma$ are satisfied by the evidence. The formulas imposed by circumscription, however, can be violated. Therefore, the closed-world assumption is softened and the initiation or the termination of a fluent when irrelevant events happen becomes likely. The lower the value of the weight on the constraint, the more probable worlds that violate the constraint become. As a result, the value of the weight affects the persistence of inertia. In other words, strong weight values in $\Sigma'$ cause the inertia to persist for longer time periods than weak ones. Despite that the fluent in this case is initiated or terminated with certainty, the softened formulas in $\Sigma'$ causes the fluent to hold with some probability.
4. All formulas are soft-constrained in both $\Sigma$ and $\Sigma'$ sets. Fluents will be initiated or terminated with some probability, as in the second case. Also the persistence of inertia is controlled by the weight of the formulas in $\Sigma'$, as in the third case.

In (23) we could have chosen a more compact representation, using equivalence, as commonly done in circumscription. However, the expanded form of the rules allows us to control separately our confidence level for each domain-dependent rule and the inertia of each fluent.

## 5 Application to Activity Recognition

To demonstrate our method (DEC-MLN)[3], we present examples and experiments[4] from the domain of video activity recognition, using the publicly available dataset of the CAVIAR project[5]. The dataset comprises 28 surveillance videos, where each frame is annotated by human experts, providing two levels of activity information. The first level contains the low-level event (LLE) annotation for individual objects or persons, using the tags *active, inactive, walking* and *running*. The second level contains the high-level events (HLE) between people and objects, using the tags *meeting, moving, fighting* and *leaving an object*. The former level provides the input LLE for our approach, while the latter the ground truth HLE. The aim of the experiments is to recognise HLE that occur among people and objects, by providing a sequence of LLE as evidence. In EC terminology, events and fluents correspond to LLE and HLE, respectively.

For comparison purposes, we use as a baseline method the activity recognition method proposed in [4] (EC-LP). EC-LP implements EC using logic programming and contains a knowledge base of HLE definitions for the CAVIAR dataset. The experiments are performed using the same HLE definitions as EC-LP, translated into first-order logic syntax using the formulation proposed in section 3 (e.g. formulas (17) - (21)) and computing the circumscription as presented in section 4. Details about the activity recognition application and a description of the HLE definitions, can be found in [4].

The input to both DEC-MLN and EC-LP consists of a sequence of LLE, along with their timestamps. Additionally, the first and the last time that a person or an object is tracked is provided as the LLE *enter* and *exit*. The input to EC-LP contains also the coordinates of the people tracked at each time-point. In DEC-MLN, the value of the *close* predicates is precomputed.

The output of both methods consists of a sequence of ground *holdsAt* predicates, indicating which HLE are recognised. EC-LP performs crisp reasoning, and thus all HLE are recognised with absolute certainty. On the other hand, DEC-MLN performs conditional probabilistic inference. Consequently, all recognised HLE are associated with a probability. In the following experiments, we consider any result with probability above 0.5 as a recognised HLE.

In the first experiment, we wanted to confirm that our method behaves like a crisp EC method, if required. For this purpose, we assigned the same strong weight value (high confidence) to each HLE definition in $\Sigma$ and hard-constrained the resulting formulas of circumscription in $\Sigma'$. As expected, DEC-MLN produced exactly the same results as EC-LP in this experiment.

In the second experiment, we demonstrate the effect of soft constraints on event recognition. For this purpose, we adjusted the weight values for the HLE *meet* and studied two cases. The first case (DEC-MLN$_a$) demonstrates the benefits of having soft-constrained domain-dependent definitions only in $\Sigma$. The

[3] The HLE definitions of the method can be found in: `http://www.iit.demokritos.gr/~anskarl`

[4] For our experiments we used the open-source MLN framework Alchemy, which can be found in: `http://alchemy.cs.washington.edu`

[5] The CAVIAR dataset can be found in: `http://homepages.inf.ed.ac.uk/rbf/CAVIARDATA1`

**Table 2.** Results for HLE "*meet*" using soft constraints. Number of True Positives (TP), False Positives (FP), False Negatives (FN), Precision and Recall rates are measured per frame.

| Method | TP | FP | FN | Precision | Recall |
|---|---|---|---|---|---|
| EC-LP | 3099 | 2258 | 525 | 0.578 | 0.855 |
| DEC-MLN$_a$ | 3048 | 1762 | 576 | 0.633 | 0.841 |
| DEC-MLN$_b$ | 3048 | 1154 | 576 | 0.725 | 0.841 |

second case (DEC-MLN$_b$) demonstrates a potential use of soft-constrained circumscription rules in $\Sigma'$, in addition to the soft-constrained rules of DEC-MLN$_a$. The evaluation results are presented in Table 2.

In the DEC-MLN$_a$ case, each initiation and termination rule in (17) - (21) is associated with a weight value that indicates its confidence. More specifically, the HLE *meet* is rarely initiated rule (18) and therefore this rule is assigned a weak weight value, indicating low confidence. On the other hand, the initiation rule (17), as well as the termination rules (19) - (21), are assigned weight values that indicate high confidence, as they are tightly associated with the HLE. The additional formulas that result from circumscription are hard-constrained, in order to fully retain the inertia. Compared to EC-LP, the low confidence value in rule (18) reduces significantly the number of false positives. The cost is a small loss of true positives, as can be seen in Table 2. As a result, precision is improved by 5.5 percentage points, while recall falls by 1.4 points, without any effect on the recognition of other HLE.

As noted in [4], the definitions of HLE *meet* and *move*, share the same termination constraints in the knowledge base. As a result, the HLE *meet* and *move* that are detected by EC-LP may overlap. According to the HLE annotation, however, *meet* and *move* do not happen concurrently. Consider, for example, a situation where two people meet for a while and thereafter they move together. During the interval where *move* is detected, *meet* will also remain detected, as it is not terminated and the *law of inertia* holds. However, there are no LLE that initiate *meet* in this interval and its probability is not reinforced. By softening the circumscription formulas for terminating *meet* in $\Sigma'$, worlds not satisfying these rules will become likely. As a result, when there is no further evidence from LLE that initiates *meet*, e.g. when *move* starts in the above example, the detection of *meet* will gradually become less likely as desired. As a side effect, this change reduces the detection probability of *meet* in cases where *meet* is initiated. To overcome this issue, we increased the weight values of the initiation rules in $\Sigma$. In summary, in the DEC-MLN$_b$ case the circumscription of termination rules in $\Sigma'$ for HLE *meet* are soft-constrained, while the circumscription of initiation rules remain hard-constrained and the weights of the initiation rules in $\Sigma$ are soft-constrained. Compared to EC-LP, the number of false positives is further reduced, increasing the precision rate by 9.2 percentage points, without any loss of recall or any effect on the recognition of other HLE.

The two cases presented here, DEC-MLN$_a$ and DEC-MLN$_b$, illustrate the benefits to be gained by softening the constraints and performing probabilistic logical reasoning in event recognition.

## 6 Related work

Symbolic methods can naturally and compactly represent high-level event (HLE) definitions for event recognition and model complex event relations, such as concurrency and persistency — see [2] for a list of applications. The chronicle recognition system [9], for example, is a symbolic event recognition method that can efficiently recognise HLE. Event Calculus (EC) is another logic-based formalism that has been recently applied to event recognition [4, 3]. The formal declarative semantics of symbolic methods, allow the compact representation of structured HLE, as well as the integration of background domain knowledge that helps to improve the event recognition performance. However, symbolic methods cannot handle uncertainty, which naturally exists in many real-world applications and may seriously compromise the event recognition results. In our work, we combine EC and Markov Logic Networks (MLN) in a method that supports the definition of HLE in EC and performs probabilistic event recognition with MLN. Unlike crisp-logic EC [4, 3], our method allows to control the level of the persistency of fluents. As noted in section 5, we have used the same HLE definitions as in [4], preprocessed appropriately to fit the MLN representation and computation.

Probabilistic graphical models, such as Hidden Markov Models and Dynamic Bayesian Networks have been successfully applied to event recognition in a variety of applications where uncertainty exists (e.g. [6, 22]). Compared to symbolic methods, such models can naturally handle uncertainty but their propositional structure provides limited representation capabilities. To model HLE that involve a large number of domain objects (e.g. interactions between multiple persons), the structure of the model may become prohibitively large and complex. The lack of a formal representation language makes the definition of such HLE complicated and the integration of domain background knowledge is very hard.

A logic-based method that handles uncertainty is presented in [21]. The method incorporates rules that express HLE in terms of input low-level events (LLE). Each HLE or LLE is associated with two uncertainty values, indicating a degree of information and confidence respectively. The underlying idea of the method is that the more confident information is provided, the stronger the belief about the corresponding HLE becomes. Accordingly, in our work, the more initiations (or terminations) we have, the higher (or lower) the probability that the corresponding HLE holds. In contrast to that method, our work employs MLN that provide formal probabilistic semantics, as well as EC to represent complex HLE.

Recently, MLN have also been used for event recognition. The method in [5], uses MLN to combine the information from low-level classifiers, in order to recognise HLE. A more expressive method that can represent persistent and concurrent HLE, as well as their starting and ending points, is proposed in [10]. However, that method has a quadratic complexity to the number of timepoints. Also, the methods in [5] and [10] focus on HLE that do not involve relations among multiple domain objects. Additionally, they cannot handle situations where nothing is happening, as their axioms require that at each timepoint at

least one HLE must occur. Due to those limitations, these methods are difficult to scale up in real-world event recognition applications.

In [23, 11] a knowledge base of common sense rules, expressing HLE, is defined in first-order logic. Each rule is associated with a weight value that indicates its confidence. Additionally, the method takes into account the confidence value of the input LLE, which may be due to noisy sensors. Probabilistic inference is performed by MLN, in order to recognise the HLE. Although, the method represents HLE that involve relations among multiple domain objects, the HLE definitions have a limited temporal representation. A more expressive method that uses interval-based temporal relations, is proposed in [15]. The aim of the method is to determine the most consistent sequence of HLE, based on the observations of low-level classifiers. Similar to [23, 11], the method expresses HLE using common sense rules. However, it employs temporal relations that are based on Allen's Interval Algebra (IA) [1]. In order to avoid the combinatorial explosion of possible intervals that IA may produce, as well as to eliminate the existential quantifiers in HLE definitions, a bottom-up process eliminates the unlikely HLE hypotheses. That process can only be applied to domain-dependent axioms, as it is guided from the observations and the IA relations. In our work, we address the combinatorial explosion problem in a more generic manner, by representing the EC domain-independent axioms efficiently.

## 7 Conclusions

In this work, we address the issue of uncertainty that naturally exists in many levels of event recognition, such as the input LLE and the imprecise HLE definitions. We propose a probabilistic extension of Event Calculus (EC) based on Markov Logic Networks (MLN). The method has formal, declarative semantics and inherits the properties of the EC. The domain-independent axioms of EC are hard-constrained, while the domain-dependent HLE definitions can be associated with a confidence level. Moreover, by exploiting the probabilistic nature of MLN, we show how the persistency of fluents can be controlled. We place emphasis on the efficiency and effectiveness of our approach to meet the requirements of real-world applications, by simplifying the axioms of the EC and therefore reducing the size of the underlying ground network built by MLN.

Due to the use of MLN, our method lends itself naturally to learning the weights of event definitions from data. We believe this is an important next step, as the manual setting of weights is suboptimal and cumbersome. Furthermore, we plan to perform additional experiments with other real-world datasets, in order to demonstrate further the potential of our method.

**Acknowledgements** This work has been partially funded by EU, in the context of the PRONTO project (FP7-ICT 231738).

## References

1. Allen, J.F.: Maintaining knowledge about temporal intervals. Commun. ACM 26(11), 832–843 (1983)

2. Artikis, A., Paliouras, G., Portet, F., Skarlatidis, A.: Logic-based representation, reasoning and machine learning for event recognition. In: DEBS. pp. 282–293 (2010c)
3. Artikis, A., Sergot, M., Paliouras, G.: A logic programming approach to activity recognition. In: ACM Workshop on Events in Multimedia (2010b)
4. Artikis, A., Skarlatidis, A., Paliouras, G.: Behaviour recognition from video content: a logic programming approach. IJAIT 19(2), 193–209 (2010a)
5. Biswas, R., Thrun, S., Fujimura, K.: Recognizing activities with multiple cues. In: Elgammal, A.M., Rosenhahn, B., Klette, R. (eds.) Workshop on Human Motion. Lecture Notes in Computer Science, vol. 4814, pp. 255–270. Springer (2007)
6. Brand, M., Oliver, N., Pentland, A.: Coupled hidden markov models for complex action recognition. In: CVPR. pp. 994–999. IEEE Computer Society (1997)
7. Doherty, P., Lukaszewicz, W., Szalas, A.: Computing circumscription revisited: A reduction algorithm. J. Autom. Reasoning 18(3), 297–336 (1997)
8. Domingos, P., Lowd, D.: Markov Logic: An Interface Layer for Artificial Intelligence. Morgan & Claypool Publishers (2009)
9. Dousson, C., Maigat, P.L.: Chronicle recognition improvement using temporal focusing and hierarchization. In: Veloso, M.M. (ed.) IJCAI. pp. 324–329 (2007)
10. Helaoui, R., Niepert, M., Stuckenschmidt, H.: Recognizing interleaved and concurrent activities: A statistical-relational approach. In: PerCom. pp. 1–9. IEEE (2011)
11. Kembhavi, A., Yeh, T., Davis, L.S.: Why did the person cross the road (there)? scene understanding using probabilistic logic models and common sense reasoning. In: ECCV (2). pp. 693–706 (2010)
12. Kowalski, R., Sergot, M.: A logic-based calculus of events. New Generation Computing 4, 67–95 (1986)
13. McCarthy, J.: Circumscription - a form of non-monotonic reasoning. Artificial Intelligence 13, 27–39 (1980)
14. Miller, R., Shanahan, M.: Some alternative formulations of the event calculus. In: Computational Logic: Logic Programming and Beyond. LNCS, vol. 2408, pp. 452–490 (2002)
15. Morariu, V.I., Davis, L.S.: Multi-agent event recognition in structured scenarios. In: Computer Vision and Pattern Recognition (CVPR)
16. Mueller, E.T.: Event calculus. In: Handbook of Knowledge Representation, FAI, vol. 3, pp. 671–708 (2008)
17. Nonnengart, A., Weidenbach, C.: Computing small clause normal forms. Handbook of automated reasoning 1, 335–367 (2001)
18. de Salvo Braz, R., Amir, E., Roth, D.: A survey of first-order probabilistic models. In: Innovations in Bayesian Networks, pp. 289–317. SCI (2008)
19. Shanahan, M.: Solving the frame problem: a mathematical investigation of the common sense law of inertia. MIT Press (1997)
20. Shanahan, M.: The event calculus explained. Artificial intelligence today: Recent trends and developments pp. 409–430 (1999)
21. Shet, V.D., Neumann, J., Ramesh, V., Davis, L.S.: Bilattice-based logical reasoning for human detection. In: CVPR (2007)
22. Shi, Y., Bobick, A.F., Essa, I.A.: Learning temporal sequence model from partially labeled data. In: CVPR (2). pp. 1631–1638. IEEE Computer Society (2006)
23. Tran, S.D., Davis, L.S.: Event modeling and recognition using markov logic networks. In: ECCV (2). pp. 610–623 (2008)