The temporal representation and reasoning of complex events

Francesco Mele[1], Antonio Sorgente[1],

[1] Istituto di Cibernetica, Consiglio Nazionale delle Ricerche,
Via Campi Flegrei, 34 Pozzuoli(Naples) Italy
{f.mele, a.sorgente}@cib.na.cnr.it

Abstract. This paper introduces a formalization of complex events. In particular, a formalism is presented to represent intentional and causal events in narrative contexts, and in their mechanisms of composition. Complex events have been defined through classes of a formal ontology that has been called the Ontology of Complex Events (OntoCE). This approach has allowed for the applications (reuse) of existing axiomatizations belonging to a large repertoire of temporal reasoning techniques and, the definition of new axiomatizations presented and discussed in this work. The focus in this work has been placed on three particular temporal aspects: the analysis of consistency, the discovery of new temporal relations in a knowledge base of events, and the causal reasoning in narrative contexts.

1 Introduction

The concept of event has been highly examined and much debated in philosophy [CAS, DAV] and Artificial Intelligence. In this area, some well-founded formalisms, like the Event Calculus [MIL], the Situation Calculus [LEV, LIN], and ALAN [BAR, Gon] have been proposed. Recently, a new point of attention, that regards the concept of the "complex event" [WIN], has been born (although this name is not explicitly mentioned by all the research projects that deal with these issues).

This concept emerges, particularly in the context of the Internet, where the broad set of information in unstructured form, hides a multitude of events that are connected by relationships extremely difficult to detect, but where one feels that these events are components of an implicit totality (suggested without being directly expressed).

The particular aim of this research was to build a model and a formalism to represent three main types of complex events (intentional events, causal events, and narrative events) and their mechanisms of composition.

The paper introduces a representation of complex events, in which an event is not only an aggregation of simple events (how it would be in the case of a narrative of events, consisting of a set of simple events and a set of relationships between those events). The modeling of narratives that have as components other complex events has been addressed. For example, casual events are considered such as: (the church of Santa Chiara was built (e1) for desire (e2) of Roberto D'Angiò) (e0), where e0 is composed by the events e1 and e2 and could also represent the component of a narrative. A mechanism for determining the interval in which a complex event (intensional, narrative or causal) occurs has also been proposed. Such intervals have been calculated considering the intervals of happening of its component events (e0): in the example the event e0 has an occurrence interval that is calculated as union of occurrence intervals of events e1 and e2.

By inserting intentional events, in representation of a narrative, one can not only annotate or discover causal connections between events, but with appropriate axioms (eg *(e1 cause e2) implies (e1 precedes e2)*) one can convert the causal relations into temporal order relations, thus eliminating the existing deficiencies of connectivity among the events of a narrative.

A formalism has been constructed to represent the complex events in explicit form, with the main motivation that such a representation can be used as an ontological reference for various types of semantic annotations, in particular:

- to aggregate, as complex events, multimedia elements (photos, video or texts whose contents represent events (historical events, news, cultural events, etc.) in the same way as proposed in the Event-Centric in [GIU] and [MEL]; and,
- to annotate and aggregate complex events in natural language, starting from annotations represented by TimeML [PUS03, PUS08] formalism.

An annotation process of natural language texts or media, especially if this is done through a process of multiple annotation (by more than one operator), can easily generate some inconsistencies or lack of connections between events in the bases of annotations. For these reasons, it is necessary to identify inconsistencies and non-connected events in order to remove such anomalies among the annotated events (see Fig. 1).

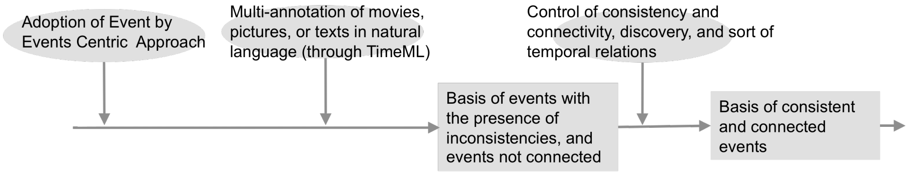
Adoption of Event by Events Centric Approach
Multi-annotation of movies, pictures, or texts in natural language (through TimeML)
Control of consistency and connectivity, discovery, and sort of temporal relations
Basis of events with the presence of inconsistencies, and events not connected
Basis of consistent and connected events

Fig. 1. Phases to control the consistency and connectivity of a narrative

The formalism (OntoCE) that has been defined in this paper is represented by a formal ontologies, where each entity has been defined as a class of an ontology. This methodological approach was chosen to facilitate the building of modules (algorithms) for the discovery of temporal relations, with the objective also, to reuse existing axiomatizations and facilitate the creation of new ones (some of these proposals are in this paper).

In this paper the logic programs have been used to represent the events (simple and complex) and their relations, to analyse the temporal consistency of a complex event, to discover new temporal relations between events, to apply causal reasoning to events, and to integrate the latter with axioms of temporal reasoning.

Related work

In recent research [WIN] there are several proposals for representing events. The basic motivation of this research stands from the claim that events can constitute an excellent framework for aggregating knowledge. The large quantity of data and (fragmented and unstructured) knowledge on the Internet, makes this research very attractive. An emerging methodology for representing through events knowledge distributed on the Internet has been named Event-Centric [WIN]. In this methodology, an event is a structure of reference that is independent from the metadata of media that one intends to annotate. An example of the Event-Centric approach, which uses high-level ontology DOLCE, is the F model [SHE]. In the F model the methodological choices are motivated by a number of functional and non-functional requirements.

With respect to the functional requirements, the representation of an event must have the attribute for the participants. It is also necessary to be able to define relationships between parts and wholes of an event. It must be possible to define cause-effect relationships between two events (no matter of the degree of difficulty of the automatic process discovery), and, finally, it must be possible to represent correlation relationships between events (two events that have a common cause).

The non-functional requirements of F, instead, include extensibility, formal precision (axiomatization), modularity, and reusability.

Among the proposed Event-Centric is the one proposed in [GIU]. This methodology adopts the slogan "Aggregation via Media Events". In this proposal, the events are the reference structures for aggregating the media. In [GIU] an implicit model of complex events (without explicit constructs of representation) and a simple mechanism to determine the "where" a complex event happens, starting from the "where" of the components' events, are introduced.

To represent the events, some formalisms were inspired from a model that has its roots in journalism. This model called "W's and one H" adopts six attributes for the representation of events: Who, When, Where, What, Why, and How. The project Eventory [WAN] adopts a model "W's and one H". Eventory has a particular structure of the "When" attribute, having two references for time: the first referring to the chronological time of "real events", the second, to the temporal attributes of some metadata (such as the length of a movie or the time during which a picture must be shown).

The decision to include the knowledge of the media in the attributes of the events, violates the constraint that characterizes the Event-Centric models, whereby the independency between the event representation and that of media is fixed. In fact, in the case of Eventory, the description contains information about the execution time of the media.

In this work, in relation to the representation of temporal intervals, the classification given in [MAJ], where all the possible combinations that exist between instants or time intervals are shown, when they represent a temporal relationship between two events, has been taken into consideration.

2 The representation of the events

In this work, an ontology for complex events has been defined: OntoCE. OntoCE has an abstract superclass (`AnythingInTime`) common to all entities that happen over time. Two subclasses are specializations of `AnythingInTime`: `Event`, that represents the class of simple events, and `ComplexEvent`, that represents the class of complex events. In Fig. 2 a sketch representation is given.

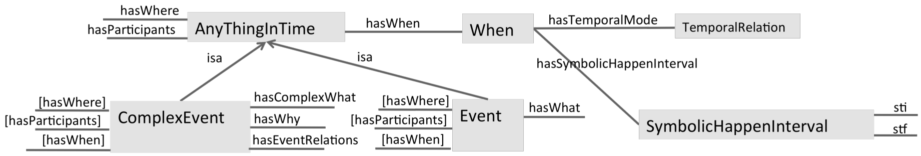

Fig. 2. The taxonomy of simple and complex events

In Fig.2, in brackets, the attributes that are inherited from their respective superclasses are reported. Formally, they are represented in Flora2[1][FLO](this formalism combines the advantages of conceptual modeling with object-oriented, owns a declarative syntax, allowing to build complex inferential apparatus in simple manner):

```
Event::AnyThingInTime.
ComplexEvent::AnyThingInTime.
AnyThingInTime[
   hasWhen*=>When, hasWhere*=>Where,
   hasParticipants*=>Participant].
Event[hasWhat*=> Action_Property].
ComplexEvent[     hasComplexWhat*=> AnyThingInTime,

      hasEventRelations*=>EventRelation].
```

`AnyThingInTime` is an abstract class (without instances) which is the superclass of the concrete classes: `Event` and `ComplexEvent`. The latter classes are the key concepts of the formalism OntoCE. The `Event` class has the descriptor `hasWhat`, which is associated to the class What. Generally, this class describes the action (which happens over time) that characterizes the event or describes a property that is true in a specific time interval. The attribute `hasComplexWhat` is a specific descriptor of `ComplexEvent`. The latter also has the attribute `hasWhy` that describes the causal relations between events. `hasWhy` is a relation between two events, so it can only be the attribute of a complex event. Attributes `hasWhen`, `hasWhere`, `hasParticipants`, `hasWhat` (or `hasComplexWhat`), and `hasWhy` correspond to the descriptors with which journalists describe their articles.

[1] To make reading simpler, some key constructs of Flora2 language are here reported. X:: Y (class X is a subclass of class Y), X: Y (X is an instance of class Y), X => Y (X is an attribute of type Y), X-> Y (Y is the value of X), X *=> Y (as X => Y, but the attribute is inherited by subclasses). In Flora2, chain of alphanumeric literal, starting with a capital letter are variables. The symbols ":-", the comma (",") and semicolon (";") have the same interpretation as the homologue constructs of Prolog language.

2.1 Instant and interval representations

The representation of time that has been adopted is mixed and based on points and time intervals. In OntoCE all temporal entities are represented as classes. `Time` is the main class and has several specializations: date or partial dates (`DateValue`), time instants or combinations of them with date (`TimeValue`), symbolic times (`Symbolic`), and time intervals (`Interval`). The definition is as follows:

```
DateCalendar[
  day*=>Integer, month*=>Integer, year*=>Integer].
DateWeek[week*=>Integer, year*=>Integer].
DateQualitative[value*=>String].
Clock[
  hour*=>Integer,minute*=>Integer,date*=>DateCalendar].
TimeQualitative[value*=>String].
Interval[
  hasBeginTime*=>(DateValue;TimeValue;SymbolicTime),
  hasEndTime*=>(DateValue; TimeValue; SymbolicTime)].
SymbolicHappenInterval[
  sti*=>SymbolicTime,stf*=> SymbolicTime].
```

2.2 The When class

In OntoCE there is a particular structure, the class `When` (see Fig. 2), which describes when an event happens using the effective symbolic interval (ESI) in which the event happens, and a temporal modality of happening, described by one (or more) temporal order relations (before, after, during, etc.) between ESI and some temporal interval of reference (or also another event). These relations have the objective of anchoring an event on the chronological axis or with another event, through a temporal order relation. The approach requires, therefore, that when an event (simple or complex) is created, it automatically generates a type identifier ESI, represented by two attributes: `sti`, the (effective) symbolic time in which the event starts (or in which the property is true), and `stf`, the time in which the event ends. The choice of having an effective time when an event happens and a temporal modality of happening, for it is motivated by the fact that often the effective time in which an event happens is not known, but one can easily know one or more relations (modality of happening) for it (after a time tx, dtx before a certain range, etc.). Thus, even if one does not exactly know the exact value of the start and end/or of an event, one can annotate (or, automatically discover) relationships with other time intervals, as soon as they become available. The class `When` has the following definition:

```
When[
  hasSymbolicHappenInterval*=>SymbolicHappenInterval,
  hasTemporalMode*=>TemporalRelation].
SymbolicHappenInterval[sti*=> SymbolicTime,
                       stf*=> SymbolicTime].
```

2.3 Simple Events

In OntoCE, the class `Event` represents simple events. This class inherits the attributes of the superclass `AnyThingInTime` and has the attribute `hasWhat`, which describes an action that happens or a property that is true in a temporal interval.

```
Event[hasWhat*=>Action-Property].
```

The attribute `hasWhat` has values in the `Property_Action` domain and describes exactly what happens (action) or what is true (property) in a temporal interval.

2.4 Complex events

`ComplexEvent` is a class defined as a set of events (simple or complex) described by `ComplexWhat` slot, and a set of relationships between events described by `hasEventRelations` slot. Also, the class is described by the method `hasWhy(AnyThingInThing)`, a function that given an input event belonging to a complex event, returns a set of causal relations that are the justification of why the event occurred.

```
ComplexEvent[ hasComplexWhat{2:*}*=>AnythingInTime,
              hasWhy(AnyThingInThing)*=> CausalRelation,
              hasEventRelations*=>EventRelation].
```

Complex events are defined by a temporal mode, described by the descriptor `hasWhen`, the same attribute used for the description of simple events. Fig. 3 shows the taxonomy of the complex events of the OntoCE ontology, where the narrative events, the causal events, the intentional events, and the perceptual events have been labeled and represented as complex events.

Narrative Events

The complex narrative events (`NarrativeEvent`) are represented as a set of events and temporal relations between events. The components of `NarrativeEvent` are simple events that describe actions that occur over time, or properties that are true in a temporal interval, or other complex events such as causal events (`CausalEvent`) or intentional events (`MentalEvent`).

`NarrativeEvent`, like all subclasses of `AnyThingInTime` inherits the attributes When, Where, and Who, and like all the subclasses of `ComplexEvent`, inherits the method `hasWhy(AnyThingInThing)`. The characterization of `NarrativeEvent` is given by the restriction of the slot `hasEventRelations`, which can only have instances of temporal relations as a value.

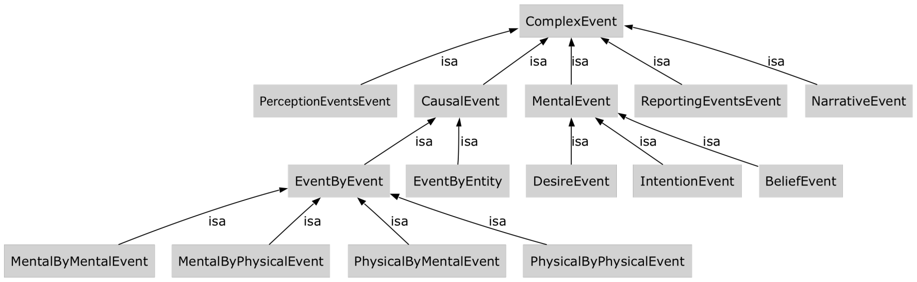

Fig. 3 The taxonomy (subclasses) of complex events.

Mental events

The class `MentalEvent` represents mental events of an agent (participant at the event), i.e., beliefs, desires, and intentions that occur over time. These entities allow for the representation as a causal events such as "*The church and monastery of Santa Chiara was built between 1310 and 1340 for desire of Roberto d'Angiò and Queen Sancha of Aragon*". Mental events are defined, through to the attributes When, What, Where, and Participants (inherited by `AnyThingInTime)` by a slot that describes a relation (`MentalRelation`) between a mental event and a physical event.

Causal events

The complex event `CausalEvent` describes events that relate to a cause-effect relation: the occurrence of an event (event cause) caused the occurrence of another event (event effect). In OntoCE a classification of causal events has been defined in accordance with the nature of the events involved, or that is, if the cause-effect relationship is defined by physical and/or mental events. We report some examples of the categories:

- *"I think it's a good book, I'll buy it"*, and "*I would like something hot, I'll take a cup of tea"*, are examples of causal relations of a mental event that causes a physical event (classified in OntoCE as a causal `PhysicalByMentalEvent`);
- "*He laughed and I thought he was joking*" is an example of causal event where a physical event (perception) caused a mental event (in OntoCE labeled as `MentalEventByPhysical`);
- "*He bumped the glass with his elbow and broke it*", "*It's raining and the road is wet*" (causal events labeled in OntoCE as `PhysicalEventByPhysical`); and,
- "*I think it's the best team and I think it will win the championship*" (causal events labeled in OntoCE as `MentalEventByMental`).

Causal events are defined by a causal relationship between events and like all events subclasses of `AnyThingInTime` inherit the attributes When, What, Where, and Participant.

For causal events, a widely shared relation (axiom), that brings together the causal relations with temporal relations, has been defined:

```
BeforeEE[Ex, Ey]:- CausalRelation[Ex, Ey].          (1)
```

If Ex is the cause of Ey, then the event Ex precedes temporally the event Ey.

2.5 The When Attribute of a complex event

For the simple event, the time of happening (hasSymbolicHappenInterval) defines the temporal interval in which the action occurs, while the time interval of occurrence of the complex event defines the minimum time interval in which all events belong to the complex event occurring. Therefore, the occurrence interval of complex events is not continuous, or that is, not in all temporal subintervals is there an event that happens. In addition, the temporal mode of a complex event is represented as the union of the temporal mode of event's components. It is obvious that, starting from the all temporal mode, one can define various algorithms that can determine, for example, the entire period of a complex event or the frequency of a particular action or category of action, etc.

The descriptor When can be calculated according to the descriptors of the component events, or it is instantiated interactively. In the latter case the compatibility checks (defined by constraints), with respect to the attributes When of the components events, must be run .

Informally, the interval of occurrence of a complex event is made up of the intervals of occurrence of the events' components and a set of temporal relations between these intervals. The rule for determining the minimum time of occurrence of a complex event, in accordance with the event components is reported. Let Ec be a complex event (a narrative), then its time of occurrence (defined as an instance of the class When) is calculated using the following rule:

```
01. Ec:NarrativeEvent[ hasComplexWhat->{E1, E2},
                       hasWhen->Wx]:-
02.   newId(Ec,E1,E2,Wx),
03.   genSymbolicInterval(I,T1,T2),
04.   insert{I:SymbolicHappenInterval[sti->T1,stf->T2]},
05.   E1:AnythingInTime[hasWhen->W1],
06.   E2:AnythingInTime[hasWhen->W2],
07.   minimun(W1, W2,Tmin), maximum(W1, W2,Tmax),
08.   tm_union(W1,W2,
           [EqualTT[T1,Tmin],EqualTT[T2,Tmax]], Tmx),
09.   insert{Wx:When[ hasSymbolicHappenInterval->I,
                      hasTemporalMode->Tmx]}.
```

To establish the time of occurrence of the narrative event Ec (01), the rule generates the structure Wx of the When attribute (02) and defines the symbolic interval of occurrence of the complex event (03-04). Then, it identifies the time of occurrence of the component events (05-06), and calculates the minimum and maximum time of occurrence of these events (07). The minimum and maximum calculated are correlated with the symbols' times with relations EqualTT[T1,Tmin] and EqualTT[T2,Tmax], which together with the temporal mode of the components (08) define the temporal mode of the complex event (09).

Others rules (not reported here) have been defined that consider the cases where the minimum and/or maximum of events cannot be determined, for example, because the annotation of the temporal mode is not complete.

In these cases, to overcome the inability to calculate the minimum and/or maximum, the rule defines temporal relations between the component events, E1 and E2, and the occurrence interval I of complex event. The relations are DuringEI[E1,I] and DuringEI[E2,I], which, together with the temporal mode pattern of the components, define the temporal mode of the complex event. Similar rules relating to the When for all types of complex events have been implemented.

3 Temporal reasoning for complex events

The classes of events that have been defined are appropriate for applying axiomatizations (expressed in the form of Horn clauses) to temporal reasoning. However, to apply these algorithms, one must follow a specific methodological statute that is associated with the ontological approach implemented.

Let Ei be a knowledge base of events, the algorithms (to check consistency and connectivity, and to discover new temporal relations) are applicable to all events that are not related to mental events: beliefs, desires and goals. This is because a mental event can be a component of a narrative, but the events that are among its arguments should not be involved in the analysis of the consistency and connectivity of a narrative.

Consider the following example: [*during the summer of 2010 a fan (Px) wants Inter to win the next season (2010-11)*]. The mental event of Px belongs to a narrative like this: 1.[*in summer 2010, a fan Px wants Inter to win the league*], 2.[*in summer 2011, Milan won the championship*]. Of course, there is no contradiction between the two events, because the event "*Inter won the league*” is something that belongs to the mind of a person, and does belong to events of real history. However, it is necessary to ensure a consistency in the set of events believed by a person. It is clear that the events to which we apply the algorithms to check consistency must belong to appropriate categories, and it’s believed that this approach (to define events as classes) is appropriate for this purpose.

3.1 How to check the consistency of events

For consistency checks of temporal relations, axioms of Russell and Kramp [LAM] (RK) have been adopted, which allows for performance of reasoning about the events through the relations between the events $\mathrm{prec}_{ev}(E1,E2)$ and $\mathrm{over}_{ev}(E1,E2)$, where $\mathrm{prec}_{ev}(E1,E2)$ means that the event E1 precedes the event E2, and $\mathrm{over}_{ev}(E1,E2)$ means that the event E1 overlaps event E2.

The RK language, with the primitive $\mathrm{prec}_{ev}(E1,E2)$ and $\mathrm{over}_{ev}(E1,E2)$, is not expressive enough to represent the temporal relations in our ontological representation (DuringEE[E1,E2], StartEE[E1,E2], etc.). In fact, the relations like DuringEE[E1, E2] cannot be defined only through the primitive $\mathtt{prec}_{\mathtt{ev}}$ and $\mathtt{over}_{\mathtt{ev}}$, because they also require the conditions between time instants. For this reason, a core with relations between time instants using the primitive $\mathtt{prec}_{\mathtt{t}}$(T1,T2) (the time instant T1 precedes the time instant T2) and $\mathtt{eq}_{\mathtt{t}}$(T1,T2) (the time instant T1 is equal to the time instant T2) has been defined. Through such primitives all temporal relations of OntoCE (between events, between events and time instant, etc.) have beeen defined and applied to the consistency analysis redefining the relations RK. Therefore, a set of axioms have been defined, that allow one to discover inconsistencies, not in order relations between events, but in order relations between time instants: $\mathtt{prec}_{\mathtt{t}}$(T1,T2) and $\mathtt{eq}_{\mathtt{t}}$(T1,T2). The following axioms for the verification of the inconsistencies is provided[2]:

Axiom schema

1: $prec_t(T1,T3)$ <- $prec_t(T1,T2) \wedge prec_t(T2,T3)$.
2: $eq_t(Tx,Tx)$.
3: $eq_t(T1,T2)$ <- $eq_t(T2,T1)$.
4: $eq_t(T1,T2)$ <- $eq_t(T1,T2) \wedge eq_t(T2,T3)$.
5: $prec_t(T1,T3)$ <- $prec_t(T1,T2) \wedge eq_t(T1,T3)$.
6: $\neg prec_t(T2,T1)$ <- $prec_t(T1,T2)$.
7: $\neg eq_t(T1,T2)$ <- $prec_t(T1,T2)$.

Implementation using Stable Models

```
prect(T1,T2)  :- prectD(T1,T2), not prectN(T1,T2).
prectN(T1,T2) :- time(T1), time(T2), prect(T2,T1).
prect(T1,T3) :- prect(T1,T2), prect(T2,T3),
   not prectN(T1,T3), time(T1),
   time(T2), time(T3).
prectN(T1,T2) :- time(T1), time(T2), eqt(T1,T2).
eqt(T1,T2) :- eqtD(T1,T2), not eqtN(T1,T2).
eqt(T1,T3) :- eqt(T1,T2), eqt(T2,T3),
   not eqtN(T1,T3), time(T1), time(T2), time(T3).
eqt(T1,T1) :- time(T1).
eqt(T2,T1) :- eqt(T1,T2), time(T1), time(T2).
eqtN(T1,T2) :- time(T1), time(T2), prect(T1,T2).
eqtN(T1,T2) :- time(T1), time(T2), prect(T2,T1).
prect(T1,T3) :- prect(T1,T2), eqt(T2,T3),
   not prectN(T1,T3),
   time(T1), time(T2),time(T3).
```

Fig. 4: Axiom schema and relative implementation for inconsistency checking

Axioms 1-4 define the properties of order relations: the transitivity of relation $\mathtt{prec}_{\mathtt{t}}$(T1,T2) and $\mathtt{eq}_{\mathtt{t}}$(T1,T2), the reflexivity and symmetry of the relation $\mathtt{eq}_{\mathtt{t}}$(T1,T2). As one can see, axioms 1-7 at the heads of the clauses are negated, so it is not possbile to implement them through monotonic logic programs and then through the traditional Horn clauses. For this reason a logic program for the axioms 1-7, through stable model semantics [GEL] by using the SModels system [SYR], has been implemented (in section 4 we report a running example of Stable Model Program of Fig. 4). In addition, there is the axiom:

prect(T1,T2) v prect(T2,T1) v eqt(T1,T2)

which translates the timeline by forcing every time point to have a relation of precedence or equality with another time. This axiom has not been implemented because it is not a useful consistency check. The axiom, however, would be useful to ensure the full connection of the time points of a narrative. For this purpose, an algorithm (shown in [MEL]) for controlling the connection has been defined. The definition of temporal relations between events in terms of primitive $\mathtt{prec}_{\mathtt{t}}$(T1,T2) and $\mathtt{eq}_{\mathtt{t}}$(T1,T2) is shown:

[2] In this paper we will use the symbols "Λ" and "<-" to denote the conjunction and implication in generic expressions of Horn Clauses, we will use instead the symbols "," and ":-" in the corresponding expressions Prolog and Flora2.

```
BeforeEE[3][E1,E2]   <-> dt(E1,T1,T2), dt(E2,T3,T4),
                         prec_t(T2,T3).
AfterEE[E1,E2]       <->  BeforeEE[E2,E1].
MeetsEE[E1,E2]       <->  dt(E1,T1,T2), dt(E2,T3,T4),
                          eq_t(T2,T3).
Meet_byEE[E1,E2]     <->  MeetsEE[E2,E1].
EqualsEE[E1,E2]      <->  dt(E1,T1,T2), dt(E2,T3,T4),
                          eq_t(T1,T3), eq_t(T2,T4).
OverlapsEE[E1,E2]    <->  dt(E1,T1,T2), dt(E2,T3,T4),
                          prec_t(T1,T3), prec_t(T3,T2),
                          prec_t(T2,T4).
Overlapped_byEE[E1,E2] <->  OverlapsEE[E2,E1].
DuringEE[E1,E2]       <-> dt(E1,T1,T2), dt(E2,T3,T4),
                          prec_t(T2,T4), prec_t(T3,T1).
ContainsEE[E1,E2]     <-> DuringEE[E2,E1].
StartsEE[E1,E2]       <-> dt(E1,T1,T2), dt(E2,T3,T4),
                          eq_t(T1,T3), prec_t(T2,T4).
Started_byEE[E1,E2]   <-> StartsEE[E2,E1].
FinishesEE[E1,E2]     <-> dt(E1,T1,T2), dt(E2,T3,T4),
                          eq_t(T2,T4), prec_t(T3,T1).
Finished_byEE[E1,E2]  <-> FinishesEE[E2,E1].
```

dt(E,T1,T2) provides the value of the start time T1 and end time T2 of the event E. In addition, for each occurrence interval of event E there is the following relation:

```
prect(T1, T2) <- dt(E,T1,T2).
```

Similary, all temporal relations have been defined.

Once the temporal relations through the primitive prec$_t$(T1,T2) and eq$_t$(T1,T2) have been redefined, one can check the inconsistencies of a set of temporal relations expressed by the relationship between time points, by using the axioms 1-7.

The program, given a knowledge base defined through the relations between time points (the input of the program), calculates the stable models, or rather, groups of consistent sets of relations (satisfying the axioms 1-7). If the program returns more than one stable model, then the relations are inconsistent.

In addition, among the events causal relation can be annotated and this produces new temporal relation between events.

Thus, to have a consistency check for all the relations annotated in a narrative, the temporal relation derived/obtained from the causal relations must be definend through the relations prec$_t$(T1,T2) and eq$_t$(T1,T2). In this way, it is possible to check

[3] In order to ease reading (so there is no ambiguity), simplified Flora2 notation, omitting the names of attributes, i.e., instead of the notation of instances id:ClassName[attr1->val1,attr2->val2, ..,attrn->valn], the following notation id:ClassName[val1,val2,..,valn] has been adopted.

the consistency of a narrative, according to the methodological approach represented in Fig. 5.

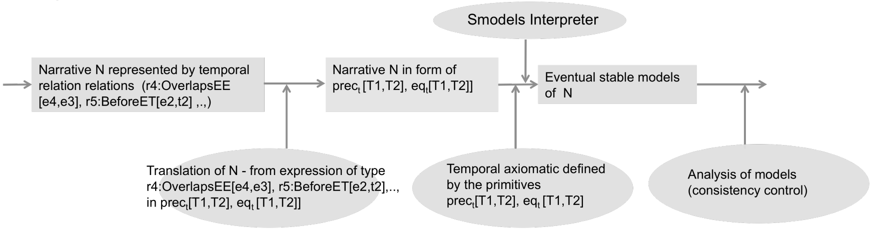
Smodels Interpreter
Narrative N represented by temporal relation relations (r4:OverlapsEE [e4,e3], r5:BeforeET[e2,t2] ,.,)
Narrative N in form of $prec_t$ [T1,T2], eq_t[T1,T2]]
Eventual stable models of N
Translation of N - from expression of type r4:OverlapsEE[e4,e3], r5:BeforeET[e2,t2],.., in $prec_t$[T1,T2], eq_t [T1,T2]]
Temporal axiomatic defined by the primitives $prec_t$[T1,T2], eq_t [T1,T2]
Analysis of models (consistency control)

Fig. 5: Transformation of temporal relations in the primitive $prec_t$, eq_t.

3.2 How to discover new temporal relations between events

Through the formalism defined for the representation of annotations, a new technique to discover new temporal relations between events has been developed. The methodology uses the results of the process of the consistency check. The axiomatization checks the consistency of the annotations and for the same model, provides all consistent derivations of set relations. Then, once the consistency of temporal relationships has been evaluated and (under the assumption of consistency) the only possible stable model (SM) has been identified, the SM can be used to identify temporal relations between events, using the equivalence relation between the temporal relations and order relations defined on time points showed in section 3. For this purpose, a rule for each temporal relation among events (all subclasses of TemporalRelationEE) has been defined. Each rule tries to find a particular temporal relationship (before, after, meets, etc.). Each rule has in input two events (E1 and E2) and the totality of relations between time points (stable model SM). The rule for finding BeforeEE relations between two events is shown:

```
1 findRel(E1,E2,SM) :-
2  not BeforeEE[E1,E2],
3  dt(E1,T1,T2), dt(E2,T3,T4),
4  subset({prec_t(T1,T2),prec_t(T3,T4),prec_t(T2,T4)},SM),
5  insert{BeforeEE[E1,E2]}.
```

In this case, the rule checks if there already exists a BeforeEE relation between two events E1 and E2 (2), then it reads the end points of occurrence intervals of events E1 and E2 (3), and verify the existence of conditions to discover a BeforeEE relation, or rather, if the relations on time points, which describe the condition, are a subset of the stable SM (4). Finally, the rule asserts the relation discovered in the knowledge base (5).

The above rules have been defined for all temporal relations between events: AfterEE[E1,E2],MeetsEE [E1,E2],etc.

3.3 Causal reasoning

For causal reasoning, an axiomatization (a variant of axiomatization defined in [BOC]) has been defined. The axioms that have been expressed (in a simplification of Flora2, see note 2) are reported as follows:

```
Id1:CausalRelation[A, B]:-                     Strengthening
    Id2:BeforeEE[A, B], demo(A,B),
    Id3:CausalRelation[B,C], newId(Id1, Id2, Id3).
Id1:CausalRelation[A, C]:-                     Weakening
    Id2:CausalRelation[A,B], Id3:BeforeEE[B, C],
    demo(B,C), newId(Id1,Id2,Id3).
Id1:CausalRelation[A,B∧C]:-                    And
    Id2:CausalRelation[A,B], Id3:CausalRelation[A,C],
    newId(Id1,Id2,Id3).
Id1:CausalRelation[A∨B,C]:-                    Or
    Id2:CausalRelation[A,C], Id3:CausalRelation[B,C],
    newId(Id1,Id2,Id3).
Id1:CausalRelation[A, C]:-                     Cut
    Id2:CausalRelation[A,B], Id3:CausalRelation[A∧B,C],
    newId(Id1,Id2,Id3).
Id1:CausalRelation[A∧C,B]:-              Left Monotonicity
    Id2:CausalRelation[A, B], C:Event,
    Id3:BeforeEE[C,B], newId(Id1, Id2, Id3).
Id1:CausalRelation[A,B∨C]:-              Right Monotonicity
    Id2:CausalRelation[A,B], C:Event,
    Id3:BeforeEE[A,C], newId(Id1, Id2, Id3).
```

The predicate `newId(Id1, Id2, Id3)` generates a new id `Id1` depending on `Id2` and `Id32`.

In the axiom Left Monotonicity the condition BeforeEE[C,B] has been included, because C is an event that cause B, and for this reason, C must precede B; otherwise it generates a contradiction.

For the axioms of *Weakening* and *Strengthening*, the meta-predicate `demo(A,B)` (implements the relation "B is deducible from A" [BOC]) have been defined, which was implemented as a variant of the meta-interpreter [BAT].

The axioms for causal relationships shown above have been defined for the class of causal events and are applied to all subclasses of that class.

4 An example of the application of the axiomatization for checking the temporal consistency

In this section, an example of the application of reasoning provided above is presented. In particular, as an example, a generic consistent narrative (without specifying in detail the actions of the events) has been defined.

Let e1, e2, e3 and e4 be four events and r1:DuringEI[e1,i1], r2:AfterET[e3,t2], r3:MeetEE [e1,e2], r4:OverlapsEE[e4,e3], and r5:BeforeET[e2,t2] be the temporal relations annotated.

t1 and t2 are the endpoints of interval i1; [st1, st2] is the symbolic interval of occurrence of e1, [st3, st4] is the symbolic interval of occurrence of e2, [st5, st6] is the symbolic interval of occurrence of e3, [st7, st8] is symbolic interval of e4.

Before performing the consistency check, the temporal relations were translated into temporal relations between time points as described in paragraph 3, obtaining the following: e prec$_t$(t1,t2),prec$_t$(st1,st2),prec$_t$(st3,st4), prec$_t$(st5,st6),prec$_t$(st7,st8),prec$_t$(st7,st5),prec$_t$(st5,st8), prec$_t$(st8,st6),prec$_t$(st2,t2),prec$_t$(t1,st1),prec$_t$(st4,t2), prec$_t$(t2,st5),prec$_t$(st2,st3).

In this set of relations the algorithm for checking consistency(paragraph 3) has been applied. Because the relations are consistent, the program returns to output only one stable model: **stableModelSet**([prect(st3,st4), prect(t2,st5), ..., eqt(st3,st2), eqt(st2,st2), eqt(st1,st1), eqt(t2,t2)]).

From this result, new temporal relations through the axioms for discovering new temporal relations (paragraph 3.2) have been identified, in particular, the rules identified the temporal relations beforeEE[e1,e3], and beforeEE[e2,e3].

If we add the relation beforeEI[e2,i1], intentionally making the knowledge base inconsistent, applying the algorithm for the consistency check obtains a more stable model. This response highlights the presence of inconsistencies.

Conclusions

In this paper a formalism for the representation of complex events has been proposed. The formalism for the representation of events is based both on time points and temporal intervals. Associated with this representation, temporal reasoning for checking the consistency and discovering new temporal relations has been constructed.

In this approach, unlike [MAN], the phase for checking the consistency is processed separately from the process for discovering the temporal relations. For this purpose, the axioms of Russell-Kamp [LAM] have been used to discover inconsistencies, and then evaluate the possible extensions of temporal relations. Furthermore, the algorithms must be applied separately, because they relate to different stages of the process of annotation and its use. Consistency checking is related to the content resulting from a process of semantic annotation, in which temporal inconsistencies could arise, while the discovery of temporal relations is a process that can be activated only after the consistency check.

In this approach, then, the process for checking the consistency of the annotation and discovering temporal relations have been separated. The motivation for this choice is so that one can apply different axiomatizations separately and divide complex problems into simple problems.

References

[ALL] J. Allen, Maintaining knowledge about temporal intervals, Communications of the ACM vol. 26 (11), p. 832-843, 1983.

[BAR] C. Baral, M. Gelfond, A. Provetti, *Representing Actions: Laws, Observations and Hypotheses*. J. Log. Program. 31(1-3): 201-243. 1997

[BAT] R. Barták, P. Stepánek, *Extendible Meta-Interpreters*, in: Journal KYBERNETIKA, Volume 33, Number 3, pages 291-310. 1977

[CAS] R. Casati, A. Varzi, "*Events*", The Stanford Encyclopedia of Philosophy (Spring 2010 Edition), E. N. Zalta (ed.),http://plato.stanford.edu/archives/spr2010/entries/events

[DAV] Davidson D., *Essay on Actions and Events*, New York, Oxford university Pers, 1980

[FLO] FLORA-2: *An Object-Oriented Knowledge Base Language* http://flora.sourceforge.net/

[GEL] M. Gelfond, Vladimir Lifschitz, *The Stable Model Semantics For Logic Programming*. In Proceedings of the Fifth International Conference on Logic Programming, pages 1070-1080, Seattle USA, August 1988.

[GIU] F. Giunchiglia, P. Andrews, G. Trecarichi, and R. Chenu-Abente, *Media Aggregation via Events*, Proceedings of the Workshop on Recognising and Tracking Events on the Web and in Real Life, Athens, Greece. 2010.

[GON] G. Gonzalez, C. Baral, M. Gelfond, *Alan: An Action Language For Modelling Non-Markovian Domains*. Studia Logica 79(1): 115-134 (2005)

[LAM] M.V. Lambalgen, F. Hamm. *The Proper Treatment of Events*. Blackwell, Oxford and Boston, 2004.

[LEV] Levesque H. J. , Pirri F., Reiter R., *Foundations for the Situation Calculus*. Electron. Trans. Artif. Intell. 2: 159-178 (1998)

[LIN] Lin F., Reiter R, *Rules as Actions: A Situation Calculus Semantics for Logic Programs*. J. Log. Program. 31(1-3): 299-330 (1997)

[MAJ] J. Ma, *Ontological considerations of time, meta-predicates and temporal propositions*. Applied Ontology, vol. 2, pp. 37-66, 2007.

[MAN] I. Mani, B. Wellner, M. Verhagen, and J. Pustejovsky, *Three approaches to learning Tlinks in timeml*, Technical Report, 2007.

[MEL] Francesco Mele, Antonio Sorgente, Giuseppe Vettigli, *Designing and Building Multimedia Cultural Stories Using Concepts of Film Theories and Logic Programming*. In Proceedings of Cognitive and Metacognitive Educational Systems (MCES) symposium. pag 57-63. ISSN 1613-0073 Vol-598. 2010.

[MIL] R., Miller, M. Shanahan, *The event-calculus in classical logic – alternative axiomatizations*. In Electronic Transactions on AI 3(1): 77-105. 1999

[PUS03] Pustejovsky, J.; Castaño, J.; Ingria, R.; Saurí, R.; Gaizauskas, R.; Setzer, A; G. Katz, *TimeML: A Specification Language for Temporal and Event Expressions*. Netherlands, Kluwer Academic Publishers. 2003.

[PUS08] J. Pustejovsky, K. Lee, H.B. Harry, B. Boguraev, and N. Ide, *Language Resource Management—Semantic Annotation Framework (SemAF)—Part 1: Time and events*, International Organization, 2008.

[SHO] Shoham, Y. *Artificial Intelligence Techniques in Prolog*. Morgan Kaufmann Publishers.

[SHE] Scherp, A. Franz, T.Saathoff, C., Staab, S., *F--a model of events based on the foundational ontology dolce+DnS ultralight*, Proceedings of the fifth international conference on Knowledge capture - K-CAP '09

[SYR] T. Syrjänen, *Lparse 1.0 User's Manual*, http://www.tcs.hut.fi/Software/smodels/

[WAN] H.S. X. Wang, S. Mamadgi, A.Thekdi, A. Kelliher, *Eventory - An Event Based Media Repository*, International Conference on Semantic Computing, 2007, pp. 95-102.

[WIN] T. Winkler, A. Artikis, Y. Kompatsiaris, and P. Milonas, eds., *Workshop Recognising and Tracking Events on the Web and in Real Life,* Athens, Greece: 2010.