Using a general theory of time and change in patient monitoring: experiment and evaluation.

Luca Chittaro, Michel Dojat

USING A GENERAL THEORY OF TIME AND CHANGE IN PATIENT MONITORING: EXPERIMENT AND EVALUATION

LUCA CHITTARO[*] AND MICHEL DOJAT[†]

[*]Dipartimento di Matematica ed Informatica, Università di Udine,Via delle Scienze, 33100 Udine, Italy; and

[†]INSERM U.296, Faculté de Médecine, 8, Rue du Général-Sarrail, 94010 Créteil Cedex, FRANCE

E-MAIL: chittaro@dimi.uniud.it; and dojat@laforia.ibp.fr

Abstract

In this paper, we propose to use one of the well-known general theories of time and change, namely the Event Calculus [1], to represent temporal aspects in intelligent medical monitoring systems. In particular, we explore the application of CEC [2] (an efficient implementation of the Event Calculus) to the management of mechanical ventilation. First, we present the prototype we have built, which has been extensively tested on patient's data from real clinical cases. Then, we provide a thorough evaluation of the obtained results, pointing out both strengths and weaknesses of the approach, and identifying a number of extensions which can be extremely useful to scale up the medical application of the approach.

KEYWORDS: temporal reasoning, Event Calculus, mechanical ventilation management, patient monitoring, knowledge-based systems, PROLOG.

1 Introduction and Motivation

In data-rich clinical environments such as Intensive Care Units (ICUs) or operating rooms, there is a crucial need for intelligent monitoring systems that can help the clinician to deal with the massive flux of information. These systems should be able: (i) to acquire and exploit the mass of data available to propose a diagnosis of the patient's state, (ii) to filter the numerous alarms from monitors to indicate only those that require a human intervention, and (iii) to propose specific therapeutic strategies depending on the evolution of the patient's state. Explicit time and change representation is essential for building such intelligent monitoring systems. In this Section, we point out the motivations for our work, both from the temporal reasoning and the application point of view.

1.1 The Temporal Dimension

Time is a central factor in intelligent monitoring systems that are supposed to interact with real dynamic environments. The need for time representation in these systems concerns two major aspects:

- *Modelling of temporal concepts and inferences performed by the physician*: the physician i) builds dynamically an interpretation of the time course of the patient's disease, ii) predicts the patient's evolution with regard to previous states, and iii) constructs and executes a plan of actions to drive the patient to an expected state. The physician adapts his/her strategy to the history of the patient's disease and to the time the patient spent in a given state. To build a global dynamic interpretation of the patient's behaviour, he/she must construct gradually specific abstractions at several levels, recognizing relevant changes for each level [3].

- *Respect of real-time constraints*: the system should be able to i) acquire physiological data

provided by several monitors and the clinical staff, ii) plan the sequencing of the three fundamental tasks in medical reasoning [4] - observation, diagnosis, and then therapy - each task being in turn decomposable in several sub-tasks, and iii) have a prompt reaction in alarming situations, which impose to short-cut some sub-tasks.

The need for an explicit representation of time in medical systems has been advocated by several researchers, but very few clinical decision support systems incorporate clear formalisms for temporal reasoning [5]. Recently, several research efforts have been devoted to define ontologies and mechanisms for medical temporal reasoning and abstraction independently of a specific medical application [3, 6, 7, 8, 9]. From this point of view, the Event Calculus (EC), the well-known general theory of time and change proposed by [1], appears to be an interesting framework. In particular, EC (i) is a general and thus likely to be re-usable approach, (ii) is well-founded and has been thoroughly formally studied (some works dealing with formal aspects of EC are [10, 11, 12, 13], just to mention the most recent ones), and (iii) adheres to a model with events, states and cause-effect relationships, which seems well adapted for temporal reasoning in medical domains [3, 6, 14]. The Situation Calculus [15] shares the same general aim of EC, i.e. to formalise common sense reasoning about the initiation and persistence of properties and relationships over the course of time. We prefer EC for our domain, because, as pointed out by [16]: (i) whereas EC was intended primarily for reasoning about actual events, the Situation Calculus was designed primarily for reasoning about hypothetical actions and situations (which is not our case), and (ii) whereas the Situation Calculus deals with transitions between global situations, EC allows to deal with the effect of actions on local states.

Although EC derives a significant representational power from its roots in Logic Programming augmented with negation-as-failure as a mechanism for default reasoning, it was found to be rather inefficient to meet the harsh requirements of fast decision-support response [12]. For this reason, EC has not been used for the design of intelligent monitoring systems up to now, but an efficient implementation [2] of EC has been recently proposed. In this paper, we exploit this implementation in order to explore the application of EC in the context of mechanical ventilation management, with the general aim of testing the adequacy of EC to model temporal reasoning in medical monitoring.

1.2 Intelligent Systems for Mechanical Ventilation Management

A typical clinical application of intelligent monitoring systems is the management of the mechanical respiratory assistance provided to patients who suffer from a lung disease and are hospitalised in ICUs. Recent physiological studies [17, 18] have convinced physicians to mechanically ventilate patients as soon as possible with partial assistance modalities: a variable level of mechanical assistance is added to the spontaneous respiratory activity of the patient. Although partial mechanical support such as *pressure support ventilation* (PSV) [19] is simple in

its principle, its use generally requires the presence of a trained and experienced physician who adapts the level of assistance to the evolution of the patient's state. This is emphasised when the physician, applying specific strategies, tries to decrease gradually the assistance and appreciates the patient's capability to breathe alone. This procedure (called *weaning procedure*) must be performed carefully to improve the quality and the success rate of such a difficult process.

Many decisions and adjustments performed on the ventilator settings are based on objective data and can be formalised and modeled with appropriate knowledge representation techniques. Ideally, the advantages of a knowledge-based system for the management of ventilator therapy are: (i) to function on a 24 hours per day basis, allowing a continuous adaptation of the level of the assistance and a reduction of total duration of ventilation, and (ii) to develop specific weaning strategies, including a gradual decrease of the mechanical support, difficult to obtain in clinical practice without the assistance of a computerised system. Such a system must work in a closed-loop to be useful to the clinical staff. Recently, clinical studies have validated this approach [20, 21].

Since the precursor work of Fagan [22], several systems have been designed to assist respiratory management (see [23] for a broad survey on recent intelligent patient monitoring projects). Briefly, systems can be divided in two categories:

- *Systems solving a practical clinical problem*: for example, [24] deals with the complex task of ventilating patients with Acute Respiratory Distress Syndrome, while [25] tackles the problem of assisting therapists and nurses in weaning post-operative cardiovascular patients from mechanical ventilation. These systems are hardly adaptable to other clinical contexts.

- *General architectures for intelligent monitoring*: these are long term research projects, such as [26], which proposes a general architecture for intelligent agents and is applied to intensive care monitoring problems, and [27], which mixes qualitative and quantitative computation in a ventilator-management advisor. The first objective of these researches is not to design a prototype working at the patient's bedside, but to explore novel AI techniques potentially useful to solve medical problems.

Our project is intermediate. On one hand, our goal is to solve a clinical problem (i.e. the management of PSV) and to test a closed-loop prototype at the patient's bedside. On the other hand, we aim at generic reasoning mechanisms for temporal medical reasoning validated in the clinical environment. Existing systems often use an embedded implicit temporal representation (e.g. [22], [28]). Unlike these systems, we adopt an explicit and general representation of temporal knowledge and reasoning (i.e., the Event Calculus [1]), to monitor and control ventilator therapy in real-time. Therefore, the prototype we built serves also the purpose of evaluating strengths and weaknesses of the Event Calculus in the medical monitoring domain.

2 Temporal Ontology and Inference in EC

Kowalski and Sergot's Event Calculus (EC) is a general approach to representing and reasoning about events and their effects in a logic programming framework [1]. From a description of events which occur in the real world and properties they initiate or terminate, EC derives the maximal validity intervals (MVIs) over which properties hold. It takes the notions of event, property, time-point and time-interval as primitives and defines a model of change in which *events* happen at *time-points* and initiate and/or terminate *time-intervals* over which some *property* holds. It embodies a notion of *default persistence* according to which properties are assumed to persist until an event occurs that interrupts them*. A model of the world based on events, whose occurrence modifies the state of the world and properties that have a tendency to persist during time, is well adapted to our applications [3].

In the following axioms, we adopt PROLOG's convention of beginning variables' names with an uppercase letter and constants with a lowercase letter. In order to help the reader who is not familiar with PROLOG syntax, we provide a natural language formulation to the right of each axiom we introduce.

2.1 Basic Event Calculus

Formally, we represent an event occurrence by means of the `happens_at(event, timePoint)` clause† . The relation between events and properties is defined by means of `initiates_at` and `terminates_at` clauses:

```
initiates_at(event1,prop,T):-
  happens_at(event1,T),
  holds_at(prop_a1,T),
  ...,
  holds_at(prop_aN,T).
```

"event1 initiates property prop at time T
if event1 happens at T,
and property prop_a1 holds at T,
and ...,
and property prop_aN holds at T"

```
terminates_at(event2,prop,T):-
  happens_at(event2,T),
  holds_at(prop_b1,T),
  ...,
  holds_at(prop_bM,T).
```

"event2 terminates property prop at time T
if event2 happens at T,
and property prop_b1 holds at T,
and ...,
and property prop_bM holds at T"

The `initiates_at` (`terminates_at`) clause states that each event of type `event1` (`event2`) initiates

* The implicit default persistance assumption of EC may not be natural in some medical problems. For these cases, some authors (e.g. [3], [9], [22]) allow also the use of explicit persistence functions to determine intervals of validity.

† In general, the requirement of knowing the exact time point when each event happened can be relaxed (we dealt with the case of partially ordered events in [29]) at the expense of efficiency. Fortunately, all the events in the considered application can be time-stamped.

(terminates) a period of time during which property `prop` holds, provided that a number (possibly zero) of given conditions hold at instant `T`. In EC, both `initiates_at` and `terminates_at` are context-independent predicates: they do not admit preconditions. However, even when modeling very simple real-world examples, the context is essential to decide which properties are initiated or terminated by the occurrence of an event. For example, the event "turn on the switch" in a simple light bulb circuit, initiates the property "the light is on" at a given instant, only if the property "electrical power is supplied" holds at that instant [2]. Thus, as indicated above, we added preconditions to `initiates_at` and `terminates_at` in order to model complex domains, such as mechanical ventilation management.

Initial conditions describe the (possibly partial) initial state of the world and are specified by means of a number of events of type `initially(prop)`.

The EC model of time and change is defined by means of the following axioms:

```
mholds_for(P,[Start,End]):-
  initiates_at(Ei,P,Start),
  terminates_at(Et,P,End),
  Start < End,
  \+broken_during(P,[Start,End]).
```

"property P maximally holds over interval [Start,End],
if event Ei initiates P at instant Start,
and event Et terminates P at instant End,
and Start is before End,
and P is not broken during [Start,End]"

```
mholds_for(P,[Start,infPlus]):-
  initiates_at(Ei,P,Start),
  \+broken_during(P,[Start,infPlus]).
```

"property P maximally holds over interval [Start,+∞],
if event Ei initiates P at instant Start,
and P is not broken during [Start,+∞]"

```
broken_during(P,[Start,End]):-
  terminates_at(E,P,T),
  Start < T, T < End.
```

"property P is broken during [Start,End],
if event E terminates P at instant T,
and T is between Start and End"

The `mholds_for` axiom states that a property `P` maximally holds between events `Ei` and `Et`, if `Ei` initiates `P` and occurs before `Et` that terminates `P`, provided there is no known interruption in between (the negation involving the `broken_during` predicate is indeed interpreted using negation-as-failure). The interval `[Start,End]` is thus a Maximal Validity Interval (MVI) for property `P`. The `broken_during` axiom states that a given property `P` is interrupted between `Start` and `End` if there is an event `E` that happens between them and terminates `P`. This axiom provides a so-called *weak interpretation* [30] of `initiates_at` clauses (in a *strong interpretation* [30], the `broken_during` axiom would consider also initiating events for `P` and not just terminating ones as possible interruptions for the interval `[Start,End]`). In general, the choice between weak and strong interpretation is domain-dependent [30].

Finally, the `holds_at` axiom relates a property to a time-point rather than to a time-interval: