A Language for Defining Events in Multi-Dimensional Time Series: Application to a Medical Domain†

Juan A. Lara[1], África López-Illescas[2], Aurora Pérez[1], Juan P. Valente[1]

[1] Facultad de Informática, Universidad Politécnica de Madrid, Campus de Montegancedo, 28660, Boadilla del Monte, Madrid, Spain.
j.lara.torralbo@upm.es, {aurora, jpvalente}@fi.upm.es

[2] Centro Nacional de Medicina del Deporte, Consejo Superior de Deportes, C/ El Greco s/n, 28040, Madrid, Spain.
africa.lopez@csd.mec.es

Abstract. In many domains, like seismography or medicine, time series analysis focuses on particular regions of interest in the time series, known as events, whereas the remainder of the times series contains hardly any useful information. Research into the field of time series events definition has proposed techniques that are only applicable to specific domains. In this paper, we propose an events definition language that is general enough to be able to simply and naturally define time series events in any domain. The proposed language has been applied to the definition of time series events generated by stabilometric systems within the branch of medicine dealing with balance-related functions in human beings.

Keywords: Data Mining, Time Series, Event Definition.

1 Introduction

Most time series data mining techniques consider whole time series. However, there are many problems where, rather than analysing the entire time series, modelling should be confined to particular regions of interest, known as events [1]. This applies to domains like seismography, the stock market or medicine. In seismography, for example, the only moments of interest are when the time series indicates an earthquake, volcanic activity leading up to the quake or replications. The lengthy periods between these events provide hardly any information.

There are several papers dealing with the topic of events identification in time series. In [2] Lan and Ma propose a way of defining events using a function of interest that evaluates each and every subsequence in the time series with the resulting computational cost. Apart from this, there are not many proposals focusing on time

† This work was funded by the Spanish Ministry of Education and Science as part of the 2004-2007 National R&D&I Plan through the *VIIP* Project (DEP2005-00232-C03).

series events definition. What we more often come across are methods for dividing a time series into subsequences. These methods are based, for example, on statistical concepts like the change point [3] or using genetic algorithms [4]. Additionally, the proposed techniques tend to be domain specific and are not easily applicable to other domains. This is the case of the TSDM framework [5], proposing a set of methods for identifying events in the financial domain. Likewise, most event processing methods are applicable to one-dimensional time series. This way, they obviate the complexity of multi-dimensional time series, where there are possible dependencies among the different time series attributes (dimensions).

In this paper we propose an events definition language for multi-dimensional time series. This language is designed to be general enough to be applied in any domain. To test the validity of this language, it was applied to time series generated in the medical field of stabilometry. Stabilometry is a discipline looking into human balance. A device, called posturograph, is used to study balance-related functions. The patient stands on a platform and completes a series of tests (Figure 1). We used a static Balance Master posturograph. In a static posturograph, the platform on which the patient stands does not move. The platform has four sensors, one at each of the four corners: right-front (RF), left-front (LF), right-rear (RR) and left-rear (LR). While the patient is completing a test, these sensors record the pressure intensity being exerted on the platform, generating four interrelated time series.

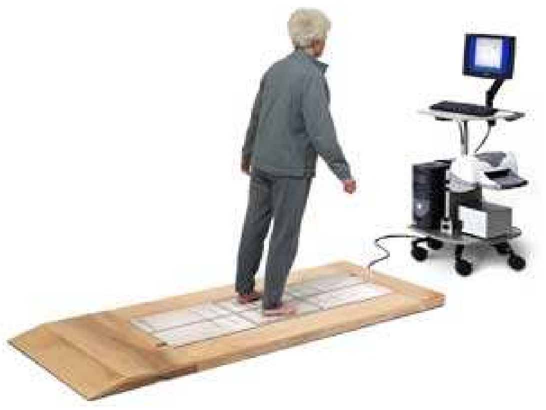

Fig. 1. Person doing a test on the posturograph.

The remainder of the article is organized as follows. Section 2 describes the events definition language. Section 3 details the application of the language to the reference domain. Section 4 discusses the results and conclusions.

2 Events definition language

Most methods dealing with time series events pose a portability problem, as events identification and characterization is extremely domain dependent. Consequently, profound very low level changes are required to be able to apply these methods to different domains. To solve this problem, this article sets out a framework for defining multi-dimensional time series events that is designed to overcome the gaps of existing techniques. Existing techniques are not easily applicable to any domain. Neither are they designed to deal with multi-dimensional time series. This framework is based on a language that uses basic set theory, logic, algebra and descriptive statistics techniques. The language has a series of predefined elements that are potentially useful in most domains and other elements that will have to be defined for each particular domain.

To define events, the user of the language proposed here will have to establish the following elements in this order:

1. Basic Elements

As applies to any high level language, for example, programming languages, the language proposed here contains operators (arithmetic, relational and set) and other basic elements like reserved words, identifiers, numerical and logical constants or basic arithmetic functions. Other basic predefined elements have been conceived apart from the above. These elements, like time series, statistical measures calculated on time series (mean, mode or median) and predefined data sets (like, for example, the set of all maximums in a time series) which tend to be valuable in all fields, will be usable in any domain. In the particular case of multi-dimensional time series, each dimension will be treated as a time series.

2. Sets of interesting points in time series

The sets of points of interest have to be established in each domain. These sets are the basis for defining events. The proposed language can make use of any of the basic language elements to define these sets. The proposed syntax for defining these sets of points is:

set name {point in Defined_Set such that Condition};

In other words, the syntax indicates the name of the set composed of those points of an earlier defined set that meet a particular condition. In the case where there are multi-dimensional time series, the condition will also cover all the time series.

3. Events

To define a type of event, we can use the sets of particular points of each domain and the basic language elements.

An event is conceived as a start point, an end point and a peculiar point, which falls between the other two. For the three points to form an event, they have to meet a particular condition. The syntax for defining events is:

**events name {peculiar_point in Defined_Set, start in Defined_Set’,
end in Defined_Set’’ such that Condition };**

That is, the events definition indicates the name of the event type, followed by the peculiar point, the start point (*start*) and the end point (*end*).

3 Case study: application of the events definition language to the stabilometric domain

The proposal detailed here is especially suitable for domains where we are interested not in the time series as a whole but exclusively in those regions where a particular event takes place. An event is part of a time series that is of interest to the expert of the domain in question.

Throughout this research we have worked on time series from what is known as the Unilateral Stance Test (UNI), one of the tests output using the posturograph. This test aims to measure how well a patient standing on one leg is able to keep his or her balance with both eyes open and closed.

The ideal thing for this test would be for the patient not to wobble at all but to keep a steady stance throughout the test. The interesting events of this test occur when the patient loses balance and puts the lifted foot down on the platform. This type of event is known in the domain as a fall. When there is a fall, the sensors for the leg that should be lifted will register the pressure increase. Figure 2 shows an example of a UNI test series. The plots at the top of the graph are the values recorded by the RR and RF sensors, that is, the right leg sensor, the leg the patient was standing on. The plots at the bottom of the graph are the values recorded by the LR and LF sensors, that is, the left leg sensors, the leg that should be lifted. The pressure peaks generated when a fall event is registered are highlighted.

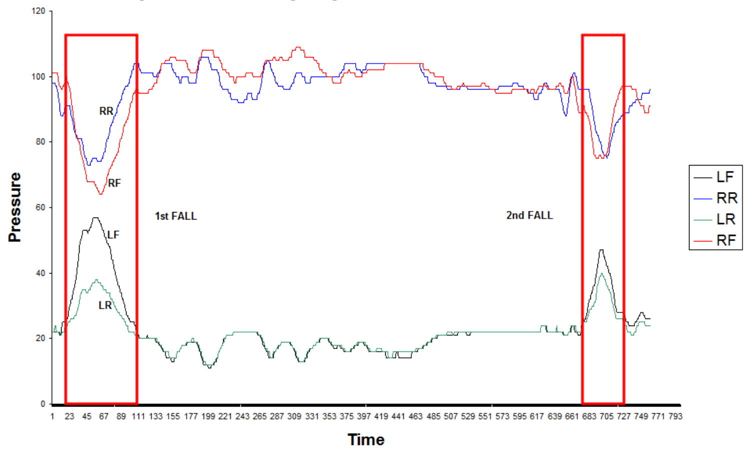

Fig. 2. UNI test time series, highlighting two events (falls).

Figure 2 shows that the LR and LF time series are almost static (with a small variation margin) throughout, except when there is a fall. Therefore, we could define a stability value for the pressure exerted on the respective sensors for the above time series. This stability value, which in the case of this example would be around 20, could be a statistical measure like the mode.

Every time there is a fall, there is a particular time instant where the LF and LR sensors record a local maximum and the RR and RF sensors record a local minimum. This point is more or less midway through the fall.

Therefore, we could define the points that meet the following conditions as the midpoint of a fall:

1) RR and RF record a local minimum and LF and LR a local maximum simultaneously.
2) The local maximum of LF is relatively distant from the stability value (mode) of the LF time series.
3) The local maximum of LR is relatively distant from the stability value (mode) of the LR time series.

If we locate the points that meet the above conditions, we will have identified the event. To determine where the event starts and ends, suffice it to consider the points of intersection of the time series with their respective modes. The start point of the fall will be the point of intersection with the mode immediately before the fall peculiar point, whereas the end point of the fall will be the point of intersection with the mode immediately after the peculiar point of the fall.

Now that we have explained the concepts required to define fall events in the UNI test, let us now describe how to specify these events using the proposed events definition language.

The first basic elements for use are the actual time series. The time series are considered as mathematical functions (*lf*, *lr*, *rf*, *rr*). According to the events definition, we are also going to use the statistical measures *mode(lr)* and *mode(lf),* and the predefined sets *max(lf), max(lr), min(rr), min(rf)* and *timestamp(lf)*. *timestamp(lf)* represents the set of all time instants in the *lf* time series. To define the time series, we use the language reserved word *ts*, whereas the language provides the reserved words *stat* and *basicset* to define the statistics and predefined sets, respectively:

```
ts lf;
ts lr;
ts rf;
ts rr;
stat modlf mode(lf);
stat modlr mode(lr);
basicset maxlf max(lf);
basicset maxlr max(lr);
basicset minrr min(rr);
basicset minrf min(rf);
basicset tslf timestamp(lf);
```

As mentioned above, we are interested in searching those time instants where *rf* and *rr* are minimum and *lf* and *lr* are maximum. This set of points shall be called *cand1*:

```
set cand1
{
    x in tslf
    such that
          (x in maxlr)&& (near(x in maxlf))&& (near(x in
          minrf)) && (near(x in minrr))
};
```

However, we are only interested in those points of *cand1* where the value of the time series for the sensors on the side of the posturograph where the patient’s leg is lifted is relatively distant from the mode. This set of points is *cand2* and could be defined as follows:

```
set cand2
{
          y in cand1
          such that
             muchGreater(lf(y).value + lr(y).value,
             modlf + modlr)
  };
```

We have used the predefined operator *muchGreater* in the definition of set *cand2*. This operator receives two numbers and returns *true* if the first is relatively distant from the second.

Finally, we need the intersections of one of the bottom two time series (for example, *lf*) with its stability value (mode). This will then be used to define the event start and end. This set of points shall be called *intersec*:

```
set intersec
{
          z in tslf
          such that
             lf(z).value == modlf
};
```

Now that we have defined the basic elements and the sets of points required, it remains to define the events. In this case, the peculiar point of the event will be a member of *cand2*, whereas the event start and end will be members of the *intersec* set. As specified above, the falls would be:

```
events falls
{
    peculiar_point in cand2, start in intersec, end in
    intersec
    such that
          (icl(start,peculiar_point)) &&
          (icr(peculiar_point,end))
};
```

Clearly, we search for the fall peculiar point (a point in *cand2*) and, then, we search the start and end of the fall within the set *intersec*. This defines the event perfectly. We have used the *icl* and *icr* operators in the definition of *falls* to assure the *start-peculiar_point-end* time sequence. These operators are predefined in the language. Specifically, *icl* assures that start is the point from the *intersec* set that is previous and closest to the *peculiar point*. The operator *icr* is similar to the above, but it checks whether *end* it is the closest element that happens after the *peculiar point*.

4 Results and Conclusions

In this article we proposed a language that is general enough to be able to simply and naturally define multi-dimensional time series events in any domain. This language was applied to events definition in time series from the stabilometry domain, thereby demonstrating the feasibility of the language for defining events in such complex domains.

In the follow-on stages of the project we intend to develop a translator that will process events definitions. After checking that the events definitions conform to the language rules, the translator will automatically generate source code implementing an events identification program matching a particular definition.

This translator is now under development. Consequently, we have simulated the translator output to test the utility of the language. We have applied this simulated source code to identify events that have been used to create posturographic time series models using a technique outlined in [6].

We used two groups of top-competition sportspeople to create these models. The first group was composed of 15 professional basketball players, whereas the second was made up of 15 young elite skaters. Thirty is a reasonable number of patients, taking into account that top-competition athletes do not abound and the tests are complex.

The first model ($M_{basketball}$) was created from a training set composed of 10 of the 15 basketball players. The other five constituted the test set for the above model. The second model ($M_{skating}$) was again generated from a training set composed of 10 of the 15 skaters. The other five formed the test set for the model. The sportspeople in the test set were chosen at random from all the sportspeople in each group.

Then we compared the 10 sportspeople from both test sets against the two created models. Nine of the athletes turned out to be more similar to the model created from sportspeople practising their discipline. Only one of the sportspeople was wrongly classified. In this case, the comparison method found the skater to be closer to the model for basketball players.

This experiment demonstrates the feasibility of the events definition language in times series in a complex domain like stabilometry. The development of the tool to translate events specifications to source code will avoid the need of ad hoc events detection methods for each particular domain having to be used.

References

1. Povinelli, R.: Time Series Data Mining: identifying temporal patterns for characterization and prediction of time series. PhD. Thesis, Milwaukee (1999)
2. Lan, Q., Ma, C.: A Method of Discovering Patterns to Predict Specified Events from Financial Time series. Fourth International Conference on Natural Computation (2008)
3. Guralnik, V., Srivastava, J.: Event Detection from Time Series Data. Proceedings of the fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (1999)
4. Chung, F.-L., Fu, T.-C., Ng, V., Luk, R.: An Evolutionary Approach to Pattern-Based Time Series Segmentation. IEEE Transactions on Evolutionary Computation, Vol. 8 (2004)
5. Povinelli, R.: Identifying Temporal Patterns for Characterization and Prediction of Financial Time Series Events. Proceedings of the First International Workshop on Temporal, Spatial, and Spatio-Temporal Data Mining (2000)
6. Lara, J. A., Moreno, G., Perez, A., Valente, J. P., Lopez-Illescas, A., Comparing Posturographic Time Series through Event Detection, 21st IEEE International Symposium on Computer-Based Medical Systems (CBMS’08), Page(s):293 – 295 (2008)