Semiqualitative Temporal Patterns in Time-Series Databases

J.A. Ortega, R.M. Gasca, M. Toro, F.J. Galán and J.M. Cañete
Departamento de Lenguajes y Sistemas Informáticos.
University of Sevilla
Avda. Reina Mercedes s/n. Sevilla (Spain)
{ortega, gasca,mtoro,galanm,canete}@lsi.us.es

Abstract

A way to obtain behaviour patterns of semiqualitative models of dynamic systems automatically is proposed in this paper. The temporal evolution of these models is stored into a database. This is a time series database. This database may be obtained by means of sensor data or by means of semiqualitative simulations. In any way, the database contains the values of state variables and parameters. Searching for similar patterns in such database is essential, because it helps in predictions, hypothesis testing and, in general, in data mining and rule discovery.
A language to carry out queries about the qualitative and temporal properties of this time-series database is proposed. The language is also intended to classify the different qualitative behaviours of a model. This classification may be carried out according with a specific criterion or automatically by means of clustering algorithms. The semiqualitative behaviour of a system is expressed by means of hierarchical rules obtained by means of machine learning algorithms.
The methodology is applied to a logistics growth model with a delay.

1 Introduction

In real systems studied in science and engineering, it is difficult to find mathematical models that represent them in an appropriate way. The modelling techniques should obviate certain aspects of the system. The simulation of these models helps us to study the evolution of the real system. A way to carry out these simulations is described in [11] in depth. However, it is not always possible to obtain a mathematical model of a system. Thus, it is necessary to apply other techniques in order to carry out its study. A possibility may be placing sensors in the real system. The analysis of these data allows to study the system evolution.

On the other hand, knowledge about dynamic systems may be quantitative, qualitative, and semiqualitative. When these models are studied all this knowledge should be taken into account. Different levels of numeric abstraction have been considered: purely qualitative [8], semiqualitative [6] [10], and quantitative.

In this paper, a technique to carry out the analysis of dynamic systems with qualitative and quantitative knowledge is proposed. The idea follows: *the quantitative behaviours of a real system are stored into a database and techniques of Knowledge Discovery in Databases (KDD) are applied to study the system.* The way to obtain the behaviours does not matter: by means of the simulation of a model or by means of the data sensors.

The term KDD is used to refer to the over-all process of discovering useful knowledge from data. The problem of knowledge extraction from databases involves many steps, ranging from data manipulation and retrieval to fundamental mathematical and statistical inference, search and reasoning. Although the problem of extracting knowledge from data (or observations) is not new, automation in the context of databases opens up many new unsolved problems. We are interested in time-series databases corresponding to the evolution of semiqualitative dynamic systems. Databases theories and tools provide the necessary infrastructure to store, access, and manipulate data.

A new way to study dynamic systems that evolve in the time is proposed merging "data mining", "time-series" and "databases engine". The proposed perspective tries to discover the underlying model in the database by means of a query/classification language. It is also possible to obtain the behaviour patterns of these systems automatically by means of clustering techniques. Clustering is a discovery process in data mining. These discovered clusters can help to explain the features of the underlying data distribution. The semiqualitative behaviour of a system is expressed by means of hierarchical rules obtained by means of machine learning algorithms [1]. The methodology is applied to a logistics growth model with a delay.

2 Our approach

In this paper, we focus on those dynamic systems where there may be qualitative knowledge in their parameters, initial conditions and/or vector field. They constitute the semiqualitative differential equations of the system. A semiqualitative model is represented by means of

$$\Phi(\dot{x}, x, q, t), \qquad x(t_0) = x_0, \qquad \Phi_0(q, x_0) \quad (1)$$

being $x \in I\!R^n$ the state variables, q the parameters, t the time, $\dot{x}$ the derivative of the state variables, Φ the interval constraints among $\dot{x}, x, q, t$, and Φ_0 constraints with the initial conditions. These models are described in [11] in depth.

There is enough literature that studies stationary states of dynamic systems, however, the study of transient states is also necessary. Stationary and transient states of a semiqualitative dynamic system may be studied with the proposed approach (figure 1).

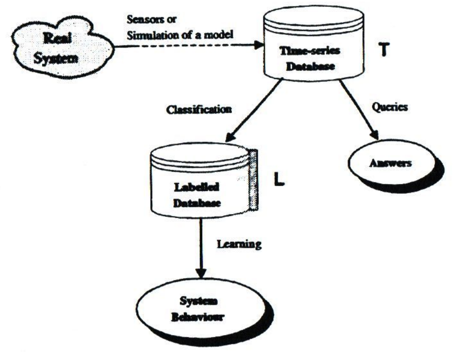

Figure 1: Our approach.

We begin with a time-series database. It may be obtained by means of semiqualitative simulations [11] or by means of data sensors. This is a trajectory database. A trajectory contains the values of the parameters and the values of all state variables from their initial value until their final value. Therefore, every trajectory stores the values of the transient and the stationary states of these variables of the system. Every trajectory is a set of time-series of state variables.

We propose a language to carry out queries about the qualitative properties of the set of trajectories included in the database. A labelled database is obtained when these trajectories are classified according to some criteria. It is also possible to classify the database by means of an automatically process. In such case, it is necessary to apply clustering techniques. Qualitative behaviours patterns of the system may be automatically obtained from this database by applying rule discovery based on genetic algorithms [1].

3 Qualitative knowledge

In this paper, we are interested in applying qualitative operators to carry out semiqualitative queries with the language.

3.1 Unary Qualitative Operators

Every magnitude of the problem with qualitative knowledge has its own unary operators defined. Let U_x be the unary operators for a variable x, i. e., $U_x = \{VN_x, MN_x, LN_x, AP0_x, LP_x, MP_x, VP_x\}$. They denote for x its qualitative labels: *very negative, moderately negative, slightly negative, approximately zero, slightly positive, moderately positive*, and *very positive* respectively.

The transformation rule for a unary operator is

$$op_u(e) \equiv \begin{cases} e - r = 0 \\ r \in I_u \end{cases} \quad (2)$$

being r a new generated variable, and I_u the interval associated with operator op_u which is established in accordance with [13].

3.2 Binary Qualitative Operators

Let e_1, e_2 be two arithmetic expressions. A binary qualitative operator $b(e_1, e_2)$ denotes the qualitative order relationship between e_1 and e_2. These operators are classified into:

- Operators related to the difference $\geq, =, \leq$. The following transformation rules are applied:

$$e_1 = e_2 \equiv e_1 - e_2 = 0 \quad (3)$$

$$e_1 \leq e_2 \equiv \begin{cases} e_1 - e_2 - r = 0 \\ r \in [-\infty, 0] \end{cases} \quad (4)$$

$$e_1 \geq e_2 \equiv \begin{cases} e_1 - e_2 - r = 0 \\ r \in [0, \infty] \end{cases} \quad (5)$$

- Operators related to the quotient $\ll, -<, \sim, \approx, \gg, Vo, Ne, \ldots$. The applied transformation rule is

$$op_b(e_1, e_2) \equiv \begin{cases} e_1 - e_2 * r = 0 \\ r \in I_b \end{cases} \quad (6)$$

being r a new variable and I_b the interval associated to op_b in accordance with [13].

4 Query/classification language

We propose a language to carry out queries and to classify with labels a time-series database T. Therefore, this language allows us to classify the behaviour patterns of the system.

Table 1: Queries abstract syntax.

Q:	$\forall r \in C \bullet P$	P:	P_b
|	$\exists r \in C \bullet P$	|	$P \wedge P$
		|	$P \vee P$
		|	$\neg P$
P_b :	P_d	P_d :	EQ
|	O_T	|	CL
|	$e_b(O_L([F], \{L\}))$	|	...
O_T :	*always F*	O_L :	*increase*
|	*sometime F*	|	*decrease*
|	*always F before F*	|	*periodic*
|	*sometime F until F*	|	*length*
|	...	|	...
F :	F_b	F_b :	e_b
|	$F \& F$	|	$e \in I$
|	$F \mid F$	|	$u(e)$
|	$F \Rightarrow F$	|	$b(e_1, e_2)$
|	$!F$		
C :	$[r \in C, automatic]$		
|	$[r \in C, P_A] \Rightarrow A, e_{n1}, \ldots$		
|	$\ldots$		

4.1 Abstract Syntax

Let T be the time-series database and let r be every trajectory in this database. The abstract syntax of the language is in the table 1.

A query Q on the database T may be a quantifier $\forall, \exists$ applied to T. This query try to determine if all the trajectories ($\forall$) or if at least one ($\exists$) verify the property P. This property may be formulated by means of the composition of other properties using the Boolean operators $\wedge, \vee, \neg$ and its result is the application of these operators among the partial properties.

A basic property P_b may be: a predefined property P_d, a temporary operator O_T, or a Boolean expression e_b applied to a list operator $O_L([F], \{L\})$. This operator returns the list L of points or intervals of the trajectory which verify the formula F.

A temporary operator O_T is used to describe properties in a concrete time of a trajectory, or to compare among different times of a trajectory, or to establish a sequence of behaviours of a trajectory.

A defined property P_d is one whose formulation is automatic. They are queries commonly used in dynamic systems. There are two predefined: EQ, which is verified when the trajectory ends up in a stable equilibrium; and CL that it is verified when it ends up in a limit cycle.

A formula F may be composed of other formulas combined by means of Boolean operators $\&, |, !$ and its result is the application of these operators among the partial formulas. A basic formula F_b may be: a Boolean expression e_b, or if a numeric expression e belongs to an interval I, or a unary u or binary b qualitative operator.

Finally, a classification rule C is formulated: automatically $[r \in C, \mathit{automatic}]$, or according with a specific criterion as a set of basic queries $[r \in C, P_L]$ with labels L and possibly other expressions e.

4.2 Semantics

The semantics of every proposed statement is translated into a query of the database. A query $\forall\ r \in C \bullet P$ is *true* when all trajectories $r \in C$ verifies property P. To prove that an $\exists$ statement is *true*, it is necessary to find at least one trajectory $r \in T$ that verifies the property P.

A property P, which is formulated by means of the application of Boolean operators $\wedge, \vee, \neg$, is *true* when the result of the application of these operators among the partial properties is true.

The result of the evaluation of a temporary operator O_T depends on its semantics. For example, *always* F returns a *true* value if all the values of r verify F. If this operator is *sometime* F returns a *true* value when at least a value of r verifies F. The semantic of these temporary operators are described in accordance with [9].

Let $e_b(O_L([F], \{L\}))$ be a basic property. This property is *true* if the Boolean expression applied to the list operator O_L returns a *true* value. The operator O_L returns the intervals or points of a trajectory which verify a formula F. In order to evaluate a formula F, it is necessary to substitute its variables by their values. These values are obtained from T.

Let $[r, P_A] \Rightarrow A, e_{A1}$ be a classification rule. A trajectory $r \in T$ is classified with the label A if

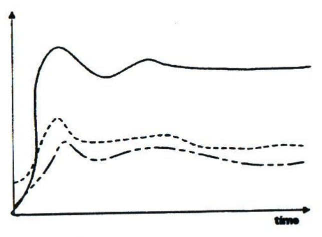

Figure 2: Similar qualitative behaviours.

it verifies property P_A. It is also included into the database for this trajectory the result of the evaluation of e_{A1}. This process is repeated with every classification rule. When the classification rule is $[r \in C, \mathit{automatic}]$, it is necessary to apply clustering techniques in order to classify the database automatically.

5 Behaviour patterns

A behaviour pattern is each one of the possible behaviours that may appears in a dynamic system from its initial state until its final state. A qualitative model has a group of qualitative behaviours. Each one of them is a behaviour pattern. In figure 2 there are three different quantitative trajectories but they follow the same qualitative pattern: ***begins with a value next to zero, then it goes growing until it reaches a maximum and next it oscillates until arriving to a positive value where it remains stable.***

It is possible, that there are different ways to reach a stationary state. Therefore, it is necessary to further out the study with the transient state to discover these behaviours.

6 Labeling of the database

In order to obtain the patterns, the database must be labeled. The labeling assigns one or more labels to every trajectory of the database. These labels may be assigned by a certain criterion provided by the user or by means of

an automatic process. The data normalization is necessary to classify automatically the database. This normalization helps us to assign the labels to the trajectories by means of clustering algorithms.

The result of the classification is a bidimensional dynamic array where the rows are the trajectories and the columns are the parameters, the labels and optionally other expressions.

6.1 Labelling with Specific Criteria

Let T be the database for the following classification rules: $C = \{C_1, C_2, ...\}$:

$$\begin{array}{l} C_1 \equiv [r \in T, P_{L_1}] \Rightarrow L_1, e_1 \\ C_2 \equiv [r \in T, P_{L_2}] \Rightarrow L_2, ... \\ ... \end{array} \qquad (7)$$

being r the trajectories of T, P_{L_i} the i-th property of the classification rule C_i, L_i labels and e expressions.

A label L is assigned to trajectory r when it verifies the property P. In order to apply the behaviour pattern algorithms, it is necessary that all trajectories have at least one label, in other case it is assigned an empty label.

6.2 Automatic Labelling

The automatic labelling classifies the database in accordance with the different behaviours of the dynamic system. It is necessary to supply the *similarity degree* to carry out this classification. It is a number in the interval [0,1] and its functionality is explained bellow.

Let T be the trajectories database. The idea is to find the different patterns that appear in T. Every trajectory is classified with a label L in accordance with its behaviour pattern.

Each trajectory $r \in T$ contains a temporary sequence for every state variable, therefore, a trajectory is a vector of time sequences. It is necessary to apply a clustering algorithm on these time series of the database to discover the behaviours. These sequences may be previously normalized. Any clustering algorithm needs to define a metric among the elements to be classified.

6.3 NORMALIZATION

Two trajectories with a similar behaviour may have a distance next to 0. In figure 2, the three trajectories follow the same qualitative pattern, but if we apply among them a distance like the Euclidean, it is concluded that they follow different behaviours. It is necessary to normalize these data to obtain the same pattern for similar qualitative behaviours. Therefore, these data will be scaled, translated and weighted.

6.3.1 Scaling

Let r be a trajectory, and let $x_1, ..., x_k$ be the values stored in the database of the state variables of r. These values will be scaled to the interval $[0, 1]$. For every variable $x \in r$, it is necessary to obtain its maximum and minimum values x_{max}, x_{min} respectively. The value $x_r = x_{max} - x_{min}$ is the range of values of x. Let x_t be the value of x at time t. Let x'_t be the escalated value. It is obtained by means of the expression

$$x'_t = \frac{x_t - x_{min}}{x_r} \qquad (8)$$

6.3.2 Translation

The trajectories $r_2, ..., r_N$ are translated to the first trajectory r_1 with $T_1 = t_2 - t_1$. Let s be a trajectory $s \in \{r_2, ..., r_N\}$. For each trajectory s is defined $T_s = s_2 - s_1$. The expression T_i is: the time between two consecutives maximum points in oscillate trajectories, in other case it is the wide time.

The translation t_s of s is obtained by the following expression:

$$t_s = \frac{t_1 s_2 + t_2 s_1 - t_1 T_s}{T_1} \qquad (9)$$

6.3.3 Weighting

Once every trajectory has been translated, it is not possible to compare them directly. Therefore, it is necessary to interpolate these values. Let r be a fixed trajectory and let s be the one that will be interpolated. Let t_s be the value obtained by the translation (9). Let t_i, t_j be instant of time being $t_i = t_0 + k\Delta t$ and $t_j = t_i + \Delta t$, where k is a natural number that verifies that $t_s \in [t_i, t_j]$. Let y_i, y_j be the translated values stored in the database for t_i, t_j respectively. We are interested in calculating the value t_s. There are several ways to calculate it: linear approximations, β-splins, ... In this paper, the value y_s is calculated by means of a linear approximation as follows:

$$y_s = y_i + \frac{y_j - y_i}{\Delta t}(t_s - t_i) \qquad (10)$$

being $\Delta t = t_j - t_i$. This is the value of trajectory s that may be compared with the corresponding on the trajectory r.

7 Distance among trajectories

Let T be a normalized database. Let D be a distance matrix among the trajectories of T. It is a triangular matrix with its main diagonal equal to zero. This matrix is necessary for the clustering techniques.

Let $\delta(r_i, r_j)$ be the distance between two trajectories. The trajectories are time series, therefore is appropriate to calculate δ by means of the Fourier coefficients $a_0, a_1, \ldots$. There are several reasons to elect the Fourier coefficients [2]: the distance is preserved, they are easy to calculate, they concentrate the signal energy in a few coefficients and there is an algorithm *Fast Fourier Transform (FFT)* that calculate this coefficients efficiently. The features of the original trajectory are obtained with a few Fourier coefficients [2].

Besides the magnitude, there are other features that are interesting to take into account from a qualitative perspective: the shape (first derivative) and the concavity (second derivative). In this paper, the distance δ between two trajectories r_i, r_j is defined as an expression depending on: Fourier coefficients, weights, variable magnitudes and first and second variable derivatives, as follows:

$$\delta_x(r_i, r_j) = \begin{cases} p_0|a_{0i} - a_{0j}|^2 + \\ p_1|a_{1i} - a_{1j}|^2 + \\ p_2|a_{2i} - a_{2j}|^2 + \\ p_3 \int |x_i(t) - x_j(t)|^2 dt + \\ p_4 \int |\dot{x}_i(t) - \dot{x}_j(t)|^2 dt + \\ p_5 \int |\ddot{x}_i(t) - \ddot{x}_j(t)|^2 dt \end{cases} \qquad (11)$$

being x the state variables, $p_0, p_1, \ldots, p_5$ weights, a_{uv} the u-th Fourier coefficient, x_i, x_j normalized values stored in the database for the trajectories i, j respectively, and $\dot{x}_i, \dot{x}_j$ and $\ddot{x}_i, \ddot{x}_j$ first and second derivative of x_i, x_j values respectively. Weights are introduced to take precedence over the magnitudes, the shapes or the concavity of the trajectories.

8 Clustering and decision rules

Clustering is a discovery process in data mining. It groups a set of data in a away that maximizes the similarity within clusters and minimizes the similarity between two different clusters. These discovered clusters can help to explain the features of the underlying data distribution. In recent years, a number of clustering algorithms for databases have been proposed: $DBScan$ [4], $CURE$ [5], ...

The scalable clustering technique proposed in this paper puts together a trajectory and a label. This label determines the behaviour pattern of the trajectory. First, the distance matrix D among trajectories may be calculated, and it is calculated the mean of D. This mean distance d_{mean} is calculated to know the magnitude of the distance.

Next, a weighted graph G is obtained with the k-neighbours of every trajectory. The vertex of G are the trajectories. The arcs are weighted with the relationship between d_{mean} and the distance between two trajectories. Figure 3 shows examples of graph building: original (a) and with 1- (b), 2- (c) and 3-(d) neighbours.

Some arcs between neighbours vertices of the k-neighbours graph are broken using the similarity degree a. The number of clusters depend-

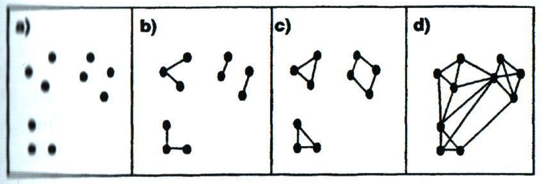

Figure 3: Graph with k-neighbours.

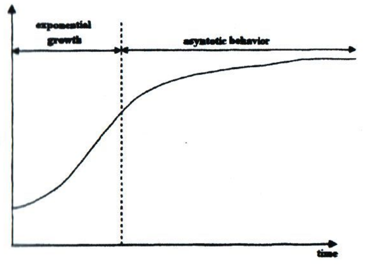

Figure 4: Logistics growth model.

ing on this number. Those whose weight w is less than

$$w < \frac{a * d_{mean}}{100} \qquad (12)$$

Finally, the obtained connected graphs represent every different behaviour pattern of the dynamic system. All the trajectories of every connected graph are classified with the same label.

Once the database has been labelled, the pattern behaviour of the system is represented by means of a set of hierarchical decision rules. These rules are obtained using the program $COGITO$ [1] to the bidimensional dynamic array described in the section 6 of this paper.

9 Application

It is very common to find growth processes in which an initial phase of exponential growth is followed by another phase of approaching to a saturation value asymptotically (figure 4). These are given the following generic names: logistic, sigmoidal, and s-shaped processes. This growth is exhibited by systems for which exponential expansion is truncated by the limitation of the resources required for this growth.

In literature, these models have been profusely studied. They abound both in natural processes, and in social and socio-technical systems. They appear in the evolution of bacteria, in mineral extraction, in world population growth, in economic development, ... There is a bimodal behaviour pattern attractor: A stands for normal growth, and O for decay. This phenomenon was first modeled by the Belgian sociologist P. F. Verhulst in relation with human population growth.

Let S be the qualitative model. If we add a delay in the feedback paths of S, its differential equations are:

$$\Phi \equiv \begin{cases} \dot{x} = x(n\,r - m), y = delay_\tau(x), \\ \quad x > 0, \quad r = h_1(y), \\ h_1 \equiv \{(-\infty, -\infty), +, (d_0, 0), +, (0, 1), \\ \qquad +, (d_1, e_0), -, (1, 0), -(+\infty, -\infty)\} \end{cases}$$

being n the increasing factor, m the decreasing factor, and h_1 a qualitative function with a maximum point at (x_1, y_0). The initial conditions are:

$$\Phi_0 \equiv \begin{cases} x_0 \in [LP_x, MP_x], \\ LP_x(m), LP_x(n), \\ \tau \in [MP_\tau, VP_\tau] \end{cases}$$

where LP, MP, VP are the qualitative unary operators ***slightly positive, moderately positive and very positive*** for x, τ variables.

We would like to know:
1. if an equilibrium is always reached
2. if there is an equilibrium whose value is not zero
3. if all the trajectories with value zero at the equilibrium are reached without oscillations.
4. To classify the database in accordance with the behaviours of the system.

Firstly, it is necessary to define the intervals associated with every qualitative operator:

$$LP_x = [0, 1] \qquad MP_x = [1, 3]$$
$$MP_\tau = [0.5, 4] \quad VP_\tau = [4, 10]$$

The methodology described in [11] is applied to obtain the trajectory database T. Applying the language, the proposed queries are formulated as follows:
1. $\forall r \in T \bullet EQ$
2. $\exists r \in T \bullet (EQ \wedge sometime\ t \simeq t_f \Rightarrow x \not\approx 0)$
3. $\forall r \in T \bullet (EQ \wedge sometime\ t \simeq t_f \Rightarrow x \approx 0 \wedge$
$\qquad length(\dot{x} = 0 \wedge \ddot{x} < 0))$

The list of points where $\dot{x} = 0$ and $\ddot{x} < 0$ is the list with the maximum points. There are no oscillations when its length is 0.

The answers to the proposed questions were:
1. $b_1 = True$, all the trajectories of T reach a stable equilibrium. Therefore, we conclude: *there is no cycle limit*.
2. $b_2 = True$, some trajectories of T reach an equilibrium whose value is not zero. Therefore, this is the first behaviour we have obtained. We know it as *recovered equilibrium*.
3. $b_3 = False$, there are at least two ways to reach this equilibrium: with oscillations (this behaviour is called as *retarded catastrophe*) and the other way is without oscillations (that it is called as *decay and extinction*).

We apply to T the described clustering technique with a similarity degree of $a = 0.1$. This algorithm found the three possible behaviours patters for this system (figure 5).

The obtained results and the behaviour patterns are in accordance with others appeared in the bibliography [3] and [7] where the results were concluded by means of a mathematical reasoning. This circumstance encourages us to continue developing our approach.

10 Conclusions and further work

In this paper, we have presented a way to obtain temporal and semiqualitative behaviour patterns of dynamic systems with qualitative and quantitative knowledge. This approach is based on a transformation process, definition of a query/classification, language on a quantitative behaviours database, and clustering techniques.

In the future, the query/classification language must be enriched with operators for comparing trajectories, spatial operators, etc. Dynamic systems with constraints and with multiple scales of time are also one of our future points of interest.

Acklowledgements This work was partially supported by the Spanish Interministerial Committee of Science and Technology by means of the program TIC98-1635-E.

References

[1] J. Aguilar, J. Riquelme and Toro M., 'Decision queue classifier for supervised learning using rotated hyperboxes'. *Lecture Notes in Artificial Intelligence* **1484**, 326–336, 1998.

[2] R. Agrawal, K.I. Lin, H.S. Sawhney and K. Shim, 'Fast similarity search in the presence of noise, scaling and translation in time series databases', *The 21th VLDB Conference*, Zurich (Switzerland), (1995).

[3] J. Aracil, E. Ponce and L. Pizarro, 'Behavior patterns of logistic models with a delay' *Mathematics and computer in simulation* **44**, 123–141, (1997).

[4] M. Ester, H.P. Kriegel, J. Sander and X. Xu, ´A density-based algorithm for discovering clusters in large spatial database with noise', *International Conference on Knowledge Discovery in Databases and Data Mining* (KDD-96), Portland (USA), 226–231, (1996).

[5] S. Guha, R. Rastogi and K. Shim 'CURE: And efficient clustering algorithm for large databases'. *Proceedings ACM SIGMOD International Conference Management of Data*, New York (USA), 73–84, (1998).

[6] H. Kay, *Refining imprecise models and their behaviors*, Ph.D. dissertation, Texas University (USA), 1996.

[7] M. Karsky, J.C. Dore and P. Gueneau, 'Da la possibilité d'apparition de catastrophes différès'. *Ecodecision* **6**, 1992.

[8] B.J. Kuipers *Qualitative reasoning. Modeling and simulation with incomplete knowledge*, The MIT Press, 1994.

[9] U.W. Lipeck and G. Saake G. 'Monitoring dynamic integrity constraints based on temporal logic' *Information Systems*, **12(3)**, 255–269, 1987.

[10] J.A. Ortega, R.M. Gasca and M. Toro, 'Including qualitative knowledge in semiqualitative dynamical systems'. *Lecture Notes in Artificial Intelligence* **1415**, 329–338, 1998.

[11] J.A. Ortega, R.M. Gasca and M. Toro, 'A semiqualitative methodology for reasoning about dynamic systems'. *The 13th International Workshop on Qualitative Reasoning* Loch Awe (Scotland), 169–177, (1999).

[12] J.A. Ortega, 'Patrones temporales de comportamiento en modelos semicualitativos con restricciones'. *Ph.D. University of Seville*, 2000.

[13] L. Travé-Massuyès, P. Dague and F. Guerrin, *Le raisonement qualitativ pour les sciences de l'ingénieur*. Hermes Ed., 1997.