# Abstracting Continuous System Behaviours into Timed Automata: Application to Diagnosis of an Anaerobic Digestion Process

**Arnaud Hélias[1,2], François Guerrin[1]** and **Jean-Philippe Steyer[2]**

**Abstract**. Abstracting 'continuous' system behaviours into discrete-event representations (*i.e.*, timed automata) for diagnosis purposes is demonstrated in this paper. As complex system dynamics are often partially known, the resulting imprecision on continuous variables is represented by means of intervals partitioning the state space according to landmarks defined by expert knowledge. Based on a continuous model simulation, an algorithm assigns discrete labels to landmark crossing by continuous variables, then, generates a timed automaton that can be further analysed by a model-checker. This procedure allows one to summarize a continuous system simulation output as a set of transitions among discrete states with qualitative interpretation (*e.g.*, high, medium, low). In order to reduce explosion in the number of states, the generated timed automaton is specifically determined according to the property of interest for the user (*e.g.*, reachability of some unwanted states). This approach has been applied to predict possible dysfunctions of a wastewater treatment process and validated using real-life data.

## 1 INTRODUCTION

More than control, fault detection, isolation and analysis in biological processes have become a challenging research area, namely for anaerobic digestion processes for which the counterpart of efficiency is probably the system instability in some circumstances.

This paper focuses on hybrid (continuous/discrete) system modelling with application to forecasting the functioning of a biological process. In order to represent the process, we have, on the one hand, a continuous model, on the other hand, expert knowledge, generally represented in a discrete form. By 'continuous' we mean a model whose variables take value on a continuum (e.g., the set of real numbers) ; this does not preclude their dynamics from exhibiting discontinuities or not. In contrast, 'discrete' denotes a model with variables taking value on a finite set, generally with a small number of elements. When, within a system, continuous variables interact with discrete ones, a classical approach to analyse this interaction is to represent the whole system within a unique formalism. In the qualitative reasoning and hybrid dynamical system communities, this problem was often approached. However, Struss [1] underlined three main difficulties: (i) explosion in the number of discrete states, (ii) rounding errors in intermediate variables and (iii) landmarks determination. Moreover, time is not often clearly represented whereas the interest of timed discrete models has been demonstrated for diagnosis purposes (*e.g.* see, [2]).

[1] CIRAD, 97408 Saint-Denis, Reunion Island (France) email: {helias, guerrin}@cirad.fr.
[2] Laboratory of Environmental Biotechnology, INRA, 11100 Narbonne, France email: steyer@ensam.inra.fr.

This study is based on a discrete-event representation similar to those developed by [3-5] with the following particularities:

- continuous system dynamics are partially known and the resulting imprecision is represented by numerical intervals;
- to tackle the issue of combinatorial explosion, our idea is to develop "on the fly" a timed discrete representation according to the system's initial states and the properties to be checked.
- landmarks defined by expert knowledge are taken as inputs of the approximate representation of the process.

The paper is organized as follows. After a brief recall of the timed automata formalism and model-checking in section 2, we briefly present the abstraction procedure in section 3. This procedure is then applied, section 4, to forecast the functioning of an anaerobic digestion process.

## 2. MODELLING FORMALISMS

### 2.1 Timed automata

Introduced by Alur and Dill [6], a timed automaton is composed of a finite state machine and the expression of continuous time. This formalism allows one to define temporal constraints using variables named *clocks*, these constraints being possibly associated with the locations or the edges of the automaton.

$x \in X$, with $X$ a finite set of clocks, is a clock whose value grows linearly with time. A clock constraint is an atomic constraint conjunction. The set $F(X)$ of atomic constraints $j$ is defined by the grammar $j := x \# c \mid x - y \# c \mid j_1 \wedge j_2$ with $x$ and $y$ two clocks, $c \in \mathbb{Q}$ a constant and $\#$, a relation symbol from the set $\{<, \leq, =, \geq, >\}$. A timed automaton $G = (S, P, X, E, A, Inv)$ is defined by:

- $S$ a finite set of locations representing all the possible states of the system, with $s_0 \in S$ the initial location;
- $P$ a map which associates to each $s \in S$, a set of atomic propositions valid in this location;
- $X$ a finite set of clocks;
- $E$ a finite set of labels;
- $A \subseteq S \times E \times F(X) \times 2^X \times S$ a finite set of edges; each edge is a tuple $(s, e, j, d, s')$ with $s \in S$ the starting location and $s' \in S$ the destination location, $j \in F(X)$ a clock constraint named guard that must be satisfied to trigger the transition and $d \subseteq X$ the clock subset that must be reinitialised during the discrete transition;
- $Inv(s) : S \rightarrow F(X)$ a map that associates to each location a clock constraint $j'$, named invariant. The system can stay in this location as long as the invariant remains true.

## 2.2 Model-checking with TCTL

Model-checking is one of the most popular techniques to automatically check the properties of a system. Model-checking methods are based on an algorithm confronting a model, describing the possible behaviours of a system, with a property expressed in a specific language (e.g. the behaviour expected from a given initial state). In the last decade, model-checkers have been extended to incorporate real-time representation using the formalism of timed automata.

The timed computational tree logic (TCTL) [7] allows one to introduce temporal information into formulae which are defined by the grammar $f := p \mid \neg f \mid \mathsf{x} - \mathsf{y} \# c \mid \mathsf{x} \# c \mid f_1 \vee f_2 \mid \exists \Diamond_{\# c} f \mid \forall \Diamond_{\# c} f$ with $p$ an atomic proposition, $\mathsf{x}$ and $\mathsf{y}$ two clocks, $c \in \mathbb{N}$ a constant and $\#$ a relation symbol from the set $\{<, \leq, =, \geq, >\}$. "$\exists\Diamond$" means "There exists at least one sequence of locations in which some property holds true" and "$\forall\Diamond$" means "For all the sequences of locations some property holds true". Model-checking tools like *Kronos* [8], *UPPAAL* [9,10] have been developed for timed discrete-event systems. In our approach, *Kronos* is coupled with the Matlab® environment, which is used for simulating the continuous part of the system. Kronos allows TCTL formulae to be checked onto one or more timed automata and, for properties like the reachability of some state, to obtain all the paths and clock constraints associated to the target states. Note that, by providing the time instants at which these states are reached, Kronos delivers more information than the Boolean results of classical model-checkers.

# 3. DISCRETE ABSTRACTION OF CONTINUOUS SYSTEM BEHAVIOURS

## 3.1 Continuous system description

### 3.1.1 Ordinary differential equations

The system considered here is a non-linear continuous system that can be described by the ordinary differential equation (ODE):

$$\begin{cases} \dot{x} = f(x, z) \\ x(t_0) = x_0 \end{cases} \tag{1}$$

where $x \in X \subseteq \mathbb{R}^n$, $X$ is an $n$-dimensional state space, $z = g(p,t)$, $z \in Z \subseteq \mathbb{R}^m$, $Z$ an $m$-dimensional input space and $p$ a set of parameters. However, it is assumed that we have only a partial knowledge of the initial state $x(t_0)$ and the input values $z$ that can be estimated by the intervals $(x_i^-, x_i^+) \in X_i^2$ and $(z_j^-, z_j^+) \in Z_j^2$ (with $X_i \in X$ and $Z_j \in Z$, $i = 1,...,n$, $j = 1,...,m$) defined by

$$\forall j, \forall t, z_j^-(t) \leq z_j(t) \leq z_j^+(t) \tag{2}$$

$$\forall i, x_i^-(t_0) \leq x_i(t_0) \leq x_i^+(t_0) \tag{3}$$

### 3.1.2 Imprecision representation

Let us define $z^<$ and $z^>$ according to the values of the derivatives as follows:

$$\begin{aligned} f_i(x, z_k, z_j^-) < f_i(x, z_k, z_j^+) &\Rightarrow z_j^< = z_j^-, z_j^> = z_j^+ \\ f_i(x, z_k, z_j^-) > f_i(x, z_k, z_j^+) &\Rightarrow z_j^< = z_j^+, z_j^> = z_j^- \end{aligned} \tag{4}$$

with $k = 1,...,m$ and $k \neq j$. From (4) we can rewrite (1) as a double ODE system:

$$\begin{cases} \dot{x}^- = f(x^-, z^<) \\ x^-(t_0) = x_0^- \\ \\ \dot{x}^+ = f(x^+, z^>) \\ x^+(t_0) = x_0^+ \end{cases} \tag{5}$$

with the property:

$$\forall t, \forall i, x_i^-(t) \leq x_i(t) \leq x_i^+(t) \tag{6}$$

Eq. (5) approximates Eq. (1) while accounting for imprecision on the initial state (*cf.* Eq. (2)) and on the system inputs (*cf.* Eq. (3)) if a mathematical property called *cooperativity* is verified on the system dynamics [11]. This property simply states that the off-diagonal elements of the Jacobian matrix of a dynamical system are positive or equal to zero. The interested reader will find additional details on these aspects in [12].

## 3.2 Landmarks

To represent the continuous system dynamics with a discrete formalism, the first step is to translate the continuous state space into a discrete state space. To this end, landmarks are defined and each $x_i$ domain is divided into a finite number of intervals that can be interpreted as qualitative states. In the example described here, the landmarks result from expert knowledge; *e.g.*, a level can be qualitatively characterised as "low", "medium", or "high". Figure 1 shows the partition of a three-dimensional space with three landmarks for each dimension.

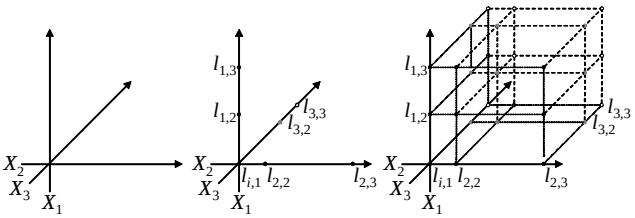


**Figure 1.** Translation of a continuous state space (a) by landmarks (b) into a discrete space state (c).

The basic ideas are (i) to take the faces of the cells partitioning the state space as discrete states (not the cells themselves) and (ii) to determine the transitions between states from the analysis of an output of the continuous system simulated within each cell, as it is done in [4]. Using such an approach, Kowalewski et al. defined a grid on each cell's faces and simulated all the trajectories issued from or bounded to each grid-point. In comparison, our approach is focused on the propagation of the interval $\left[x^-, x^+\right]$ within each cell by simulating Eqs. (5).

## 3.3 Trajectories within a cell

3.3.1 Landmark crossing

Let $t$ be the time instant when the system (1) reaches $v$, a landmark value defined in the $i^{th}$ dimension, *i.e.*, $x_i(t) = v, v \in \mathbb{R}$. Because $x_i$ is estimated by the interval $\left[x_i^-, x_i^+\right]$, the time window of crossing the $v$ landmark can be described by $t^-$ and $t^+$ such that $x_i^-(t^-) = x_i^+(t^+) = v$.

From Eqs. (5), it comes: $\left(\dot{x}_i^-(t^-) > 0\right) \wedge \left(\dot{x}_i^+(t^+) > 0\right) \Rightarrow t^+ < t^-$ and $\left(\dot{x}_i^-(t^-) < 0\right) \wedge \left(\dot{x}_i^+(t^+) < 0\right) \Rightarrow t^+ > t^-$.

Let $t_{min} = \min(t^-, t^+)$, $t_{max} = \max(t^-, t^+)$ and $F$ the map that associates a label to each landmark crossing defined by:

$$F(v) \rightarrow \begin{cases} \text{"}v^{\Delta}\text{" if } t_{min} = t^+ \\ \text{"}v^{\nabla}\text{" if } t_{min} = t^- \end{cases} \quad (7)$$

with the symbols $\Delta$ and $\nabla$ denoting landmark crossing with increasing and decreasing trends, respectively.

3.3.2 Timed automaton representation

The system trajectory between two landmarks $w$ and $v$ is represented by the time window $\left[t_{min}, t_{max}\right]$ which can be modelled in the timed automata formalism (*cf.* figure 2 where $w < v$) by:

- $s_1$ and $s_2$, two locations with associated propositions "$w^{\Delta}$" and "$v^{\Delta}$" respectively,
- $x$, a clock,
- $x \le t_{max}$, the invariant of $s_1$,
- $\left(s_1, \varnothing, x \ge t_{min}, \varnothing, s_2\right)$, the edge.

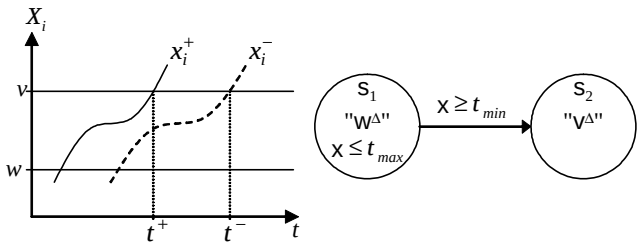


**Figure 2.** Timed automaton representing a trajectory from landmark $w$ to $v$.

## 3.4 Global timed automaton

Starting from a simulation run of the continuous model on a defined period of time, for each dimension, the continuous system dynamics is approximated by a timed automaton where:

- each landmark crossing is assigned a location and a proposition with crossing characteristics and trend by means of Eq. (7),
- invariants are defined by the latest time points $t_{max}$ at which landmark crossing may occur,
- guards of edges are defined by the earliest time points $t_{min}$ at which landmark crossing may occur,
- the clock set is a singleton.

Using this approach and performing the synchronised product with Kronos (more details are in [8]) of all the timed automata generated for each dimension of the system, yields a single timed automaton, with one clock without reinitialisation, globally representing the system dynamics for the simulated period. The complete procedure can be found in [13].

# 4 APPLICATION TO AN ANAEROBIC DIGESTION PROCESS

The above described procedure is applied to an anaerobic digestion process used in wastewater treatment. The aim is to forecast the possible dysfunctions of the process by using a discrete approximation of a mechanistic model together with partial knowledge of the system's dynamics and parameters.

## 4.1 Process description

Anaerobic digestion is a biological process used for carbon removal from wastewater. The principle is to transform organic matter into biogas (*i.e.*, a mixture of methane $CH_4$ and carbon dioxide $CO_2$) in absence of oxygen. This process has advantages like:

- methane may be used as a renewable energy source,
- sludge production is low,
- anoxic reactions have a low energetic demand.

The Laboratory of Environmental Biotechnology of INRA in Narbonne (France) runs a 1 $m^3$ fixed-bed anaerobic reactor since 1997 [14]. This semi-industrial pilot has been used for modelling, diagnostic and control purposes. This complex process involves many parallel and serial reactions that consume a substrate by means of a succession of bacteria communities. However, the process can be summarised by (i) an acidogenic phase, where a first bacterial biomass ($X_1$) converts organic matter ($S_1$) into volatiles fatty acid (VFAs, $S_2$) and (ii) a methanogenic phase, where another biomass ($X_2$) converts $S_2$ into methane and $CO_2$:

$$k_1 S_1 \xrightarrow{r_1} X_1 + k_2 S_2 + k_4 CO_2 \quad (10)$$

$$k_3 S_2 \xrightarrow{r_2} X_2 + k_5 CO_2 + k_6 CH_4 \quad (11)$$

with $k_1$- $k_6$ the yield coefficients and $r_1$, $r_2$ the reaction rates.

## 4.2 Process model

4.2.1 Mass-balance model

A classical mass-balance analytical model is used [15]:

$$\begin{cases} \dot{X}_1 &= (m_1 - a\,D) X_1 \\ \dot{X}_2 &= (m_2 - a\,D) X_2 \\ \dot{S}_1 &= D(S_{1in} - S_1) - k_1 m_1 X_1 \\ \dot{S}_2 &= D(S_{2in} - S_2) + k_2 m_1 X_1 - k_3 m_2 X_2 \\ \dot{Z} &= D(Z_{in} - Z) \\ \dot{C} &= D(C_{in} - C) - q_{CO_2} + k_4 m_1 X_1 + k_5 m_2 X_2 \end{cases} \quad (12)$$

with $Z$ the total alkalinity, $C$ the total inorganic carbon concentration, $q_{CO_2}$ the molar flow of $CO_2$, $D$ the dilution rate by

the input flow, $a$ the fraction of bacteria in the liquid phase, $k_1$ to $k_6$ the yield coefficients of the reactions, $m_1$ and $m_2$ the growth rate of bacteria in the acidogenic and methanogenic phases and $S_{1in}$, $S_{2in}$, $Z_{in}$ and $C_{in}$ the concentrations of elements in the influent.

From this ODE system, and assuming (i) the methane molar flow is only dependent upon the methanogenic biomass activity, (ii) the biogas is mainly composed of methane and $CO_2$ and (iii) the ideal gas law, the following variables can be determined:

$$q_{\mathrm{CH_4}} = k_6 m_2 X_2 \tag{14}$$

$$q_{gas} = q_{\mathrm{CH_4}} + q_{\mathrm{CO_2}} \tag{15}$$

$$q_{\mathrm{CO_2}} = \frac{k_6 m_2 X_2 P_{\mathrm{CO_2}}}{P_{\mathrm{T}} - P_{\mathrm{CO_2}}} \tag{16}$$

$$\mathrm{COD} = S_1 + k_C S_2 \tag{17}$$

with $P_T$, the total pressure in the reactor, $P_{CO_2}$, the partial pressure of $CO_2$, COD, the chemical oxygen demand and $k_c$, a conversion constant.

#### 4.2.2 Model simplification

Using the model and introducing imprecision in it require that simplifications are made to comply with the property (6). First of all, with respect to the length of the simulation period (some hours or some days), the biomasses $X_1$ and $X_2$ are assumed to be constant. Moreover, if in [15], the growth rate $m_2$ is modelled by a Haldane-type kinetics (inhibition of the reaction by $S_2$ is assumed), we use here a Monod-type kinetics in the form $m_2 = m_{2_{\max}} \frac{S_2}{S_2 + K_{S_2}}$ no inhibition), with $m_{2_{max}}$ the maximum bacterial growth rate and $K_{S_2}$ the half-saturation constant associated to $S_2$. The acidogenic growth rate $m_1$ is also taken as a Monod-type kinetics

Finally Eqs. (13-17) are rewritten to yield:

$$\begin{cases} \dot{x} = f(x,z,t) \\ y = g(x,z,t) \end{cases} \tag{18}$$

where $x^T = [S_1, S_2, Z, C]$, $x \in \mathbb{R}_+^4$, $y^T = [\mathrm{DCO}, q_{gaz}]$, $y \in \mathbb{R}_+^2$ and $z^T = [S_{1in}, S_{2in}, Z_{in}, C_{in}, D, P_{\mathrm{CO_2}}]$, $z \in \mathbb{R}_+^6$. $\dot{x} = f(x,z,t)$ is defined by:

$$x = \begin{cases} \dot{S}_1 &= D(S_{1in} - S_1) - k_1 m_1 X_1 \\ \dot{S}_2 &= D(S_{2in} - S_2) + k_2 m_1 X_1 - k_3 m_2 X_2 \\ \dot{Z} &= D(Z_{in} - Z) \\ \dot{C} &= D(C_{in} - C) + k_4 m_1 X_1 + k_P m_2 X_2 \end{cases} \tag{19}$$

with $k_P = k_5 - k_6 \frac{P_{\mathrm{CO_2}}}{P_{\mathrm{T}} - P_{\mathrm{CO_2}}}$. $y = g(x,z,t)$ is defined by:

$$y = \begin{cases} \mathrm{COD} &= S_1 + k_C S_2 \\ q_{gas} &= k_6 m_2 X_2 \frac{P_T}{P_T - P_{\mathrm{CO_2}}} \end{cases} \tag{20}$$

#### 4.2.3 Construction of the system bounds

The off-diagonal elements of the Jacobian matrix of (19) are always positive or null apart of $\frac{\partial f_4(x,z,t)}{\partial x_2}$ for which the condition

$$\frac{P_{\mathrm{CO_2}}}{P_T} \le \left( \frac{k_5}{k_5 + k_6} \right) \tag{21}$$

must be checked. Note that condition (21) depends only on parameters and inputs. Subject to the satisfaction of condition (21), the ODE system (19) is cooperative, so, it is possible to bound by intervals the input concentration values and obtain:

$$\begin{cases} \dot{S}_1^+ &= D(S_{1in}^+ - S_1^+) - k_1 m_1^+ X_1^- \\ \dot{S}_2^+ &= D(S_{2in}^+ - S_2^+) + k_2 m_1^+ X_1^+ - k_3 m_2^+ X_2^- \\ \dot{Z}^+ &= D(Z_{in}^+ - Z^+) \\ \dot{C}^+ &= D(C_{in}^+ - C^+) + k_4 m_1^+ X_1^+ + k_P m_2^+ X_2^+ \\ \dot{S}_1^- &= D(S_{1in}^- - S_1^-) - k_1 m_1^- X_1^+ \\ \dot{S}_2^- &= D(S_{2in}^- - S_2^-) + k_2 m_1^- X_1^- - k_3 m_2^- X_2^+ \\ \dot{Z}^- &= D(Z_{in}^- - Z^-) \\ \dot{C}^- &= D(C_{in}^- - C^-) + k_4 m_1^- X_1^- + k_P m_2^- X_2^- \end{cases} \tag{22}$$

Given that $\forall i, x_i^-(t) \le x_i(t) \le x_i^+(t)$, Eqs. (20) can be approximated by :

$$\begin{cases} COD^+ &= S_1^+ + k_C S_2^+ \\ q_{gas}^+ &= k_6 m_2^+ X_2^+ \frac{P_T}{P_T - P_{\mathrm{CO_2}}} \\ COD^- &= S_1^- + k_C S_2^- \\ q_{gas}^- &= k_6 m_2^- X_2^- \frac{P_T}{P_T - P_{\mathrm{CO_2}}} \end{cases} \tag{23}$$

## 4.3 Discrete abstraction and analysis of the model

#### 4.3.1 Input and initial state of the system

To forecast possible dysfunctions of the plant, it is possible to verify reachability and safety properties so as to envisage the intervention of an operator when the system is potentially in a critical state. Let us assume that the process is in a given situation and that the following events are expected (*cf.* figure 3):

- day 1, low dilution rate and low input concentrations,
- day 2, high dilution rate and low input concentrations,
- day 3, high dilution rate and high input concentrations,
- day 4, low dilution rate and low input concentrations,

Based on expert knowledge, the following relations are used to determine $P_{CO_2}$ (with $k_5$ and $k_6$ values, condition (21) is satisfied):

$$\begin{aligned} D \le 0.04 &\Rightarrow P_{CO_2} = 0.2 \mathrm{mmol.L^{-1}} \\ D > 0.04 &\Rightarrow P_{CO_2} = 0.3 \mathrm{mmol.L^{-1}} \end{aligned} \tag{24}$$

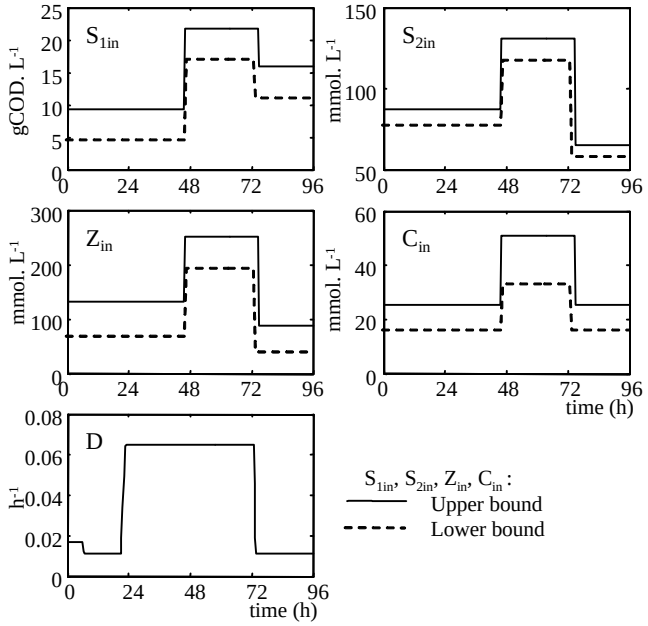


**Figure 3.** Forecasted profiles of inputs used during the experimentation on the process.

4.3.2 Landmarks and associated qualitative states

Figure 4 shows the qualitative states associated with the intervals defined by the landmarks assigned to the state variables domains. The functioning aims are:

- COD concentration (representing the residual pollution after treatment) should be low or normal,
- biogas production should be sufficient to be further used,
- organic overload due to a high substrate concentration S2 should be avoided as it denotes a process dysfunction.

Functioning criteria are established by expert knowledge using the qualitative states defined in figure 4. So, the COD concentration allows the following relations to be established:

- normal COD, normal $S_2$, and normal $q_{gas}$: normal functioning of the reactor,
- low COD: underload, a bigger quantity of organic matter could be treated,
- high COD: the process does not manage well to treat the whole organic matter, pollution still remains after treatment,
- critical COD: an important pollution is present in the effluent.

The VFAs concentration allows organic overload risks to be defined:

- normal $S_2$: very small risk,
- high $Z$ and high $S_2$: small risk,
- high $Z$ and critical $S_2$: risk,
- low $Z$ and high $S_2$ important risk,
- low $Z$ and critical $S_2$: very important risk.

The production of biogas influences the possibilities of using the energy produced by the process:

- low $q_{gas}$: small production of gas,
- high $C$ and high $q_{gas}$: normal production of gas,
- high $C$ and critic $q_{gas}$: normal production of gas, but possible difficulty for its use (the quality of the biogas may be altered),
- low $C$ and high $q_{gas}$: problematic production of gas, difficulty for its use (biogas composition may be modified),
- low $C$ and critic $q_{gas}$: critical production of gas, significant risk during its use.

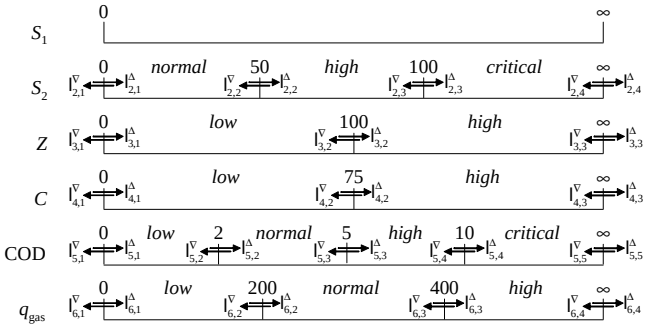


**Figure 4.** Landmark on different state variables with associated propositions.

4.3.3 Timed automaton approximation

The results of four-day simulations with and without imprecision are shown in figure 5 together with experimental data obtain from the pilot plant. If the simulation without imprecision crosses landmarks sometimes earlier or later than on-line data, the double system (23) allows an imprecise but a good and correct estimation of the process features. The model with imprecision intervals is thus more robust for state prediction.

The resulting timed automaton has 1152 locations and 4608 transitions but the set of reachable states has only 118 elements. All the procedure is achieved in 26 seconds on a 1.6 GHz/Windows 2000 computer.

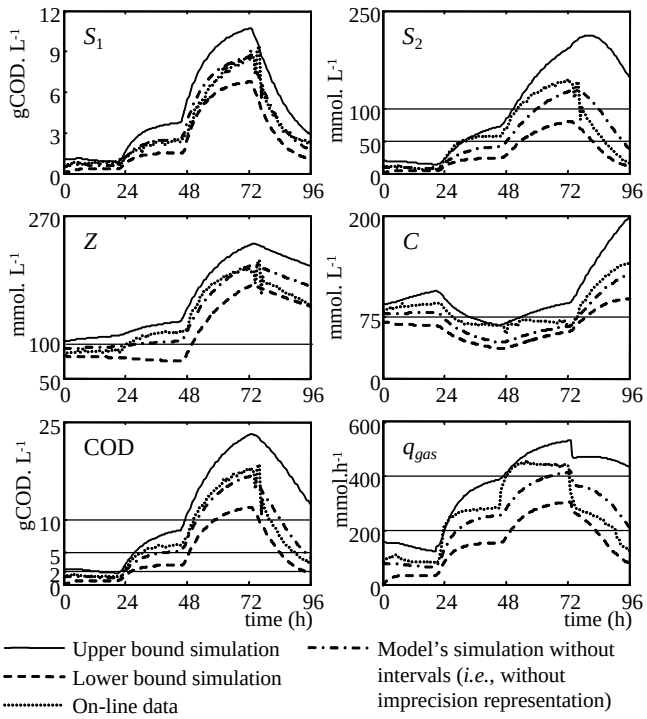


**Figure 5.** Model simulation.

4.3.4 Property checking

Table 1 presents the results of checking the properties enumerated in § 4.3.2. The time of reachability of some state is calculated by the model-checker itself (see [13] for more details). For example, in the first line of Table 1, checking the reachability of the state "normal *COD*, normal $S_2$, and normal $q_{gas}$" is expressed by:

$$\left(l_{2,0} \wedge l_{5,0} \wedge l_{6,0}\right) \Rightarrow \exists \Diamond \left(\left(l_{2,1}^{\Delta} \vee l_{2,2}^{\nabla}\right) \wedge \left(l_{5,2}^{\Delta} \vee l_{5,3}^{\nabla}\right) \wedge \left(l_{6,2}^{\Delta} \vee l_{6,3}^{\nabla}\right)\right) \quad (26)$$

where $l_{i,j}^{\Delta}$ and $l_{i,j}^{\nabla}$ represent crossing of landmark *j* in the *ith* dimension, respectively with increasing and decreasing trends. Eq. (26) means, "from the initial state, is it possible to reach a state corresponding to normal *COD* (i.e., between landmark 1 and 2), normal $S_2$ (i.e., between landmark 2 and 3), normal $q_{gas}$ (i.e., between landmark 2 and 3)?" (*cf.* figure 4). The same approach is used to check all the properties listed in Table 1.

**Table 1.** Property checking on the four-day period.

| Properties | State reachability times (hours) |
|---|---|
| normal *COD*, normal $S_2$, and normal $q_{gas}$ | [24 49] and [84 96] |
| low *COD* | [0 26] and [94 96] |
| high *COD* | [28 59] and [76 96] |
| critical *COD* | [47 96] |
| normal $S_2$ | [0 54] and [81 96] |
| high *Z* and high $S_2$ | [32 96] |
| high *Z* and critical $S_2$ | [51 96] |
| low *Z* and high $S_2$ | [32 50] |
| low *Z* and critical $S_2$ | unreachable |
| low $q_{gas}$ | [0 50] and [83 96] |
| high *C* and high $q_{gas}$ | [53 96] |
| high *C* and critic $q_{gas}$ | [53 96] |
| low *C* and high $q_{gas}$ | [24 79] |
| low *C* and critic $q_{gas}$ | [47 79] |

But it is also possible to check properties like "the system is always in a normal state" by checking the reachability of the negation of the normal states:

$$\left(l_{2,0} \wedge l_{5,0} \wedge l_{5,0}\right) \Rightarrow \exists \Diamond \neg \left(\left(l_{2,1}^{\Delta} \vee l_{2,2}^{\nabla}\right) \wedge \left(l_{5,2}^{\Delta} \vee l_{5,3}^{\nabla}\right) \wedge \left(l_{6,2}^{\Delta} \vee l_{6,3}^{\nabla}\right)\right) \quad (27)$$

Model-checking allows us to verify that this statement is true at all times. As a consequence, no guarantee exists that the system will normally run during the four-day period.

## 5 DISCUSSION AND CONCLUSION

This paper describes an implemented procedure to abstract the dynamics of a continuous system into a timed automaton, where (i) inputs and initial states are estimated by intervals, which comes down to simulate the system behaviour by simulating its upper and lower bounds represented as an extremal ODE system and (ii) the generated behaviour is transformed into a timed automaton (discrete formalism with continuous time), which is made possible by partitioning the state variables' domains by landmarks according to expert knowledge.

The principal characteristics are to take landmark crossing as discrete states and to use this procedure with an initial state, imprecise inputs and a landmark definition set.

The final size of the automaton depends upon the system's dimension, the number of landmarks and landmark crossings. Consequently, this approach is inappropriate when the system oscillates around a landmark value and if a large number of landmarks are defined. Nevertheless, this approach has been devised to be executed "on the fly" when one wishes to verify a specific property of a continuous system. The idea is not to create an automaton that can check all the possible properties but, rather, to create an automaton according to the property to be checked by defining adequate landmarks.

It was shown how this approach can be used to forecast qualitative states of a real anaerobic digestion process by approximating an analytical mass-balance model. An interesting extension would be to couple this representation to the discrete part of a hybrid dynamical system in order to check properties on mixed continuous and discrete elements.

## ACKNOWLEDGEMENTS

This work was partially supported by the Région Réunion and the European Social Fund.

## REFERENCES

[1] P. Struss. 'Automated Abstraction of Numerical Simulation Models - Theory and Practical Experience', *proceedings of Model Based Systems and Qualitative Reasoning for Intelligent Tutoring Systems*, 161-168, (2002).

[2] M.-O. Cordier and C. Largouët. 'Using model-checking techniques for diagnosing discrete-event systems'. *proceedings of DX'01, International Workshop on Principles of Diagnosis*, (2001).

[3] P. Supavatanakul, C. Falkenberg and J. Lunze. 'Identification of timed discrete event models for diagnosis', *proceedings of DX'01, International Workshop on Principles of Diagnosis*, (2003).

[4] S. Kowalewski, S. Engell, J. Prußig and O. Stursberg. 'Verification of logic controller for continuous plants using timed condition/event-system models', *Automatica*, **35**, 505-518, (1999).

[5] T. A. Henzinger, P.-H. Ho and H. Wong-Toi. 'Algorithmic Analysis of Nonlinear Hybrid Systems'. *IEEE Transactions on Automatic Control*, **43**, 540-554, (1998).

[6] R. Alur and D. Dill. 'The theory of timed automata', *Theoretical Computer Science*, **126**, 183-235 (1994).

[7] T. A. Henzinger, X. Nicollin, J. Sifakis and S. Yovine. 'Symbolic Model Checking for Real-Time Systems', *Information and Computation*, **111**, 193-244, (1994).

[8] S. Yovine. 'Kronos: A verification tool for Real-Time Systems', *Journal of Software Tools for Technology Transfer*, **1**, 123-133, (1997).

[9] P. Pettersson and K. G. Larsen. 'Uppaal2k', *Bulletin of the European Association for Theoretical Computer Science* **70**, 40-44, (2000).

[10] K. G. Larsen, P. Pettersson and W. Y. I. Springer. 'UPPAAL in a Nutshell', *International Journal of Software Tools for Technology Transfer*, **1**, 134-152, (1997).

[11] H. L. Smith. 'Monotone dynamical systems: An introduction to the theory of competitive and cooperative systems', *AMS Mathematical Surveys and Monographs*, **41**, 31-53, (1995).

[12] J. L. Gouzé, A. Rapaport and M. Z. Hadj-Sadok. 'Interval observers for uncertain biological systems', *Ecological Modelling*, **133**, 45-56 (2000).

[13] A. Hélias. 'Agrégation/abstraction de modèles pour l'analyse et l'organisation de réseaux de flux : application à la gestion des effluents d'élevage à la Réunion', PhD thesis, ENSAM, Montpellier (France), (2003).

[14] J.-P. Steyer, J.-C. Bouvier, T. Conte, P. Gras and P. Sousbie. 'Evaluation of a four year experience with a fully instrumented anaerobic digestion process', *Water Science and Technology*, **45**, 495-502, (2002).

[15] O. Bernard, Z. Hadj-Sadok, D. Dochain, A. Genovesi and J.-P. Steyer. 'Dynamical model development and parameter identification for anaerobic wastewater treatment process', *Biotechnology and Bioengineering*, **75**, 424-438 (2001).