

DIAGNOSIS OF DISCRETE EVENT SYSTEMS USING TIMED AUTOMATA

Zineb Simeu-Abazi*, Maria Di Mascolo*, Michal Knotek**

**Laboratoire G-SCOP INPG-UJF*

***Honeywell HTSL – Brno Aerospace*

E-mail: Zineb.simeu-Abazi@g-scop.inpg.fr

Abstract: This paper proposes an effective way for diagnosis of discrete–event systems by using a timed automaton. It is based on the model-check technique, thanks to the time analysis of the timed model. This paper gives a method to construct all timed models and details the different steps used to obtain the diagnosis path in the timed automaton. The proposed approach could be applied to dynamical systems, where input and output measurement can be not available. The modeling formalism permits us a formal verification of the model. The model-checking became favorite method for the checking of timed automata, our extension is an adaptation of this method for diagnoser verification. A simple real-world batch process is used to demonstrate the modeling and verification task. Finally, a backward time analysis is performed to reach diagnostic results. *Copyright © 2002 IFAC*

Keywords: Diagnosis, Discrete–Event Systems (DES), Fault Location, timed automata, model-checking.

1. INTRODUCTION

Modern industry deals with efficient diagnosis to improve reliability relevant functions and reduce high maintenance cost. Our diagnosis objective is to detect and identify the possible faults occurring in the dynamical system. That leads to determine the way of locating a fault, and its time occurrence (Blanke, 2003). In continuous systems, *residuals* are used for diagnosis (Schullerus, 2002). Residuals describe inconsistencies responses between the model and the current system behaviour. The residual value is used to detect any fault.

In the Discrete-Event Systems (or DES) area, the most common diagnosis approach is the so-called *model-based diagnosis*, which uses the inputs and outputs of the system under supervision to **detect** the fault and **isolate** (locate, distinguish) the source of failure (Supavatanakul, 2003; Zad, 1999). The measured signals are processed by means of a dynamical model in order to decide whether the system behaves normally, and which faults have occurred (Lunze, 2000). Model-based diagnosis algorithms use an explicit model of the dynamical system under investigation. This model incorporates the knowledge about the faultless and the faulty system behaviour, in systematic way, in order to analyse the fault symptoms.

Recently, process diagnosis methods have been developed for DES by using timed Petri nets, stochastic automata (Lunze, 2002), timed automata (Hristov, 2004), template languages (Pandalai, 2000) or Semi-Markov processes. The main idea of most of these diagnosis methods is to simulate nominal or faulty system behaviours with the discrete model.

Another approach is based on the control of tasks duration, and was introduced by (Simeu, 2003; Simeu, 2004; Rayhane, 2003). This diagnosis provides temporal distances between the events, corresponding to the beginning or the end of the tasks execution. Authors define three functioning modes. The system is in the *normal mode* for faultless functioning, the *degraded mode* for functioning when the temporal distances are in the acceptable margins, and the *failure mode* when the defined tolerance margin is exceeded.

In this paper, we focus on diagnosis of DES, where inputs and outputs are discrete values. A fault refers to a non permitted deviation in the behaviour of the system. We consider the “drastic” type of failure, such as valve stuck-close. Other types of partial failures (such as drift in sensor or small changes in the dynamics of actuators) are not considered.

In our approach a dynamic model with temporal transitions is proposed in order to model the system under study. By “dynamical model”, we mean an extension of timed automata for which the faulty states are identified. Then, the global model contains the faultless functioning states and all the faulty states. Our method is based on the backward exploitation of the dynamic model, where all possible reverse paths are searched. The reverse path is the connection of the faulty state to the initial state. The

diagnosis method is based on the coherence between the occurrence time of the fault and the reverse path length. It consists in three main steps:

1. Modeling : build the model by using **timed automata**
2. Analysis : define the **diagnosis algorithm** based on **time analysis**
3. Implementation: develop a **diagnoser**

In this paper we focus on the first two sub-problems. In section 2, we present the diagnosis principle. In Section 3 we give some basic notions on the timed automaton which is used in the modeling step presented in section 4. An academical example is then used (in section 5) to illustrate our approach. Finally, a conclusion is presented with some perspectives.

2 –DIAGNOSIS PRINCIPLE

In the existing research in the diagnosis area, the common approach uses the inputs and outputs of the system. These measured signals are processed in order to decide whether the system behaves normally and which faults have occurred (Lunze, 2001). The common diagnosis approach is shown on Figure 1a. In the approach we propose here, the diagnosis system is based on checking the consistency between the time of failure occurrences and the inputs sequences (see figure 1b). It is thus not necessary to know the I/O sequence, but only the time trajectories.

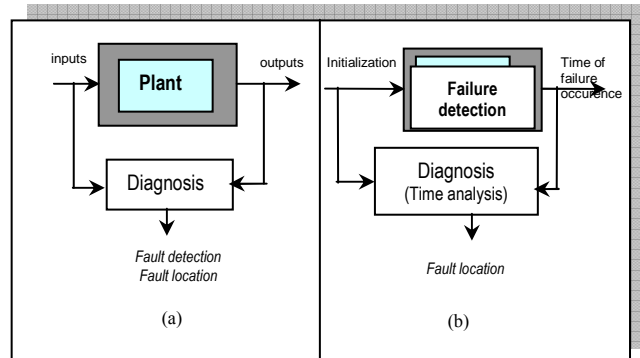


Figure 1- Diagnosis approach: (a) Approach based on input/output; (b) the proposed approach

The diagnosis approach requires the four following steps:

1. **State space model** – As the studied systems correspond to DES, the adapted modelling tool is the timed automata. It is a right tool for description of dynamical system by temporal transitions.
2. **Fault model** – The fault tree analysis is used to identify a set of faults. The considered faults are implemented into the timed-automaton.
3. **Dynamic model** – The temporal parameters are identified for faultless and faulty modes.
4. **Backward time analysis** – The algorithm for the backward time analysis algorithm is detailed in section 4.

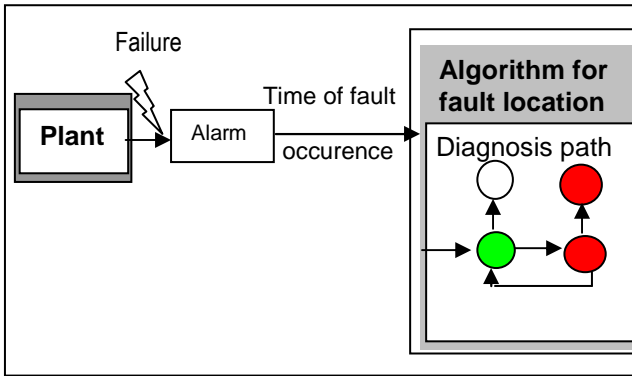


Figure 2- Principle of the fault location

The objective is to design a diagnosis tool for a given plant. We consider a plant equipped with an alarm and with a global clock for synchronization. Alarm produces an error signal when a fault is detected. Our diagnosis task is to locate and identify all faults which can occur (Figure 2).

3- MODELING TOOL: TIMED AUTOMATON

The fault diagnosis problem in DES is generally solved by using the model-based approach. In other words, an algorithm for fault detection and isolation will analyse and compare the model with the observed behaviour. The timed automaton is a well suited tool for modelling all the evolutions of a DES.

3.1 Finite automata

The formal language theory offers the possibility to model system behaviour. A formalism of finite-state machine (FSM, Finite State Machine) has been studied for many years and provides a qualitative way to model a system. Engineers have found that the use of finite-state automata and corresponding state transition diagrams makes design and diagnosis of complex systems easier.

An automaton is a mathematical model for a finite state machine (FSM) [TRI-02]. A FSM is a graph in which, given an input, we can jump through a series of states according to a transition function (which can be expressed as a table). In the common “Mealy” variety of FSMs, this transition function tells the automaton which is the next state to go to, given a current state and a current symbol. The input is read symbol by symbol, until it is consumed completely. Once the input is depleted, the automaton is said to have stopped. Depending on the state in which the automaton stops, it is said that the automaton either accepts or rejects the input. If it stops in an “admitted” state, then the automaton accepts the word. If, on the other hand, it lands on a “non-admitted” state, the word is rejected. The set of all the words accepted by an automaton is called the language accepted by the automaton. The following terms are very often used in the automata theory:

Symbol: Analogy with a single letter, although it needs not to be a letter. It may be any symbol, as long as it is a single token (no words here yet), and can be distinguished from other symbols.

Word: A finite string formed by the concatenation of a number of symbols.

Alphabet: A finite set of symbols.

Language: A set of words, formed by symbols in a given alphabet. May be infinite or not.

3.2 Timed automata

Timed automata are a tool for the modelling and verification of real time systems (Alur, 1994; Bengtsson, 2004). A timed automaton is essentially a finite automaton (FSM), extended with real-valued variables. Such an automaton may be considered as an abstract model of timed systems. This expressive modelling tool offers the possibilities of model analysis like verification, controller synthesis and also faults detection and isolation. In the original theory of timed automata, a timed automaton is a finite state extended with a set of real-valued variables modelling clocks. Constraints on the clock variables are used to restrict the behaviour of timed automaton, and conditions are used to enforce progress properties. A simplified version, namely Timed Safety Automata, is introduced in (Henziger, 1994) to specify progress properties using local invariant conditions. Due to its simplicity, a Timed Safety Automaton has been adopted in several verification tools for timed automata. Using timed automata, the system is described in a qualitative and quantitative way. This is illustrated by the example on the figure 3, where the qualitative parameters represent the sequence of events while quantitative ones relate to temporal parameters. The problem of automata analysis is considerably more difficult in the timed case than in the discrete case: in the discrete case, one deals with classical regular languages which have robust closure properties.

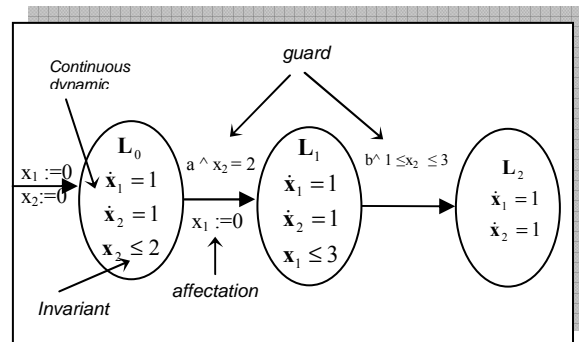


Figure 3: Example of the timed automata

The timed automata tool (Sava 2001; Simeu, 2003) is defined as a finite state machine with a set of continuous variables that are named clock. These variables evolve continuously in each location of the automaton, according to an associated evolution function. As long as the system is in one state L_i , the clock x_i is continuously incremented. Its evolution is described by $\dot{x} = 1$. The clocks are synchronized and change with the same step. An invariant is associated to each state. It corresponds to the conditions needed to remain in the state. The number of clocks depends on the parallelism in the system. The automaton can stay in one state as long as the invariant condition is checked. Each transition of an automaton is conditioned by an event or temporisation called “guard” and its execution determines the discrete evolution of the variables according to its associated assignment.

Let us consider the timed automaton given in figure 3. This automaton has two clocks x_1 and x_2 . The continuous evolution of time in this model is represented by $\dot{x}_1 = 1$ and the labelled arcs in the graph represent the model of discrete evolution. The guard in each arc is a transition labelling function that assigns firing conditions with the transitions of the automaton. The affectation is a function that associates with each transition of the automaton one relation that allows actualizing the value of continuous state space variables after the firing of a transition. The invariant in the state L_0 and L_1 are respectively $x_2 \leq 2$ and $x_2 \leq 3$. The initial state of this system is represented by an input arc in the origin state (L_0). In the dynamic model, active clocks are found in each state. A graphical interpretation of the timed automaton is the automaton graph (figure 3).

4 EXPLOITATION OF THE TIMED AUTOMATA FOR THE MODELLING

For a DES diagnosis, two different modes are defined. The dynamic model can be represented by a set of states. One of them corresponds to the faultless functioning mode, and the other to faulty functioning modes.

4.1 Modelling principles

The difference with the approach developed by Lunze (Lunze, 2002) relies in the implementation of the fault. We propose the implementation in the sense of extension of the states. Lunze defines the faults as a set of the fault and this set of fault implemented into the behavioural relation. Finally, Lunze imprints dynamic for each considered fault to its own automata, then the coherent automata sequence of inputs and outputs is searched. And as said before, our approach is not based on the I/O sequence but only on the time analysis. Then, the dynamic model is obtained from the state space model with considered faulty states by identification. The identification includes two tasks:

- Extraction of the **trajectory**
- Measure of the **temporal transitions** of the trajectory.

The trajectory refers to state evolution in the dynamical model (in the state space). The results of the identification are a set of arcs from one state to another with time affectation (temporal transition). The identification is needed for every considered fault (evolution) of the system and for the nominal faultless case as well. This identification can be done using simulation or analytical techniques. The reachability of the faulty states allows to obtain the set of trajectories for each fault model. We must associate the faulty state with the fault origin; that is the state in which the fault can occurs.

4.2 Modelling example

Modelling with Timed automata will be illustrated on a trivial example also known as batch process.

Instrumentation: Neutralisation batch process consists in one tank that is equipped by two level sensors and three valves. Valve V1, V2 as input valves, output valve V3. Figure 4-a, shows the placement of two levels sensors.

Control: The production sequence takes is the following: First phase is preparation of the chemical product. Firstly, the valve V1 is open; an ingredient 1 flows into tank 1. When the tank level L_1 is reached, the valve V1 is closed and V2 is opened. Then, the tank level L_2 is waited for. After the positive edge of sensor L2, the valve V2 is closed..

Control sequence:

- (1) S0: When the process is initialized, tank should be empty.
- (2) S1: First, valve V1 is open, an ingredient 1 flows into tank 1.
- (3) S2: If level L1 is reached then valve V1 is closed and V2 is opened.
- (4) S3: If Level L2 is reached then V2 is closed.

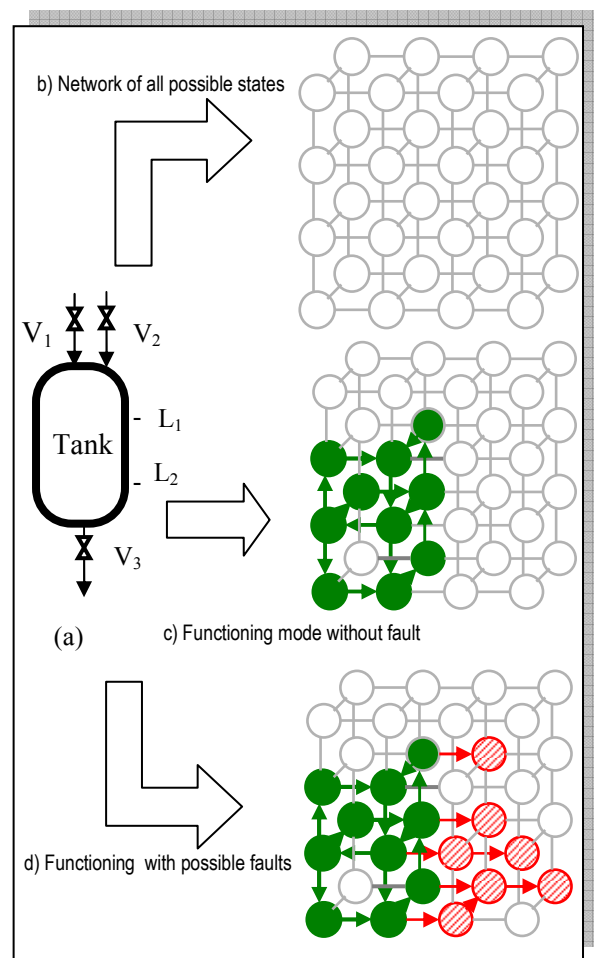


Figure 4- The “Tank-example” and the timed models

- a- Schema of tank system
- b- State space of physical system
- c- Global faultless model - faultless controller projection into state space
- d- Dynamic global model - controller projection with fault consideration

4.3 Dynamic Global model

The model with all considered faulty and faultless evolution is called Global model (G). The global model incorporates information from plant, sensors, actuator and controller. The global model for a complex system can be represented in a model checker (Knotek, 2006).

The description of different states of this process can be represented by the network (figure 4-b) where all possible states are reachable. The control part must bound the evolution of the process to only faultless functioning according to the control sequence (figure 4-c). For diagnosis purposes of timed diagnosis, the last step is the most important. The obtained global model must be extended by temporal parameters described the dynamic of the real system e.g. by opening the valve, L_1 is reached normally at $10tu$, etc. This extended model by time is called “dynamic global model” and this model denotes base for time diagnosis. Hence, time is taken into account to help us distinguish which state could be reached. This fact is important to control the diagnosability of the system. See time is associated with transitions in figure 4.-d.

4.4 Evolution of dynamic Global model

In the DES framework, we consider discrete variables. For example, when valve V1 is closed, the variable V_1 takes the value 0. It takes the value 1 when the valve is open. To illustrate fault modelling with Timed Automata, we consider in this example the two following faults:

- V1 can be stuck close ($V_1=2$)
- V1 can be stuck open ($V_1=3$)

Where, the notation used is the following:

- The state of a valve Vi is defined by the value of a variable V_i .
- The state of the tank is defined by the vector (V_1, L_1, L_2) , where V_1 is the state of valve V1 and L_1, L_2 are level variables giving the state of sensors L1 and L2.

It means that valve V1 can be in one of four states: 0,1,2,3. The both variables L_1 and L_2 can be in one of following states: empty tank (00) (i.e.: sensors L_1 and L_2 indicate 0), intermediate (01) and full (11).

The combination of all possible states together creates the state space composed by the state of V1 (V_1), the state of sensors L1 and L2. State space describes all the possible evolutions (including faulty evolutions).

See states 000, 001, . . . , 311 at figure 5-a. This composition is also called network (as illustrated in figure 4-b).. Applying control sequence on the state space (projection) gives the path which presents a faultless behaviour. In Figure 5-b, see that system starts from the state 000 (V1 close, L_1, L_2 is 0). By opening the V1 the system state changed to 100. When sensor L1 is reached, the system state is changed to 101. Thus, the state 111 corresponds to the state where V1 is open and L_2 is also reached.

Our controller closes V1, therefore system ends in the state 011.

The fault behaviour stuck close of V1 is modelled by an unobservable event called $Fault_1$, which changed state 000 to 200 instead of 100. Another fault stuck open of V1 is modelled by unobservable event $Fault_2$: from state 111 to 311, instead of 011.

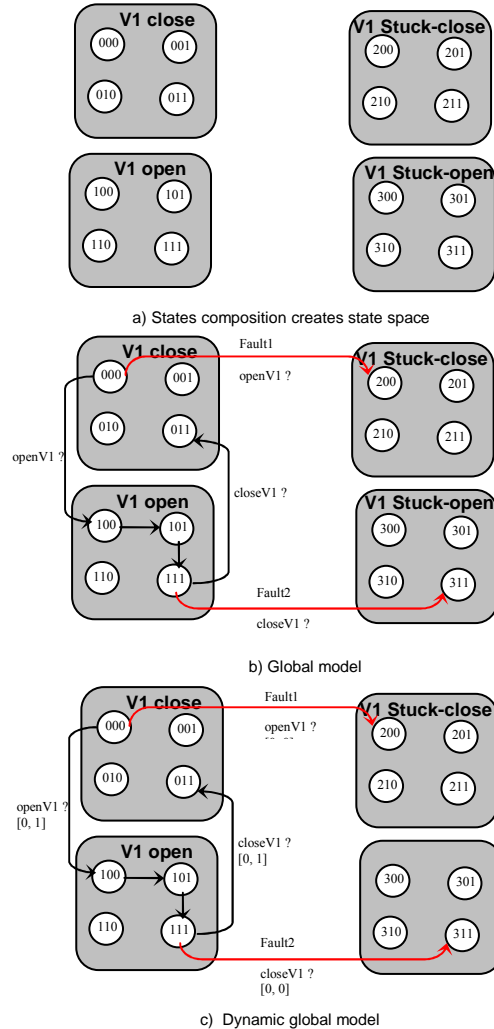


Figure 5. Global model for the tank example

5. BACKWARD TIME ANALYSIS

The aim of backward time analysis of timed automata is to locate (isolate) a fault. In our case exploitation means searching accessible trace according the time from a final faulty state to the initial state of automaton denoted by **reverse path**. Therefore the initial state must be known. Our task can be seen as retrace the automaton graph from the faulty states to the known origin state. The aim is to find from the set of reverse path the coherent ones.

5.1 – Principle of the analysis

See automaton graph with fault model in figure 6. From fault model one can see that fault F1 can occurs from states 1 or 3, and the fault F2 from the state 2. The diagnostic model must be find and locate which fault occurs in the system, according the time occurrence. If the fault occurs in the time 3 or 7 time unit, it's fault located

as F1. In another case, the fault occurs in the time 5, the fault F2 is located.

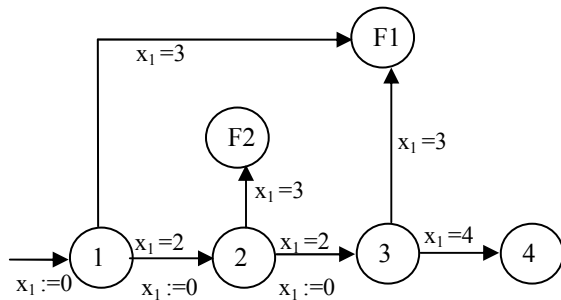


Figure 6: Time analysis of the timed automata

5.2 Application to the "Tank-example"

The fault isolation algorithm will be shown for the following faults:

- f1: Fault valve V1 being stuck close. Practically it means, that tank stays in the initialized state. Controller waits to event L1 which can not occur because of the stuck valve.
- f2: Fault valve V1 being stuck open. This fault can physically cause an overflow.
- f3: The third considered fault is sensor L1 which stays in close position. It means when the level L1 is reached, this sensor does not indicate it.

Take the case where the state waitingV1 is reached and we consider faulty state detected by invariant violation in time $t = 20tu$. Now, let us explore the phase of fault isolation. The detection of the fault is represented by faulty flash $!L1 \text{ AND } x=20$. The state N_LongFilling is reached, timer is set to the value $T.SetV \text{ alue} = 20tu$. If the event L2 appears in the time $t = [38, 40]tu$ the fault f3 is indicated (Faulty sensor L1). If the event L2 does not appear, the considered fault is f2 (V1StuckClose).

6. CONCLUSION

The algorithm of diagnosis based on backward time analysis was presented. This algorithm makes analysis of the model of dynamical system (timed automata), where considered faults are implemented as extension of states. The trajectory and temporal transition of the model must be identified for all considered modes (faultless and faulty modes). The time of occurrence of fault is considered. The backward time analysis searches the possible reverse path to localise the fault according the time of fault occurrence. Our approach deals only with one extended timed automata, comparing to mentioned approaches, where for every fault is built one automaton model. The next step in our work is the model checker. It means to verify if all faulty states in the dynamic model are reachable or it is necessary to add some other sensors to isolate the faults.

REFERENCES

alur R, Dill D., "A theory of timed automata," Theoretical Computer Science (TCS), 126(2), 1994, pp.183–235.

- Bengtsson J. and Wang Yi, "Timed automata: Semantics, algorithms and tools," In Lecture Notes on Concurrency and Petri Nets. W. Reisig and G. Rozenberg (eds.), LNCS 3098, Springer-Verlag, 2004.
- Blanke M., Kinnaert, Lunze J., and Staroswiecki M.. *Diagnosis and Fault-tolerant Control*. Springer Verlag, 2003.
- Cordier M.O., Grastien A., Largouët Ch., and Pencolé Y., "Efficient trajectories computing exploiting inversibility properties," Proceedings of the Fourteenth International Workshop on Principles of Diagnosis, pp. 93–98, 2003.
- Knotek, M. "Fault diagnosis based on temporal analysis," PhD thesis, UJF Grenoble France and FECC BUT Czech Republic, 2006.
- Hristov I, J.Lunze, and P.Supavatanakul. *A time discrete-event approach to diagnosis of the damadics actuator benchmark*. In 5th DAMADICS Workshop, 2004.
- Lunze J. and Schröder J.. *Sensor and actuator fault diagnosis of systems with discrete inputs and outputs*. Automatica, 2000.
- Lunze J., Schröder, J. and Supavatanakul. P. *Diagnosis of discrete event systems: the method and an example*. In Proceedings of the Workshop on Principles of Diagnosis, DX'01, pages 111–118, Via Lattea, Italy, 2001.
- Lunze. J. *Qualitative methods for fault diagnosis*. Research report Nr. 2002.31, December 2002. [PAN-00] D.N. Pandalai and L.E. Holloway. *Discrete-event models of quantised systems for diagnosis*. IEEE Transactions on Automatic Control, 45(5):868–882, May 2000.
- Rayhane H. and Simeu-Abazi Z.and Bennani. T. *Surveillance des systèmes de production par automates temporisés*. Conférence Internationale en Productique, CIP'03, 14-16 Octobre 2003.
- Sava. A.T. *Sur la synthèse de la commande des systèmes à événements discrets temporisés*. PhD thesis, Laboratoire d'Automatique de Grenoble, november 2001.
- Simeu-Abazi Z., Rayhane H., Bennani T., Bouredji. Z. *Optimisation des temps de détection dans la surveillance des systèmes*, 5ème Congrès International de Génie Industriel, 25-29 Octobre 2003.
- Simeu-Abazi Z. and Zikmund J. and Bouredji Z.. *Monitoring and predictive maintenance: optimisation of fault latency*. IMS, Arles, July 2004.
- Schullerus G. and Supavatanakul P and Krebs V. and J.Lunze. *Efficient hierarchial diagnosis for timed discrete-event systems*. Research report Nr. 2002.4, Ruhr-Universität Bochum, 2002.
- Supavatanakul P., Falkenberg C., and Lunze J.. *Identification of timed discrete-event models for diagnosis*. In International Workshop on Principles of Diagnosis (DX'03), 2003.
- Tripakis S., "Fault diagnosis for timed automata," In Springer, editor, In Proc. 7th Int. Symp. Formal Techniques in Real-Time and Fault Tolerant Systems (FTRTFT 02), volume 2469 of LNCS, pages 205–224, 2002.
- Zad S.H.. *Fault diagnosis on discrete-event and hybrid systems*. PhD thesis, University Of Toronto, 1999.