Health Monitoring of Hybrid Systems Using Hybrid Particle Petri Nets

Quentin Gaudel[1], Elodie Chanthery[2], and Pauline Ribot[3]

[1,2,3] *CNRS, LAAS, 7 avenue du colonel Roche, F-31400 Toulouse, France*
quentin.gaudel@laas.fr, elodie.chanthery@laas.fr, pauline.ribot@laas.fr

[1,2] *Univ de Toulouse, INSA, LAAS, F-31400 Toulouse, France*

[3] *Univ de Toulouse, UPS, LAAS, F-31400 Toulouse, France*

ABSTRACT

This paper presents an approach of model-based diagnosis for the health monitoring of hybrid systems. These systems have both continuous and discrete dynamics. Modified Particle Petri Nets, initially defined in the context of hybrid systems mission monitoring, are extended to estimate the health state of hybrid systems. This formalism takes into account both uncertainties about the system knowledge and about diagnosis results. The generation of a diagnoser is proposed to track online the system health state under uncertainties by using particle filter. To include more complex characteristics of the system, as its degradations for prognosis purpose, an enriched formalism called Hybrid Particle Petri Nets is defined.

1. INTRODUCTION

Systems have become so complex that it is often impossible for humans to capture and explain their behaviors as a whole, especially when they are exposed to failures. It is therefore necessary to develop tools that can support operator tasks but that also reduce the global costs due to unavailability and repair actions. An efficient diagnosis technique has to be adopted to detect and isolate faults leading to failures. Diagnosis uses a behavioral model of the system and online observations to determine the behavioral state of the system. Uncertainties in diagnosis can be taken into account by giving as much information as possible about the ambiguous state likelihood. On the other side, systems are continuously degrading depending on operational conditions. Knowing available information on the system, it is possible to establish physical degradation laws or time-dependent fault probability distributions based on the feedback. It is then interesting to take into account this temporal and/or stochastic information about the system degradation. Health monitoring consists in evaluating the current health state of the system through a diagnosis and a degradation law value. The health state is represented by a degradation measure for the system in a specific behavioral state (Vinson, Ribot, Prado, & Combacau, 2013). Its estimation is the first step to perform later prognosis and to compute the remaining useful life (RUL) of the system. A formal generic modeling framework for health monitoring of complex heterogeneous systems has been presented in (Ribot, Pencolé, & Combacau, 2013) and encapsulates knowledge about the system behavior and degradation used by diagnosis and prognosis. Uncertainties in the model and diagnosis results are taken into account by estimating interval ranks for parameters. Another common framework for diagnosis and prognosis has been proposed in (Roychoudhury & Daigle, 2011). This article presents a state model that specifies the nominal behavior of the system and fault progression over time. However, it only represents systems with a continuous dynamics without discrete or hybrid aspect.

Recent industrial systems exhibit an increasing complexity of dynamics that are both continuous and discrete. It has become difficult to ignore the fact that most systems are hybrid (Henzinger, 1996). In previous works (Chanthery & Ribot, 2013), we extended the diagnosis approach proposed in (Bayoudh, Travé-Massuyes, & Olive, 2008) in order to integrate diagnosis and prognosis for hybrid systems. The approach uses hybrid automata and stochastic models for the system degradation. The main drawback of this approach is that the discrete event system oriented diagnosis framework explodes in number of states and it does not seem the best suited for the incorporation of the highly probabilistic prognosis task. To have a more compact representation and to capture all uncertainties related to the system, to the observations and to the diagnosis results, we propose to consider the formalism of Modified Particle Petri Nets (MPPN) defined in (Zouaghi, Alexopoulos, Wagner, & Badreddin, 2011a). Moreover this representation is intuitive and facilitates the modeling task. MPPN are an extension of Particle Petri nets (Lesire & Tessier, 2005) that combines a discrete event model (Petri net) and a continuous model (differential equations). The main advantage of MPPN is that uncertainties and hybrid dynamics are taken into account. Particle filter is used to integrate probabilities in the continuous state estimation process. MPPN have been used for supervision and planning, but never for health monitoring, diagnosis and/or prognosis.

MPPN representation is useful in capturing all uncertainties about the state knowledge and about the observations. As wide as can be the range of feature representations provided by MPPN, we did not succeed in modelling a characteristic that depends on a discrete state and a continuous state of the system. That is why we propose to define what we call Hybrid Particle Petri Nets (HPPN) in order to model both behavior and degradation of hybrid systems in the context of health monitoring. The HPPN formalism enriches MPPN to model available knowledge about hybrid characteristics of the system. The paper is organized as follows. Section 2 recalls the MPPN framework and presents how it can be used for behavioral health monitoring. In Section 3 a hybrid diagnosis technique is proposed based on the generation of a behavioral diagnoser using the MPPN formalism. The MPPN enrichment is defined in Section 4 as Hybrid Particle Petri Nets to take the system degradation into account by interacting with the hybrid behavioral model. Some conclusions and future work are discussed in the final section.

2. MODIFIED PARTICLE PETRI NETS FOR MONITORING

In this section, the Modified Particle Petri Nets (MPPN) formalism is described according to the work of (Zouaghi et al., 2011a). First the model structure and its online process are detailed and then a way to use it to represent system health model is presented.

2.1. Definition

Modified Particle Petri Nets are defined as a tuple $< P, T, Pre, Post, X, C, \gamma, \Omega, M_0 >$ where:

- P is the set of places, partitioned into numerical places P^N and symbolic places P^S.
- T is the set of transitions (numerical T^N, symbolic T^S and mixed T^M).
- Pre and $Post$ are the incidence matrices of the net, of dimension $|P| \times |T|$.
- $X \subset \Re^n$ is the state space of the numerical state vector.
- C is the set of dynamics equations of the system associated with numerical places, representing continuous state evolution.
- $\gamma(p^S)$ is the application that associates tokens with each symbolic places $p^S \in P^S$.
- Ω is the set of conditions associated with the transitions (numerical Ω^N and symbolic Ω^S).
- M_0 is the initial marking of the net.

MPPN can model system behaviors. A basic example of a system behavior modeled with MPPN is illustrated in Figure 1.

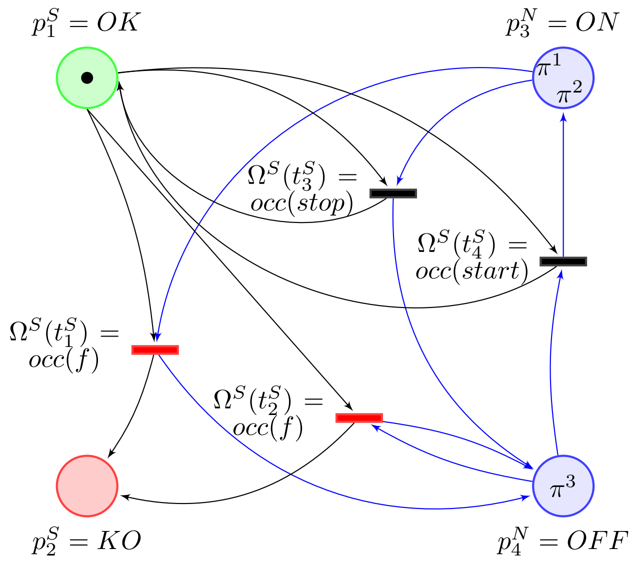
$p_1^S = OK$
$p_3^N = ON$
π^1
π^2
$\Omega^S(t_3^S) = occ(stop)$
$\Omega^S(t_4^S) = occ(start)$
$\Omega^S(t_1^S) = occ(f)$
$\Omega^S(t_2^S) = occ(f)$
π^3
$p_2^S = KO$
$p_4^N = OFF$

Figure 1. Example of MPPN.

There are four places in this MPPN: $P = \{p_1^S, p_2^S, p_3^N, p_4^N\}$. Two symbolic places p_1^S and p_2^S represent the two discrete modes of the system, respectively when the system is working well (OK) and when the system has failed (KO). Two numerical places p_3^N and p_4^N represent the two continuous behaviors of the system, respectively when it is turned on (ON) and when it is turned off (OFF). There are four symbolic transitions is this MPPN: $T = \{t_1^S, t_2^S, t_3^S, t_4^S\}$. They represent occurences of discrete events. t_1^S and t_2^S represent the occurence of a fault event f respectively when the system is turned on and turned off and let the system go from the OK mode to the KO mode. t_3^S represents the occurence of a mission event $stop$ that turns off the system when it is turned on and is in OK mode. Finally, t_4^S represents the occurence of a mission event $start$ that turns on the system when it is turned off and is in OK mode.

A numerical place $p^N \in P^N$ is associated with a set of dynamics equations representing the continuous behavior of the system. Numerical places thus model continuous dynamics of the system. Numerical places are marked by a set of particles $\pi_k^i = [x_k^i, \mathrm{w}_k^i]$ with $i \in \{1, ..., |M_k^N|\}$ where M_k^N is the set of all the particles in the net at time k. Particles are defined by their corresponding numerical state vector $x_k^i \in X$

and their weight $w_k^i \in [0,1]$ at time k. The set of particles represents an uncertain distribution over the value of the numerical state vector.

Symbolic places model the behavioral modes of the system. A symbolic place $p^S \in P^S$ is marked by configurations δ_k^j with $j \in \{1, ..., |M_k^S|\}$ where M_k^S is the set of configurations in the net at time k. The set of configurations represents all the possible current modes of the system.

The marking of the net is composed of tokens, that can be numerical tokens (particles) or symbolic tokens (configurations). The marking M_k of the MPPN at time k consists of both kinds of tokens:

$$M_k = \{M_k^S, M_k^N\} \quad (1)$$

For example, in Figure 1, the numerical place p_3^N is marked by the set of particles $\{\pi^1, \pi^2\}$ and the symbolic place p_1^S is marked by a configuration. The marking of the illustrated MPPN is then $M_k = \{[0,1]', [\{\pi^1, \pi^2\}, \{\pi^3\}]'\}$.

A transition models a change in the continuous dynamic and/or a change of the system mode. A symbolic transition is conditioned by an observable discrete event. A numerical transition is conditioned by a set of constraints on continuous observable variables. Finally, a mixed transition is conditioned by an observable discrete event and a set of constraints on continuous observable variables.

2.2. Firing Rules

This section recalls the basic ideas of MPPN firing rules. More formal details about the firing rules of the different transitions can be found in (Gaudel, Chanthery, Ribot, & Le Corronc, 2014). A numerical transition $t_j^N \in T^N$ is associated with conditions $\Omega^N(t_j^N)$, where $\Omega^N(t_j^N)(\pi) = 1$ if the particle satisfies the conditions. For example, if $\pi = [x, \text{w}]$ follows the constraint equation c and b is a trigger value, a numerical condition can be defined as $\Omega^N(t_j^N)(\pi) = (c(x) > b)$. $\Omega^S(t_j^S) = occ(e)$ represents the conditions assigned to a symbolic transition $t_j^S \in T^S$. $occ(e)$ is a boolean indicator of the occurrence of the discrete event e : $occ(e) = 1$ if e has occurred. Then, a configuration δ satisfies the condition $\Omega^S(t_j^S)$ when $\Omega^S(t_j^S)(\delta) = 1$, ie. when the event e has occurred.

The numerical firing uses the concept of classical firing with the particles satisfying the numerical condition and the concept of pseudo-firing (ie. duplication) for the configurations. The duplication of configurations represents uncertainty about the occurrence of an unobservable discrete event. An example of a numerical firing from marking at time k to marking at time $k+1$ is illustrated in Figure 2(a). In this example, t_1^N only has a numerical condition because it is a numerical transition. Particle π^3 satisfies the numerical condition $\Omega^N(t_1^N)$ and thus is moved through the transition t_1^N to p_4^N. The configuration in place p_1^S is duplicated in p_2^S.

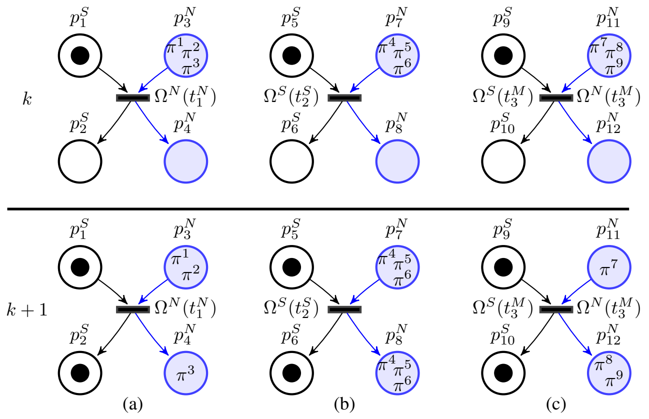

Figure 2. Illustration of firing rules of numerical (a), symbolic (b) and hybrid (c) transitions.

The symbolic firing uses the concept of pseudo-firing for particles and configurations. The pseudo-firing of all the tokens models uncertainty about the occurence and the non occurrence of an observable discrete event. Figure 2(b) illustrates an example of a symbolic firing. The symbolic transition t_2^S only has a symbolic condition. No token satisfies the condition $\Omega^S(t_2^S)$, however all tokens are duplicated.

Mixed transitions are introduced in (Zouaghi et al., 2011a) to model the interaction between discrete events and system continuous dynamics. In the referred article, they were called "hybrid transitions". A mixed transition merges a symbolic transition with a numerical transition to correlate discrete observations with continuous observations. The firing of the symbolic transition only depends on a discrete event, but the simultaneous firing of the numerical transition models the dependency of the mixed transition on the symbolic part because discrete events are part of the process behavior. A mixed transition $t_j^M \in T^M$ is then associated with both numerical conditions $\Omega^N(t_j^M)$ and symbolic conditions $\Omega^S(t_j^M)$.

The mixed firing uses the concept of classical firing with the particles satisfying the numerical condition and the concept of pseudo-firing with the configurations satisfying the symbolic condition. The pseudo-firing of configurations models uncertainty about the occurrence of an observable discrete event which is supported by a change of continuous dynamics. An example of a mixed firing is illustrated in Figure 2(c). t_3^M is a mixed transition therefore it has a symbolic condition and a numerical condition. The configuration in place p_9^S is duplicated because it satisfies the symbolic condition $\Omega^S(t_3^M)$. Regarding the numerical part, particles π^8 and π^9 satisfy $\Omega^N(t_3^M)$ and so they are moved through t_3^M. Furthermore, π^7 stays in place p_{11}^N because it does not satisfy $\Omega^N(t_3^M)$.

Heterogeneous systems are defined as systems that have a discrete, continuous or both discrete and continuous dynamics.

MPPN can easily model heterogeneous systems by using only the symbolic or numerical subpart of the model or both in the case of hybrid systems.

2.3. State Estimation

The problem of hybrid state estimation in MPPN has been introduced in (Zouaghi et al., 2011a) and consists of a prediction step and a correction step, illustrated in Figure 3.

For the sake of clarity in this paper we assume that a hybrid state is represented by a couple (p_i^S, p_j^N) of a symbolic place and a numerical place. The initial marking of the MPPN is $M_0 = \{M_0^S, M_0^N\}$ and the estimated marking at time k is $\hat{M}_k = \{\hat{M}_k^S, \hat{M}_k^N\}$ where $\hat{M}_k = \hat{M}_{k|k}$. The observations start at time $k = 1$, $O_1 = (O_1^S, O_1^N)$ where O^S and O^N respectively represent the observations corresponding to the symbolic part and the numerical part.

(1) The prediction is based on the evolution of the MPPN marking and on the estimation of the particle values. It aims at determining all possible next states of the system $\hat{M}_{k+1|k} = \{\hat{M}_{k+1|k}^S, \hat{M}_{k+1|k}^N\}$. A noise is added during the particle values update to take into account uncertainty about the dynamics equations and thus about the continuous system model.

(2) The correction is based on the update of the prediction according to new observations on the system.

(a) A numerical correction, based on particle filter algorithms, produces a probability distribution Pr_{DN} of the particles $\hat{M}_{k+1|k+1}^N$ over the value of the numerical state vector. At this step, particle weights are updated using a probability distribution function depending on a random noise that models uncertainty about continuous observations O_{k+1}^N.

(b) A symbolic correction then computes a probability distribution Pr_{DS} over the symbolic states of the system, depending on discrete observations O_{k+1}^S and on Pr_{DN} making the process hybrid.

Finally, in order to update the complete predicted marking $\hat{M}_{k+1|k}$, a decision making method is required. The result of the whole state estimation process is the estimated marking at time $k+1$, $\hat{M}_{k+1|k+1} = \{\hat{M}_{k+1|k+1}^S, \hat{M}_{k+1|k+1}^N\}$.

Modified Particle Petri Nets have been originally designed to monitor hybrid system mission in (Zouaghi, Alexopoulos, Wagner, & Badreddin, 2011b). The main advantage provided by MPPN is the way they manage uncertainties. In this article, we will focus on a way to use them in a context of health monitoring.

2.4. Application to Health Monitoring

The main objective of the system health monitoring is to determine the health state of the system at any time (Chanthery & Ribot, 2013). Diagnosis is used to identify the probable

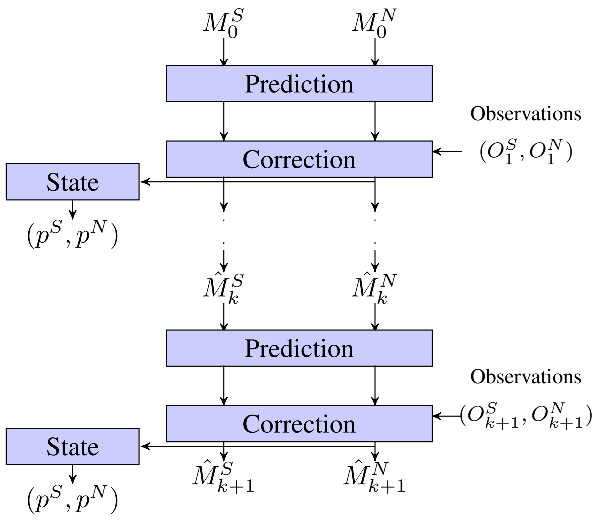

Figure 3. Hybrid sate estimation process of MPPN.

causes of the failures by reasoning on system observation. Thus diagnosis reasoning consists in detecting and isolating faults that may cause a system failure. Results of the diagnosis function lead to the current health state of the system. To perform model-based health monitoring of a hybrid system, it is necessary to represent both behavioral model and degradation model of the system. We are interesting in representing changes in system dynamics when one or several anticipated faults happen. Thinking that way, we define a *health mode* by a discrete health state coupled to a continuous behavior. Then health state estimation partially relies on common techniques for continuous variable estimation. As long as the system does not encounter any fault, it is in a *nominal mode*. We assume that tracked faults are permanent. This means that once a fault happens, the system moves from a nominal mode to a *degraded mode* or faulty mode. Without repair, system evolution is unidirectional and ends with a *failure mode* whereas the system is not operational anymore. This evolution is illustrated in Figure 4.

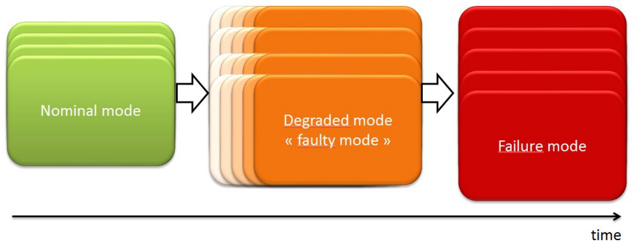

Figure 4. Unidirectional system evolution without maintenance or repair action.

Regarding the degradation model, we consider that faults in the system age depending on a stress level that is relative not to a behavior but to a health mode.

With the definition of the MPPN abstraction provided in previous sections, it is possible to model hybrid system behavior. Indeed, MPPN numerical places can be used to represent system dynamics, and symbolic places can be used to represent the different discrete health states of the system. Systems dynamics are then represented by differential equations. Thus, a hybrid state (p_i^S, p_j^N) will represent a health mode of the system. We designate by $Q = \{q_m\}$ the set of health modes of our system:

$$q_m = (p_i^S, p_j^N) \in Q \;\; if \;\; \exists t_l \in T, (p_i^S, p_j^N) \in (Post(t_l))^2 \tag{2}$$

where $Post(t_l)$ is the set of output places of t_l.

Using places that way, it becomes possible to use the symbolic conditions to model the occurrence of observable discrete events belonging to Σ_o and unobservable discrete events belonging to Σ_{uo} (faults, mission events, interaction with the environment, etc ...). $\Sigma = \Sigma_o \cup \Sigma_{uo}$ is defined as the set of discrete events of the system.

An example of a system behavioral model is described in Figure 5. In this example, the system has three different dynamics represented by p_5^N, p_6^N, p_7^N and four different health states p_1^S, p_2^S, p_3^S and p_4^S. By using Equation 2, five health modes are distinguishable. Health modes $q_1 = (p_1^S, p_5^N)$ and $q_2 = (p_1^S, p_6^N)$ are two nominal modes changing from the one to the other when condition $\Omega^S(t_1^S) = occ(e_1)$ or condition $\Omega^S(t_2^S) = occ(e_2)$ is satisfied. These conditions represent respectively the occurrence of observable events $e_1 \in \Sigma_o$ and $e_2 \in \Sigma_o$ supporting a change of behavior between p_5^N and p_6^N. Health modes $q_3 = (p_2^S, p_6^N)$ and $q_4 = (p_3^S, p_6^N)$ are two degraded modes reachable from health mode q_1 by satisfying the conditions $\Omega^S(t_3^S) = occ(f_1)$ and $\Omega^S(t_4^S) = occ(f_2)$ respectively. These two conditions represent respectively the occurrence of two unobservable fault events $f_1 \in \Sigma_{uo}$ and $f_2 \in \Sigma_{uo}$. Finally, $q_5 = (p_4^S, p_7^N)$ is a failure mode in which both f_1 and f_2 occurred and is reachable from the two degraded modes. Therefore $\Omega^N(t_5^S) = occ(f_1)$ is associated to the occurrence of f_1 and $\Omega^S(t_6^S) = occ(f_2)$ is associated with the occurrence of f_2.

While the design of the degradation model and its interaction with the behavioral model will be presented in Section 4, Section 3 will present a methodology to build a state tracker object called a diagnoser from the behavioral system model.

3. BEHAVIORAL DIAGNOSIS

In health monitoring, diagnosis is used to track system current health state. To do so, a common way is to generate a diagnoser of the system from the system model (Sampath, Sengupta, Lafortune, Sinnamohideen, & Teneketzis, 1995). The diagnoser is basically a monitor that is able to process any possible observable event on the system. It consists in recording these observations and providing the set of possible faults whose occurrence is consistent with these observations.

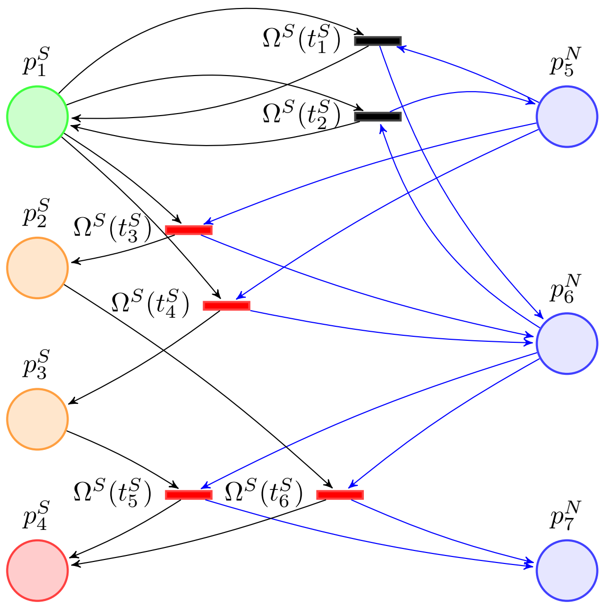

Figure 5. Example of system behavioral model using MPPN.

Concerning hybrid systems, one approach is to build a hybrid diagnoser (Bayoudh et al., 2008) from a hybrid automaton describing the system. The major idea is to abstract the continuous part of the system to only work with a discrete view of the system. This abstraction is done by using consistency tests, that take the form of a set of analytical redundancy relations (ARR). The diagnoser method is then directly applied on the resulting discrete event system. In previous works (Chanthery & Ribot, 2013), we extended this approach in order to integrate diagnosis and prognosis for hybrid systems. The main drawback of this approach is that the DES oriented diagnosis framework seems not the best suited for the incorporation of the highly probabilistic prognosis task. With the MPPN representation, we succeed in capturing all the uncertainties about the state knowledge, but also about the observations. Consequently, we have to develop a new diagnoser build from an MPPN. Moreover, the classical diagnoser is a finite state machine. If this theoretical object is very interesting for studying properties on system, like diagnosability or controllability, it is absolutely not suited for embedded systems, because the number of states of the diagnoser explodes for large models. Consequently, we choose to build a diagnoser based on a MPPN model for the following reasons:

- there is no lack of information during the diagnoser generation,
- MPPN model captures all the uncertainties,
- this representation is more compact than hybrid automa-

ton description, so the problem of embeddability of the diagnoser is reduced.

The diagnoser takes as input the MPPN specifying the behavior of the system and the set of online observations on the system. The output of the diagnoser is an estimation of the health state of the system. Next sections describe how to generate a diagnoser from an MPPN specifying the behavior of a system, then define what is finally called a diagnosis and how this object may be used for health monitoring.

3.1. Diagnoser Generation Based on MPPN

The goal of this section is to generate a MPPN that is able to monitor the system current health state thanks to the observations. Let suppose that the MPPN specifying the behavior of the system is a tuple $< P, T, Pre, Post, X, F, \gamma, \Omega, M_0 >$ as defined in Section 2.1. The set of places of the diagnoser remains the same as the one of the system. Concerning the transitions, there are two aspects to take into account.

First, it is necessary to follow the continuous behavior of the system with information issued from the observed variables of the system. A set of analytical redundancy relations (ARR) can be generated from the set of differential equations C of the system model. In the linear case, ARRs can be computed by using the parity space approach (Staroswiecki & Comtet-Varga, 2001). The parity space approach has been extended to multi-mode systems in (Cocquempot, El Mezyani, & Staroswiecki, 2004). In our case, a relation ARR_i is associated to each numerical place p_i^N. A numerical condition $\Omega^N(t_l)$ associated with a transition t_l linking two numerical places p_i^N and p_j^N carries ARR_{ij} satisfaction test, with $(i, j) \in \{1, ..., |P^N|\}^2$ and $l \in \{1, ..., |T|\}$. This means that $\Omega^N(t_l)(\pi)$ is satisfied when ARR_{ij} is satisfied for π. ARRs are satisfied if the observations satisfy the model constraints. Since ARRs are constraints that only contain observable variables, they can be evaluated online with the incoming observations given by the sensors. It is thus possible to check the consistency of the observed system behavior with the predicted one.

Secondly, because the diagnoser only captures the observable behavior of the system, a condition representing the occurence of an unobservable discrete event would never be satisfied. Consequently, all the symbolic conditions representing the occurences of unobservable events are removed from Ω without loss of information. Concerning the observable discrete part of the system, occurrences of observable discrete events will be used as symbolic condition triggers.

Once the system behavioral model is defined and all numerical conditions are computed from the ARRs generation, the corresponding diagnoser can be generated with the following steps:

Step 1: Add corresponding numerical conditions $\Omega^N(t_j^S)$ to every symbolic transition $t_j^S \in T^S$, with $j \in \{1, ..., |T|\}$. As a result, the symbolic transition t_j^S will be upgraded into a mixed transition $t_j^M \in T^M$.

Step 2: Remove, from any mixed transition $t_j^M \in T^M$, symbolic conditions $\Omega^S(t_j^M)$ covering the occurrence of an unobservable event, because these conditions would never be satisfied. Consequently, the mixed transition t_j^M is transformed in a numerical transitions $t_j^N \in T^N$.

Ambiguity: Hybrid system diagnosis consists in determining the health state of the system wherein observations are consistent. Diagnosis challenge is the ability to diagnose anticipated but unobservable faults in the system. In this context, modeling unobservable events can lead to ambiguity in the diagnoser. Indeed, the occurrence of several faults that can not be distinguishable with the observations of the systems will lead to ambiguous health states for the diagnoser. Therefore, a third step is needed during the diagnoser generation to track ambiguity. To do so, it is necessary to define a merger property to merge two numerical transitions. Two numerical transitions are mergeable if they are conditioned by the same dynamics change and if they share the same symbolic places in their sets of inputs places. In a more formal way, let $Pre(t_j)$ be the set of input places of a transition $t_j \in T$:

$$Pre(t_j) = \{p_i | Pre(i, j) \neq 0, i \in \{1, ..., |P|\}\} \tag{3}$$

As well, $Post(t_j)$ is the set of its output places:

$$Post(t_j) = \{p_i | Post(i, j) \neq 0, i \in \{1, ..., |P|\}\} \tag{4}$$

Definition 1 *Two numerical transitions $(t_i^N, t_j^N) \in (T^N)^2$, with $(i, j) \in \{1, ..., |T^N|\}^2$ and $i \neq j$ are mergeable if :*

$$(Pre(t_i^N) = Pre(t_j^N)) \ \wedge \ (Post(t_i^N) \cap P^N \cap Post(t_j^N) \neq \emptyset) \tag{5}$$

Note that condition (5) implies that the two transitions share the same numerical condition: $\Omega^N(t_i^N) = \Omega^N(t_j^N)$.

Step 3: Merge all mergeable transitions while there is at least two mergeable transitions using the following merging definition:

Definition 2 *The merging of two mergeable numerical transitions $(t_i^N, t_j^N) \in (T^N)^2$, with $(i, j) \in \{1, ..., |T^N|\}^2$ and $i \neq j$ is defined by two steps as follows:*

(1) *Creation of a new transition t_{ij}^N characterized by:*

$$\begin{cases} Pre(t_{ij}^N) &= Pre(t_i^N) \\ Post(t_{ij}^N) &= Post(t_i^N) \cup Post(t_j^N) \\ \Omega^N(t_{ij}^N) &= \Omega^N(t_i^N) \end{cases} \quad (6)$$

(2) *Introduction of t_{ij}^N and deletion of t_i^N and t_j^N in T:*

$$T = (T \backslash \{t_i^N, t_j^N\}) \cup \{t_{ij}^N\} \quad (7)$$

The resulting diagnoser of the model in Figure 5, after computing the third steps above, is presented in Figure 6.

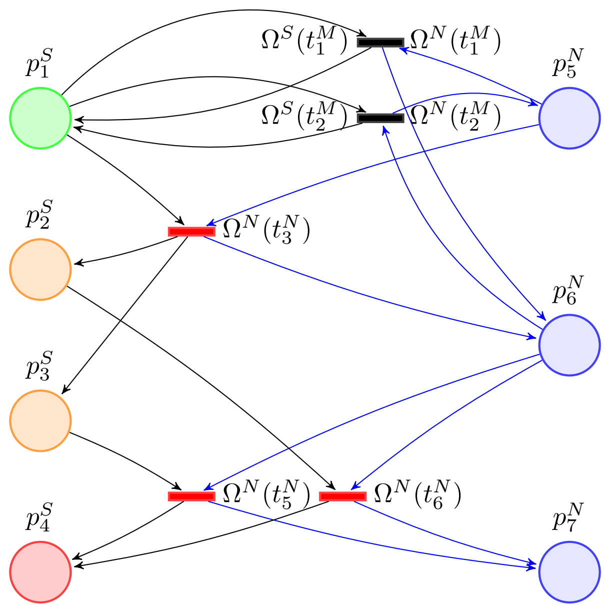

Figure 6. Example of diagnoser of system using MPPN.

In Figure 6, performing Step 1 has generated numerical condition Ω^N to every transition. Indeed, all transitions where supported by a change of dynamics that can be observed with the generation of the ARR. After this step on this example, all transitions are upgraded into mixed transitions. As there were unobservable events, symbolic conditions associated with the occurrence of f_1 and f_2 have been removed from the diagnoser model during Step 2, transforming t_3, t_4, t_5 and t_6 into numerical transitions. Finally, because transitions t_3 and t_4 were generating a change of dynamics from p_1^N to p_2^N, they were mergeable and thus have been merged into one single numerical transition t_3^N.

3.2. Behavioral Diagnosis Results

The behavioral diagnosis is defined at each clock tick as the state of the diagnoser. By using the MPPN, the diagnosis Δ_k at time k is the distribution of health mode believes that depends on particle values and weights and is deduced from the marking of the diagnoser at time k :

$$\Delta_k = \hat{M}_k = \{\hat{M}_k^S, \hat{M}_k^N\} \quad (8)$$

The marking $\hat{M}_k$ indicates the belief on the fault occurrences. It gives the same information than a classical diagnoser mode in terms of faults occurrences, with the same ambiguity. The difference is that in a classical diagnoser, every possible diagnosis has the same belief degree. With MPPN-based diagnoser, the ambiguity is valued by the knowledge about the weights of each particle of the marking.

Consequently, using the diagnosis results for health management becomes easier. Indeed, in the case of classical diagnoser, it is very difficult to ”choose” a belief state for the system in case of decision making. It is then very important to obtain the less ambiguous diagnosis as possible. In the case of MPPN-based diagnoser, each possible state of the system is valued, so it is easy to evaluate the more probable state at each clock tick.

4. DEGRADATION DIAGNOSIS

The previous part describes a way to use MPPN to monitor health state of the system based on its behavioral model. It is often interesting to take into account another level of representation to illustrate a different level of dynamics, or a more aggregate view of the system. For instance, in the framework of health monitoring, it is worth to look at the system at another level to take into account the degradation dynamics. Getting some information about the degradation of the system is a huge advantage for elaborating a more precise diagnosis and to perform prognosis.

Next sections describes what we call *Hybrid Particle Petri Nets* (HPPN). HPPN give a theoretical framework to represent MPPN at a higher level called the *hybrid level*. The purpose of this hybrid level is to represent some hybrid states characteristics, and not only continuous behavior or discrete state. A set of dynamics equations is used to follow hybrid information we are focused on. To point out this new hybrid level, we assume that places, transitions, conditions and tokens used in Section 2 and Section 3 are part of the *behavioral level*. Because of the new hybrid level, the enriched formalism is called Hybrid Particle Petri nets. The set of dynamics equations we focus on with the hybrid level represent component degradation laws, that depend on the health modes of the system. The update of the degradation value at each clock tick defines a degradation diagnosis function. The application of HPPN for health monitoring is then illustrated on an example.

4.1. Hybrid Level

A Hybrid Particle Petri Net is described as an enriched MPPN $< P, T, Pre, Post, X, C, H, \mathcal{F}, \gamma, \Omega, M_0 >$ where:

- P is the set of places, partitioned into numerical places P^N, symbolic places P^S and hybrid places P^H.
- T is the set of transitions (numerical T^N, symbolic T^S, mixed T^M and hybrid T^H).
- $H \subset \Re^n$ is the state space of the hybrid state vector.
- $\mathcal{F}$ is the set of dynamics equations of the system associated with hybrid places, representing hybrid state evolution.
- Ω is the set of conditions associated with the transitions (numerical Ω^N and symbolic Ω^S and hybrid Ω^H).

Hybrid places are used to compose the hybrid level and represent possible hybrid states of system. In HPPN, a hybrid state is a couple (p_i^S, p_j^N). For the sake of clarity in the paper, we will use $p_l^H = (p_i^S, p_j^N)$ to indicate that hybrid place p_l^H represents the hybrid state (p_i^S, p_j^N). Because hybrid states are combinations of symbolic places and numerical places, the set of hybrid states for a given behavioral model is always finite. However, only couples that are part of the set of output places of the same transition are considered as hybrid states. Formally:

$$p_l^H = (p_i^S, p_j^N) \in P^H \ if \ \exists t_m \in T, (p_i^S, p_j^N) \in (Post(t_m))^2 \tag{9}$$

Hybrid states that do not satisfy Condition 9 are considered as *intermediate states*. This means there is no information in the model about these hybrid states.

A hybrid place is marked by hybrid tokens $h_k^i = [s_k^i, \eta_k^i]$ with $i \in \{1, ..., |M_k^H|\}$ where M_k^H is the set of all the hybrid tokens in the net at time k. A hybrid token is defined by a couple $s_k^i = (\delta_k^j, \pi_k^l)$ of tokens running in the behavioral level and its corresponding hybrid state vector $\eta_k^i \in H$. The whole marking at time k of the HPPN is $M_k = \{M_k^S, M_k^N, M_k^H\}$. Now that hybrid tokens have been described, we are going to detail their creation and deletion rules.

Creation: Because of their dependencies on configurations and particles, new hybrid tokens are created at the same time of creation of a configuration or a particle. If a hybrid token h^i depends on a particle π^l that is duplicated during the particle filter step in a new particle π'^l, then h^i is also duplicated in h'^i but h'^i depends on the new particles π'^l.

Deletion: A hybrid token h^i depending on a configuration δ^j and a particle π^l is deleted when δ^j or π^l is deleted during the online process of the behavioral level.

Considering the two rules above, the hybrid level online process totally depends on the behavioral level online progress. However, the two processes are simultaneous.

Any hybrid place is linked with all other hybrid places through a hybrid transition $t_j^H \in T^H$.

$\forall p_i \in P$, $M_k(p_i)$ is the set of tokens in p_i at time k and $m_k(p_i) = |M_k(p_i)|$ is the number of tokens in p_i at time k.

Definition 3 *A hybrid transition $t_j^H \in T^H$ is fire-enabled at time k if:*

$$\exists p_i^H \in Pre(t_j^H), \quad m_k(p_i^H) \geq Pre(i,j) \tag{10}$$

A hybrid place is associated with a set of dynamics equations representing a hybrid state characteristic. The idea is to let evolve a hybrid token $h^i = [s^i, \eta^i]$ in the hybrid level in accordance to the symbolic and numerical places in which are evolving its associated configuration δ_k^j and its associated particle π_k^l, with $s^i = (\delta^j, \pi^l)$.

To formally define the firing of hybrid transitions, we need to define the following notations. $P(\delta^j) = p_j^S$ and $P(\pi^l) = p_l^N$ denote the projections of δ^j and π^l on the set of places P. Then, $P(s^i) = (p_j^S, p_l^N)$ denote the hybrid place of a couple $s^i = (\delta^j, \pi^l)$.

Every hybrid transition carries a hybrid condition $\Omega^H(t_j^H)$ which is satisfied if $\Omega^H(t_j^H)(h^i) = 1$. Hybrid tokens h^i are moved to another hybrid place p'^H if $P(s^i) = p'^H$. Formally:

$$\forall h^i = [s^i, \eta^i], \quad \Omega^H(t_j^H)(h^i) = \begin{cases} 1 & if \ \ P(s^i) = p'^H \\ 0 & otherwise \end{cases} \tag{11}$$

$\mathcal{S}_k^H(p^H)$ is the set of hybrid tokens in p^H satisfying the condition $\Omega^H(t_j^H)$ at time k:

Equation 11 implies that every transition t_j^H has only one hybrid output place p'^H and sees all the other hybrid places p^H as input places. More formally:

Definition 4 *The firing of a fire-enabled hybrid transition $t_j^H \in T^H$ at time k is defined by:*

$$\begin{cases} M_{k+1}^H(p^H) = M_k^H(p^H) \backslash \mathcal{S}_k^H(p^H) \\ M_{k+1}^H(p'^H) = M_k^H(p'^H) \cup \mathcal{S}_k^H(p^H) \end{cases} \tag{12}$$

An example of hybrid transition firing in a hybrid level is shown in Figure 7. In the example, there are two hybrid places $p_1^H = (p_1^S, p_1^N)$ and $p_2^H = (p_2^S, p_2^N)$. At time k, the two hybrid tokens $h_k^1 = [s_k^1, \eta_k^1]$ and $h_k^2 = [s_k^2, \eta_k^2]$ are following the characteristic of the hybrid state represented by p_1^H, so hybrid transitions t_1^H and t_2^H are fire-enabled. $P(s_k^1)$ is (p_1^S, p_1^N) but $P(s_k^2)$ is (p_2^S, p_2^N) so $\Omega^H(t_2^H)(h^2)$ is satisfied and h^2 is moved through t_2^H. Thus, h^2 is in the hybrid place p_2^H at time $k+1$ and follows the characteristic of the hybrid state (p_2^S, p_2^N).

This enrichment evolves all the possible hybrid states of the system alongside according to their corresponding laws. Indeed, because tokens in the behavioral level are changing of places during the prediction step (see Section 2.3 (1)), hybrid

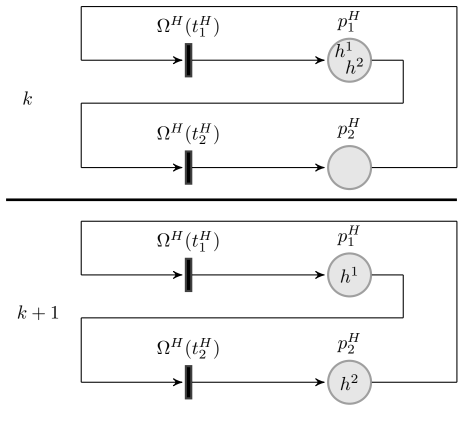

Figure 7. Illustration of firing rules of hybrid fire-enabled transitions.

tokens are simultaneous changing of places and their values are updated as follows:

$$\forall h^i_{k+1|k} \in \hat{M}^H_{k+1|k}(p^H_j), \quad \eta^i_{k+1} = F^j_{k+1}(\eta^i_k) \tag{13}$$

where $F^j_k \in \mathcal{F}$ is the set of dynamics equations associated with the hybrid place p^H_j. Because η^i_{k+1} depends on η^i_k, the continuity of the value η^i can be ensured. Figure 8 illustrates the evolution of the value η^2 of hybrid token h^2 of Figure 7. It shows that η^2_{k+1} is computed with the dynamics equation F^2_{k+1}. F^2_{k+1} is associated with p^H_2 and depends on η^2_k the value of η^2 at time k. This dependency ensures the continuity between F^1_k and F^2_{k+1} at time $k+1$.

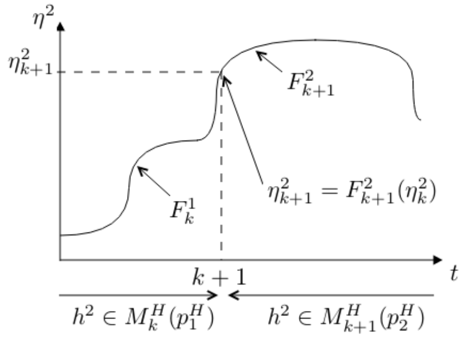

Figure 8. Illustration of the continuities of hybrid token values.

If $\mathcal{F}$ is not empty, the values η^i_k can be taken into account in the decision making process at time k that determine the marking at time $k+1$ of the behavioral level.

If the set of hybrid characteristics $\mathcal{F}$ is empty, the hybrid level directly monitors the hybrid state of the system over a distribution of hybrid tokens considering the particles weights. Moreover, considering a HPPN A, if $\mathcal{F} = \emptyset$ hybrid tokens has no value ($\eta^i = 0$) so they can be considered as configurations in another HPPN B. As well, if $\mathcal{F} \neq \emptyset$, values η^i of hybrid tokens evolve depending on the hybrid places and thus they can be considered as particle for HPPN B. By this way, hybrid tokens can go through a particle filter, making the hybrid level values having an effect on the configurations and particles of the behavioral level of HPPN A. Following this reasoning, we understand that the HPPN formalism is recursive. MPPN/HPPN can model hybrid systems, so by using only numerical places, numerical transitions and particles, its is possible to monitor continuous systems. As well, by using only symbolic places, symbolic transitions and configurations, it is possible to monitor discrete systems. This means that the HPPN formalism is also generic and can model different kind of systems such as heterogeneous systems. Finally, because HPPN is recursive, generic and can model discrete, continuous and hybrid systems, HPPN can be considered as a holistic method.

4.2. HPPN for Health Monitoring

This section introduces a way to represent uncertainty about degradation for each health mode of the system using probability measures.

The system description is enriched with a set of degradation laws modeling the degradation depending on hybrid state stress levels. The set of degradation laws is supposed to be accurately known. $\mathcal{F} = \{F^{q_m}, q_m \in Q\}$ is the set of degradation laws associated with health modes of the system. F^{q_m} is a vector of degradation laws for each anticipated fault in the health mode $q_m = (p^S_i, p^N_j)$. For example, in a system where n_f faults are considered:

$$F^{q_m}(t) = \begin{bmatrix} f_1^{q_m}(t) \\ f_2^{q_m}(t) \\ \vdots \\ f_{n_f}^{q_m}(t) \end{bmatrix} \tag{14}$$

where $f^{q_m}_j$ represents the probability distribution of the fault f_j at any time in the health mode q_m.

In the context of health monitoring, we need the formalism of the hybrid level to include health mode degradation laws in our model. We propose to consider health modes as hybrid states of an HPPN. Thus health modes are represented by hybrid places (see Section 2.4) and the set of degradation laws will be the set of dynamics equations associated with hybrid places.

Figure 9(b) represents the degradation laws model of the example of Figure 5. This system has five health modes (see Section 2.4), thus the corresponding hybrid level has five hybrid places $p^H_8 = (p^S_1, p^N_5)$, $p^H_9 = (p^S_1, p^N_6)$, $p^H_{10} = (p^S_2, p^N_6)$, $p^H_{11} = (p^S_3, p^N_6)$ and $p^H_{12} = (p^S_4, p^N_7)$. Therefore five hybrid transitions t^H_7, t^H_8, t^H_9, t^H_{10} and t^H_{11} deliver accesses to the five

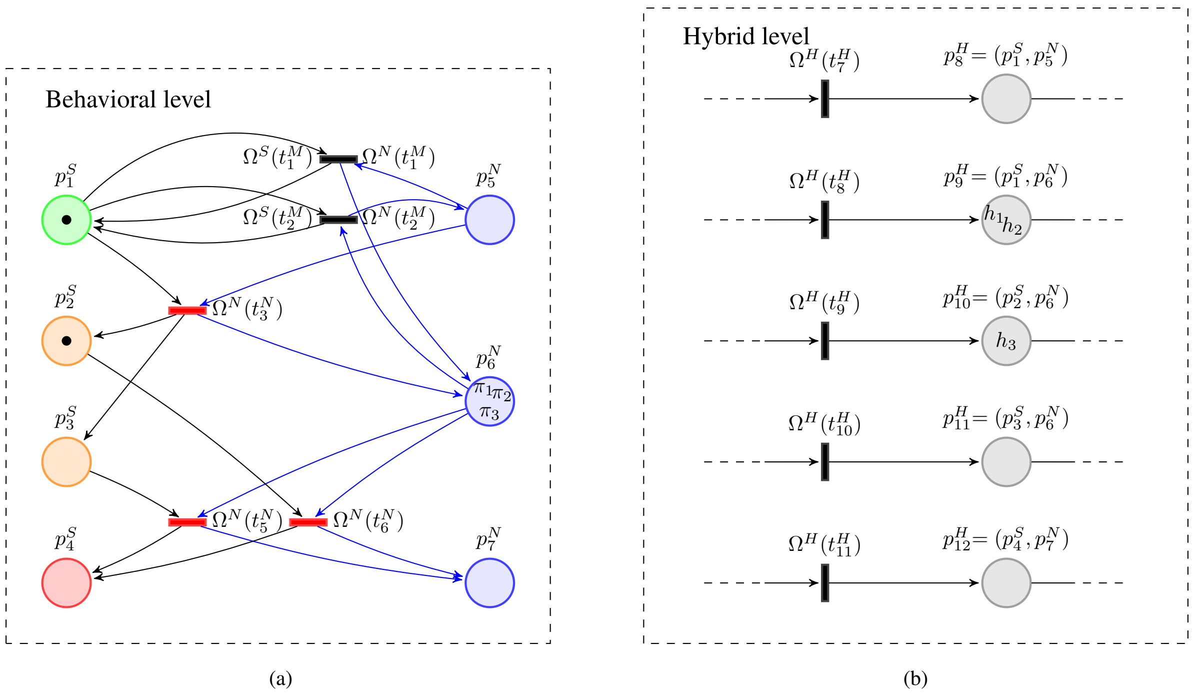

Figure 9. Example of diagnoser of system using HPPN.

hybrid places when associated hybrid conditions are satisfied (Equation 11). All the transitions are not represented in the figure because of the complexity of the representation.

4.2.1. Diagnoser Generation Based on HPPN

The diagnoser generation step does not change the degradation model during its computation. The degradation model is added to the behavioral diagnoser (Section 3.1) as a hybrid level. The result of the whole generation step is a HPPN-based diagnoser that monitors both the behavior and the degradation of the system.

Figure 9 shows the complete diagnoser of the system example presented in this paper. It illustrates the interactions between the behavioral level (a) and the hybrid level (b) of the diagnoser. Two configurations and three particles are running in the behavioral level. One configuration is in the symbolic places p_1^S and the other one in the symbolic place p_2^S. All three particles π^1, π^2 and π^3 are in numerical place p_6^N. Therefore, three hybrid tokens are running in the hybrid level. h^1 and h^2 are in the hybrid place p_9^H because they are linked to configuration in p_1^S and respectively π^1 and π^2. However, h^3 is in the hybrid place p_{10}^H because it is linked to the configuration in p_2^S and π^3.

4.2.2. Diagnosis Results

Using HPPN-based diagnoser, the diagnosis Δ_k of the system at time k is the complete marking of the diagnoser, indicating the distribution of health mode believes depending on particle values and weights and hybrid token values:

$$\Delta_k = \hat{M}_k = \{\hat{M}_k^S, \hat{M}_k^N, \hat{M}_k^H\} \tag{15}$$

The marking $\{\hat{M}_k^S, \hat{M}_k^N\}$ represents the belief on the health modes through a probabilty distribution. The marking $\hat{M}_k^H$ represents a degradation distribution over the health modes. Because each hybrid token depends on a particle and a configuration, its degradation value is linked with the belief of its health mode. Consequently, the belief and the degradation value can be correlated in case of decision making in the context of health management.

5. CONCLUSION AND FUTURE WORK

This paper formally introduces the HPPN approach to model the monitoring of hybrid systems. The MPPN method is enriched to consider another level to represent a hybrid dynamics. The method takes into account uncertainty about the knowledge of the system and uncertainty during the online process, such as continuous and discrete observations. The article then proposes to use HPPN to build a diagnosis methodology in a health monitoring context. HPPN can be used to model a diagnoser to monitor both discrete and continuous behaviors of the system, but also to consider the system degradation depending on the hybrid state of the system. The methodology is illustrated with an academic example. The building of such a diagnoser is a first step to perform prognosis and health management of hybrid systems under uncertainty. Moreover, diagnosis results can be used as probability distributions for decision making.

In future works, we will implement this work and test it on an embedded system. The prognosis methodology will be formally described considering the InterDP framework introduced in (Chanthery & Ribot, 2013) that interleaves diagnosis and prognosis methods to let results be more accurate.

REFERENCES

Bayoudh, M., Travé-Massuyes, L., & Olive, X. (2008). Hybrid systems diagnosis by coupling continuous and discrete event techniques. In *Proceedings of the IFAC World Congress* (pp. 7265–7270). Seoul, Korea.

Chanthery, E., & Ribot, P. (2013). An integrated framework for diagnosis and prognosis of hybrid systems. In 3rd *Workshop on Hybrid Autonomous System (HAS).* Roma, Italy.

Cocquempot, V., El Mezyani, T., & Staroswiecki, M. (2004). Fault detection and isolation for hybrid systems using structured parity residuals. In 5th *Asian Control Conference.* (Vol. 2, pp. 1204–1212).

Gaudel, Q., Chanthery, E., Ribot, P., & Le Corronc, E. (2014). Hybrid systems diagnosis using modified particle petri nets. In *Proceedings of the* 25th *International Workshop on Principles of Diagnosis (DX'14).* Graz, Austria.

Henzinger, T. (1996). The theory of hybrid automata. In *Proceedings of the* 11th *Annual IEEE Symposium on Logic in Computer Science* (pp. 278–292).

Lesire, C., & Tessier, C. (2005). Particle petri nets for aircraft procedure monitoring under uncertainty. In *Applications and Theory of Petri Nets* (pp. 329–348). Springer.

Ribot, P., Pencolé, Y., & Combacau, M. (2013). Generic characterization of diagnosis and prognosis for complex heterogeneous systems. *International Journal of Prognostics and Health Management*, *4*.

Roychoudhury, I., & Daigle, M. (2011). An integrated model-based diagnostic and prognostic framework. In *Proceedings of the* 22nd *International Workshop on Principle of Diagnosis (DX'11).* Murnau, Germany.

Sampath, M., Sengupta, R., Lafortune, S., Sinnamohideen, K., & Teneketzis, D. (1995). Diagnosability of discrete-event systems. *IEEE Transactions on Automatic Control*, *40*(9), 1555–1575.

Staroswiecki, M., & Comtet-Varga, G. (2001). Analytical redundancy relations for fault detection and isolation in algebraic dynamic systems. *Automatica*, *37*(5), 687–699.

Vinson, G., Ribot, P., Prado, T., & Combacau, M. (2013). A generic diagnosis and prognosis framework: application to permanent magnets synchronous machines. In *IEEE Prognostics and System Health Management Conference (PHM)* (pp. 1039–1044). Milano, Italy.

Zouaghi, L., Alexopoulos, A., Wagner, A., & Badreddin, E. (2011a). Modified particle petri nets for hybrid dynamical systems monitoring under environmental uncertainties. In *IEEE/SICE International Symposium on System Integration (SII)* (pp. 497–502).

Zouaghi, L., Alexopoulos, A., Wagner, A., & Badreddin, E. (2011b). Probabilistic online-generated monitoring models for mobile robot navigation using modified petri net. In 15th *International Conference on Advanced Robotics (ICAR)* (pp. 594–599).