Evaluating the robustness of scheduling in uncertain environment with Petri nets

Dimitri Lefebvre

Normandie Univ, UNIHAVRE, GREAH, 76600 Le Havre, France
e-mail: dimitri.lefebvre@univ-lehavre.fr

ABSTRACT

This paper is about the incremental computation of control sequences in uncertain environments where uncontrollable events may occur. Timed Petri nets under earliest firing policy are used for that purpose. The aim is to drive the marking of the net from an initial value to a reference one, in minimal or near-minimal time, by avoiding forbidden markings, deadlocks and dead branches. The approach is inspired from model predictive control and at each step only a small area of the reachability graph is explored in order to limit the computation effort. For each computed sequence the probability of uncontrollable events is estimated to evaluate the robustness of the resulting trajectory. A sufficient condition is provided to compute robust trajectories. The proposed results are applicable to a large class of discrete event systems in the domains of flexible manufacturing, communication, computer science, transportation and traffic. In particular, they are suitable for dynamical deadlock-free scheduling and reconfiguration problems in uncertain environments.

CCS CONCEPTS

• **Computer systems organization** → Dependable and fault tolerant systems and networks• **Computing methodologies** → modeling and simulation

KEYWORDS

Discrete event systems, timed Petri nets, stochastic processes, scheduling problems, manufacturing systems

1 INTRODUCTION

The design of controllers that optimize a specific cost function is a basic objective in many control problems in particular in scheduling problems that aim to allocate a limited number of resources within several users or servers. In the domains of flexible manufacturing, communication, computer science, transportation and traffic, the makespan is commonly used as an effective cost function because it leads directly to the design of cycles of tasks with minimal duration. The difficulty is that scheduling problems are known to be NP-hard due to multi-layer resource sharing and routing flexibility of the jobs. Thus, a large literature has been devoted to such optimization in the operations research, automatic control and computer science communities. In operations research community, flow-shop and job-shop problems have been investigated for a long time [1] [2] and a lot of contributions have been proposed, based either on heuristic methods (like Nawaz, Enscore and Ham or Campbell, Dudek and Smith heuristics) or artificial intelligence and evolutionary theory [3] [4] [5]. In the automatic control community, automata, Petri nets (PNs) and (max,+) algebra have been used to solve scheduling problems for discrete event systems (DESs) [6] [7]. In particular, with PNs, the pioneer contributions for scheduling problems have been provided by some adaptations of the Dijkstra and A* algorithms to the PNs [8] [9]. Such algorithms explore the reachability graph of the net, in order to generate optimal or sub-optimal schedules. Numerous improvements have been developed: pruning of non-promising branches [10] [11], backtracking limitation [12], determination of lower bounds for the makespan [13], best first search with backtracking and heuristic [14] or dynamic programming [15]. In order to avoid deadlocks, a few results also combine scheduling and supervisory control in the same approach: search in the partial reachability graph [16], genetic algorithms [17] and heuristic functions based on the firing vector [13] [18]. The performance of operations research approaches are good in general compared to the automatic control approaches as long as static scheduling is considered. The advantage of PNs and other tools developed in control theory is to use a common formalism to describe a large class of problems. This makes such approaches suitable for dynamic and robust scheduling in uncertain environments. But modularity and genericity is usually paid by a large computational effort that disqualifies the approaches for numerous large systems

This work proposes a method for timed PNs under earliest firing policy that incrementally computes control sequences in uncertain environments. Uncertainties are assumed to result from system failures or other unexpected events and robustness with respect to such uncertainties is obtained thanks to a model predictive control (MPC) approach [25] [26]. As a consequence, it is suitable for real time control, dynamical scheduling and reconfiguration. The system and the uncertain environment are modeled with Timed PNs where some transitions are controllable and others are not. The controller can select the controllable transition that should fire next such that the whole computed control sequence reaches a reference state from an initial one. This sequence avoids forbidden states, unknown deadlocks and dead-branches with a trajectory of minimal

VALUETOOLS 2017, December 5–7, 2017, Venice, Italy

ACM ISBN 978-1-4503-6346-4/17/12…$15.00
https://doi.org/10.1145/3150928.3150944

or near-minimal duration, under earliest firing policy. The approach is based on a partial exploration of the PN reachability graph but limits this exploration to the neighborhood of the current marking. Moreover, for each computed sequence the risk to deviate from the reference is estimated. Finally robust (i.e. zero-risk) trajectories are computed. Compared to our previous works [19] [20] [21] [22], the main contribution is to include explicitly uncertainties by means of uncontrollable stochastic transitions in the PN model, to evaluate the risk of the computed control sequences and to propose a sufficient conditions for the existence of a robust trajectory. This paper improves our preliminary results in uncertain environments [28].

The paper is organized as follows. In Section 2, timed PNs with uncontrollable transitions and risk of control sequences are introduced. Section 3 presents the approach to compute non-robust and robust firing sequences with minimal duration. Section 4 is a case study. Section 5 sums up the conclusions and perspectives.

2 TIMED PNS WITH UNCONTROLLABLE TRANSITIONS

2.1 Petri nets

A PN structure is defined as $G = <\boldsymbol{P}, \boldsymbol{T}, W_{PR}, W_{PO}>$, where $\boldsymbol{P} = \{P_1,\ldots, P_n\}$ is a set of n places and $\boldsymbol{T} = \{T_1,\ldots, T_q\}$ is a set of q transitions of labels $\{1,...,q\}$, $W_{PO} \in (\mathbf{N})^{n\times q}$ and $W_{PR} \in (\mathbf{N})^{n\times q}$ are the post and pre incidence matrices ($\mathbf{N}$ is the set of non-negative integer numbers), and $W = W_{PO} - W_{PR} \in (\mathbf{Z})^{n\times q}$ ($\mathbf{Z}$ is the set of positive and negative integer numbers) is the incidence matrix. $<G, M_I>$ is a PN system with initial marking M_I and $M \in (\mathbf{N})^n$ represents the PN marking vector. The enabling degree of transition T_j at marking M is given by $n_j(M)$:

$$n_j(M) = \min\{\lfloor m_k / w^{PR}_{kj} \rfloor : P_k \in {}^{\circ}\boldsymbol{T_j}\} \tag{1}$$

where ${}^{\circ}\boldsymbol{T_j}$ stands for the set of T_j upstream places, m_k is the marking of place P_k, w^{PR}_{kj} is the entry of matrix W_{PR} in row k and column j. A transition T_j is enabled at marking M if and only if (iff) $n_j(M) > 0$, this is denoted as $M\,[T_j>$. When T_j fires once, the marking varies according to $\Delta M = M' - M = W(:, j)$, where $W(:, j)$ is the column j of incidence matrix. This is denoted by $M\,[T_j > M'$ or equivalently by $M' = M + W.X_j$ where X_j denotes the firing count vector of transition T_j [7]. A firing sequence σ is defined as $\sigma = T(j_1)T(j_2)\ldots T(j_h)$ where $j_1,\ldots j_h$ are the labels of the transitions. $X(\sigma) \in (\mathbf{N})^q$ is the firing count vector associated to σ, $|\sigma| = \|X(\sigma)\|_1 = h$ is the length of σ ($\|\,.\,\|_1$ stands for the 1-norm), and $\sigma = \varepsilon$ stands for the empty sequence. The firing sequence σ fired at M leads to the marking trajectory (σ,M):

$$(\sigma,M) = M(0)\,[T(j_1) > M(1)\ldots. > M(h\text{-}1)\,[T(j_h) > M(h) \tag{2}$$

where $M(0) = M, M(1),...,M(h\text{-}1)$ are the intermediate markings and $M(h)$ is the final marking (in the next, we write $M(k) \in (\sigma,M)$, $k = 0,\ldots h$). A marking M is said ***reachable*** from initial marking M_I if there exists a firing sequence σ such that (st) $M_I\,[\sigma>M$ and σ is said ***feasible*** at M_I. $\boldsymbol{R}(G, M_I)$ is the set of all reachable markings from M_I.

2.2 Forbidden, dangerous and legal markings

For control issues, the set of transitions $\boldsymbol{T}$ is divided into 2 disjoint subsets $\boldsymbol{T_C}$, and $\boldsymbol{T_{NC}}$ such that $\boldsymbol{T} = \boldsymbol{T_C} \cup \boldsymbol{T_{NC}}$. $\boldsymbol{T_C}$ is the subset of q_C controllable transitions, and $\boldsymbol{T_{NC}}$ the subset of q_{NC} ***uncontrollable*** transitions. Without loss of generality $\boldsymbol{T_C} = \{T_1,\ldots,T_{qC}\}$ and $\boldsymbol{T_{NC}} = \{T_{qC+1},\ldots,T_{qC+qNC}\}$. The firing of enabled controllable transitions are enforced or avoided by the controller whereas the firing of uncontrollable transitions are not and fire spontaneously according to some unknown random processes. A set of ***marking specifications*** is also defined with the function *SPEC*: for any marking $M \in \boldsymbol{R}(G, M_I)$, $SPEC(M) = 1$ if M satisfies the marking specifications otherwise $SPEC(M) = 0$. Considering a reference marking M_{ref} to be reached from M_I, the 2 disjoint sets $\boldsymbol{F}(G, M_I, M_{ref})$ and $\boldsymbol{L}(G, M_I, M_{ref})$ of respectively forbidden and legal markings are introduced:

$$\boldsymbol{L}(G, M_I, M_{ref}) = \{M \in \boldsymbol{R}(G, M_I) \text{ st } \exists\ \sigma \in (\boldsymbol{T_C})^* \text{ with } M\,[\sigma > M_{ref} \text{ with } (SPEC(M') = 1) \text{ for all } M' \in (\sigma,M)\} \tag{3}$$

$$\boldsymbol{F}(G, M_I, M_{ref}) = \boldsymbol{R}(G, M_I)/\boldsymbol{L}(G, M_I, M_{ref}) \tag{4}$$

In other words, a marking $M \in \boldsymbol{R}(G, M_I)$ is ***legal*** with respect to (wrt) M_{ref} if a marking trajectory exists from M to M_{ref} that contains only controllable transitions and intermediate markings that satisfy the specifications. In addition, a legal marking M is ***robust*** wrt to $\boldsymbol{T_C}$ if $\boldsymbol{M^{\circ}} \subseteq \boldsymbol{T_C}$, where $\boldsymbol{M^{\circ}}$ stands for the set of transitions enabled at M otherwise M is ***dangerous*** (Fig. 1). On the contrary, a ***forbidden*** marking is a marking from which no controllable trajectory exists to the reference. Examples of forbidden markings are deadlocks or markings that do not satisfy the system specifications or markings that enable only uncontrollable transitions (Fig. 1).

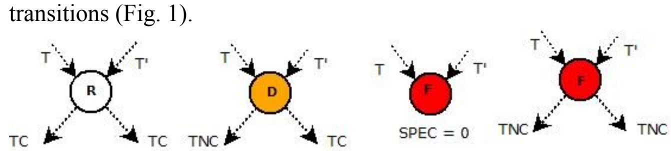

Figure 1: Robust, dangerous and forbidden markings

The previous definitions are extended to marking trajectories. A robust marking trajectory is a legal trajectory that visits only robust markings. On the contrary a dangerous marking trajectory is a legal trajectory that visits at least one dangerous marking.

2.3 TPNs with uncontrollable transitions

Timed Petri nets are PNs whose behaviors are constrained by temporal specifications [7]. For this reason, timed PNs have been intensively used to describe DESs like production systems [6]. This paper concerns partially controlled timed PNs under infinite server semantic where controllable transitions fire according to an earliest firing policy with firing preselection (computed by the controller) and time specifications similar to the one used for T-timed PNs

[23]: if $T_j \in \boldsymbol{T_C}$, the firing of T_j occurs at earliest after a minimal delay $d_{min\ j}$ from the date it has been enabled. If no time specification exists for T_j, then $d_{min\ j} = 0$ and the firing may occur immediately once the transition is enabled. On the contrary, the firings of uncontrollable transitions are unpredictable: if $T_j \in \boldsymbol{T_{NC}}$, the firings of T_j occur according to an unknown arbitrarily random process at any time from the date it has been enabled. Consequently, partially controlled timed PNs (PCont-TPNs) are defined as $<G, M_I, D_{min}>$ where $D_{min} = (d_{min\ j}) \in (\mathbf{R}^+)^{qC}$ and $\mathbf{R}^+$ is the set of non-negative real numbers. If in addition, the stochastic dynamics of the uncontrollable transitions are driven by exponential probability density functions (pdfs) of parameters $\mu = (\mu_j) \in (\mathbf{R}^+)^{qNC}$, with an infinite server semantic, a race policy and a resampling memory [24] then partially controlled stochastic timed PNs (PCont-SPNs) defined as $<G, M_I, D_{min}, \mu>$ will be used instead of PCont-TPNs. The introduction of PCont-TPNs is motivated when uncertainties correspond to failure processes that are represented with exponential distributions.

A timed firing sequence σ of length $|\sigma| = h$ and of duration t_h is defined as $\sigma = T(j_1, t_1)T(j_2, t_2)\ldots T(j_h, t_h)$ where $j_1,\ldots j_h$ are the labels of the transitions and $t_1,\ldots,t_h$ represent the dates of the firings that satisfy $0 \leq t_1 \leq t_2 \leq \ldots \leq t_h$. The timed firing sequence σ fired at M leads to the timed marking trajectory (σ,M):

$$(\sigma,M) = M(0)\ [T(j_1, t_1) > M(1)\ldots.\ M(h\text{-}1)\ [T(j_h, t_h) > M(h) \quad (5)$$

As long as earliest firing policy is considered for controllable transitions, Algorithm 1 transforms an untimed marking trajectory of the form (2) that contains only controllable transitions, in a straightforward way, into a timed marking trajectory of the form (5) of minimal duration [20] [21]. Algorithm 1 also returns $DURATION(\sigma,M) = \tau$. In simple words, Algorithm 1, uses the chronological firing order of the untimed marking trajectory and the earliest firing policy to update at each new marking M the remaining durations of the transitions enabled at M. These transitions and there remaining firing durations are stored in a calendar CAL that changes at each intermediate marking of the trajectory. The algorithm searches the earliest occurrence of the next transition in CAL and computes its firing date from the current date and the remaining firing duration. It transforms step by step the untimed trajectory in a timed one.

Algorithm 1

1. initialization: $\tau \leftarrow 0$; $CAL \leftarrow \{(T_j, d_{min\ j})$ st $M\ [\ T_j >\}$,
   $\sigma' \leftarrow (\varepsilon,0)$, $h \leftarrow |\ \sigma|$
2. for k from 1 to h
3. find in CAL the date τ_k of the earliest occurrence of the k^{th} transition $T(j_k)$ in σ
4. $\tau \leftarrow \tau_k$, remove entry $(T(j_k), \tau_k)$ in CAL
5. $CAL_{new} \leftarrow \emptyset$, $M' \leftarrow M - W_{PR}.X(T(j_k))$
6. for all T' st $M'\ [\ T' >$
7. compute the enabling degree $n'(T',M')$ of T' at M'
8. for j from 1 to $n(T',M')$
9 find the j^{th} occurrence (T', τ'_j) of T' in CAL
10. $CAL_{new} \leftarrow CAL_{new} \cup (T', \max(\tau'_j, \tau))$
11. end for
12. end for
13. $M'' \leftarrow M' + W_{PO}.X(T(j_k))$
14. for all t'' st $M''\ [\ T'' >$
15. compute the enabling degree $n''(T'',M'')$ of T'' at M''
16. for j from 1 to $n''(T'',M'') - n'(T'',M')$
17. $CAL_{new} \leftarrow CAL_{new} \cup (T'', \tau + d_{min}(T''))$
18. end for
19. end for
20. $CAL \leftarrow CAL_{new}$, $\sigma' \leftarrow \sigma'\ (T(j_k), \tau_k)$, $M \leftarrow M''$
21. end for
22. $\tau \leftarrow \tau_h$

2.4. Belief and probability of trajectory deviation

The objective of the proposed control design is (1) to reach the reference marking M_{ref} starting from an initial robust legal marking M_I with a legal trajectory (σ,M_I) of minimal duration; (2) to evaluate the risk that the trajectory will deviate from the reference. Such a risk can be trivially estimated with the belief $RB(\sigma,M_I,\boldsymbol{T_C})$ that uncontrollable firings occur during the execution of (σ,M_I):

$$RB(\sigma,M_I,\boldsymbol{T_C}) = h_{NC} / h \quad (6)$$

where h_{NC} is the number of intermediate dangerous markings in (σ,M_I) and h is the number of markings visited by (σ,M_I). This very trivial risk assessment can be used if no additional knowledge about the uncertainties exists. For PCont-SPNs, the uncertainties are modelled with transitions that fire according to exponential pdfs and the belief $RB(\sigma,M_I,\boldsymbol{T_C})$ is replaced by the probability $RP(\sigma,M_I,\boldsymbol{T_C})$ that satisfies Proposition 1:

Proposition 1: Let $<G, M_I, D_{min}, \mu>$ be a PCont-SPN with M_I a legal robust marking. Let M_{ref} be a reference marking and (σ,M_I) be a legal marking trajectory to M_{ref}. The probability $RP(\sigma,M_I,\boldsymbol{T_C})$ that (σ,M_I) deviates from the reference is given by:

$$RP(\sigma, M_I, \boldsymbol{T_C}) = \sum_{1 \leq k_1 \leq h} \pi(k_1) - \sum_{1 \leq k_1 < k_2 \leq h} \big(\pi(k_1).\pi(k_2)\big) + \cdots + (-1)^{h-1}. \sum_{1 \leq k_1 < \ldots < k_{h-1} \leq h} \big(\pi(k_1) \ldots \pi(k_{h-1})\big) + (-1)^h.\pi(1)\ldots\pi(h) \quad (7)$$

with:

$$\pi(k) = 1 - \exp\Big(-\textstyle\sum_{T_j \in \boldsymbol{T_{NC}} \cup (\boldsymbol{M(k)})^\circ} \big(\mu_j.n_j(M(k))\big).d_{j_k}\Big) \quad (8)$$

where $d_{jk} = t_{k+1} - t_k$ is the remaining time to fire $T(j_{k+1}, t_{k+1})$ at date t_k.

Proof: $RP(\sigma,M_I,\boldsymbol{T_C})$ is the probability to fire uncontrollable transitions when the trajectory (σ,M_I) is executed.

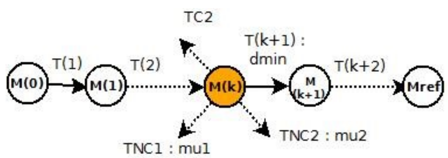

Figure 2: An example of dangerous trajectory

Consider the trajectory in Fig. 2. As long as the uncontrollable transitions T_1 and T_2 behave according to exponential pdfs with an infinite server policy, their mean durations are respectively $\mu_1.n_1(M(k))$ and $\mu_2.n_2(M(k))$. The probability that the uncontrollable transition T_{NC1} or T_{NC2} fires before $T(j_{k+1}, t_{k+1})$ and that the trajectory deviates from M_{ref} at $M(k)$ is given by:

$$\pi(k) = Prob(T_{NC1} \text{ or } T_{NC2} \text{ fires before } T(j_{k+1}, t_{k+1}))$$

$$= 1 - \exp(-\mu_1.n_1(M(k)).d_{j_k} - \mu_2.n_2(M(k)).d_{j_k})$$

Alternatively the probability that the trajectory continues to $M(k+1)$ at $M(k)$ is given by:

$$1 - \pi(k) = Prob(T(j_{k+1}, t_{k+1}) \text{ fires before } T_{NC1} \text{ and } T_{NC2})$$
$$= \exp\left(-\mu_1.n_1(M(k)).d_{j_k} - \mu_2.n_2(M(k)).d_{j_k}\right) \qquad (9)$$

Thus, $RP(\sigma, M_I, \boldsymbol{T_C})$ is finally given by:

$$RP(\sigma, M_I, \boldsymbol{T_C}) = \pi(0) + (1 - \pi(0))(\pi + (1 - \pi(1)) \ldots \pi(h)))$$

which exhaustive development is easily rewritten as in (7). Note that the duration of other controllable transitions enabled at $M(k)$ (for example, T_{C2} in Fig. 2) is not considered because this transition does not belong to (σ, M_I).

If the controllable transition $T(j_{k+1}, t_{k+1})$ fires at earliest after a duration d_{jk}, then the probability $\pi(k)$ can be approximated by considering the mean firing rate $1/d_{jk}$ of $T(j_{k+1}, t_{k+1})$. To evaluate the probability $\pi(k)$, $1/d_{jk}$ is compared to the mean firing duration of the stochastic uncontrollable transitions T_{NC1} and T_{NC2} and (8) may be replaced by:

$$\pi(k) \cong \frac{\sum_{T_j \in \boldsymbol{T_{NC}} \cup (\boldsymbol{M(k)})^{\circ}} (\mu_j.n_j(M(k)))}{\sum_{T_j \in \boldsymbol{T_{NC}} \cup (\boldsymbol{M(k)})^{\circ}} (\mu_j.n_j(M(k)) + (d_{j_k})^{-1})} \qquad (10)$$

2.5. Example

PN1 is considered with $\boldsymbol{T_C} = \{T_1, T_2, T_3, T_4, T_5, T_6\}$, $\boldsymbol{T_{NC}} = \{T_7\}$, $D_{min} = (1, 1, 1, 1, 1, 1)^T$ and $\mu = \mu_7$ (Fig. 3). The control objective is to reach $M_{ref} = (5\ 0\ 0\ 0)^T$ from $M_I = (1\ 0\ 0\ 0)^T$ and no additional marking constraint is considered. The cycles $\{p_1, t_1, p_2, t_2\}$ and $\{p_1, t_3, p_3, t_4\}$ are both marking producers due to the weighted arcs: the execution of $\{p_1, t_3, p_3, t_4\}$ multiplies each token by 5 compared to $\{p_1, t_1, p_2, t_2\}$ that multiplies it by 2 only. Thus, sequences with cycle $\{p_1, t_3, p_3, t_4\}$ will reach more rapidly the reference. But the uncontrollable transition T_7 may fire during execution of this cycle that may lead to an excessive production of tokens. Finally the cycle $\{p_1, t_5, p_4, t_6\}$ is marking consumer.

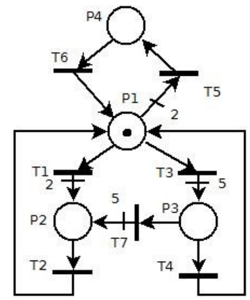

Figure 3: Example PN1

The optimal timed sequence to reach M_{ref} is given by $\sigma_1 = T(3, 1)(T(4, 2))^5$ with duration $DURATION(\sigma_1) = 2$ time units (TUs). A suboptimal sequence $\sigma_2 = T(1, 1)(T(2, 2))^2(T(1, 3))^2(T(2, 4))^4T(1, 5)(T(2, 6))^2$ that also reaches M_{ref} with duration $DURATION(\sigma_2) = 6$ TU is also considered. In order to illustrate the interest to introduce the risk evaluation in the control strategy and to motivate the advantage to use the risk probabilities (*RP*) better than the risk beliefs (*RB*) when the dynamics of the uncontrollable firings are known, Table 1 reports the indicators *RB* and *RP* for both sequences according to several values of μ. From Table 1, one can notice that the sequence σ_2 that is suboptimal in time has the advantage to be robust compared to σ_1. When the risk is evaluated as the belief (6), an unexpected firing of T_7 may occurs at 5 intermediate markings over a trajectory of length 6 and consequently $RB(\sigma, M_I, \boldsymbol{T_C}) = 5/6$. When the risk is evaluated as a probability, the results depends strongly on the dynamic of the stochastic firing of uncontrollable transition T_7. Evaluating the probability $RP(\sigma, M_I, \{T_7\})$ instead of $RB(\sigma, M_I, \{T_7\})$ with approximation (10) provides a better evaluation of the deviation risk.

Table 1: Deviation risk for σ1 and σ2

	RB	*RP*		
		$\mu = 0.1$	$\mu = 1$	$\mu = 10$
σ_1	5/6=0.83	1/3=0.33	5/6=0.83	50/51=0.98
σ_2	0	0	0	0

3. Model predictive control for PCont-TPNs

The determination of control sequences for PNs that contains only controllable transitions has been considered in our previous works [19] [20] with a model predictive control (MPC) approach adapted for DESs. In this section this approach is extended to PCont-TPNs (and consecutively to PCont-SPNs). The basic idea of MPC is to anticipate the evolution of the system in order to achieve the control objective. At each step, the future trajectory is predicted from the current state. A sequence of control actions is computed by minimizing a cost function. The first action of this sequence is stored and the prediction starts again from the new state reached by the system [25] [26].

3.1. Cost function based on the firing count vector

The cost function $J_{FC}(M,M_{ref}) = (D_{min})^T.X$ based on the estimation of the firing count vector X required to reach the reference M_{ref} from the marking M has been introduced in our previous work [21] to estimate the time to the reference. In this section, this cost function is rewritten for PCont-TPNs. For this purpose let us define G_C, $W_C \in (\mathbf{N})^{n \times qC}$ and $X_C \in (\mathbf{N})^{qC}$ as the restrictions of G, W and X to the set of controllable transitions $\boldsymbol{T_C}$. The controllable firing count vector X_C that satisfies $M_{ref} - M = W_C.X_C$ and minimize $J_{FC}(M,M_{ref}) = (D_{min})^T.X_C$ is obtained according to an integer optimization problem:

$$\text{Min } \{(D_{min})^T.X_C : X_C \in (\mathbf{Z})^{qC} \text{ st } X_C \geq 0 \text{ and } W_C.X_C = (M_{ref} - M)\} \qquad (11)$$

The linear optimization problem (11) has a solution with integer values as long as $M_{ref} \in \boldsymbol{R}(G_C, M)$ and the cost function $J_{FC}(M,M_{ref})$ based on firing count vector X_C and on D_{min} is defined by:

$$J_{FC}(M,M_{ref}) = (D_{min})^T.X_C \qquad (12)$$

As long as X_C corresponds to a feasible and legal firing sequence σ to the reference (ie. X_C does not encode a spurious solution of (11)), $J_{FC}(M,M_{ref})$ provides an upper bound of the duration of σ as proved with Proposition 2.

Proposition 2: Let consider a PCont-TPN (or PCont-SPN) of parameter D_{min} under earliest firing policy. Let M_{ref} be a reference marking, (σ,M_I) a legal marking trajectory to M_{ref} with $\sigma \in \boldsymbol{T_C}^*$ and $X_C(\sigma) \in (\mathbf{N})^{qC}$ be the firing count vector of σ. The minimal duration $DURATION(\sigma,M_I)$ to execute (σ,M_I) satisfies:

$$DURATION(\sigma,M_I) \leq (D_{min})^T.X_C(\sigma) \qquad (13)$$

Proof: (σ,M_I) is written as in (5). $T(j_1, t_1)$ is enabled at date 0 and fires at date $t_1 = d_{min\,j1}$ to result in marking $M(1)$. $T(j_2, t_2)$ is enabled at date 0 or t_1 and fires not later than at date $t_1 + d_{min\,j2}$. Thus $t_2 \leq d_{min\,j1} + d_{min\,j2}$. The same reasoning is repeated h times. $T(j_h, t_h)$ is enabled at latest at date t_{h-1} and fires not later than at date $t_{h-1} + d_{min\,jh}$. Thus $t_h \leq d_{min\,j1} + \ldots + d_{min\,jh}$. The minimal duration of (σ,M_I) is t_h, thus (13) holds.

3.2. Prediction and control action updating

The basic idea is to use $J_{FC}(M,M_{ref})$ to drive iteratively the search of the controllable firing sequence of minimal duration that leads to the reference. At each step (i.e. for each intermediate marking), the prediction is obtained with a local exploration of the controllable part of the reachability graph and an estimation of the remaining duration to the reference obtained with cost function $J_{FC}(M,M_{ref})$ is computed. Then the first control action (i.e. firing of the next controllable transition) is stored. If an uncontrollable firing occurs, the trajectory deviates from the predicted one and the system enters in an unexpected state. But the deviation is immediately taken into account by the controller that updates the control sequence at the next step. For this reason the proposed strategy is suitable to propose dynamical and robust scheduling. Two algorithms detailed in our previous works [21] [22] are adapted for control issues in uncertain environments.

A preliminary algorithm (Algorithm 1 in [21] [22]) encodes as a tree $\boldsymbol{Tree}(M,H)$ a small part of depth H of the reachability graph rooted at M (Fig. 4).

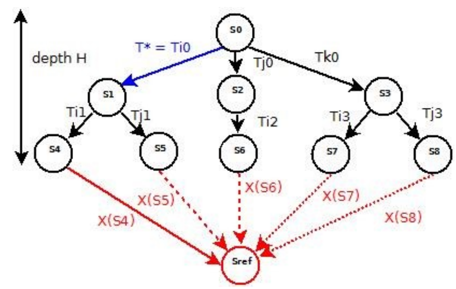

Figure 4: Prediction with Algorithm 2 [21] [22]

Each node $S = \{m(S), \sigma(S), s(S), l(S), e(S)\} \in \boldsymbol{Tree}(M,H)$ is tagged with a marking $m(S)$, the firing sequence $\sigma(S)$ st $M\,[\sigma(S) > m(S)$, and the sequence of nodes $s(S)$ in the tree from M to $m(S)$. In addition, the flags $l(S)$ and $e(S)$ are introduced st $l(S) = 0$ if S is forbidden, otherwise $l(S) = 1$ and $e(S) = 1$ if S is a terminal node of the tree, otherwise $e(S) = 0$. At each intermediate marking, the algorithm returns the next transition T^* to fire. In simple words, Algorithm 2 explores some reachable states in the neighborhood of the current marking and for each explored state it evaluates the criterion J_{FC} as a distance to the reference in order to select the state with the minimal value and to return the first transition of the sequence that reach that state. The explored neighborhood is limited by two parameters: the maximal number H of successive firings and a maximal duration H_τ of encoded sequences. Once, one of the two previous limits is reached, the exploration stops and the state is tagged as a terminal node for the considered neighborhood.

Algorithm 2 [21] [22]

1. if $M \in \boldsymbol{F}$, $S_0 \leftarrow \{M, \varepsilon, S_0, 0, 1\}$, $converge \leftarrow -2$,
   else $S_0 \leftarrow \{M, \varepsilon, S_0, 1, 0\}$, end if
2. if $M = M_{ref}$, $S_0 \leftarrow \{M, \varepsilon, S_0, 1, 1\}$, $converge \leftarrow 1$,
   else $S_0 \leftarrow \{M, \varepsilon, S_0, 1, 0\}$, end if
3. $\boldsymbol{Tree} \leftarrow S_0$, $\boldsymbol{\Sigma} \leftarrow S_0$, $T^* \leftarrow \varepsilon$, $exhaustive \leftarrow 1$
4. while $\exists\, S \in \boldsymbol{Tree}$ st $l(S) = 1$ and $e(S) = 0$,
5. for each $T \in \boldsymbol{T_C}$ st $m(S)\,[T >$
6. compute the successor S' of S by firing T,
   M' st $m(S)\,[t > M'$, $\sigma' \leftarrow \sigma(S)\,T$, $s' \leftarrow s(S)\,S'$
7. if $(SPEC(M')=0) \vee ((\boldsymbol{M'})^{\circ} \cup \boldsymbol{T_C} = \varnothing)$, $\boldsymbol{F} \leftarrow \boldsymbol{F} \cup \{m(S)\}$, end if
8. if $(M' \in \boldsymbol{F}) \vee (S' \in s(S))$, $l \leftarrow 0$, else $l \leftarrow 1$, end if
9. if $(l = 0) \vee (M' = M_{ref}) \vee (|\sigma'| = H) \vee (DURATION(\sigma',M) > H_\tau)$,

$e \leftarrow 1$, else $e \leftarrow 0$, end if
10. $\boldsymbol{Tree} \leftarrow \boldsymbol{Tree} \cup \{M', \sigma', s', l, e\}$
11. end for
12. end while
13. for h from H-1 to 0
14. for each $S \in \boldsymbol{Tree}$ st $|\sigma(S)|=h$
15. if ($l(S')=0$ for all direct successors S' of S in $\boldsymbol{Tree}$), $l(S) \leftarrow 0$, $e(S) \leftarrow 1$, end if
16. end for
17. end for
18. for each $S \in \boldsymbol{Tree}$ st $(l(S)=0) \wedge (e(S)=0)$
19. if $\exists\perp S' \in \boldsymbol{Tree}$ st $(S' \neq S) \wedge (m(S') = m(S)) \wedge (l(S') = 1)$, $\boldsymbol{F} \leftarrow \boldsymbol{F} \cup \{m(S)\}$, end if
20. end for
21. for each $S \in \boldsymbol{Tree}$ st $e(S)=1$, $\boldsymbol{\Sigma} \leftarrow \boldsymbol{\Sigma} \cup \{S\}$, end if
22. $\boldsymbol{\Sigma}^* \leftarrow \{S^*$ st $J_{FC}(m(S^*),M_{ref}) = \min(J_{FC}(m(S),M_{ref})$, for all $S \in \boldsymbol{\Sigma}\}$
23. $\boldsymbol{\Sigma}^{**} \leftarrow \{S^*$ st $DURATION(\sigma(S^*),M) = \min(DURATION(\sigma(S),M))$ for all $S \in \boldsymbol{\Sigma}^*\}$
24. if $\{S_0\} = \boldsymbol{\Sigma}^{**}$, $converge \leftarrow -1$, $T^* \leftarrow \varepsilon$, else select T^* as the first transition of $\sigma(S^*)$ with $S^* \in \boldsymbol{\Sigma}^{**}$, $converge \leftarrow 0$, end if
25. for each $S \in \boldsymbol{\Sigma}$
26. if $(l(S)=1) \wedge (e(S)=1) \wedge (DURATION(\sigma(S),M) < H_\tau)$, $exhaustive \leftarrow 0$, end if
27. end for

The complete control sequence σ^* is obtained with a second algorithm (Algorithm 2 in [21] [22]) that adapts the parameter H in range $[1 : \overline{H}]$ where $\overline{H}$ is an input parameter (Fig. 5). This algorithm starts at initial marking M_I, with no forbidden marking (ie $\boldsymbol{F} = \emptyset$) and with minimal depth (ie $H = 1$). As long as convergence is ensured, T^* is added to σ^* and the current marking M is updated. In simple words, Algorithm 3 iterates the use of Algorithm 2 in order to compute the complete sequence from the initial marking to the reference. When Algorithm 2 finds a deadlock or any other bottleneck, the forbidden marking is added in set $\boldsymbol{F}$ and Algorithm 3 backtracks to remove the last part of the already computed trajectory, enlarges the explored neighborhood by increasing the value of parameter H and starts the search again from the last valid marking. Note that the complexity of Algorithm 3 is at most $O(|\sigma^*|.q_C^{\overline{H}})$.

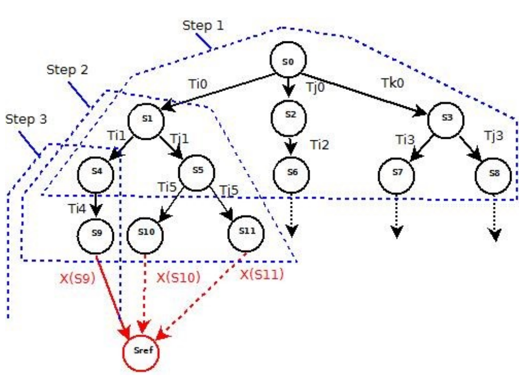

Figure 5: MPC global schema with Algorithm 3 [21] [22]

Algorithm 3 [21] [22]
1. $M \leftarrow M_I$, $converge \leftarrow 0$, $\sigma^* \leftarrow \varepsilon$, $H \leftarrow 1$, $\boldsymbol{F} \leftarrow \emptyset$, $success \leftarrow 1$
2. while $(converge < 1)$
3. compute $converge$, $exhaustive$ and $T^* \in \boldsymbol{T_C}$ and update $\boldsymbol{F}$ with Algorithm 2
4. if $(converge = 0)$^$((exhaustive=1) \vee ((exhaustive=0)$ ^$(H=\overline{H})))$,
5. compute $\sigma^* \leftarrow \sigma^* T^*$ and M st $M_I [\sigma^* > M$
6. $H \leftarrow \max(1, H\text{-}1)$
7. end if
8. if $((converge = -1) \wedge (H = \overline{H})) \vee (converge = -2)$,
9. if $(M \neq M_I)$,
10. remove last transition in σ^* and compute M st $M_I[\sigma^* > M$
11. else
12. if $(converge = -2)$, $success \leftarrow -2$, else $success \leftarrow -1$, end if
13. break
14 end if
15. end if
16. if $((H=\overline{H}) \wedge (converge=0) \wedge (exhaustive=0))$, $success \leftarrow 0$, end if
17. if $(H<\overline{H}) \wedge ((converge=-1) \vee ((converge=0) \wedge (exhaustive=0)))$, $H \leftarrow H+1$, end if
18. end while

3.3. Robust scheduling

In order to compute robust marking trajectories that cannot deviate from the reference, the controller should avoid dangerous intermediate markings and consider only legal trajectories with robust markings. The difficulty in this computation is that the intermediate markings are computed step by step and these markings are known in advance only within a small time windows provided by the partial reachability graph of depth H. During prediction phase of MPC, only the remaining firing count vector to the reference is determined and this vector does not provide the risk belief or risk probability of the future trajectory. Proposition 3 provides a sufficient condition to ensure that the computed trajectory visits only robust markings. For this purpose, let us define $\boldsymbol{T_{RC}} = \{T_j \in \boldsymbol{T_C}$ st $(\boldsymbol{T_j}^\circ)^\circ \subseteq \boldsymbol{T_C}\}$ where $(\boldsymbol{T_j}^\circ)^\circ = \cup\{\boldsymbol{P_i}^\circ : P_i \in \boldsymbol{T_j}^\circ\}$.

Proposition 3: Let consider a PCont-TPN (or PCont-SPN). Let (σ, M_I) be a marking trajectory such that $(\boldsymbol{M_I})^\circ \subseteq \boldsymbol{T_C}$. If $\sigma \in \boldsymbol{T_{RC}}^*$ then (σ, M_I) is a robust legal trajectory.

Proof: Note at first that $(\boldsymbol{M_I})^\circ \subseteq \boldsymbol{T_C}$ implies that the net has no uncontrollable source transition (i.e. ${}^\circ\boldsymbol{T_j} \neq \emptyset$ for all $T_j \in \boldsymbol{T_{NC}}$). Then, (σ, M_I) is written as in (5) : $\sigma = M_I\ [T(j_1, t_1) > M(1) \ldots. > M(h)$ Assume that there exists $T_j \in (\boldsymbol{M(1)})^\circ$ such that $T_j \in \boldsymbol{T_{NC}}$. T_j is necessarily enabled by the firing of $T(j_1, t_1)$ because T_j is not enabled at M_I. As T_j is not a source transition, there exists a place $P_i \in {}^\circ\boldsymbol{T_j}$ whose marking increases by firing $T(j_1, t_1)$ and consequently $P_i \in (\boldsymbol{T(j_1, t_1)})^\circ$. As $T_j \in \boldsymbol{P_i}^\circ$, $T_j \in ((\boldsymbol{T(j_1, t_1)})^\circ)^\circ$. Thus,

$T_j \in \boldsymbol{T_C}$ that is contradictory with assumption and $(\boldsymbol{M(1)})° \subseteq \boldsymbol{T_C}$. Repeating successively the same reasoning up to $M(h)$, one can conclude that $(\boldsymbol{M(k)})° \subseteq \boldsymbol{T_C}$, $k = 1,\ldots,h$, and that (σ,M_I) is a robust legal trajectory.

Note that the set $\boldsymbol{T_{RC}}$ is easy to obtain by checking for each transition T_j if the condition $X_j.(W_{PO})^T.W_{PR}.(0 \mid I_{qNC})^T = 0$ is satisfied or not, with X_j the firing count vector of T_j and I_{qNC} the identity matrix of size q_{NC}.

Note also that robust legal trajectories are computed with the same algorithms by considering only robust controllable transitions $T \in \boldsymbol{T_{RC}}$ in line 5 of Algorithm 2 and by replacing $J_{FC}(M,M_{ref})$ by $J_{RFC}(M,M_{ref})$:

$$J_{RFC}(M,M_{ref}) = (D_{Rmin})^T.X_{RC} \tag{12}$$

where D_{Rmin} and X_{RC} are the restrictions of D_{min} and X_C to the set of robust transitions in $\boldsymbol{T_{RC}}$.

3.4. Example

Consider again PN1 with $\boldsymbol{T_C} = \{T_1, T_2, T_3, T_4, T_5, T_6\}$, $\boldsymbol{T_{NC}} = \{T_7\}$, $D_{min} = (1, 1, 1, 1, 1, 1)^T$, $M_I = (1\ 0\ 0\ 0)^T$ and $M_{ref} = (5\ 0\ 0\ 0)^T$ (Fig. 3). If the uncontrollable transition T_7 does not fire, Algorithm 3 returns the sequence σ_1 (see section 2.5). If T_7 fires, Algorithm 3 updates the rest of the sequence. $\sigma_3 = T(3, 1)T(7, 1.2)(T(2, 2.2))^5(T(4, 2.2))^4(T(5, 3.2))^4(T(6, 4.2))^4$ is an example of control sequence resulting from the single firing of T_7 at date 1.2. In order to avoid any deviation, $\boldsymbol{T_{RC}} = \{T_1, T_2, T_4, T_5, T_6\}$ instead of $\boldsymbol{T_C}$ can be considered (T_3 does not belong to $\boldsymbol{T_{RC}}$ because $T_7 \in (\boldsymbol{T_3}°)°$ and $T_7 \in \boldsymbol{T_{NC}}$). The same method applied by considering only robust controllable transitions leads to $\sigma_2 = T(1, 1)(T(2, 2))^2(T(1, 3))^2(T(2, 4))^4T(1, 5)(T(2, 6))^2$ that cannot be perturbed by any unexpected firing.

4 CASE STUDY

Pcont-SPN2 in Fig. 6 is the model of a production system that processes a single type of products according to two possible jobs [27]. The first job is composed of the transitions t_1 to t_8, and the second one by the transitions t_9 to t_{14}. Job 1 could be altered by a server failure whereas Job 2 could not. The occurrence of this failure is represented by the firing of the subsequence T_2T_5 instead of T_3T_4. Note that the faults under consideration are not blocking the system, but they delay the cycle time. Consequently the nominal sequence $T_1\ T_3\ T_4\ T_6\ T_7\ T_8$ may be altered when an unexpected firing of T_2 occurs that leads to the perturbed behavior $T_1\ T_2\ T_5\ T_6\ T_7\ T_8$ with an excessive global duration. The six resources p_{14} to p_{19} have limited capacities: $m(p_{14}) = m(p_{15}) = m(p_{16}) = m(p_{17}) = m(p_{18}) = m(p_{19}) = 1$. The places p_{20} and p_{21} represent the input and output buffers, respectively. p_{20} contains the number of products to be processed either by Job 1 or Job 2. The temporal specifications are given by $D_{min} = (1\ 1\ 2\ 20\ 1\ 1\ 1\ 3\ 3\ 3\ 3\ 3\ 3)^T$ for $\boldsymbol{T_C} = \boldsymbol{T}/\{T_2\}$ and by $\mu_2 = 1$. Control sequences are computed with $M_I = 3P_1 + 3P_8 + 1P_{14} + 1P_{15} + 1P_{16} + 1P_{17} + 1P_{18} + 1P_{19} + kP_{20}$ and $M_{ref} = 3P_1 + 3P_8 + 1P_{14} + 1P_{15} + 1P_{16} + 1P_{17} + 1P_{18} + 1P_{19} + kP_{21}$ where k is a varying parameter. The results are reported in Table 2 for $\overline{H} = 5$ and $H_\tau = 20$.

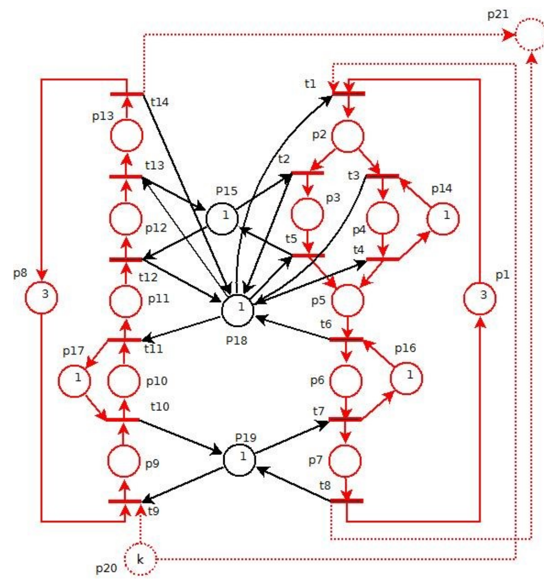

Figure 6. Pcont-SPN2 [27].

Table 2. Performance of Algorithm 3 with Pcont-SPN2: average sequence duration (TUs).

k	Scenario 1	Scenario 2	Scenario 3
5	45	103.4	72
10	142	213.5	147
15	236	321.9	222
20	325	427.1	297

Three scenarios are considered: in scenario 1 all transitions, including T_2, are assumed to be controllable with $d_{min\ 2} = 1$. In scenario 2, $\boldsymbol{T_C} = \boldsymbol{T}/\{T_2\}$ and Algorithm 3 is applied with $\boldsymbol{T_C}$. In scenario 3 Algorithm 3 is applied with $\boldsymbol{T_{RC}} = \boldsymbol{T}/\{T_1, T_2\}$. Note, at first, that due to the numerical values of the firing parameters, the scenarios 1 and 2 prefer Job 1 that has a global duration of 7 TUs to process one product compared to Job 2, which has a global duration of 18 TUs (without considering the constraints due to the limited resources). For scenario 2, if an unexpected firing of T_2 occurs the long firing duration $d_{min\ 5} = 20$ of T_5 alters the global duration required to process the product. This explains that scenario 2 leads to longer sequences compared to scenario 1. Scenario 3 is also tested in a stochastic context with the same value of parameters. The restriction of the control actions in set $\boldsymbol{T_{RC}}$ prefers systematically Job 2 that is robust to the perturbations. As mentioned in the previous section, the solutions returned by Algorithm 3 are not optimal solutions in a systematic way. The performance of the algorithm depends on the two input parameters: $\overline{H}$, that limits the exploration in depth, and H_τ, that limits the search in duration. If the depth H is too small, Algorithm 2 returns the flag *converge* = −1 or *exhaustive* = 0 and Algorithm 3 increases H in the range $[1:\overline{H}]$. On the contrary, if H is too large, then the iterative use of Algorithm 2 certainly reaches M_{ref} but the computational effort is uselessly high. In that case, Algorithm 3 decreases H in the range $[1:\overline{H}]$. Consequently, the aim of Algorithm 3 is to adapt at each step the depth of the search to maintain *converge* = 0 and *exhaustive* = 1 or *converge* = 1. Table 3 reports the performance in function of the parameters $\overline{H}$ and H_τ for Pcont-SPN2 for $k = 5$ and $\boldsymbol{T_C} = \boldsymbol{T}$. The duration of the control sequences and the computational time required to compute the sequences with Algorithm 3 are reported for an Intel Core i7-46000 CPU at 2.1–2.7 GHz.

Table 3. Performance of Algorithm 3 for PCont-SPN2: sequence duration (TUs) and computational time (s).

$\overline{H}/H_\tau$	1	3	5
5	82 (0.9 s)	68 (1 s)	68 (1.3 s)
10	82 (0.9 s)	76 (1.5 s)	76 (4.7 s)
15	82 (1 s)	63 (2.1 s)	63 (9.4 s)
20	82 (1 s)	45 (2.6 s)	45 (10.7 s)

5 CONCLUSIONS

A method that incrementally computes control sequences that approach the minimal duration for timed PNs has been proposed in the context of uncertain environments. The method uses PNs as a systematic formalism easy to adapt to various problems. It limits the part of the reachability graph that is expanded even if the initial marking and reference marking are far one from each other and if deadlocks and dead branches are a priori unknown for the controller. It evaluates the risk of deviation from the expected planning and it results in a robust scheduling under some additional assumptions.

In our next works, the research effort will concern at first the refinement of the cost function to obtain a better approximation of the remaining time to reference. The sensitivity of the performance with respect to H will be also studied. We will also include the risk evaluation investigate in the cost function to obtain trajectories of low risk level.

ACKNOWLEDGMENTS

The Project MRT MADNESS 2016-2019 has been funded with the support from the European Union with the European Regional Development Fund (ERDF) and from the Regional Council of Normandie.

REFERENCES

[1] M. R. Garey, D. S. Johnson, and R. Sethi, The complexity of flowshop and jobshop scheduling. *Mathematics of operations research*, 1(2), 117-129, 1976

[2] S. M. Johnson, Optimal two-and three-stage production schedules with setup times included. *Naval research logistics quarterly*, 1(1), 61-68, 1954.

[3] K.R. Baker, D. Trietsch, *Principles of Sequencing and Scheduling*, John Wiley & Sons, 2009.

[4] P. Lopez, F. Roubellat, *Production Scheduling* ISTE, 2008.

[5] J. Y-T. Leung, *Handbook of Scheduling: Algorithms, Models, and Performance Analysis*, Chapman & Hall/CRC Computer & Information Science Series, 2004.

[6] 1 C. Cassandras, *Discrete Event Systems: Modeling and Performances Analysis*, Aksen Ass. Inc. Pub., 1993.

[7] R. David and H. Alla, *Petri nets and grafcet – tools for modelling discrete events systems*, London: Prentice Hall, 1992.

[8] P. Chretienne, Timed Petri nets: A solution to the minimum-time-reachability problem between two states of a timed-event-graph, *Journ. Systems and Software,* vol. 1, no. 2, pp.95–101, 1986.

[9] D.Y. Lee, F. DiCesare, Scheduling flexible manufacturing systems using Petri nets and heuristic search, *IEEE Trans. Robot. Autom*. vol. 10, no. 2, pp.123–133, 1994.

[10] T.H. Sun, C.W. Cheng, L.C. Fu, Petri net based approach to modeling and scheduling for an FMS and a case study. *IEEE Trans. Ind. Electron*., vol. 41, no. 6, pp. 593–601, 1994.

[11] A. Reyes-Moro, H. Hu G. Kelleher, Hybrid Heuristic Search for the Scheduling of Flexible Manufacturing Systems Using Petri Nets, *IEEE Trans. Robotic and Autom*., vol. 18, no. 2, pp. 240-245, 2002.

[12] H.H. Xiong, M.C. Zhou, Scheduling of semiconductor test facility via Petri nets and hybrid heuristic search. *IEEE Trans. Semicond. Manuf. vol.* 11, no. 3, pp. 384–393, 1998.

[13] M.D. Jeng, S.C. Chen, Heuristic search approach using approximate solutions to Petri net state equations for scheduling flexible manufacturing systems. *Int J FMS*, vol. 10, no. 2, pp. 139–162, 1998.

[14] Q. Wang, Z. Wang, Hybrid Heuristic Search Based on Petri Net for FMS Scheduling, *Energy Procedia*, vol. 17 pp. 506 – 512, 2012.

[15] W. Zhang, T. Freiheit, H. Yang, Dynamic scheduling in flexible assembly system based on timed Petri nets model, Robotics and Computer-*Integrated Manufacturing*, vol. 21, pp. 550–558, 2005.

[16] H. Hu and Z. Li, Local and global deadlock prevention policies for resource allocation systems using partially generated reachability graphs, *Computers & Industrial Engineering*, 57: 1168–1181, 2009.

[17] B. Abdallah, H.A. ElMaraghy, T. ElMekkawy, Deadlock-free scheduling in flexible manufacturing systems. *Int J. Prod. Res. vol.* 40, no. 12, pp. 2733–2756, 2002.

[18] H. Lei, K. Xing, L. Han, F. Xiong, Z. Ge, Deadlock-free scheduling for flexible manufacturing systems using Petri nets and heuristic search, *Computers & Industrial Engineering*, vol. 72, pp. 297–305, 2014.

[19] D. Lefebvre and E. Leclercq, Control design for trajectory tracking with untimed Petri nets, *IEEE Trans. Aut. Contr.*, vol. 60(7), pp. 1921-1926, July 2015.

[20] D. Lefebvre , Approaching minimal time control sequences for timed Petri nets, *IEEE Trans. Automation Science and Engineering*, vol. 13, no. 2, pp. 1215-1221, 2016.

[21] D. Lefebvre, Deadlock-free scheduling for Timed Petri Net models combined with MPC and backtracking, Proc. IEEE WODES 2016, Invited session "Control, Observation, Estimation and Diagnosis with Timed PNs", pp. 466-471, Xi'an, China, 2016.

[22] D. Lefebvre, Deadlock-free scheduling for flexible manufacturing systems using untimed Petri nets and model predictive control, Proc. IFAC – MIM, Invited session "DES for manufacturing systems", Troyes, France, June 2016.

[23] C. Ramchandani, *Analysis of asynchronous concurrent systems by timed Petri nets*, Ph. D, MIT, USA, 1973.

[24] M. K. Molloy Performance analysis using stochastic Petri nets, *IEEE Tran. Comp. C*, 31, pp. 913 – 917, 1982.

[25] J. Richalet, A. Rault, J. Testud and J. Papon, Model predictive heuristic control: Applications to industrial processes, *Automatica*, 14: 413-428, 1978.

[26] E. Camacho and A. Bordons, *Model predictive control, London: Springer Verlag*, 2007.

[27] Y. Chen, Z. Li, M. Khalgui, O. Mosbahi, Design of a Maximally Permissive Liveness-Enforcing Petri Net Supervisor for Flexible Manufacturing Systems, *IEEE Trans. Aut. Science and Eng*, 8, 374–393, 2011.

[28] D. Lefebvre, Dynamical Scheduling and Robust Control in Uncertain Environments with Petri Nets for DESs, *MDPI Processes*, 5(54), doi:10.3390/pr5040054