O P E R A T I O N S  R E S E A R C H  A N D  D E C I S I O N S
No. 4 2017
DOI: 10.5277/ord170402

Cherki DAOUI[1]
Dimitri LEFEBVRE[2]

# CONTROL DESIGN FOR UNTIMED PETRI NETS USING MARKOV DECISION PROCESSES

Design of control sequences for discrete event systems (DESs) has been presented modelled by untimed Petri nets (PNs). PNs are well-known mathematical and graphical models that are widely used to describe distributed DESs, including choices, synchronizations and parallelisms. The domains of application include, but are not restricted to, manufacturing systems, computer science and transportation networks. We are motivated by the observation that such systems need to plan their production or services. The paper is more particularly concerned with control issues in uncertain environments when unexpected events occur or when control errors disturb the behaviour of the system. To deal with such uncertainties, a new approach based on discrete time Markov decision processes (MDPs) has been proposed that associates the modelling power of PNs with the planning power of MDPs. Finally, the simulation results illustrate the benefit of our method from the computational point of view.

**Keywords:** *discrete event systems, Petri nets, control design, Markov decision process, value iteration algorithm*

## 1. Introduction

Performance in most manufacturing settings is affected by operative decisions related to job scheduling, such as: a) selection of jobs in a queue, b) priority when choosing a machine for processing a job (among parallel machines), and c) assignment of resources in the execution of a production plan. Such decisions have a significant impact on the efficiency of a system, operational costs and fulfilling orders, and are frequently

[1]Sultan Moulay Slimane University, TIAD, Avenue Mohamed V, Quartier Taqaddoum 591-23000 Béni Mellal, Morocco, e-mail address: daouic@yahoo.com
[2]Normandie Université, UNIHAVRE, GREAH, 76600 Le Havre, 25 rue Philippe Lebon, France, e-mail address: dimitri.lefebvre@univ-lehavre.fr

taken intuitively by operators, based on their experience. In order to support this decision process and guarantee the efficiency of a system, computational applications need to be developed. Such applications should be able to (i) provide modelling aids, (ii) apply scheduling techniques and (iii) define a production plan.

The design of control procedures for the scheduling and planning of discrete event systems is a major challenge in numerous domains such as flexible manufacturing, communication, computer science, transportation, robotics, biotechnologies, and business [24]. One of the major challenges faced by controllers is to provide more efficient decisions even when the environment of a system is uncertain and when its behaviour may be disturbed by unexpected changes, both internal and external. In this paper, we consider the particular problem of controlling discrete time event systems in uncertain environments that include control errors, as well as unexpected and uncontrollable events. Petri nets (PNs) are used to model the system and its uncertain environment and a MDP-based control approach is proposed to generate the control actions.

On the one hand, the potential of PNs for analysing and modelling complex systems has encouraged its use in the design of control systems for DESs, particularly for scheduling problems applied in the field of manufacturing systems 3, 15, 22, 23, 27. The motivation to use Petri nets as a modelling tool is that such models are physics--based and provide a realistic and comprehensive representation of the systems considered. They are modular and flexible in the sense that they can easily be updated when the system specifications change. In addition, the proposed extension of PNs takes into account not only the system operations but also the faults, as well as processes of aging and degradation that progressively lead to system failure. Another important advantage of Petri nets is that specific properties such as conflicts, deadlocks, limited buffer sizes, and constraints on resources can be easily represented within a single formal model [26]. This model is also suitable for control issues. In earlier works, the authors propose a model for a predictive approach to control issues in DESs [18–21].

On the other hand, a Markov decision process (MDP) is a well-known formalism providing a simple mathematical model to express optimization problems in random environments. In particular, a discrete time finite MDP is an extension of a Markov chain which allows non-deterministic choices/actions, and a reward/cost function expressing an objective function to be minimized/maximized. For each non-deterministic action allowed in a given state, a reward/cost and a transition probability distribution are defined. Hence, the evolution of an MDP can be described as a sequence of non-deterministic transitions. MDP theory has been proved to provide systematic low-cost decisions for stochastic processes satisfying the Markov property. They are used in a wide range of disciplines: production planning, automated control, artificial intelligence and economics [9].

In most real distributed systems, it is possible to perform non-deterministic choice among the set of possible actions, e.g., the scheduling of tasks, when the effect of the selected action is probabilistic, e.g., the random duration of a task. Modelling these

systems directly at the level of an MDP may be a hard task. To cope with this problem, a number of higher-level formalisms have been proposed in the literature (e.g., [2, 4–6, 12, 14, 28]. Almost all of these approaches have been defined as extensions of PNs for highly specified MDPs but they do not consider problems where control is required to be robust to faults [13, 29], i.e., for the derivation of sequences that steer the system from its current state to a reference state in the case of a fault. More particularly, they do not consider control issues in uncertain environments where control errors or unexpected events disturb system behaviour.

MDPs with discrete-time will be considered in this paper: each decision is taken at a given moment. After the probabilistic consequence of a decision has been realised, a new decision moment starts. Therefore, the goal is to associate the modelling power of PNs with the planning power of MDPs that is useful for systems and environments that obey the Markov assumption. This paper is organized as follows. Section 2 is about the modelling of DESs in uncertain environments using PNs. Section 3 presents the main results. Section 4 gives an illustrative example. Section 5 sums up the conclusions and perspectives.

## 2. PNs with uncertain control actions

In this section, PNs are used to formalize the problem of controlling DESs in uncertain environments.

### 2.1. Petri nets

A PN structure is defined as $G = <\mathbf{P}, \mathbf{T}, W_{PR}, W_{PO}>$, where $\mathbf{P} = \{p_1, ..., p_n\}$ is a set of $n$ places and $\mathbf{T} = \{t_1, ..., t_q\}$ is a set of $q$ transitions with indexes $\{1, ..., q\}$, $W_{PO} \in (\mathbf{N})^{n \times q}$ 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 $W = W_{PO} - W_{PR}$ 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 (which can be interpreted as the vector of the number of "tokens" at each place). Each marking $M$ represents a state of the DES. A transition $t_j$ is enabled given the marking $M$ if $\min\{\lfloor m_k / w_{kj}^{PR} \rfloor : p_k \in {}^\circ t_j\} > 0$ (if $w_{kj}^{PR} = 0$, then it is assumed that $\lfloor m_k / w_{kj}^{PR} \rfloor > 0$), where ${}^\circ t_j$ stands for the set of upstream places for the transition $t_j$, $m_k$ is the marking of place $p_k$, $w_{kj}^{PR}$ is the entry of matrix $W_{PR}$ in row $k$ and column $j$. 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 a column $j$ of the incidence matrix. This is denoted by $M[t_j > M'$ or equivalently by $M' = M + W \cdot X_j$, where $X_j$ denotes the firing count vector of transition $t_j$ [11]. Hence, for a transition to occur, the numbers of tokens at the upstream sites must be large enough (according to the

matrix *WPR*) to enable firing and the numbers of tokens delivered to the downfield sites are given by the matrix $W_{PO}$. Each firing of a transition represents an event (controlled or unexpected) that changes the state of the DES. A firing sequence $\sigma$ with initial marking $M_I$ is defined as $\sigma = t(j_1)t(j_2) \ldots t(j_h)$, where $j_1, \ldots, j_h$ are the indexes of the transitions. $X(\sigma) \in (\mathbf{N})^q$ is the firing count vector associated with $\sigma$, $|\sigma| = h$ is the length of $\sigma$, and $\sigma = \varepsilon$ denotes the empty sequence. The firing sequence $\sigma$ starting from $M$ leads to the marking trajectory $(\sigma, M)$, where:

$$(\sigma, M) = M\,[t(j_1) > M(1) \ldots M(h-1)[t(j_h) > M(h)) \tag{1}$$

where $M(1), \ldots, M(h-1)$ are the intermediate markings and $M(h)$ is the final marking (hereafter, we write $M(k) \in (\sigma, M)$, $k = 1, \ldots, h$). A marking $M$ is said to be reachable from initial marking $M_I$ if there exists a firing sequence $\sigma$ such that (s.t.) $M_I\,[\sigma > M$ and $\sigma$ is said to be feasible at $M_I$. Reach($G$, $M_I$) is the set of all reachable markings from $M_I$ (i.e., a set of discrete states). Assumption **A1** is considered in the sequel:

**A1.** The set of reachable markings, Reach($G$, $M_I$), of the considered PN is of finite cardinality.

As a consequence, the PN system $<G, M_I>$ is bounded. Additionally, the function *TR* is defined by:

$$\begin{aligned} &\text{For all}\left(MM'\right) \in \text{Reach}\left(G, M_I\right) \times Reach\left(G, M_I\right) : TR\left(M, M\right) = e \\ &TR\left(M, M'\right) = t_j \ \text{ if } M[t_j > M' \\ &\text{otherwise } TR\left(M, M'\right) = \varnothing \end{aligned} \tag{2}$$

### 2.2. Controlled PNs in uncertain environments

For control issues in uncertain environments, PNs with specific interpretations are considered. For this purpose, a set $\mathbf{C} = \{c_1, \ldots, c_{kc}\}$ of $k_c$ distinct control actions is defined. In addition, $\theta$ stands for the specific control action "there is nothing to do" (the null action). Control actions are controlled events. Uncontrollable events also exist and **D** stands for the set of uncontrolled events: uncontrolled events are either the occurrence of faults that lead to a system failure or events that result from the interaction between the environment and the system. Such uncontrolled events are not detailed. The set of transitions **T** is also divided into 2 disjoint subsets $\mathbf{T}_C$, and $\mathbf{T}_{NC}$ such that $\mathbf{T} = \mathbf{T}_C \cup \mathbf{T}_{NC}$. $\mathbf{T}_C$ is the subset of $q_c$ controllable transitions, and $\mathbf{T}_{NC}$ the subset of $q_{nc}$ uncontrollable transitions. The firing of enabled controllable transitions are enforced or avoided by the controller according to the control actions, whereas the firing of uncontrollable transitions cannot be enforced or avoided and fire spontaneously according to the occurrence of

uncontrolled events. A sequence is said to be controllable if it contains only controllable transitions. Note that two or more control actions may enforce the same transition, but in order to obtain deterministic control strategies, Assumption A2 is adopted:

**A2.** Two transitions $t_j$ and $t_k$ with $j \neq k$ are not associated with the same controlled event. *M*ore formally, the function $A_C$ maps controllable events to controlled transitions: for $c \in \mathbf{C}$, $A_C(c) \in \mathbf{T}_C$ is the controlled transition whose firing is enforced by $c$ when the transition $A_C(c)$ is enabled. In addition $A_C(\theta) = \varepsilon$. Reversely, for $t_j \in \mathbf{T}_C$, $\mathbf{A}_C^{-1}(t_j) \subseteq \mathbf{C}$ is the set of controlled events that enforce the firing of the transition $t_j$ when $t_j$ is enabled and $\mathbf{A}_C^{-1}(\varepsilon) = \{\theta\}$. In addition, for any couple $(M, c) \in \text{Reach}(G, M_I) \times (\mathbf{C} \cup \{\theta\})$, $\Gamma_c^{+}(M)$ (resp. $\Gamma_c^{-}(M)$) refers to the successor of marking $M$ (resp. the predecessor of $M$) when $c$ is executed. $\mathbf{\Gamma}_C^{+}(M)$ and $\mathbf{\Gamma}_D^{+}(M)$ refer respectively to the sets of successors of the marking $M$ for all controlled events and all uncontrolled ones. $\mathbf{\Gamma}_C^{-}(M)$ and $\mathbf{\Gamma}_D^{-}(M)$ refer to the sets of predecessors of $M$ obtainable by applying any controlled event and any uncontrolled event, respectively.

Hereafter, faults will be considered that correspond either to the wrong execution of some controlled event or to the unexpected occurrence of some uncontrolled event. Such faults lead to an uncertain environment that the controller has to deal with. Such faults will occur according to the set PROB of probabilities (3):

$$\text{PROB} = \{\text{prob}(M, M', c)$$

$$\text{for all } (M, M', c) \in \text{Reach}(G, M_I) \times \text{Reach}(G, M_I) \times (\text{C} \cup \{\theta\})\} \tag{3}$$

prob($M$, $M'$, $c$) is the probability of $M'$ being the successor of $M$ when the control action $c$ is undertaken by the controller. The previous definitions and assumptions are suitable to represent a large variety of uncertain environments. Four particular cases will be studied below.

**Case 1.** There is no uncontrollable transition (i.e., $\mathbf{T}_C = \mathbf{T}$) and no fault. The set PROB is defined by:

$$\text{prob}(M, M', c) = 1 \text{ if } A_C(c) = TR(M, M')$$

$$\text{prob}(M, M', c) = 0, \text{ otherwise.} \tag{4}$$

In this case, the controlled system behaves deterministically and there is no uncertainty at all.

**Case 2.** There is no uncontrollable transition (i.e., $\mathbf{T}_C = \mathbf{T}$), but faults are considered that correspond to the incorrect execution of a control action: a control action $c \in \mathbf{C} \cup \{\theta\}$ is executed by the controller when the marking is $M$ and no transition fires or an unexpected enabled controllable transition $t_j \neq A_C(c)$ fires. For any couple $(c, t) \in (\mathbf{C} \cup \{\theta\}) \times (\mathbf{T}_C \cup \{\varepsilon\})$, prob$(c, t)$ is the probability that $t$ fires when $c$ is executed. The probabilities prob$(c, t)$, $t \in \mathbf{T}_C \cup \{\varepsilon\}$ obviously satisfy:

$$\sum_{t \in T_c \cup \{\varepsilon\}} \text{prob}(c, t) = 1$$

For simplicity, prob$(c, t)$ does not depend on the current marking $M$. Consequently, the set PROB is defined by:

$$\begin{aligned} &\text{prob}(M, M', c) = \text{prob}(c, TR(M, M'))/S_2(M, c) \\ &\text{if } TR(M, M') \in \mathbf{T}_C \cup \{\varepsilon\} \\ &\text{prob}(M, M', c) = 0, \text{ otherwise} \end{aligned} \tag{5}$$

where $S_2(M, c)$ is the normalization coefficient for case 2, i.e.:

$$S_2(M, c) = \sum_{t \in \mathbf{T}_c \cup \{\varepsilon\},\, M[t>} (\text{prob}(c, t))$$

**Case 3.** Uncontrollable transitions exist (i.e., $\mathbf{T}_C \subset \mathbf{T}$) and faults are considered that correspond to the unexpected occurrence of some uncontrolled events. A control action $c \in \mathbf{C} \cup \{\theta\}$ is undertaken by the controller when the marking is $M$. This action should enforce the firing of $A_C(c) \in \mathbf{T}_C \cup \{\varepsilon\}$, but an uncontrollable transition $t \in \mathbf{T}_{NC}$ fires instead. For any couple $(c, t) \in (\mathbf{C} \cup \{\theta\}) \times (\mathbf{T}_{NC} \cup \{A_C(c)\})$, prob$(c, t)$ is the probability that $t$ fires when $c$ is decided. Prob$(c, t)$, $t \in \mathbf{T}_{NC} \cup \{A_C(c)\}$ obviously satisfy:

$$\text{prob}(c, A_C(c)) + \sum_{t \in \mathbf{T}_{NC}} \text{prob}(c, t) = 1$$

In this case, the set PROB is defined by:

$$\begin{aligned} &\text{prob}(M, M', c) = \text{prob}(c, TR(M, M'))/S_3(M, c) \\ &\text{if } TR(M, M') \in \mathbf{T}_{NC} \cup \{A_C(c)\} \\ &\text{prob}(M, M', c) = 0, \text{ otherwise} \end{aligned} \tag{6}$$

where $S_3(M, c)$ is the normalization coefficient for case 3, i.e.:

$$S_3(M, c) = (\text{prob}(c, A_C(c)) + \sum_{t \in \mathbf{T}_{NC},\, M[t>} (\text{prob}(c, t))$$

**Case 4.** Faults are considered that correspond either to the wrong execution of a control action or to the unexpected occurrence of some uncontrollable event. Case 4 combines cases 2 and 3, and the set PROB is defined by:

$$\begin{aligned} &\text{prob}(M, M', c) = \text{prob}(c, TR(M, M'))/S_4(M, c) \\ &\text{if } TR(M, M') \in \mathbf{T}_C \cup \mathbf{T}_{NC} \cup \{\varepsilon\} \\ &\text{prob}(M, M', c) = 0, \text{ otherwise} \end{aligned} \tag{7}$$

where $S_4(M, c)$ is the normalization coefficient for case 4, i.e.:

$$S_4(M, c) = \sum_{t \in \mathbf{T}_C \cup \mathbf{T}_{NC} \cup \{\varepsilon\},\, M[t>} (\text{prob(c, t)})$$

## 2.3. Example

Let us consider PN1 as described in Fig. 1 as an example of a controlled DES. The initial marking is $M_I = (2\ 0\ 0\ 0\ 0)^T = 2p_1$ and the objective of the controller is to compute a trajectory of minimal length to reach the reference marking $M_{\text{ref}} = (0\ 0\ 0\ 0\ 2)^T = 2p_5$. The transitions $\{t_1, t_2, t_3, t_4, t_5, t_6\}$ are mapped to controllable events according to the function $A_C$.

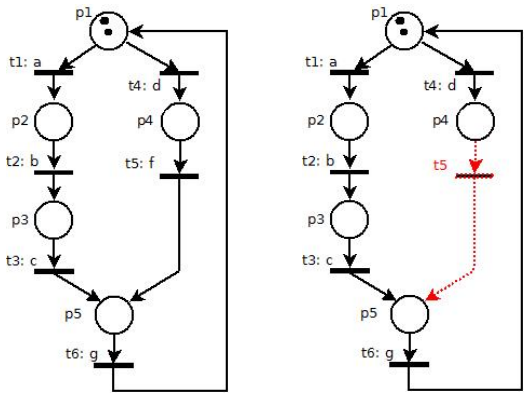


Fig. 1. Example of controlled PNs: PN1 (left), PN2 (right)

Different environments are successively considered according to the four different cases previously described. In cases 1 and 2, the set $\mathbf{C} = \{a, b, c, d, f, g\}$ of control

actions is mapped to controllable transitions according to the function $A_C$: $A_C(a) = t_1$, $A_C(b) = t_2$, $A_C(c) = t_3$, $A_C(d) = t_4$, $A_C(f) = t_5$, $A_C(g) = t_6$ (Fig. 1, left). Whereas in cases 3 and 4, the set $C' = \{a, b, c, d, g\}$ and the mapping $A'_C$ are considered, where: $A'_C(a) = t_1$, $A'_C(b) = t_2$, $A'_C(c) = t_3$, $A'_C(d) = t_4$, $A'_C(g) = t_6$ (Fig. 1, right).

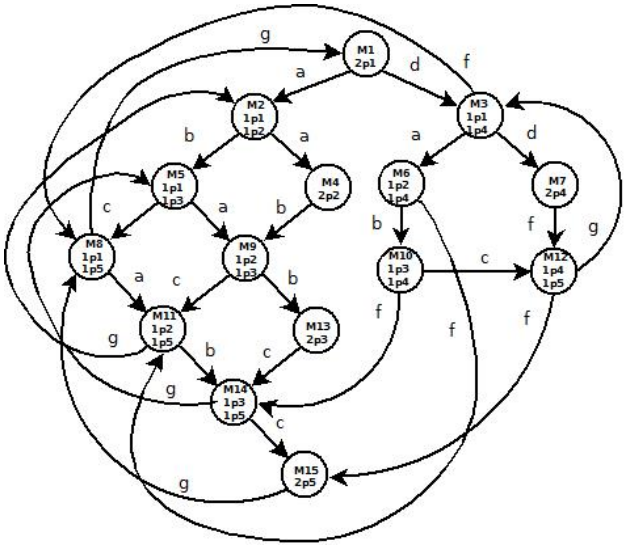


Fig. 2. Reachability graph for PN1 in cases 1 and 2

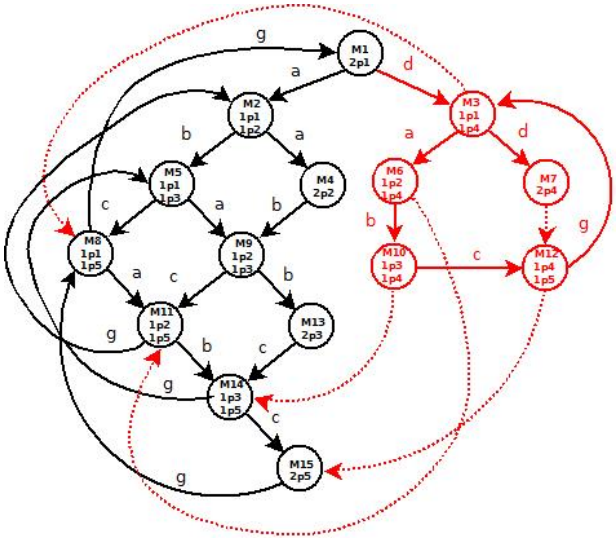


Fig. 3. Reachability graph for PN2 in cases 3 and 4

In cases 2 and 4, control errors are assumed to occur with probability prob($c$, $t$) = 0.1 for $c \in \mathbf{C} \cup \{\theta\}$ and $t \in \mathbf{T}_C \cup \{\varepsilon\}$. In cases 3 and 4, an uncontrollable transition $t_5$ is considered which occurs with probability prob($c$, $t_5$) = 0.2 for $c \in \mathbf{C} \cup \{\theta\}$. The reachability graphs of the PNs are given in Fig. 2 for cases 1 and 2, and in Fig. 3 for cases 3 and 4. The dashed lines represent the occurrence of an unexpected event.

## 3. Control design with MDPs

In this section, the theory of Markov decision processes [25] is adapted to the PN--based control design of DESs in uncertain environments.

### 3.1. Markov decision processes

Basically, in MDP models an agent interacts with his/her environment, taking the state of the environment as the input and generating actions as outputs. MDPs are defined using controllable stochastic processes which satisfy the Markov property and assigning rewards to each state transition. We consider a stochastic dynamic system which is observed at discrete time points ($k$ = 0, 1, ...). At moment $k$, the system is in state $S_k \in S$ ($S$ denotes the state space of the system), an action $a \in A$ ($A$ denotes the set of actions) is chosen by optimizing the expected reward $R(S_k, a)$. This action is executed and the state of the system changes according to the transition probabilities prob($S_k$, $S_{k+1}$, $a$). MDPs are special cases of decision processes where the transition probability, reward function and optimal action depend only on the current state and are otherwise independent of the history of the process.

A strategy $\pi$ is defined as a sequence $\pi = (\pi^1, \pi^2...)$ where $\pi^k : H_k \to \Psi$ is a decision rule, $H_k = (S \times A)^{k-1} \times S$ is the set of all histories up to time $k$, and

$$\Psi = \left\{ (q_1, ..., q_{|A|}) \in \Re^{|A|} : \sum_{i=1}^{|A|} q_i = 1,\ q_i \geq 0,\ 1 \leq i \leq |A| \right\}$$

is the set of probability distributions over $A = \bigcup_{i \in S} A(i)$.

A Markov strategy is a strategy $\pi$ in which $\pi^k$ depends only on the current state at time $k$, a stationary strategy is a Markov strategy with identical decision rules at each time $k$, and a deterministic or pure strategy is a stationary strategy whose single decision rule is non-stochastic.

The core problem of MDPs is to find an optimal policy that specifies which action should be taken in each state. Optimality is decided according to an optimality criterion

that evaluates the efficiency of the different candidate policies. The criterion is to maximize the expected reward. There are several standard methods for finding optimal policies for MDPs. Widely employed approaches include linear programming (LP), the value iteration (VI) algorithm and the policy iteration (PI) algorithm. Details about MDPs and optimization algorithms can be found in [1, 10, 25].

### 3.2. Markov decision process for untimed PNs

We consider untimed PNs in uncertain environments as stochastic dynamic systems which are observed at discrete time points ($k$ = 0, 1, ...). At each time point $k$, if the system is in state $M \in \text{Reach}(G, M_I)$, a control action $c \in \mathbf{C} \cup \{\theta\}$ has to be chosen by the controller and executed. Such an action changes the state of the system. In this case, two things happen: a reward $R(M, c)$ is earned immediately, and the system moves to a new state $M'$ according to the probability $\text{prob}(M, M', c)$ defined in Section 2. The rewards are defined by:

$$\begin{aligned} &R(M, c) = +N \text{ if } (M \in \Gamma_C{}^{-}(M_{\text{ref}}) \text{ and } c \neq \theta) \text{ or } (M = M_{\text{ref}} \text{ and } c = \theta) \\ &R(M, c) = 0 \text{ if } M \neq M_{\text{ref}} \text{ and } c = \theta \\ &R(M, c) = -1 \text{ otherwise} \end{aligned} \tag{8}$$

The reward function $R(M, c)$ gives a payoff of $N$ to concrete actions (i.e., $c \neq \theta$) chosen for any predecessor of $M_{\text{ref}}$ and for the null action chosen for $M_{\text{ref}}$. It gives no gain nor penalty for the null action (i.e., $c = \theta$) when applied in state $M \neq M_{\text{ref}}$. Finally, it gives a penalty in all other cases. The reward obtained from a trajectory $(\sigma, M)$ of the form (9) depends linearly on the length $h$ of the firing sequence $\sigma$:

$$R(M, \sigma) = N - h + 1 \tag{9}$$

Consequently, determining a trajectory of minimal length from $M$ to $M_{\text{ref}}$ is equivalent to deriving a trajectory that maximizes $R(M, \sigma)$. Note that for very uncertain environments, the reward function can be adapted by adding a penalty for actions that move the state into a dangerous state (i.e., a state from which an unexpected event may occur).

To ensure the existence of an exact solution we consider discounted MDP with finite state and action spaces. Thus, the value function $V^{\pi}(M_I)$, which is the expected reward from using the policy $\pi$ when the process starts in state $M_I$, is given by

$$V^{\pi}(M_I) = E\left[\sum_k \alpha^k R(M(k), \pi(M(k)) \mid M_I\right] \tag{10}$$

The objective is to determine **V***, the maximum expected total discounted reward, over an infinite horizon where $\alpha$ is the discount factor ($0 \le \alpha < 1$). The vector **V*** satisfies the optimality equation [7]:

$$V^*(M) = \left\{ \max_{c \in C \cup \{\theta\}} R(M, c) + \alpha \sum_{M' \in \text{Reach}(G, M_I)} \text{prob}(M, M', c) V^*(M') \right\} \quad (11)$$

The actions attaining the maximum in (11) define an optimal deterministic policy $\pi$* satisfying: $\sum_{k} \alpha^k R(M(k), \pi(M(k)) \mid M_I]$

$$\pi^*(M) \in \underset{c \in C \cup \{\theta\}}{\text{argmax}} \left\{ R(M, c) + \alpha \sum_{M' \in \text{Reach}(G, M_I)} \text{prob}(M, M', c) V^*(M') \right\} \quad (12)$$

The most widely used iterative methods for finding optimal or approximately optimal policies for MDPs include value iteration (VI) and policy iteration (PI) algorithms. In this section, we present an adaptation of the VI algorithm to PNs introduced by Larach et al. [16]. This algorithm uses a list of state-action successors that leads to accelerating the iterations of the usual VI algorithm.

**Algorithm 1. VI Algorithm**

(In: Reach($G$, $M_I$), PROB, **C**, $\{\Gamma_C^+, c \in C \cup \{\theta\}\}$, $\alpha$, $\eta$; Out: $V$*, $\pi$*)

1. For all $M \in$ Reach($G$, $M_I$), *Do* $V$*($M$) $\leftarrow$ 0; end for
2. Repeat
3. For all $M \in$ Reach($G$, $M_I$), *Do*

$$V^k(M) = \max_{c \in C \cup \{\theta\}} \left\{ R(M, c) + \alpha \sum_{M' \in \Gamma_C^+} \text{prob}(M, M', c) V^{k-1}(M) \right\}$$

4. End for
5. Until $(\|V^k - V^{(k-1)}\| \le \eta)$
6. $V^* \leftarrow V_k$
7. For all $M \in$ Reach($G$, $M_I$), *Do*

$$\pi^*(M) \in \underset{c \in C \cup \{\theta\}}{\text{argmax}} \left\{ R(M, c) + \alpha \sum_{M' \in \Gamma_C^+} \text{prob}(M, M', c) V^*(M') \right\}$$

8. End for
9. Return $V$*, $\pi$*

The use of the sets $\{\Gamma_C^+, c \in \mathbf{C} \cup \{\theta\}\}$ instead of Reach($G$, $M_I$) avoids the need to include the states $M'$ that are not reachable from state $M$ with control action $c$ in (12). This reduces the complexity of the classical VI algorithm [16].

### 3.3. Example

Let us consider again PN1 and PN2 as given in Fig. 1. Four MDP-based controllers $\pi_1$–$\pi_4$, will be considered below, corresponding to the four cases previously described. The controllers generate only controllable actions in the set $\mathbf{C} \cup \{\theta\}$. The definition of the controller depends on the set PROB defined by (5)–(7) and on the reward function (8) according to the individual cases. The control actions taken by $\pi_1$–$\pi_4$ are given in Table 1.

Table 1. MDP-based controllers for PN1 and PN2

| $M$ | $\pi_1$ | $\pi_2$ | $\pi_3$ | $\pi_4$ |
|---|---|---|---|---|
| $M1$ | $d$ | $d$ | $a$ | $a$ |
| $M2$ | $b$ | $d$ | $a$ | $a$ |
| $M3$ | $d$ | $d$ | $a$ | $a$ |
| $M4$ | $b$ | $b$ | $b$ | $b$ |
| $M5$ | $c$ | $d$ | $a$ | $a$ |
| $M6$ | $b$ | $b$ | $b$ | $b$ |
| $M7$ | $f$ | $f$ | $\theta$ | $\theta$ |
| $M8$ | $d$ | $d$ | $a$ | $a$ |
| $M9$ | $b$ | $b$ | $b$ | $b$ |
| $M10$ | $c$ | $f$ | $\theta$ | $\theta$ |
| $M11$ | $b$ | $b$ | $b$ | $b$ |
| $M12$ | $f$ | $f$ | $\theta$ | $\theta$ |
| $M13$ | $c$ | $c$ | $c$ | $c$ |
| $M14$ | $c$ | $c$ | $c$ | $c$ |
| $M15$ | $\theta$ | $\theta$ | $\theta$ | $\theta$ |

Let us consider again the initial marking $M_I = 2p_1$ and the reference marking $M_{\text{ref}} = 2p_5$. The policies $\pi_1$ and $\pi_2$ are used under PN1 when no unexpected events are considered. These policies decide at the first step to enforce the firing of $t_4$ using the control action $d$. When no control error occurs (Case 1), the complete trajectory is obtained by the sequence of decisions $\sigma_\pi = d\ d\ f\ f$ that leads to the firing sequence $\sigma = t_4\ t_4\ t_5\ t_5$. The policy $\pi_1$ leads to the shortest sequence and the sequence $t_4\ t_5$ is preferred to the sequence $t_1\ t_2\ t_3$ to move each token in $p_1$ into $p_5$. When control errors occur (Case 2), the controller adapts the sequence of actions depending on these errors. For example, the incorrect execution of the second action $d$ as $t_1$ leads to the firing sequence $\sigma' = t_4\ t_1\ t_2\ t_5\ t_3$. The policies $\pi_3$ and $\pi_4$ are used under PN2 when unexpected events are considered. In these cases, the sequence $t_1\ t_2\ t_3$ is preferred to the sequence $t_4\ t_5$ to move each token in $p_1$ into

$p_5$ because the transition $t_5$ is no longer controllable. When no control error occurs (Case 3), the complete trajectory is obtained by the sequence of decisions $\sigma''_\pi = a\ a\ b\ b\ c\ c$ that leads to the firing sequence $\sigma'' = t_1\ t_1\ t_2\ t_2\ t_3\ t_3$. When control errors can also occur (Case 4), the controller adapts the sequence of actions. For example, if the last action $c$ is executed as $t_6$, the sequence of decisions leads to the firing sequence $\sigma''' = t_1\ t_1\ t_2\ t_2\ t_3\ t_6\ t_1\ t_2\ t_3$.

## 4. Case study

PN3 is the untimed model of a manufacturing system that produces two types of products corresponding to two jobs [9] (Fig. 4). The first job is defined by transitions $t_1$–$t_8$ and the second one is defined by transitions $t_9$–$t_{14}$. The six resources $p_{14}$–$p_{19}$ have limited capacities: $m(p_{14}) = m(p_{15}) = m(p_{16}) = m(p_{17}) = m(p_{18}) = m(p_{19}) = 1$. Given the initial marking $M_I = 6p_1 + 6p_8 + 1p_{14} + 1p_{15} + 1p_{16} + 1p_{17} + 1p_{18} + 1p_{19}$, PN3 has 282 reachable states and 16 deadlock markings (from which no transitions can be implemented).

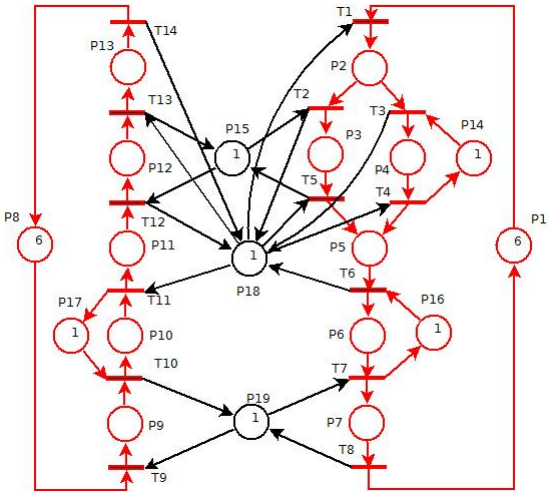


Fig. 4. PN3 model of a manufacturing system (after [9])

In order to point out the advantages of the proposed MDP-based method, the performance and complexity of the method are evaluated and compared with the results obtained with exhaustive searches based on breadth-first search (BFS) and depth-first search (DFS) approaches. Case 1 is considered for this example: there is no control error nor unexpected event, all transitions are controllable (i.e., $\mathbf{T}_C = T$). For comparison, a selection of several reference markings $M_{\text{ref}}$ is considered. The markings $M_{\text{ref}}$ are categorised into different sets $G$, depending on the length $K_{\text{opt}}$ of the optimal sequence for reaching $M_{\text{ref}}$. Note that the length of the shortest sequence from $M_I$ to any reachable

marking $M_{ref}$ is easy to obtain by calculating the successive powers of the generator of the reachability graph, but this method does not derive the sequence, it just gives its length [17]. The sets $GK_{opt}$ with $K_{opt} \in \{4, 7, 10, 13, 15, 18\}$ (18 is the maximal depth in Reach(PN3, $M_I$)) are considered in turn and $|GK_{opt}|$ denotes the cardinality of the set $GK_{opt}$. The proposed solutions are calculated using a maximal computation time, $C_{T\lim}$, of 120 s that corresponds to exhaustive exploration up to depth 10 using the BFS approach for an Intel Core i7-4600 CPU at 2.1–2.7 GHz. Table 2 summarises the global convergence performance in the untimed context: for each set $GK_{opt}$ the success rate (i.e., the proportion of reference markings in $GK_{opt}$ for which an admissible control sequence is found within $C_{T\lim}$) is reported for each method. Table 2 also reports the proportion of reference markings for which the computed control sequence is minimal in length. For example, the performance using the DFS method for set $G4$ is 100/10. This means that 100% of the reference markings in the set $G4$ are reached and that 10% are reached using a sequence of length 4. Table 2 illustrates that the MDP-based method always succeeds and leads to the best solution, whereas the BFS only succeeds up to depth 10 and then fails because $C_{T\lim}$ is exceeded. In comparison, the DFS approach provides poor solutions, because DFS continues exploring each branch until a marking that has been previously visited in the same branch is reached. Thus, the returned solutions are often non-optimal, because only a few branches are explored within $C_{T\lim}$.

Table 2. Performance for PN3

| $GK_{opt}$ | $\|GK_{opt}\|$ | MDP | DFS | BFS |
|---|---|---|---|---|
| $G4$ | 10 | 100/100 | 100/10 | 100/100 |
| $G7$ | 26 | 100/100 | 92/8 | 100/100 |
| $G10$ | 32 | 100/100 | 91/9 | 100/100 |
| $G13$ | 12 | 100/100 | 92/8 | 0 |
| $G15$ | 4 | 100/100 | 100/0 | 0 |
| $G18$ | 4 | 100/100 | 50/0 | 0 |

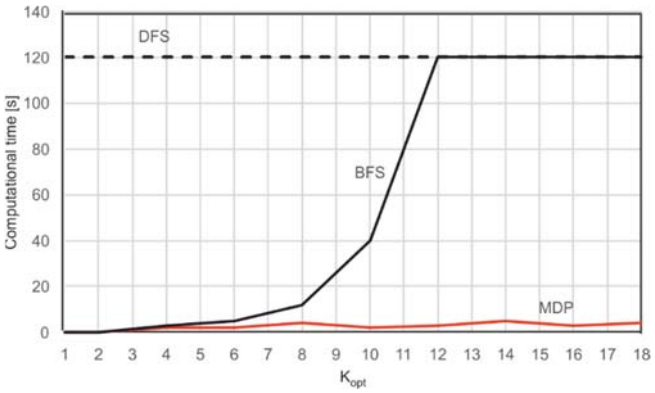


Fig. 5. Mean computation time in dependence on the depth $K_{opt}$ for PN3

Figure 5 reports the computational time using the tested methods w.r.t. the depth $K_{\text{opt}}$. One can notice that the complexity of the MDP-based methods is low compared to exhaustive search, which has an exponential complexity: it never exceeds 5 seconds. This illustrates the advantage of MDP-based methods from the computational point of view.

## 5. Conclusion

Control issues for DESs in uncertain environments have been considered. These systems have been modelled using untimed PNs that include unexpected events and control errors encoded in a set of probabilistic equations. The design of control sequences is obtained by using discrete time Markov decision processes (MDPs) leading to minimal size sequences according to an appropriately defined reward function. The main advantage of the proposed methods is that they provide an optimal solution even if the environment is uncertain but satisfies the Markovian property. Another advantage is that the optimal policy is applicable for any initial marking and only depends on the reference to be reached.

This work is a preliminary study about the use of MDPs for control problems modelled using PNs and numerous perspectives will be considered in future work. Above all, the reward function will be adapted to include an evaluation of the risk of firing uncontrollable transitions in the computed trajectory. The structural analysis of PNs will be used for that purpose. For DESs with very uncertain environments (i.e., including numerous unexpected events), policies that consider unexpected events as possible actions will also be studied. To accelerate convergence, a reverse computation of the reachability graph will be studied.

In the next step, the method will be extended to timed PNs and to unbounded nets. Finally, unknown environments will be considered by considering Markov decision PNs, Markov decision well-formed nets, bounded-parameter MDPs and by adding learning to the MDP framework.

### Acknowledgements

The Project MRT MADNESS 2016–2019 is funded by the European Union from the European Regional Development Fund (ERDF), as well as by the Regional Council of Normandy and the Erasmus Battuta program.

### References

[1] ABBAD M., DAOUI C., *Hierarchical algorithms for discounted and weighted markov decision processes*, Math. Meth. Oper. Res., 2003, 58 (2), 237–245.

[2] ALUR R., HENZINGER T., *Reactive modules*, Formal Meth. Syst. Des., 1999, 15 (1), 7–48.

[3] BAKER K.R., TRIETSCH D., *Principles of Sequencing and Scheduling*, Wiley, 2009.
[4] BECCUTI M., FRANCESCHINIS G., HADDAD S., *A framework to design and solve Markov decision Petri nets*, Int. J. Perform. Eng., 2011, 7 (5), 417–442.
[5] BECCUTI M., FRANCESCHINIS G., HADDAD S., *Markov decision Petri net and Markov decision well-formed net formalisms*, Lecture Notes in Comp. Sci., 2007, 4546, 43–62.
[6] BECCUTI M., CODETTA-RAITERI D., FRANCESCHINIS G., HADDAD S., *A framework to design and solve Markov decision well-formed net models*, Proc. 4th IEEE Int. Conf. Quantitative Evaluation of Systems (QEST 2007), Scotland, UK, 2007, 165–166.
[7] BELLMAN R.E., *Dynamic Programming*, Princeton University Press, Princeton, NJ, 1957.
[8] CASSANDRAS C., *Discrete event systems. Modeling and performance analysis*, Aksen Ass. Inc. Pub., Homewood 1993.
[9] CHEN Y., LI Z., KHALGUI M., MOSBAHI O., *Design of a maximally permissive liveness-enforcing Petri net supervisor for flexible manufacturing systems*, IEEE Trans. Automation Science and Engineering, 2011, 8 (2), 374–393.
[10] DAOUI C., ABBAD M., TKIOUAT M., *Exact decomposition approaches for Markov decision processes. A survey*, Adv. Oper. Res., 2010, 1–19.
[11] DAVID R., ALLA H., *Petri Nets and Grafcet. Tools for Modelling Discrete Events Systems*, Prentice Hall, London 1992.
[12] EBOLI M.G., COZMAN F.G., *Markov decision processes from colored Petri nets*, SBIA 2010, Lecture Notes in Comp. Sci., Springer, 2010, 6404.
[13] FENG Y., XING K., GAO Z., WU Y., *Transition cover-based robust Petri net controllers for automated manufacturing systems with a type of unreliable resources*, IEEE Trans. Syst., *M*an Cyber. Syst., 2016, 1–11.
[14] GIVAN R., LEACH S., DEAN T., *Bounded-parameter Markov decision processes*, J. Art. Int., 2000, 122 (1–2), 71–109.
[15] JENG M.D., CHEN S.C., *Heuristic search approach using approximate solutions to Petri Net state equations for scheduling flexible manufacturing systems*, Int. J. Flex. Manuf. Syst., 1998, 10 (2), 139–162.
[16] LARACH A., DAOUI C., CHAfiK S., *Accelerated decomposition techniques for large discounted Markov decision processes*, J. Ind. Eng., Springer, 2017.
[17] LEFEBVRE D., *On-line fault diagnosis with partially observed Petri nets*, IEEE Trans. Aut. Control, 2014, 59 (7), 1919–1924.
[18] LEFEBVRE D., LECLERCQ E., *Control design for trajectory tracking with untimed Petri nets*, IEEE Trans. Aut. Control, 2015, 60 (7), 1921–1926.
[19] LEFEBVRE D., *Approaching minimal time control sequences for timed Petri nets*, IEEE Trans. Aut. Sci. Eng., 2016, 13 (2), 1215–1221.
[20] LEFEBVRE D., *Deadlock-free scheduling for timed Petri net models combined with MPC and backtracking*, Proc. IEEE WODES 2016, Control, Observation, Estimation and Diagnosis with Timed PNs, Xian, China, 2016, 466–471.
[21] LEFEBVRE D., *Deadlock-free scheduling for flexible manufacturing systems using untimed Petri nets and model predictive control*, Proc. IFAC – MIM, DES for Manufacturing Systems, Troyes, France, 2016.
[22] LEI H., XING K., HAN L., XIONG F., GE Z., *Deadlock-free scheduling for flexible manufacturing systems using Petri nets and heuristic search*, Comp. Ind. Eng., 2014, 72, 297–305.
[23] LEUNG Y.-T., *Handbook of Scheduling. Algorithms, Models, and Performance Analysis*, CRC Computer and Information Science Series, Chapman Hall, 2004.
[24] LOPEZ P., ROUBELLAT F., *Production Scheduling*, ISTE Wiley, London 2008.
[25] PUTERMAN M., *Markov Decision Processes. Discrete Stochastic Dynamic Programming*, Wiley, New York 1994.

[26] TUNCEL G., BAYHAN G.M., *Applications of Petri nets in production scheduling. A review*, Int. J. Adv. Manuf. Techn., 2007, 34, 762–773.
[27] WANG Q., WANG Z., *Hybrid heuristic search based on Petri net for FMS scheduling*, En. Proc., 2012, 17, 506–512.
[28] WIESEMANN W., KUHN D., RUSTEM B., *Robust Markov decision processes*, Math. Oper. Res., 2013, 138 (1), 153–183.
[29] YUE H., XING K., HU H., WU W., SU H., *Petri-net-based robust supervisory control of automated manufacturing systems*, Control Eng. Pract., 2016, 54, 176–189.

*Received 13 June 2017*
*Accepted 9 January 2018*