processes

MDPI

Article

Dynamical Scheduling and Robust Control in Uncertain Environments with Petri Nets for DESs

Dimitri Lefebvre

GREAH Research Group, UNIHAVRE, Normandie University, 76600 Le Havre, France; dimitri.lefebvre@univ-lehavre.fr

Received: 3 September 2017; Accepted: 21 September 2017; Published: 1 October 2017

Abstract: This paper is about the incremental computation of control sequences for discrete event systems in uncertain environments where uncontrollable events may occur. Timed Petri nets are used for this 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 similar to model predictive control with a finite set of control actions. At each step only a small area of the reachability graph is explored: this leads to a reasonable computational complexity. The robustness of the resulting trajectory is also evaluated according to a risk probability. A sufficient condition is provided to compute robust trajectories. The proposed results are applicable to a large class of discrete event systems, in particular in the domains of flexible manufacturing. However, they are also applicable to other domains as communication, computer science, transportation, and traffic as long as the considered systems admit Petri Nets (PNs) models. They are suitable for dynamical deadlock-free scheduling and reconfiguration problems in uncertain environments.

Keywords: discrete event systems; timed Petri nets; stochastic Petri nets; model predictive control; scheduling problems

1. Introduction

The design of controllers that optimize a cost function is an important objective in many control problems, in particular in scheduling problems that aim to allocate a limited number of resources within several users or servers according to the optimization of a given cost function. 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 minimal cycle times. However, due to multi-layer resource sharing and routing flexibility of the jobs, scheduling problems are often NP-hard problems. Many recent works in operations research, automatic control, and computer science communities have studied such problems. In operations research community, flow-shop, and job-shop problem have been investigated from 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 [3–5]. In the automatic control community, automata, Petri nets (PNs), and max-plus 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 are based on the Dijkstra and A* algorithms [8,9]. Such algorithms explore the reachability graph of the net, in order to generate schedules. Numerous improvements have been proposed: 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]. By combining scheduling and supervisory control in the same approach, one can also avoid deadlocks. Some approaches have been proposed: 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 problems are considered. The advantage to solving scheduling problems with PNs or other tools issued from the control theory is to use a common formalism to describe a large class of problems and to facilitate the representation from one problem to another. In particular, PNs are suitable to represent many systems in various domains as flexible manufacturing, communication, computer science, transportation, and traffic [6,7]. This makes such approaches more suitable for dynamic and robust scheduling in uncertain environments. However, modularity and genericity usually suffer from a large computational effort that disqualifies the approaches for numerous large systems.

This work aims to propose a modular and generic approach of weak complexity. It details a method for timed PNs 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. The computed control sequences aim to reach a reference state from an initial one. The forbidden states, as deadlocks and dead-branches are avoided. The trajectory duration approaches its minimal value. Thanks to its robustness, the proposed approach generates dynamical and reconfigurable schedules. Consequently, it can be used in a real-time context. Resource allocation and operation scheduling for manufacturing systems are considered as the main applications. The robustness of the resulting trajectory is evaluated as a risk belief or probability. For that purpose structural and behavioral models of the uncertainties are considered. Finally, robust trajectories are computed. Compared to our previous works [19–22], the main contributions are: including, explicitly, uncertainties by means of uncontrollable stochastic transitions in the PNs model; evaluating the risk of the computed control sequences; proposing a sufficient condition for the existence of robust trajectories.

The paper is organized as follows. In Section 2, the preliminary notions and the proposed method are developed: timed PNs with uncontrollable transitions are presented, non-robust and robust control sequences are introduced, and the approach to compute non-robust and robust control sequences with minimal duration is developed. Section 3 illustrates the method on a simple example and then presents the performance for a case study. Section 4 is a discussion about the method and the results. Section 5 sums up the conclusions and perspectives.

2. Materials and Methods

2.1. Petri Nets

A PN structure is defined as $G = \langle \boldsymbol{P}, \boldsymbol{T}, W_{PR}, W_{PO} \rangle$, 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 with indices $\{1, \ldots, q\}$ $W_{PO} \in (\mathbf{N})^{n \times q}$ and $W_{PR} \in (\mathbf{N})^{n \times q}$ are the post- and pre- incidence matrices (**N** is the set of non-negative integer numbers), and $W = W_{PO} - W_{PR} \in (\mathbf{Z})^{n \times q}$ (**Z** is the set of positive and negative integer numbers) is the incidence matrix. $\langle G, M_I \rangle$ 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 $n_j(M)$:

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

where ${}^{\circ}\boldsymbol{T_j}$ stands for the preset of T_j, 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 indices 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 trajectory (σ, M):

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

where $M(0) = M$ is the marking from which the trajectory is issued, $M(1)$, ..., $M(h-1)$ 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 to be ***reachable*** from initial marking M_I if there exists a firing sequence σ such that M_I $[\sigma > M$ and σ is said to be ***feasible*** at M_I. $\boldsymbol{R}(G, M_I)$ is the set of all reachable markings from M_I.

2.2. Forbidden, Dangerous and Robust Legal Markings

For control issues, the set of transitions $\boldsymbol{T}$ is divided into two 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 firings of enabled controllable transitions are enforced or avoided by the controller, whereas the firings of uncontrollable transitions are not, and uncontrollable transitions 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 $SPEC(M) = 0$. When no specification is considered, $SPEC(M) = 1$ for all $M \in \boldsymbol{R}(G, M_I)$. The two disjoint sets $\boldsymbol{F}(G, M_I, M_{ref})$ and $\boldsymbol{L}(G, M_I, M_{ref})$ of forbidden and legal markings respectively are introduced:

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

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

In other words, a marking $M \in \boldsymbol{R}(G, M_I)$ is ***legal*** with respect to M_{ref} if a 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*** with respect to $\boldsymbol{T_C}$ if $M^\circ \subseteq \boldsymbol{T_C}$, where M° stands for the set of transitions enabled at M, otherwise M is ***dangerous*** (Figure 1) With this definition of robust and dangerous markings, a marking that satisfies $M^\circ \subseteq \boldsymbol{T_C}$ but that has only dangerous markings as successors in $\boldsymbol{R}(G, M_I)$ is considered as robust. Note that a finer partition of the legal markings in three classes (strong robust, weak robust, and dangerous) could be used for some problems. 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 (Figure 1).

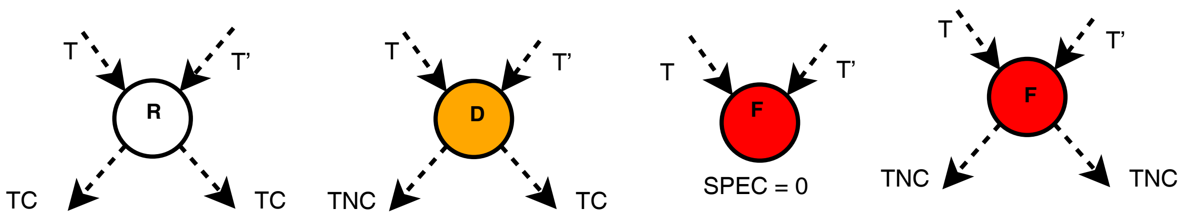

Figure 1. Examples of robust (R), dangerous (D) and forbidden (F) markings in $\boldsymbol{R}(G, M_I)$ depending on the controllable (TC) and uncontrollable transitions (TNC).

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

2.3. Timed Petri Nets 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 and infinite server semantic where the firing of controllable transitions behaves according to an earliest firing preselection policy (transitions fire earliest in the order 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 ($d_{min\,j} = 0$ if no time specification exists for T_j). 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 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 parameters $d_{min\,j}$ are set in an arbitrary time unit (TU) and the parameters μ_j are set in TU^{-1}.

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 indices 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 trajectory (σ, M):

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

with $M(0) = M$. Note that, under earliest firing policy, an untimed trajectory of the form of Equation (2) that contains only controllable transitions can be transformed in a straightforward way into a timed trajectory of the form of Equation (5) of minimal duration [20,21] using Algorithm 1. This algorithm also returns $DURATION(\sigma, M) = t_h$.

Algorithm 1. Transformation of an untimed trajectory (σ,M) into timed one (σ',M).

(Inputs: σ, M, G, D_{min},; Output: σ', τ)

initialization: $\tau \leftarrow 0$; $CAL \leftarrow \{(T_j, d_{min\,j})$ at $M\ [\ T_j >\}$, $\sigma' \leftarrow (\varepsilon,0)$, $h \leftarrow |\sigma|$
for k from 1 to h
  find in CAL the date τ_k of the earliest occurrence of the k^{th} transition $T(j_k)$ in σ
  $\tau \leftarrow \tau_k$, remove entry $(T(j_k), \tau_k)$ in CAL
  $CAL_{new} \leftarrow \varnothing$, $M' \leftarrow M' - W_{PR}.X(T(j_k))$
  for all T' at $M'\ [\ T' >$
    compute the enabling degree $n'(T', M')$ of T' at M'
    for j from 1 to $n(T', M')$
      find the j^{th} occurrence (T', τ'_j) of T' in CAL
      $CAL_{new} \leftarrow CAL_{new} \cup (T,' \max(\tau'_j, \tau))$
    end for
  end for
  $M'' \leftarrow M' + W_{PO}.X(T(j_k))$
  for all t'' at $M''\ [\ T'' >$
    compute the enabling degree $n''(T'', M'')$ of T'' at M''
    for j from 1 to $n''(T'', M'') - n'(T'', M')$
      $CAL_{new} \leftarrow CAL_{new} \cup (T'', \tau + d_{min}(T''))$
    end for
  end for
  $CAL \leftarrow CAL_{new}$, $\sigma' \leftarrow \sigma'\ (T(j_k), \tau_k)$
end for
$\tau \leftarrow \tau_h$

2.4. Belief and Probability of Trajectory Deviation

The objective of this section is to evaluate the risk that uncontrollable firings may occur during the execution of the trajectory (σ, M_I) and deviate the trajectory from the reference. For PCont-TPNs, this risk is evaluated with the belief $RB(\sigma, M_I, \boldsymbol{T_C})$:

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

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). For PCont-SPNs, the belief $RB(\sigma, M_I, \boldsymbol{T_C})$ is replaced by the probability $RP(\sigma, M_I, \boldsymbol{T_C})$ that can be computed with Proposition 1:

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

$$\begin{gathered} RP(\sigma, M_I, \boldsymbol{T_C}) = \sum_{1\le k_1\le h} \pi(k_1) - \sum_{1\le k_1<k_2\le h} (\pi(k_1).\pi(k_2)) + \cdots + \\ (-1)^{h-1}.\sum_{1\le k_1<\ldots<k_{h-1}\le h} (\pi(k_1)\ldots\pi(k_{h-1})) + (-1)^h.\pi(1)\ldots\pi(h) \end{gathered} \tag{7}$$

with:

$$\pi(k) = \frac{\sum_{T_j\in \boldsymbol{T_{NC}}\cup(\boldsymbol{M(k)})^\circ} \mu_j}{\sum_{T_j\in \boldsymbol{T_{NC}}\cup(\boldsymbol{M(k)})^\circ} \mu_j + \left(d_{j_k}\right)^{-1}}$$

if $d_{jk} \neq 0$, otherwise $\pi(k) = 0$, and $d_{jk} = t_{k+1} - t_k$ 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 dangerous markings belong to (σ, M_I).

Consider the trajectory of Figure 2. Under earliest firing policy, 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:

$$\pi(k) = Prob(T_{NC1} \text{ or } T_{NC2} \text{ fires before } T(j_{k+1},\ t_{k+1})) = \frac{\mu_1+\mu_2}{\mu_1+\mu_2+\left(d_{j_k}\right)^{-1}}$$

if $d_{jk} \neq 0$, otherwise $Prob(T_{NC1}$ or T_{NC2} fires before $T(j_{k+1}, t_{k+1})) = 0$. Note that if the controllable transition $T(j_{k+1}, t_{k+1})$ fires earliest after a duration d_{jk}, then the probability $\pi(k)$ is computed by considering the approximation $1/d_{jk}$ of the mean firing rate of $T(j_{k+1}, t_{k+1})$. Note also that the duration of other controllable transitions enabled at $M(k)$ (for example, T_{C2} in Figure 2) are not considered because this transition does not belong to (σ, M_I). Alternatively the probability that the trajectory continues to $M(k+1)$ at $M(k)$ is given by:

$$1-\pi(k) = Prob(T(j_{k+1},\ t_{k+1}) \text{ fires before } T_{NC1} \text{ and } T_{NC2}) = \frac{\left(d_{j_k}\right)^{-1}}{\mu_1+\mu_2+\left(d_{j_k}\right)^{-1}} \tag{8}$$

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

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

for which an exhaustive development is easily rewritten as in Equation (7).

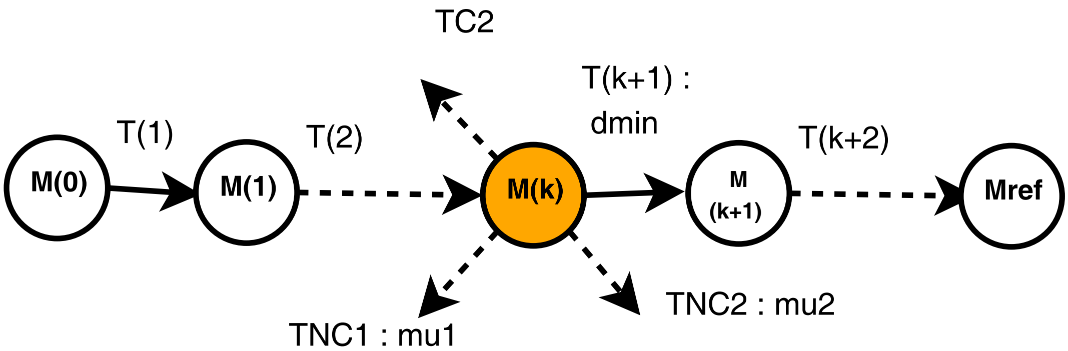

Figure 2. An example of dangerous trajectory: $M(k)$ enables two controllable transitions $T(k+1)$ and T_{C2} and two uncontrollable ones T_{NC1} and T_{NC2}.

2.5. Model Predictive Control for PCont-TPNs

The determination of control sequences for untimed and timed PNs that contain 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). At each step, the future trajectory is predicted from the current state. A sequence of control actions is computed by minimizing and the first action of the sequence is applied. Then prediction starts again from the new state reached by the system [25,26]. The cost function $J_{FC}(M, M_{ref}) = (D_{min})^T. X$ based on the temporal specification and on the evaluation X of the firing count vector, that leads to 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 and $W_C \in (\mathbf{Z})^{n \times qC}$ as the restrictions of G and W 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 minimizes $J_{FC}(M, M_{ref}) = (D_{min})^T.X_C$ is obtained by solving an optimization problem with integer variables of reduced size q_C-r where r is the rank of W_C. A regular matrix $P_L \in (\mathbf{Z})^{n \times n}$ and a regular permutation matrix $P_R \in \{0,1\}^{qC \times qC}$ exists at:

$$W_C{}' = P_L.W_C.P_R = \begin{pmatrix} W_{11} & W_{12} \\ W_{21} & W_{22} \end{pmatrix} \tag{9}$$

with $W_{11} \in (\mathbf{Z})^{r \times r}$ a regular upper triangular matrix with integer entries, and $W_{21} = 0_{(n\text{-}r) \times r}$, $W_{22} = 0_{(n\text{-}r) \times (qC\text{-}r)}$ zero matrices of appropriate dimensions. For each $M \in \boldsymbol{R}(G, M_I)$, solving Equation(10):

$$\text{Min } \{(D_{min})^T.X_C : X_C \in (\mathbf{N})^{qC} \text{ at } W_C.X_C = (M_{ref} - M)\} \tag{10}$$

is equivalent to solving Equation (11) and this leads to reduce the number of variables by r:

$$\text{Min } \{F_2.X_{C2} : X_{C2} \in (\mathbf{N})^{qC-r} \text{ at } (W_{11})^{-1}.W_{12}.X_{C2} \leq (W_{11})^{-1}.\Delta M_1\} \tag{11}$$

with $F_2 = (D_{min})^T.(P_{R2} - P_{R1}.(W_{11})^{-1}.W_{12})$, $P_R = (P_{R1} \mid P_{R2})$, $P_L = ((P_{L1})^T \mid (P_{L2})^T)^T$ and $\Delta M_1 = P_{L1}.(M_{ref} - M)$. This reformulation results from the rewriting $(\Delta {M_1}^T \; \Delta {M_2}^T)^T = P_L.(M_{ref} - M)$ and $({X_{C1}}^T \; {X_{C2}}^T)^T = (P_R)^{-1}.X_C$ with $X_{C1} = (W_{11})^{-1}.\Delta M_1 - (W_{11})^{-1}.W_{12}.X_{C2}$. The linear optimization problem (Equation (11)) has a solution with integer values as long as $M_{ref} \in \mathbf{R}(G_C, M)$ and the cost function $J_{FC}(M, M_{ref})$ based on firing count vector X_{C2} and on D_{min} is defined by Equation (12):

$$J_{FC}(M, M_{ref}) = (D_{min})^T.(P_{R1}.(W_{11})^{-1}.\Delta M_1 + P_{R2}.X_{C2}\; P_{R1}.(W_{11})^{-1}.W_{12}.X_{C2}) \tag{12}$$

As long as X_{C2} corresponds to a feasible and legal firing sequence σ to the reference (i.e., X_{C2} does not encode a spurious solution for Equation (11)), $J_{FC}(M, M_{ref})$ provides an upper bound of the duration of σ as proved with Proposition 2.

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

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

Proof. (σ, M_I) is written as in Equation (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 $t_1 + d_{min\,j2}$. 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 $t_{h-1} + d_{min\,jh}$. Thus $t_h \leq d_{min\,j1} + \ldots + d_{min\,jh}$. The minimal duration of (σ, M_I) is t_h, thus, Equation (13) holds.

The basic idea is to use $J_{FC}(M, M_{ref})$ to iteratively drive the search of the controllable firing sequence of minimal duration that leads to the reference. At each step (i.e., for each intermediate marking), a part of the controllable reachability graph is explored and a prediction of the remaining duration to the reference is obtained with cost function $J_{FC}(M, M_{ref})$ computed for each marking M of the explored graph. Then the first control action is applied (i.e., the next controllable transition fires). If an uncontrollable firing occurs, the trajectory deviates from the predicted one and the system enters in an unexpected state. However, 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 leads to a dynamical and robust scheduling. Two algorithms already developed in our previous works [21,22] are used for that purpose.

Algorithm 2 similar to the one developed in [21,22] encodes as a tree ***Tree***(M, H) a small part of the reachability graph rooted at M (Figure 3). The tree is limited in depth with parameter H and in duration with parameter H_τ.

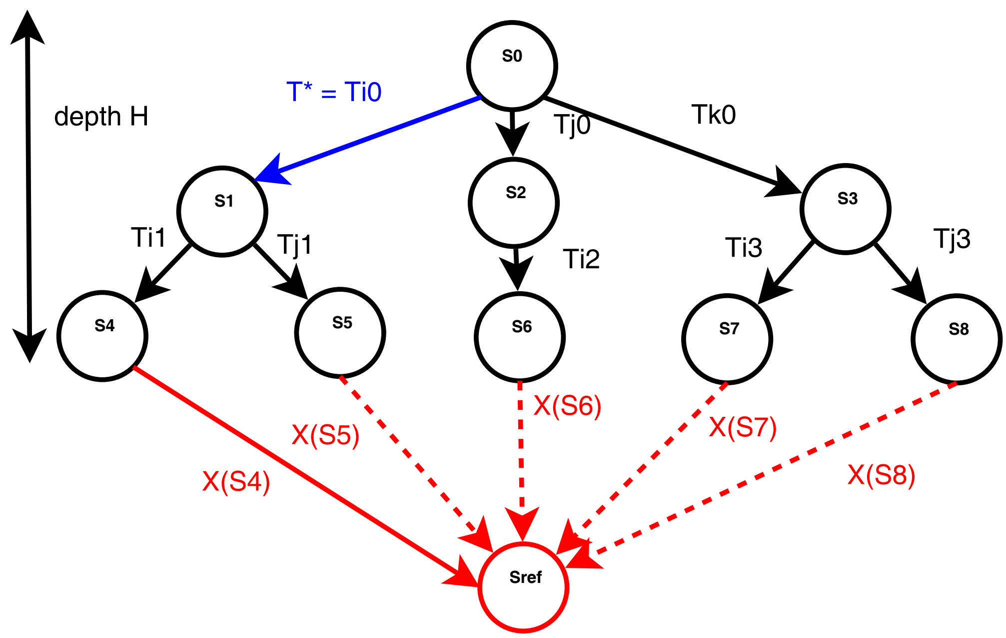

Figure 3. Computation of the next transition to fire with Algorithm 2.

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 $\sigma(S)$ at $M\ [\sigma(S) > m(S)$, 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 at $l(S) = 0$ 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, Algorithm 2 returns the next transition T^* to fire.

Algorithm 2. Computation of T^* for PCont-TPNs.

(Inputs: M, M_{ref}, G_C, $SPEC$, $\boldsymbol{F}$, H, H_τ ; Outputs: $\boldsymbol{F}$, *converge*, *exhaustive*, T^*)

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
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
$\boldsymbol{Tree} \leftarrow S_0$, $\sum \leftarrow S_0$, $T^* \leftarrow \varepsilon$, $exhaustive \leftarrow 1$
while $\exists\ S \in \boldsymbol{Tree}$ at $l(S) = 1$ and $e(S) = 0$,
  for each $T \in \boldsymbol{T_C}$ at $m(S)\ [T >$
    compute the successor S' of S by firing T, M' at $m(S)\ [t > M'$, $\sigma' \leftarrow \sigma(S)\ T$, $s' \leftarrow s(S)\ S'$
    if $(SPEC(M') = 0) \vee ((M')^{\circ} \cup \boldsymbol{T_C} = \varnothing)$, $\boldsymbol{F} \leftarrow \boldsymbol{F} \cup \{m(S)\}$, end if
    if $(M' \in \boldsymbol{F}) \vee (S' \in s(S))$,$l \leftarrow 0$, else $l \leftarrow 1$, end if
    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
    $\boldsymbol{Tree} \leftarrow \boldsymbol{Tree} \cup \{M', \sigma', s', l, e\}$
  end for
end while
for h from H-1 to 0
  for each $S \in \boldsymbol{Tree}$ at $|\sigma(S)| = h$
    if ($l(S') = 0$ for all direct successors S' of S in $\boldsymbol{Tree}$), $l(S) \leftarrow 0$, $e(S) \leftarrow 1$, end if
  end for
end for
for each $S \in \boldsymbol{Tree}$ at $(l(S) = 0) \wedge (e(S) = 0)$
  if $\exists \perp S' \in \boldsymbol{Tree}$ at $(S' \neq S) \wedge (m(S') = m(S)) \wedge (l(S') = 1)$, $\boldsymbol{F} \leftarrow \boldsymbol{F} \cup \{m(S)\}$, end if
end for
for each $S \in \boldsymbol{Tree}$ st $e(S) = 1$, $\sum \leftarrow \sum \cup \{S\}$, end if
$\sum^* \leftarrow \{S^*$ at $J_{FC}(m(S^*), M_{ref}) = \min(J_{FC}(m(S), M_{ref})$, for all $S \in \sum\}$
$\sum^{**} \leftarrow \{S^*$ at $DURATION(\sigma(S^*),M) = \min(DURATION(\sigma(S),M))$ for all $S \in \sum^*\}$
if $\{S_0\} = \sum^{**}$, $converge \leftarrow -1$, $T^* \leftarrow \varepsilon$, else select T^* as the first transition of $\sigma(S^*)$ with $S^* \in \sum^{**}$, $converge \leftarrow 0$, end if
for each $S \in \sum$
  if $(l(S) = 1) \wedge (e(S) = 1) \wedge (DURATION(\sigma(S),M) < H_\tau)$, $exhaustive \leftarrow 0$, end if
end for

The complete control sequence σ^* is obtained with Algorithm 3 similar to the one developed in [21,22] that adapts the parameter H in range $[1 : \overline{H}]$ where $\overline{H}$ is an input parameter (Figure 4) that limits the maximal depth of the search in steps. This algorithm starts at initial marking M_I, with no forbidden marking (i.e., $\boldsymbol{F} = \varnothing$) and with minimal depth (i.e., $H = 1$). As long as convergence is ensured, T^* is added to σ^* and the current marking M is updated. Finally Algorithm 3 also evaluates the risk RP of the computed trajectory.

Algorithm 3. Control sequence computation for PCont-TPNs.

(Inputs: M_I, M_{ref}, G, $\boldsymbol{T_C}$, $\boldsymbol{T_{NC}}$, $SPEC$, D_{min}, μ, $\overline{H}$,H_τ ; Outputs: σ^*, $success$, RP)

$M \leftarrow M_I$, $converge \leftarrow 0$, $\sigma^* \leftarrow \varepsilon$, $H \leftarrow 1$, $\boldsymbol{F} \leftarrow \varnothing$, $success \leftarrow 1$
while ($converge < 1$)
  compute $converge$, $exhaustive$ and $T^* \in \boldsymbol{T_C}$ and update $\boldsymbol{F}$ with Algorithm 2
  if ($converge = 0$)^(($exhaustive = 1$) $\vee$ (($exhaustive = 0$)^($H = \overline{H}$))),
    compute $\sigma^* \leftarrow \sigma^*\, T^*$ and M at $M_I\ [\sigma^* > M$
    $H \leftarrow \max(1, H\text{-}1)$
  end if
  if (($converge = -1$) $\wedge$ ($H = \overline{H}$)) $\vee$ ($converge = -2$),
    if ($M \neq M_I$),
      remove last transition in σ^* and compute M at $M_I[\sigma^* > M$
    else
    if ($converge = -2$), $success \leftarrow -2$, else $success \leftarrow -1$,end if
    break
    end if
  end if
  if (($H = \overline{H}$) $\wedge$ ($converge = 0$) $\wedge$ ($exhaustive = 0$)), $success \leftarrow 0$, end if
  if ($H < \overline{H}$) $\wedge$ (($converge = -1$) $\vee$ (($converge = 0$) $\wedge$ ($exhaustive = 0$)), $H \leftarrow H + 1$, end if
end while
compute RP with (7)

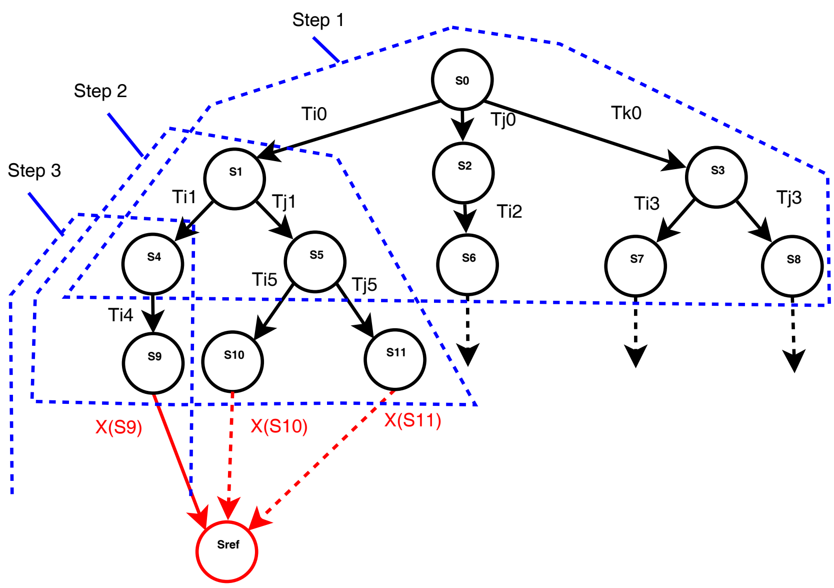

Figure 4. MPC global schema with Algorithm 3.

Note that the complexity of Algorithm 3 is at most $O(h.q_C^{\overline{H}})$ where $h = |\sigma^*|$.

Example 1. *PCont-SPN1 is considered with* $\boldsymbol{T_C} = \{T_1, T_2, T_3, T_4, T_5, T_6\}$, $\boldsymbol{T_{NC}} = \{T_7\}$, $D_{min} = (1, 1, 1, 1, 1, 5)^T$ *and* $\mu = \mu_7 = 1$ *(Figure 5). 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 token 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 the reference more rapidly. However, the uncontrollable transition* T_7 *may fire during execution of this cycle which leads to an* *excessive production of tokens. The cycle {P_1, T_5, P_4, T_6} which is a token consumer, is then used to correct the excessive number of tokens. Note that the execution of this last cycle is slow compared to the two other ones due (a) to the firing duration of T_6 that is five times larger than the duration of the other transitions; and (b) to the presence of the selfloop {T_5, P_8} that limits the number of simultaneous firings of T_5 to one (whereas the other transitions may fire several times simultaneously according to the infinite server semantic).*

The optimal timed sequence to reach M_{ref} is given by $\sigma_1 = T(3, 1)(T(4, 2))^5$ with duration DURATION(σ_1, M_I) = 2 time units (TUs). If no unexpected firing of T_7 occurs, Algorithm 3 applied with $\mathbf{T_C}$ leads to σ_1. However, if unexpected firings of T_7 occur, the trajectory is disturbed and requires more time to reach the reference. Figure 6 is an example of trajectory including one firing of T_7 at date 1.6 TUs. The rest of the control sequence is updated in order to compensate the deviation so that the marking finally reaches M_{ref} in 48.6 TUs instead of 2 TUs.

Figure 6 illustrates the systematic updating of the optimization process at each step (i.e., for each new firing). Consequently the firing of an uncontrollable transition at a given step k changes the future predictions, and the control actions computed at steps k + 1, k + 2, ... compensate the deviation as long as a controllable trajectory exists from the current marking to M_{ref}.

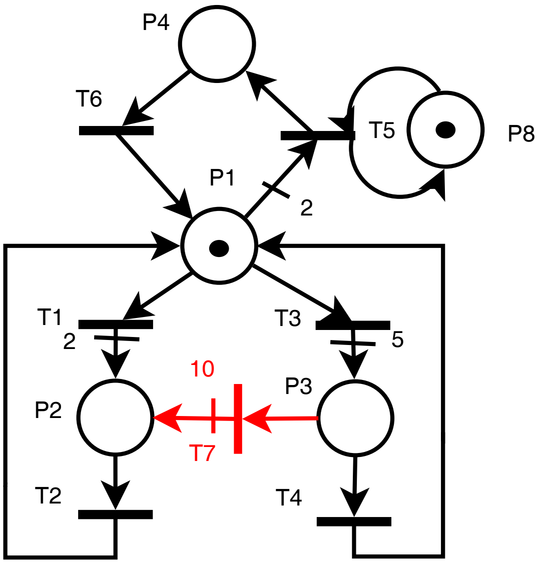

Figure 5. Example PCont-SPN1.

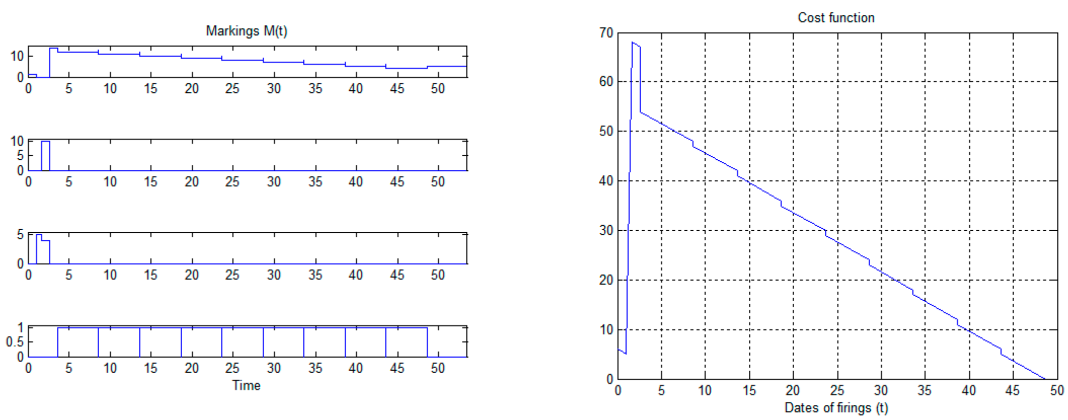

Figure 6. Cost function J_{FC} for a controlled sequence disturbed by an unexpected firing of T_7 with respect to time (TUs).

2.6. Robust Scheduling

In order to compute robust trajectories that cannot deviate from the reference, the controller should avoid dangerous intermediate markings and consider only legal trajectories with robust markings (i.e., with zero-risk belief or probability). 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 window provided by the part of the reachability graph, of depth H, explored at each step. During the 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 $\boldsymbol{T_{RC}} = \{T_j \in \boldsymbol{T_C} \text{ at } (T_j^\circ)^\circ \subseteq \boldsymbol{T_C}\}$ where $(T_j^\circ)^\circ = \cup \{P_i^\circ : P_i \in T_j^\circ\}$.

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

Proof. Note at first that $(M_I)^\circ \subseteq \boldsymbol{T_C}$ implies that the net has no uncontrollable source transition (i.e., ${}^\circ T_j \neq \varnothing$ for all $T_j \in \boldsymbol{T_{NC}}$). Then, (σ, M_I) is written as in Equation (5): $\sigma = M_I\ [T(j_1, t_1) > M(1) \ldots . > M(h)$. Assume that there exists $T_j \in (M(1))^\circ$ 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 $P_i \in {}^\circ T_j$ whose marking increases by firing $T(j_1, t_1)$ and consequently $P_i \in (T(j_1, t_1))^\circ$. As $T_j \in P_i^\circ$, $T_j \in ((T(j_1, t_1))^\circ)^\circ$. Thus $T_j \in \boldsymbol{T_C}$ that is contradictory with assumption and $(M(1))^\circ \subseteq \boldsymbol{T_C}$. Repeating successively the same reasoning up to $M(h)$, one can conclude that $(M(k))^\circ \subseteq \boldsymbol{T_C}$, $k = 0, \ldots, h$, and that (σ, M_I) is a robust legal trajectory.

Note that robust legal trajectories are computed with Algorithms 2 and 3 by replacing $W_C \in (\mathbf{Z})^{n \times qC}$ with $W_{RC} \in (\mathbf{Z})^{n \times qRC}$ (i.e., the restriction of W to the set of robust controllable transitions $\boldsymbol{T_{RC}}$) in the determination of $J_{FC}(M, M_{ref})$.

Note also 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}:

$$\boldsymbol{T_{RC}} = \{T_j \in \boldsymbol{T_C} \text{ at } X_j.(W_{PO})^T.W_{PR}.(0 \mid I_{qNC})^T = 0\} \tag{14}$$

Example 2. *Let us consider again Pcont-SPN1 of Figure 5. In order to avoid any deviation, $\boldsymbol{T_{RC}} = \{T_1, T_2, T_4, T_5, T_6\}$ is considered instead of $\boldsymbol{T_C}$. Algorithm 3 applied with $W_{RC} \in (\mathbf{Z})^{n \times qRC}$ 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 has a duration DURATION$(\sigma_2, M_I) = 6$ TUs larger than DURATION(σ_1, M_I).*

The decision to prefer the control sequence σ_2 instead of σ_1 depends on the risk of both control strategies. Table 1 reports the values of RB and RP for both sequences σ_1 and σ_2 with respect to several values of μ. From Table 1, one can notice that the sequence σ_2 that is non-optimal in time has the advantage to be robust compared to σ_1. It cannot be perturbed by any unexpected firing. Note also that the risk probability of σ_1 depends strongly on the dynamic of the random firing of uncontrollable transition T_7. Note finally that computing RP instead of RB provides a better evaluation of that risk.

Table 1. Deviation risk for σ_1 and σ_2.

	RB	***RP***		
		$\boldsymbol{\mu_7 = 0.1}$	$\boldsymbol{\mu_7 = 1}$	$\boldsymbol{\mu_7 = 10}$
σ_1	5/6 = 0.83	1/3 = 0.33	5/6 = 0.83	50/51 = 0.98
σ_2	0	0	0	0

Table 2 reports the mean duration d of control sequences depending on μ_7 for three scenarios. All sequences are computed with M_I = (1 0 0 0)^T and M_{ref} = (5 0 0 0)^T and parameters $\overline{H}$ = 1, H_τ = 1. In scenario 1 all transitions are assumed to be controllable. In scenarios 2 and 3, $\boldsymbol{T_C}$ = {T_1, T_2, T_3, T_4, T_5, T_6}. Algorithm 3 is applied with $\boldsymbol{T_C}$ in scenario 2 whereas it is applied with $\boldsymbol{T_{RC}}$ = {T_1, T_2, T_4, T_5, T_6} in scenario 3. Simulations with scenario 2 are repeated 10 times to obtain a significant average duration. One can notice the advantage to compute a robust sub-optimal trajectory with scenario 3, which provides better result from μ_7 = 0.5. When μ_7 increases, the mean duration of T_7 firings decreases and the probability to fire T_7 before T_4 increases; consequently, the number of perturbations increases and the mean duration of the global trajectory also increases due to the execution of the cycle {P_1, T_5, P_4, T_6}.

Table 2. Performance of Algorithm 3 with Pcont-SPN1: average sequence duration (TUs).

μ_7	**0.1**	**0.5**	**1**	**2**	**10**
Scenario 1	2	2	2	2	2
Scenario 2	2.0	20.3	57.2	125.4	228.5
Scenario 3	6	6	6	6	6

3. Results

Pcont-SPN2 (Figure 7) is the timed model of a production system that processes a single type of products according to two possible jobs [27,28]. 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}. In the first job the transitions T_1, T_3, T_4, T_6, T_7, T_8 represent the operations in successive machines and the places P_1, P_2, P_4, P_6, P_7, T_8 are intermediate buffers where products are temporarily stored. The initial marking of place P_1 represents the maximal number of products that can be simultaneously processed by the Job 1. In the second job the transitions T_9, T_{10}, T_{11}, T_{12}, T_{13}, T_{14} represent the operations in successive machines and the places P_8, P_9, P_{10}, P_{11}, P_{12}, T_{13} are intermediate buffers. The initial marking of place P_8 represents the maximal number of products that can be simultaneously processed by the Job 2. 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, that contain 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 3 for $\overline{H} = 5$ and $H_\tau = 20$.

Table 3. 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

Another time, 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 cost function prefers 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). Thus scenario 1 corresponds to the iterated execution of Job 1. For scenario 2, $\mu_2 = 1$ and $d_{min\ 3} = 1$: consequently the probability that an unexpected firing of T_2 occurs is 0.5. When such a firing occurs the long firing duration $d_{min\ 5} = 20$ of T_5 compared to $d_{min\ 4} = 2$ 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 $\mu_2 = 1$ and $d_{min\ 3} = 1$. However, the restriction of the control actions in set $\boldsymbol{T_{RC}}$ prefers systematically Job 2 that is robust to the perturbations. Note also that the global duration for $k = 15$ and $k = 20$ is better with scenario 3 than with scenario 2. This is due to the partial exploration of the reachability graph and to the approximation of the remaining sequence duration with cost function J_{FC} that provide solutions with no warranty of optimality.

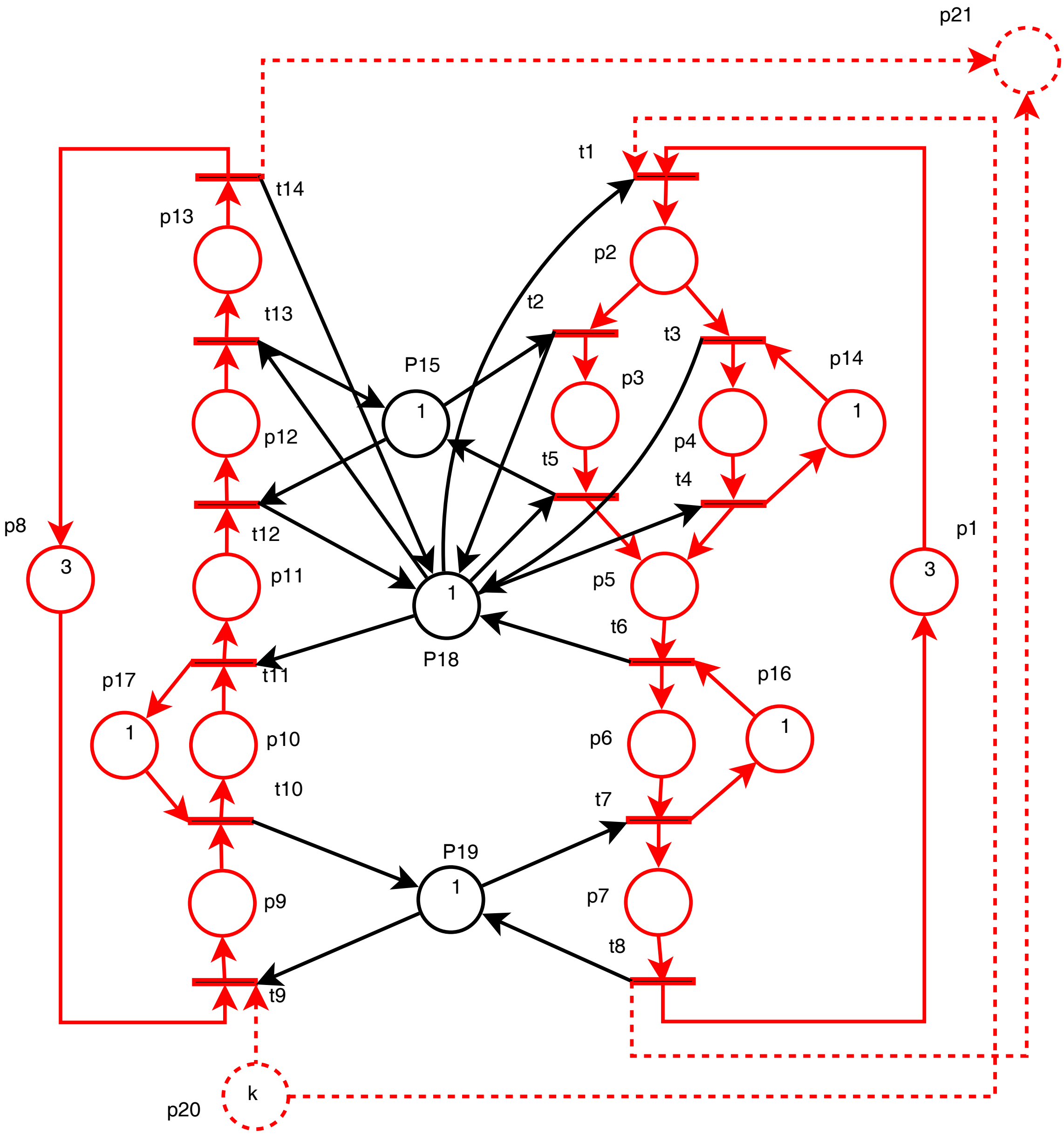

Figure 7. Pcont-SPN2 model of a manufacturing system [28].

4. Discussion

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: $\overline{H}$, which limits the exploration in depth, and H_τ, which 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 4 reports the performance in function of the parameters $\overline{H}$ and H_τ for Pcont-SPN2 with $M_I = 3P_1 + 3P_8 + 1P_{14} + 1P_{15} + 1P_{16} + 1P_{17} + 1P_{18} + 1P_{19} + 5P_{20}$, $M_{ref} = 3P_1 + 3P_8 + 1P_{14} + 1P_{15} + 1P_{16} + 1P_{17} + 1P_{18} + 1P_{19} + 5P_{21}$, 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 4. Performance of Algorithm 3 with respect to parameters $\overline{H}$ and H_τ for PCont-SPN2, sequence duration (TUs) and computational time (s).

$\overline{H}/H_\tau$	1	2	3	4	5	6
5	82 (0.9 s)	86 (0.8 s)	68 (1 s)	68 (1.2 s)	68 (1.3 s)	68 (1.3 s)
10	82 (0.9 s)	86 (0.8 s)	76 (1.5 s)	76 (2.5 s)	76 (4.7 s)	76 (7.9 s)
15	82 (1 s)	86 (0.9 s)	63 (2.1 s)	63 (3.5 s)	63 (9.4 s)	63 (16.1 s)
20	82 (1 s)	86 (0.8 s)	45 (2.6 s)	45 (4.7 s)	45 (10.7 s)	45 (20.6 s)

Note that optimal solutions can be searched in a systematic way instead of using Algorithm 3 considering the extended timed reachability graph [29–31]. Such a graph contains not only the different markings but also the different timed sequences (a given marking can be reached by several sequences with different durations). Table 5 illustrates the rapid increase of the complexity to build such a graph depending on the initial marking $M_I = 3P_1 + 3P_8 + 1P_{14} + 1P_{15} + 1P_{16} + 1P_{17} + 1P_{18} + 1P_{19} + kP_{20}$ when k increases. For each value of k, the number of nodes as the computational time required to compute the graph, are reported for the usual reachability graph and for the timed reachability graph. Table 5 shows that such a method is no longer suitable for large systems. This motivates the proposed approach.

Table 5. Complexity of the exhaustive exploration of control sequences for PCont-SPN2, the number of nodes and the computation time (s).

k	Usual Reachability Graph	Extended Reachability Graph
5	698 (1.4 s)	2208 (106 s)
10	1963 (11 s)	6848 (1827 s)
15	3268 (29 s)	...
20	...	...

5. Conclusions

A method has been proposed to compute control sequences for discrete events systems in uncertain environments. The method uses timed PNs under an earliest firing policy with controllable and uncontrollable transitions as a modeling formalism that is easy to adapt to various problems. The obtained solutions are minimal or near-minimal in duration. Moreover, for each returned solution, the risk to fire uncontrollable transitions is evaluated. Another advantage of the proposed approach is to limit the computational complexity of the algorithm by limiting the part of the reachability graph that is expanded even if the initial marking and reference marking are far from each other, and if deadlocks and dead branches are a priori unknown for the controller. Thanks to the risk evaluation, a robust scheduling becomes computable under some additional assumptions.

In our next works, the research effort will concern, at first, the definition of the cost function that will be improved to provide a more accurate approximation of the remaining time to the reference. The sensitivity of the performance with respect to H will be also studied. We will also include the risk evaluation 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.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Garey, M.R.; Johnson, D.S.; Sethi, R. The complexity of flowshop and jobshop scheduling. *Math. Oper. Res.* **1976**, *1*, 117–129. [CrossRef]
2. Johnson, S.M. Optimal two-and three-stage production schedules with setup times included. *Nav. Res. Logist. Q.* **1954**, *1*, 61–68. [CrossRef]
3. Baker, K.R.; Trietsch, D. *Principles of Sequencing and Scheduling*; John Wiley & Sons: Hoboken, NJ, USA, 2009.
4. Lopez, P.; Roubellat, F. *Production Scheduling*; ISTE: Arlington, VA, USA, 2008.
5. Leung, J.Y. *Handbook of Scheduling: Algorithms, Models, and Performance Analysis*; Chapman & Hall/CRC Computer & Information Science Series: New Delhi, India, 2004; ISBN 9781584883975.
6. Cassandras, C. *Discrete Event Systems: Modeling and Performances Analysis*; Aksen Ass. Inc. Pub.: Homewood, IL, USA, 1993.
7. David, R.; Alla, H. *Petri Nets and Grafcet—Tools for Modelling Discrete Events Systems*; Prentice Hall: London, UK, 1992.
8. Chretienne, P. Timed Petri nets: A solution to the minimum-time-reachability problem between two states of a timed-event-graph. *J. Syst. Softw.* **1986**, *6*, 95–101. [CrossRef]
9. Lee, D.Y.; DiCesare, F. Scheduling flexible manufacturing systems using Petri nets and heuristic search. *IEEE Trans. Robot. Autom.* **1994**, *10*, 123–133. [CrossRef]
10. Sun, T.H.; Cheng, C.W.; Fu, L.C. Petri net based approach to modeling and scheduling for an FMS and a case study. *IEEE Trans. Ind. Electron.* **1994**, *41*, 593–601.
11. Reyes-Moro, A.; Kelleher, H.H.G. Hybrid Heuristic Search for the Scheduling of Flexible Manufacturing Systems Using Petri Nets. *IEEE Trans. Robot. Autom.* **2002**, *18*, 240–245. [CrossRef]
12. Xiong, H.H.; Zhou, M.C. Scheduling of semiconductor test facility via Petri nets and hybrid heuristic search. *IEEE Trans. Semicond. Manuf.* **1998**, *11*, 384–393. [CrossRef]
13. Jeng, M.D.; Chen, S.C. Heuristic search approach using approximate solutions to Petri net state equations for scheduling flexible manufacturing systems. *Int. J. FMS* **1998**, *10*, 139–162.
14. Wang, Q.; Wang, Z. Hybrid Heuristic Search Based on Petri Net for FMS Scheduling. *Energy Proced.* **2012**, *17*, 506–512. [CrossRef]
15. Zhang, W.; Freiheit, T.; Yang, H. Dynamic scheduling in flexible assembly system based on timed Petri nets model. *Robot. Comput. Integr. Manuf.* **2005**, *21*, 550–558. [CrossRef]
16. Hu, H.; Li, Z. Local and global deadlock prevention policies for resource allocation systems using partially generated reachability graphs. *Comput. Ind. Eng.* **2009**, *57*, 1168–1181. [CrossRef]
17. Abdallah, B.; ElMaraghy, H.A.; ElMekkawy, T. Deadlock-free scheduling in flexible manufacturing systems. *Int J. Prod. Res. Vol.* **2002**, *40*, 2733–2756. [CrossRef]
18. Lei, H.; Xing, K.; Han, L.; Xiong, F.; Ge, Z. Deadlock-free scheduling for flexible manufacturing systems using Petri nets and heuristic search. *Comput. Ind. Eng.* **2014**, *72*, 297–305. [CrossRef]
19. Lefebvre, D.; Leclercq, E. Control design for trajectory tracking with untimed Petri nets. *IEEE Trans. Autom. Control* **2015**, *60*, 1921–1926. [CrossRef]
20. Lefebvre, D. Approaching minimal time control sequences for timed Petri nets. *IEEE Trans. Autom. Sci. Eng.* **2016**, *13*, 1215–1221. [CrossRef]

21. Lefebvre, D. Deadlock-free scheduling for Timed Petri Net models combined with MPC and backtracking. In Proceedings of the IEEE WODES 2016, Invited Session "Control, Observation, Estimation and Diagnosis with Timed PNs", Xi'an, China, 30 May–1 June 2016; pp. 466–471.
22. Lefebvre, D. Deadlock-free scheduling for flexible manufacturing systems using untimed Petri nets and model predictive control. In Proceedings of the IFAC—MIM, Invited Session "DES for Manufacturing Systems", Troyes, France, 28–30 June 2016.
23. Ramchandani, C. Analysis of Asynchronous Concurrent Systems by Timed Petri Nets. Ph.D. Thesis, MIT, Cambridge, MA, USA, 1973.
24. Molloy, M.K. Performance analysis using stochastic Petri nets. *IEEE Trans. Comput. C* **1982**, *31*, 913–917. [CrossRef]
25. Richalet, J.; Rault, A.; Testud, J.; Papon, J. Model predictive heuristic control: Applications to industrial processes. *Automatica* **1978**, *14*, 413–428. [CrossRef]
26. Camacho, E.; Bordons, A. *Model Predictive Control*; Springer: London, UK, 2007.
27. Uzam, M. An optimal deadlock prevention policy for flexible manufacturing systems using Petri net models with resources and the theory of regions. *Int. J. Adv. Manuf. Technol.* **2002**, *19*, 192–208. [CrossRef]
28. Chen, Y.; Li, Z.; Khalgui, M.; Mosbahi, O. Design of a Maximally Permissive Liveness-Enforcing Petri Net Supervisor for Flexible Manufacturing Systems. *IEEE Trans. Aut. Science and Eng.* **2011**, *8*, 374–393. [CrossRef]
29. Berthomieu, B.; Vernadat, F. State Class Constructions for Branching Analysis of Time Petri Nets. In Proceedings of the Ninth International Conference on Tools and Algorithms for the Construction and Analysis of Systems TACAS 2003, Warsaw, Poland, 7–11 April 2003; Springer: New York, NY, USA, 2003; Volume 2619, pp. 442–457.
30. Gardey, G.; Roux, O.H.; Roux, O.F. Using Zone Graph Method for Computing the State Space of a Time Petri Net. In Proceedings of the International Conference on Formal Modeling and Analysis of Timed Systems FORMATS 2003, Marseille, France, 6–7 September 2003; Springer: Berlin/Heidelberg, Germany, 2003; Volume 2791, pp. 246–259.
31. Klai, K.; Aber, N.; Petrucci, L. A New Approach to Abstract Reachability State Space of Time Petri Nets. In Proceedings of the 20th International Symposium on Temporal Representation and Reasoning, Pensacola, FL, USA, 26–28 September 2013.