Approximate analysis of non Markovian stochastic systems with multiple time scale delays

Serge Haddad (haddad@lamsade.dauphine.fr)
LAMSADE, UMR CNRS 7024, Université Paris Dauphine
Place du Maréchal de Lattre de Tassigny, 75775 PARIS Cedex 16, FRANCE

Patrice Moreaux (patrice.moreaux@univ-reims.fr)
LAMSADE and CReSTIC
Université de Reims Champagne-Ardenne, FRANCE

Abstract

We address the problem of transient and steady-state analysis of stochastic discrete event systems which include concurrent activities with multiple time scales finite support distributions (and consequently non Markovian). Rather than computing an approximate distribution of the model (as done in previous methods), we develop an exact analysis of an approximate model. The design of this method leads to a uniform handling for the computation of the transient and steady-state behaviour of the model. We extend a previous result restricted to one time scale in order to handle different time scales. Furthermore, we show that some useful classes of non ergodic systems can be analyzed in an exact way with this method. We have evaluated our algorithms on standard queuing models benchmarks. Our results demonstrate that in most of the cases the solution of the approximate model converges quickly to the solution of the exact model, and in the difficult cases (e.g. an heavy load on the queue) our method is more robust than the previous ones.

1. Introduction

The transient and steady-state analysis of Markovian discrete event systems is now well established with numerous tools at the disposal of the modelers. The main open issue is the reduction of the space complexity induced by this analysis. However in a realistic system, the distribution of the occurrence (or the duration) of some events can not be described by a exponential law (e.g. the triggering of a time-out). Theoretically any "reasonable" distribution is approximated by a phase-type distribution enabling again a Markovian analysis [Cox, 1955b]. Unfortunately the continuous time Markov chain (CTMC) associated to this approximation is so huge that it forbids its analysis (indeed even its construction). Such a phenomenon often occurs when the non exponential distribution has a finite support i.e. when the whole probability mass is included in a finite subset of $\mathbb{R}^+$ (non null Dirac, uniform, etc.); then a good phase-type approximation requires too much stages for its specification.

Hence the research has focused on alternative methods. In the case of a single realization of a non Markovian distribution at any time, successful methods have been proposed [German et al., 1995] both for the transient and steady state analysis, especially in the Stochastic Petri Net (SPN) modelling framework. Let us cite, for instance, the method of supplementary variables [Cox, 1955a, German and Lindemann, 1994] or the method of the subordinated Markov chains [Ajmone Marsan and Chiola, 1987].

The general case (i.e. simultaneous multiple realizations of such distributions) is more intricate. The method of supplementary variables is still theoretically applicable but the required space and the computation time limit its use to very small examples. An alternative approach is described for non null Dirac distributions (i.e. "deterministic" durations) in [Lindemann and Schedler, 1996]. The stochastic process, which is a General State space Markov Process (GSMP) is observed at periodic moments of time ($\{h\Delta \mid h \in \mathbb{N}\}$) and this new process is expressed by a system of integro-differential equations and solved numerically. The steady-state distributions of these processes are identical and, with another computation, one obtains the transient distribution of the original process from some transient distribution of the transformed process. This method has been implemented in the DSPNexpress tool [Lindemann et al., 1999] (but currently for only two concurrent "deterministic" events with same duration).

By imposing conditions on the simultaneous occurrences of concurrent activities, other authors have also designed efficient algorithms [German, 1999, Puliafito et al., 1998, Bobbio and Telek, 2002, Jones and Ciardo, 2001, Horváth et al., 2000] (see the Related Works section for more details).

In a previous work [Haddad et al., 2004] we have proposed a different approach to deal with *multiple concurrent* events with finite support distributions. Moreover, in contrast with other works, our solution does not require specific synchronization between these events such as non overlapped or nested events. The main idea is to define an *approximate* model on which we perform an *exact* analysis. To this end, given a time interval (say Δ) we describe the behaviour of the stochastic model by two components: a CTMC and a discrete time Markov chain (DTMC). During an interval $(h\Delta,(h+1)\Delta)$ the behaviour is driven by the CTMC which corresponds to Markovian events occurring in $(h\Delta,(h+1)\Delta)$. Non Markovian activities are taken into account at $h\Delta$ instants only: the untimed probabilistic changes of state are processed according to a DTMC.

In our approximate process, the Markovian events are in fact exactly modelled since the set $\{h\Delta \mid h \in \mathbb{N}\}$ has a null measure. The approximation comes from non Markovian events: the distribution of a non Markovian event is approximated by a discrete random variable expressing the number of points $h\Delta$ that must be reached before its occurrence. Thus the residual number of points to be met is included in the state of our approximate process. At any moment $h\Delta$, the current residual numbers are decreased and the corresponding events occur when their residues are null.

It is well known that stochastic systems with events having very different time scales often lead to difficulties during numerical transient or steady-state analysis. This is even worse when these events are non Markovian for most of the above mentioned methods. These difficulties mainly arise because we need to study the stochastic process during the largest time scale but with a precision which is driven by the smallest time scale. Thus the resulting state space is generally huge.

In this paper we introduce in contrast an extension of our basic method which efficiently deals with very different non Markovian time scales. We first define different time scales. Then we associate to each non Markovian event a time scale and the residue of the event (when activated) is measured *w.r.t. to its time scale*. For instance, in case of two time scales Δ and $\Delta' = d\Delta$, we describe the behaviour of the approximate process during intervals $(h\Delta,(h+1)\Delta)$ by a CTMC and at times $h\Delta$ with $h \bmod d \neq 0$ by a DTMC related to the events associated to the Δ scale and at times $hd\Delta$ by another DTMC related to both the events associated to the Δ scale or the Δ' scale. This approach avoids the state space explosion of the initial method.

Our approximate process may be analyzed either in transient mode or in steady-state. The transient analysis is done by successively computing the state distribution at the instants $\Delta, 2\Delta, \ldots, h\Delta, \ldots$ applying a transient analysis of the CTMC during an interval Δ (via the uniformization technique [Gross and Miller, 1984]) followed by a "step" of the appropriate DTMC depending on h. In order to smooth the effect of the discretization, we average the distribution upon the last largest interval (with length $D_{\max}$) with a variant of uniformization. Since the steady-state distribution depends on the relative position w.r.t. the points $h\Delta$, the approximate process is not ergodic but is asymptotically periodic. Hence, for the steady-state analysis, one computes the steady-state distribution at the instants $h\Delta$ and then starting from this distribution, one again averages the steady-state distribution upon the interval $D_{\max}$.

Another interest of our approach is related to non ergodic systems with finite support distributions. For instance, a transactional system including batch tasks launched at fixed time of the day is not ergodic. However, it has an asymptotical periodic behaviour, one wants to study. To the best of our knowledge, none of the standard tools allows to deal with this kind of systems. It turns out that our method deals with a significant class of such systems in an *exact* way.

The balance of the paper is the following one. In section 2 we detail our approach. Introducing a standard example, we report empirical evaluations of the method in section 3 including comparisons with the mono-scale solution. In section 4, we show that all the previous methods handle particular cases of the systems that we are able to analyse. To conclude, we present the application of our method to non ergodic systems and we give indications on future developments of our work.

2. The approximate method

The method we develop below allows any number of time scales. However, in order to keep simple notations, we limit the presentation to two time scales. The generalization to any number of time scales is straightforward.

2.1. Principle

As mentioned in the introduction, we define an *approximate* model (say $Y^{(\Delta)}$) of the initial model (say X), on which we perform an *exact* analysis. The main idea is to choose a time interval Δ and to restrict in $Y^{(\Delta)}$ the non Markovian events to only occur at times $t_h = h\Delta$. We then study in an exact way the evolution of the stochastic process $Y^{(\Delta)}$ in each interval (t_h, t_{h+1}) and during the state changes at time t_h . We stress that the starting times of the active non Markovian events are *in no way related*. We obtain such a model $Y^{(\Delta)}$ from a general model with non Markovian finite support distributions as follows. The set of non Marko-

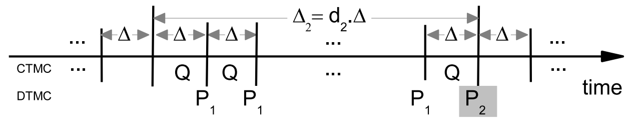

Figure 1. Time decomposition

vian events is split in two subsets E_1 and E_2 depending on the size of their support. For short sizes, the distribution is approximated by a discrete time distribution lying on points $h\Delta$. For large sizes, the distribution is approximated by a discrete time distribution lying on points $h\Delta_2 = hd_2\Delta$ with d_2 integer. We postpone the discussion about this approximation to a later section. Let us note that although Δ seems to be the approximation parameter, the appropriate parameter is the maximum number of points used to express the distribution. Moreover this indicator is the key factor for the complexity of our analysis (see the Experimentations section).

In the approximate process, the Markovian events occur during the intervals $(h\Delta, (h+1)\Delta)$. Non Markovian events always occur in $\{h\Delta \mid h \in \mathbb{N}\}$. Let us describe how they are scheduled. When a non Markovian event of E_i $(i = 1,2)$ is enabled in an interval $(h\Delta, (h+1)\Delta)$ due to the occurrence of a Markovian event, then its approximate distribution is interpreted as the number of points $kd_i\Delta$ that must be met before its occurrence (with $d_1 = 1$). Here we can choose whether we count the next point (i.e. an under-evaluation of the approximated distribution) or not (i.e. an overestimation of the approximated distribution). The impact of this choice will be discussed later. Thus the residual number of points to be met is included in the state of $Y^{(\Delta)}$. At any moment $h\Delta$, the current residual numbers corresponding to events of E_i are decreased if $h \mod d_i = 0$. If some residues are null then the corresponding (non Markovian) events occur with possibly some probabilistic choice in case of conflicts. The occurrence of these events may enable new non Markovian events. Such events are handled similarly except that the next point is always counted since now it corresponds to a complete interval. If we denote by t_h^- (t_h^+) the "time" before (after) the state change in t_h, the process $Y^{(\Delta)}$ is defined by three components:

• the subordinated process in (t_h, t_{h+1}) associated to states at t_h^+ records only exponential events. It is then a CTMC defined by its generator $\mathbf{Q}$.

• the state changes at t_h $(h \mod d_2 \neq 0)$ are defined by a stochastic matrix $\mathbf{P}_1[i,j] = \Pr(Y^{(\Delta)}(h\Delta^+) = j \mid Y^{(\Delta)}(h\Delta^-) = i, h \; mod \; d_2 \neq 0)$.

• the state changes at t_h $(h \mod d_2 = 0)$ are defined by a stochastic matrix $\mathbf{P}_2[i,j] = \Pr(Y^{(\Delta)}(h\Delta^+) = j \mid Y^{(\Delta)}(h\Delta^-) = i, h \; mod \; d_2 = 0)$.

Thus the Markov Regenerative Process (MRGP) $Y^{(\Delta)}$ is fully defined by its initial probability vector $\pi(0)$ and the matrices $\mathbf{P}_1, \mathbf{P}_2, \mathbf{Q}$ (figure 1). These four components depend on Δ since the state space includes the residual number of instants per activated event. We stress however that, even if non Markovian events occurs at $h\Delta$, *all kinds of concurrency are allowed between the activities of the system, contrary to the previous methods*.

It is important to note that in the approximate process, the Markovian events are in fact *exactly* modelled since the set $\{h\Delta \mid h \in \mathbb{N}\}$ has a null measure. The only approximation comes from non Markovian events: their approximate distribution is interpreted as the number of points $kd_i\Delta$ that must be met before their occurrence.

Our approximate process may be analyzed either in transient mode or in steady-state. The proposed analysis is an adaptation of the classical Markovian renewal theory methods.

Transient analysis The transient analysis is done by successively computing the state distribution at the instants $\Delta, 2\Delta, \ldots$ applying a transient analysis of the CTMC during an interval Δ (via the uniformization technique [Gross and Miller, 1984]) followed by a "step" of one of the two DTMCs. In order to smooth the effect of the discretization, we average the distribution on the last largest interval (with a variant of the uniformization).

Let $\pi(h\Delta^+)$ the probability vector of the process $Y^{(\Delta)}$ at time $h\Delta$ after the discrete time change. We have:

$$\pi((kd_2+l)\Delta^+) = \begin{cases} \pi((kd_2+l-1)\Delta^+)(e^{\mathbf{Q}\Delta}\mathbf{P}_1) & (0 < l < d_2) \\ \pi((kd_2+l-1)\Delta^+)(e^{\mathbf{Q}\Delta}\mathbf{P}_2) & (l = 0). \end{cases}$$

Since we want to smooth the discretization effect, we define the approximate value $\pi^{(a)}(hd_2\Delta)$ of $\pi_X(hd_2\Delta)$ as the averaged value of the probabilities of the states of $Y^{(\Delta)}$ in $[t_{hd_2}, t_{(h+1)d_2})$:

$$\begin{aligned} \pi^{(a)}(hd_2\Delta) &\overset{def}{=} \frac{1}{d_2}\sum_{k=0}^{d_2-1}\int_{(hd_2+k)\Delta}^{(hd_2+k+1)\Delta} \pi(\tau)d\tau \qquad (1) \\ &= \frac{1}{\Delta_2}\sum_{k=0}^{d_2-1}\pi((hd_2+k)\Delta^+)\int_0^{\Delta} e^{\mathbf{Q}\tau}d\tau. \end{aligned}$$

Finally, we are in general interested by performance measures defined on the states *of the system*, and not on the states of the stochastic process $Y^{(\Delta)}$. Hence, all components of $\pi^{(a)}(t)$ corresponding to a given state of the original system (i.e. when forgetting the residual numbers) are summed up to compute performance measures (see the example below).

Steady-state analysis Since the steady-state distribution depends on the relative position w.r.t. the points $h\Delta$, the approximate process is not ergodic but asymptotically periodic. We first compute the (approximate) steady-state distribution at times $k\Delta_2$: $\pi^{(\Delta_2)} \overset{def}{=} \lim_{m\to\infty}\pi(m\Delta_2^+)$ where $\pi(m\Delta_2^+) = \pi(md_2\Delta^+)$. This steady-state distribution is computed by a transient analysis stopped when the distribution is stabilized. Since $Y^{(\Delta)}$ is asymptotically periodic with Δ_2 as period, we average the steady-state distributions on an interval $[m\Delta_2,(m+1)\Delta_2)$. Let us introduce the steady-state distribution $\pi^{(\Delta_2,k)}$ at times $m\Delta_2 + k\Delta$ with $0 < k < d_2$: $\pi^{(\Delta_2,k)} \overset{def}{=} \pi^{(\Delta_2,k-1)} \left(e^{\mathbf{Q}\Delta}\mathbf{P}_1\right)^k$ for $0 < k < d_2$ with $\pi^{(\Delta_2,0)} = \pi^{(\Delta_2)}$. Then the approximate steady-state distribution is given by:

$$\pi^{(a)} \overset{def}{=} \frac{1}{\Delta_2} \sum_{k=0}^{d_2-1} \pi^{(\Delta_2,k)} \int_0^{\Delta} e^{\mathbf{Q}\tau} d\tau. \qquad (2)$$

As in the transient case, all components of $\pi^{(a)}$ corresponding to a given state of the system are summed up to compute performance indices.

2.2. Numerical considerations

Formulae (1) and (2) for transient and steady-state probabilities involve vector-matrix products with possibly very large matrices, either $e^{\mathbf{Q}\Delta}$ or $I^{(\Delta)} = \int_0^{\Delta} e^{\mathbf{Q}\tau} d\tau$. Moreover, it is well known (and reported as the "fill in" phenomenon) that, although $\mathbf{Q}$ is generally very sparse (see examples below), $e^{\mathbf{Q}\tau}$ is not sparse at all. Since these matrices are only required through products vector-matrix, the usual approach [Sidje and Stewart, 1999] is to never compute these matrices explicitly but to compute directly the products vector-matrix avoiding the fill in phenomenon. The products of a vector by an exponential matrix are based on the series expansion of the exponential matrix (uniformization) and numerical summation until a required precision level is reached. This is the method that we have implemented.

When we need $e^{\mathbf{Q}\tau}$ we follows the uniformization approach [Stewart, 1994]. If $\mathbf{P}^{(u)} = \mathbf{I} + \frac{1}{u}\mathbf{Q}$ is the uniformized matrix of $\mathbf{Q}$ with rate $u > \max_i\{|q_{ii}|\}$, we have

$$e^{\mathbf{Q}\tau} = \sum_{k\geq 0} e^{-ut} \frac{(u\tau)^k}{k!} \left(\mathbf{P}^{(u)}\right)^k. \qquad (3)$$

For the transient solution (1), $\pi(0)\left(e^{\mathbf{Q}\Delta}\mathbf{P}\right)^h$ is computed iteratively. During the algorithm only one current vector $\mathbf{W}$ indexed on the state space is required (two for the intermediate computations) and for each step we apply the vector-matrix product method to $\mathbf{W}.e^{\mathbf{Q}\Delta}\mathbf{P}$. The computation of $I^{(\Delta)} = \int_0^{\Delta} e^{\mathbf{Q}\tau} d\tau$ is based on (3). By definition, $I^{(\Delta)} = \sum_{k\geq 0} \left[\int_0^{\Delta} e^{-ut} \frac{(ut)^k}{k!} dt\right] \left(\mathbf{P}^{(u)}\right)^k$. An elementary derivation with integration by parts and summation gives us:

$$I^{(\Delta)} = \frac{1}{u} \sum_{k\geq 0} \left[1 - e^{-u\Delta} \sum_{h=0}^{h=k} \frac{(u\Delta)^h}{h!}\right] \left(\mathbf{P}^{(u)}\right)^k \qquad (4)$$

As for $e^{\mathbf{Q}\Delta}$, we only need $I^{(\Delta)}$ through products $\frac{1}{\Delta}.W.I^{(\Delta)}$. We compute these products iteratively to avoid the fill in.

Algorithm 2.1 : Computing the approximate probability distribution (time horizon h)

```
// $\varepsilon$ is the required precision
// $n_0$ is the initial value of $n = D/\Delta$,
//     the subdivision factor of the support
// $\delta$ is the additive term applied to $n$ at each step of the iteration
begin
    $n \leftarrow n_0$
    compute $\pi_n^{(L)}(h)$ and $\pi_n^{(H)}(h)$
    $\mathbf{V} \leftarrow (\pi_n^{(L)}(h) + \pi_n^{(H)}(h))/2$
    repeat
        $n \leftarrow n + \delta$
        $old\mathbf{V} \leftarrow \mathbf{V}$
        $old\pi_n^{(L)}(h) \leftarrow \pi_n^{(L)}(h)$
        $old\pi_n^{(H)}(h) \leftarrow \pi_n^{(H)}(h)$
        compute $\pi_n^{(L)}(h)$ and $\pi_n^{(H)}(h)$
        $d_n^{(L)} \leftarrow \frac{1}{\|\pi_n^{(L)}(h) - old\pi_n^{(L)}(h)\|}$
        $d_n^{(H)} \leftarrow \frac{1}{\|\pi_n^{(H)}(h) - old\pi_n^{(H)}(h)\|}$
        $\mathbf{V} \leftarrow \frac{1}{d_n^{(L)} + d_n^{(H)}} \left(d_n^{(L)}\pi_n^{(L)}(h) + d_n^{(H)}\pi_n^{(H)}(h)\right)$
        $d \leftarrow \| \mathbf{V} - old\mathbf{V} \|$
    until $d \leq \varepsilon$
    // $\mathbf{V}$ is the approximation
end
```

An analogous approach was used in [German, 2001] for steady-state solution of Deterministic Stochastic Petri Nets (DSPN) but restricted to one deterministic event at any given time.

Steady-state solution is obtained in a similar way, steps $\mathbf{W}^{(m+1)} = \mathbf{W}^{(m)}.e^{\mathbf{Q}\Delta}\mathbf{P}$ being computed until convergence.

2.2.1. Choosing an approximate probability vector

Recall that our goal is to give an approximate probability vector $\pi^{(a)}$ either at time t or in steady-state for models with finite support distributions. Let us suppose that the lengths of the short size supports (i.e. related to E_1) are bounded by D. Then the precision of the approximation is given by the parameter n leading to $\Delta = D/n$.

The computation of the approximate $\pi^{(a)}(h)$ is given in Algorithm 2.1 for the transient case with a given time horizon h. The main idea is to compute successive approximation vectors until a given level ε of precision is reached. At each step we increase the precision of the approximation by decreasing the size Δ of the elementary interval. In all the paper, we use the L_1 norm $\| \pi_2 - \pi_2 \| = \sum_i |\pi_1[i] - \pi_2[i]|$ to compare two probability distributions π_1 and π_2 and the precision of the approximation is given by the distance between successive vectors.

The special feature of the algorithm lies in the definition of our approximate vector ($\mathbf{V}$). Recall that for a given n (hence Δ) we can choose between two approximations depending whether we count or not the next $k\Delta$ to be met in the value returned by the discrete random variable corresponding the distribution of a non Markovian event. This gives us two approximate vectors at time $h\Delta$ denoted by $\pi_n^{(L)}(h)$ and $\pi_n^{(H)}(h)$. We observed during our experiments that the sequences $(\pi_n^{(L)}(h))_{n\in\{n_0+k\delta\}}$ and $(\pi_n^{(H)}(h))_{n\in\{n_0+k\delta\}}$ are both convergent but that $\| \pi_n^{(L)}(h) - \pi_n^{(H)}(h) \|_{n\in\{n_0+k\delta\}}$ does not necessarily converge to 0. Moreover, several comparisons have shown that depending on the parameters, one of the two sequences converges faster than the other and that the corresponding limit is closer to the exact distribution (when available) than the other one. These behaviours have led us to define the approximate distribution for n as a weighed sum of $\pi_n^{(L)}(h)$ and $\pi_n^{(H)}(h)$ based on their respective convergence rate as given in the algorithm.

The steady-state approximate distribution is defined similarly except that the successive approximations are computed with the method explained in the Steady-state analysis section.

Note that we compute iteratively the sums (1) and (2) so that we only store two probability vectors during computation and no (full) exponential matrix.

3. Experimentations

We study the application of our method to a queueing system with Poisson arrival (rate λ), two deterministic servers with delays D_1 and D_2 and a K sized buffer. We term such a queue as a $M/D_{1,2}/K$ station.

3.1. The approximate process

In this section, we present the method assuming D_2 is an integer multiple of D_1 ($D2 = d_2 D_1$) for sake of simplicity. The general case is described below. In this particular case, we relate each server to one time scale: Δ for the fast service (delay D_1) and Δ_2 for the slower service (delay D_2). So, we have $D_2/\Delta_2 = D_1/\Delta = n$, that is we count the *same* number of discretization points for each time scale. Recall that we define two approximate processes (H) and (L) corresponding to $n^{(*)} = n$ or $n+1$. A state of each model is $\langle c,(x_1,x_2)\rangle$ where

• c is the number of clients in the queue with $0 \le c \le K$;

• x_1 (resp. x_2) is the fast (D_1) (resp. slow, (D_2)) service state: $0 \le x_i \le n^{(*)}$.

Note that the states do not depend explicitly on the ratio D_2/D_1 which is the major advantage of our refined method. The state $\langle 0,-\rangle$ corresponds to an empty queue and it is the initial state of the system. States are ordered w.r.t. to the numbering function *sno*:

$$sno(c,x) = \begin{cases} \sum_{i=1}^{2} x_i(n^*+1)^{i-1}+1 & \forall c \le 2, \\ (n^*+1)^2+(c-3)(n^*)^2 & \\ \quad +\sum_{i=1}^{2}(x_i-1)(n^*)^{i-1}+1 & \forall c > 2. \end{cases}$$

The cardinality of the state-spaces $S_n^{(L)}$ and $S_n^{(H)}$ for a given n is $|S_n^{(*)}| = (n^*+1)^2+(K-2)(n^*)^2$.

K	$n=10$	$n=100$	$n=200$	$n=300$
5	507	41007	162007	363007
10	1112	92012	364012	816012

The table above gives the values of $S_n^{(H)}$ for some parameters.

The computation of the **Q** matrix defined in section 2.1 is straightforward. From each state $\langle c,x\rangle$ (with $c < K$) the only exponential event is the Poisson arrival of a new client (rate λ), thus we have the transitions:

$$\begin{array}{rcl} \langle 0,-\rangle & \xrightarrow{\lambda} & \langle 1,(n^*,0)\rangle, \\ \forall x>0,\ \langle 1,(x,0)\rangle & \xrightarrow{\lambda} & \langle 1,(x,n^*)\rangle, \\ \forall y>0,\ \langle 1,(0,y)\rangle & \xrightarrow{\lambda} & \langle 2,(n^*,y)\rangle, \\ \forall 2<c<K, x>0, y>0,\ \langle c,(x,y)\rangle & \xrightarrow{\lambda} & \langle c+1,(x,y)\rangle. \end{array}$$

The first transition reflects that a client chooses the fastest server when both servers are idle. If $c = K$ there is no transition from $\langle c,x\rangle$.

The $\mathbf{P}_1$ matrix records state changes due to non exponential events at times $h\Delta$ when $h \bmod d_2 \neq 0$. There is no change for the state of the second server. $\mathbf{P}_1$ has only one non null term per row (with $c > 0$) corresponding to the decrease of the remaining first service time which possibly leads to the end of the service followed possibly by the start of a new service when there is at least one waiting client. Then, the transitions of $\mathbf{P}_1$ are:

$$\begin{array}{rcl} \langle 1,(1,0)\rangle & \xrightarrow{1} & \langle 0\rangle, \\ \forall x>1,\ \langle 1,(x,0)\rangle & \xrightarrow{1} & \langle 1,(x-1,0)\rangle, \\ \forall y>0,\ \langle 1,(0,y)\rangle & \xrightarrow{1} & \langle 1,(0,y)\rangle, \\ \forall y>0,\ \langle 2,(1,y)\rangle & \xrightarrow{1} & \langle 1,(0,y)\rangle, \\ \forall 2<c\le K,\ \forall y>0,\ \langle c,(1,y)\rangle & \xrightarrow{1} & \langle c-1,(n^*,y)\rangle, \\ \forall 2\le c\le K, x>1,\ \forall y>0,\ \langle c,(x,y)\rangle & \xrightarrow{1} & \langle c,(x-1,y)\rangle. \end{array}$$

There is no transition from $\langle 0,-\rangle$ ($\mathbf{P}_1[1,1] = 0.0$).

The $\mathbf{P}_2$ matrix records state changes due to non exponential events at times $h\Delta_2$. $\mathbf{P}_2$ has only one non null term per row (with $c > 0$) corresponding to decrease of (both whenever possible) service times and possible (both) ends of service followed by starts of service for waiting clients. Hence, the transitions encoded in $\mathbf{P}_2$ are:

$$\begin{array}{rcl} \langle 1,(1,0)\rangle & \xrightarrow{1} & \langle 0\rangle, \\ \forall x>1,\ \langle 1,(x,0)\rangle & \xrightarrow{1} & \langle 1,(x-1,0)\rangle, \\ \langle 1,(0,1)\rangle & \xrightarrow{1} & \langle 0\rangle, \\ \forall y>1,\ \langle 1,(0,y)\rangle & \xrightarrow{1} & \langle 1,(0,y-1)\rangle, \\ \langle 2,(1,1)\rangle & \xrightarrow{1} & \langle 0\rangle, \\ \forall y>1,\ \langle 2,(1,y)\rangle & \xrightarrow{1} & \langle 1,(0,y-1)\rangle, \\ \forall x>1,\ \langle 2,(x,1)\rangle & \xrightarrow{1} & \langle 1,(x-1,0)\rangle, \\ \forall x>1, y>1,\ \langle 2,(x,y)\rangle & \xrightarrow{1} & \langle 2,(x-1,y-1)\rangle, \end{array}$$

$$\begin{array}{rrcl}
 & \langle 3,(1,1)\rangle & \xrightarrow{1} & \langle 1,(n^*,0)\rangle, \\
\forall x>1, & \langle 3,(x,1)\rangle & \xrightarrow{1} & \langle 2,(x-1,n^*)\rangle, \\
\forall y>1, & \langle 3,(1,y)\rangle & \xrightarrow{1} & \langle 2,(n^*,y-1)\rangle, \\
\forall x>1,y>1, & \langle 3,(x,y)\rangle & \xrightarrow{1} & \langle 3,(x-1,y-1)\rangle, \\
\forall 3<c\leq K, & \langle c,(1,1)\rangle & \xrightarrow{1} & \langle c-2,(n^*,n^*)\rangle, \\
\forall 3<c\leq K,x>1, & \langle c,(x,1)\rangle & \xrightarrow{1} & \langle c-1,(x-1,n^*)\rangle, \\
\forall 3<c\leq K,y>1, & \langle c,(1,y)\rangle & \xrightarrow{1} & \langle c-1,(n^*,y-1)\rangle, \\
\forall 3<c\leq K,x>1,y>1, & \langle c,(x,y)\rangle & \xrightarrow{1} & \langle c,(x-1,y-1)\rangle.
\end{array}$$

There is no transition from $\langle 0,-\rangle$ ($\mathbf{P}_2[1,1]=0.0$).

3.2. Numerical results

All our computations were done on a Pentium-PC 2.6Ghz, 512MB, with Python 2.3.3 [Python team, 2004] and the Numerical [Dubois, 2004] and the sparse [Geus, 2004] packages under SuSE Linux 8.2.

We first computed our approximate transient distribution for various values of K and time horizon t. Results for $K=10$, $d_2=5,50$ and various values of the load factor $\rho=\lambda D_1/2$ are reported in the table below.

	time horizon ($d_2=5/d_2=50$)		
ρ	$10D_2$	$50D_2$	$100D_2$
0.1	40/100	60/100	60/100
1.0	80/160	60/200	100/220
10.0	140/180	120/160	120/160

For $d_2=5$, we observe that the required n to get a stable approximation remains relatively low, although it increases with ρ as in the single time scale case. For $d_2=50$, the minimal n to reach stability is significantly greater than for $n=5$: this could be explained by the fact that the influence of the D_2 "events" on the transient distribution requires sufficient D_2 "cycles" to be perceptible. More results are reported in the section devoted to the comparison with the single scale method. Steady-state numerical experiments exhibits same trends. Due to lack of space, detailed numerical results including performance measures will be available from our Internet home page.

Non integer multiple delays We explain now how we deal with non integer multiple time delays. In the sequel a given interval Δ, or equivalently a given n is fixed. Let us assume that $D_2=n\Delta_2+r$ with $0<r<\Delta_2$. The problem is in fact to approximate a general distribution G by a discrete distribution concentrated on points $h\Delta_2$. Suppose that G has a continuous component F with density f and total probability $p_F\leq 1$ and a discrete component D with total probability $p_D=1-p_F$. We propose the following method to approximate G on points $h.\Delta$:

• the approximation of the continuous component is set to: $P^{(F)}(h\Delta)=\int_{(h-1/2)\Delta}^{(h+1/2)\Delta}f(t)dt$

• let p_x the probability mass of D in x with $h\Delta\leq x<(h+1)\Delta$. Then the amount of probability mass of p_x to the discrete approximation is set to $P^{(D,x)}((h+1)\Delta)=p_x.\frac{x-h\Delta}{\Delta}$ for the point $(h+1)\Delta$ and to $P^{(D,x)}(h\Delta)=p_x.\left(1-\frac{x-h\Delta}{\Delta}\right)$ for the point $h\Delta$.

The continuous part $P^{(F)}$ accounts for the F-probability of the interval centered on $h\Delta$. The discrete part $P^{(D,x)}$ is the D-probability split with weights proportional to the distance of the mass point x w.r.t. the $k.\Delta$ points. If $X(h)$ is the set of mass points of D in $[h\Delta,(h+1)\Delta)$, the discretization P of G is then:

$$P(h\Delta)=\int_{(h-1/2)\Delta}^{(h+1/2)\Delta}f(t)dt+\sum_{x\in X(h-1)\cup X(h)}P^{(D,x)}(h\Delta)\quad(5)$$

In our special case, G is the deterministic distribution with delay $D2=n.\Delta_2+r$ with $0<r<\Delta_2$, its discretization is concentrated on two points: $n.\Delta_2$ with a probability mass $(1-r)/\Delta_2$ and $(n+1)\Delta_2$ with a probability mass r/Δ_2.

3.3. Comparison with the single time scale method

We have also tried to analyse the $M/D_{1,2}/K$ station with the single time scale method [Haddad et al., 2004]. In this case, a state of the approximate process is $\langle c,(x_1,x_2)\rangle$ with $0\leq c\leq K$, $0\leq x_1\leq n^*$ and $0\leq x_2\leq d_2n^*$. We have ordered the states with the function *sno*:

$$sno(c,x)=\begin{cases}\sum_{i=1}^{2}x_i(n^*+1)^{i-1}+1 & \forall c\leq 2,\\ (n^*+1)(d_2n^*+1)+(c-3)d_2(n^*)^2 & \\ \quad+\sum_{i=1}^{2}(x_i-1)(n^*)^{i-1}+1 & \forall c>2.\end{cases}$$

Complexity comparison The cardinality of the state-spaces $S_n^{(L)}$ and $S_n^{(H)}$ is now $|S_n^*|=(n^*+1)(d_2n^*+1)+(K-2)d_2(n^*)^2$. The ratio (r) of the cardinalities of the state spaces for the single time scale and the multiple time scales methods is roughly d_2 (as this could have been predicted). For instance, with $n=100$ and $n^*=n+1$, $r(d_2=10)=9.99, r(d_2=50)=49.95$ and $r(d_2=100)=99.89$.

Computation time ratios are also approximatively linear w.r.t. d_2 when performing the transient analysis. Indeed, for each method we carry out nd_2 times the following sequence of operations: a product $W.e^{Q\Delta}P$, a computation of the integral $I^{(\Delta)}$ and one more vector-matrix product. Due to the sparse structure of the matrices of our example, a vector-matrix product requires roughly $2T$ double floating point products and sums where T is the state space size. The number of sequences of operations may be large (for instance for a $100D_2$ time horizon and n=100, we compute 10^4 sequences). Thus even when we can manage the state space for the first method, the d_2 time complexity ratio forbids its use even for medium sized parameters. Indeed, we have only computed results for $d_2=5$, $\rho=0.1$, $K=10$ and $t=10D_2,50D_2,100D_2$.

Precision of the results If we fix a given computation time, the multiple scales method outperforms the single scale method both in transient and steady-state modes. This is confirmed if we compare the two methods by fixing a given level of precision for the approximate probability vector. Let n_2 (two scales) and n_1 (one scale) the required n to reach this precision. When the convergence is very fast, we found $n_1 \simeq n_2$: for $\varepsilon = 10^{-2}, d_2 = 5, K = 10$ and $\rho = 0.1$ we found $n_1 = n_2 = 80$ for $t = 10D_2$ with a computation time ratio (c.t.r.) of 4.85 and $n_1 = 80, n_2 = 60$ for $t = 50D_2$ (c.t.r. = 10.5). If we set $\rho = 1.0$, the convergence is slower for large t and n_1 becomes larger than n_2: $n_1 = n_2 = 80$ for $t = 10D_2$ (c.t.r.= 5.20), $n_1 = 70, n_2 = 60$ for $t = 50D_2$ (c.t.r. = 7.5) and $n_1 = 130, n_2 = 100$ for $t = 100D_2$ (c.t.r. = 12.8).

Note that in all cases, the computation time ratio is significant, especially for large time horizon: although specific to our computation software environment, we experienced several hours of computation in the worst cases.

4. Related works

Since we deal with systems composed of general and exponential distributions, it is impossible, except for special cases, to derive analytical expressions of the transient or even steady-state distributions of the states. Thus most results are developed on so-called state based models and they involve numerical solution algorithms.

When the system exhibits complex synchronization, the Queueing Network (QN) framework becomes frequently too restrictive and in fact, many works have studied non exponential activities with the help of the non Markovian Stochastic Petri Nets (NMSPN) formalism, some of them being adapted to general distributions and other ones to deterministic distributions only. In this context, there are two main categories of works (see also the introduction for the approach based on GSMPs).

The first family of solutions defines conditions under which i) the underlying stochastic process is a MRGP and ii) the parameters of this MRGP can be derived from the NMSPN definition. In [German, 1999], the author introduces “Cascaded” Deterministic SPN (C-DSPN). A C-DSPN is a DSPN for which *when two or more deterministic transitions (activities) are concurrently enabled they are enabled in the same states*. With the additional constraint that the $(k+1)$th firing time is a multiple of the kth one, it is possible to compute efficiently the probability distribution as we do. In [Puliafito et al., 1998], the authors derive the elements of the MRGP underlying a SPN with general finite support distributions. However, the NMSPN must satisfy the condition that *several generally distributed transitions concurrently enabled must become enabled at the same time* (being able to become disabled at various times). The transient analysis is achieved first in the Laplace transform domain and then by a numerical Laplace anti-transformation. A simpler method is used for the steady-state solution.

The second family of solutions is based on Phase-type distributions, either continuous (CPHD) or discrete (DPHD). In [Bobbio and Telek, 2002], the authors compare the qualities of fitting general distributions with DPHD or CPHD. It is shown that the time step (the scale factor) of DPHD plays a essential role in the quality of the fitting. [Jones and Ciardo, 2001] introduces the Phased Delay SPNs (PDSPN) which mix CPHD and DCPHD (general distributions must have been fitted to such distributions by the modeller). As pointed out by the authors, without any restriction, the transient or steady-state solutions of PDSPN can only be computed by stochastic simulation. However *when synchronization is imposed between firings of a CPHD transition and resamplings of DPHD transitions* the underlying stochastic process is a MRGP and its parameters can be derived from the reachability graph of the PDSPN. The approach of [Horváth et al., 2000] is based on full discretization of the process. The distributions of the transitions are either DPHD or exponential (general distributions must be fitted with DPHD). For an adapted time step δ, all exponential distributions are then discretized as DPHD and the solution is computed through the resulting process which is a DTMC. We note that discretization may introduce simultaneous event occurrences corresponding to end of continuous Markovian activities, an eventuality with zero probability in the continuous setting.

In contrast to our approach, SPN approaches derive the stochastic process underlying the SPN which is then solved, possibly with approximate methods. However, restrictions on the concurrency between generally distributed activities are always imposed in order to design efficient methods for transient or steady-state solutions.

5. Conclusions and perspectives

Main results We have presented a new approximate method for stochastic processes with multiple non Markovian concurrent activities. Contrary to the other methods, we have approximated the model and applied an exact analysis rather than the opposite way to do. The key factor for the quality of this approximation is that the occurrences of Markovian events are not approximated as it would be in a naive discretization process. Furthermore, the design of its analysis is based on robust numerical methods (i.e. uniformization) and the steady-state and transient cases are handled similarly. Our method efficiently handles distributions whose support have different magnitude orders. Furthermore, the examples have shown that our method is more robust w.r.t. the numerical parameters of the model than the tools we have experimented.

Analysis of non ergodic systems We informally illustrate here the usefulness of our method for some classes of non ergodic systems. Let us suppose that we want to analyze a database associated to a library. At any time, interactive research transactions may be activated by local or remote clients. In addition, every day at midnight, a batch transaction is performed corresponding to the update of the database by downloading remote information from a central database. In case of an overloaded database, the previous update may be still active. Thus the new update is not launched. Even if the modeller considers that all the transactions durations are defined by memoryless distributions, this non Markovian model is not ergodic. However applying the current tools for non Markovian models will give the modeller a useless steady-state distribution with no clear interpretation. Instead we can model such a system in an exact way with our hybrid model. In this simple case, the single time scale model is sufficient. Then with a slight adaptation of our method, the modeller may analyse the asymptotic load of its system at different moments of the day in order to manage to additional load due to the batch transaction.

Another application area of our method is the real-time systems domain. Such systems are often composed by periodic tasks and sporadic tasks both with deadlines. With our hybrid model, we can compute the probability of deadline missing tasks.

Future work We are currently extending our method in the following direction. We plane to introduce tensorial expressions for the matrices in order to manage the space complexity associated to systems with many non Markovian concurrent activities. At last, we are currently developing a high-level model associated with an automatic generation of our "low-level" model. This would be the starting point for more involved case studies.

References

[Ajmone Marsan and Chiola, 1987] Ajmone Marsan, M. and Chiola, G. (1987). On Petri nets with deterministic and exponentially distributed firing times. In Rozenberg, G., editor, *Advances in Petri Nets 1987*, number 266 in LNCS, pages 132–145. Springer–Verlag.

[Bobbio and Telek, 2002] Bobbio, A. and Telek, A. H. M. (2002). The scale factor: A new degree of freedom in phase type approximation. In *International Conference on Dependable Systems and Networks (DSN 2002) - IPDS 2002*, pages 627–636, Washington, DC, USA. IEEE C.S. Press.

[Cox, 1955a] Cox, D. R. (1955a). The analysis of non-Markov stochastic processes by the inclusion of supplementary variables. *Proc. Cambridge Philosophical Society (Math. and Phys. Sciences)*, 51:433–441.

[Cox, 1955b] Cox, D. R. (1955b). A use of complex probabilities in the theory of stochastic processes. *Proc. Cambridge Philosophical Society*, pages 313–319.

[Dubois, 2004] Dubois, P. (2004). Numeric Python home page: `http://www.pfdubois.com/numpy/`. and the Numpy community.

[German, 1999] German, R. (1999). Cascaded deterministic and stochastic petri nets. In B. Plateau, W. J. S. and Silva, M., editors, *Proc. of the third Int. Workshop on Numerical Solution of Markov Chains*, pages 111–130, Zaragoza, Spain. Prensas Universitarias de Zaragoza.

[German, 2001] German, R. (2001). Iterative analysis of Markov regenerative models. *Performance Evaluation*, 44:51–72.

[German and Lindemann, 1994] German, R. and Lindemann, C. (1994). Analysis of stochastic Petri nets by the method of supplementary variables. *Performance Evaluation*, 20(1–3):317–335. special issue: Peformance'93.

[German et al., 1995] German, R., Logothesis, D., and Trivedi, K. (1995). Transient analysis of Markov regenerative stochastic Petri nets: A comparison of approaches. In *Proc. of the 6th International Workshop on Petri Nets and Performance Models*, pages 103–112, Durham, NC, USA. IEEE Computer Society Press.

[Geus, 2004] Geus, R. (2004). PySparse home page: `http://www.geus.ch`.

[Gross and Miller, 1984] Gross, D. and Miller, D. (1984). The randomization technique as a modeling tool an solution procedure for transient markov processes. *Operations Research*, 32(2):343–361.

[Haddad et al., 2004] Haddad, S., Mokdad, L., and Moreaux, P. (2004). Non markovian stochastic discrete event systems - a new approach. In *Proc. of the 7th IFAC Workshop on Discrete Event Systems (WODES'04)*, Reims, France. IFAC.

[Horváth et al., 2000] Horváth, A., Puliafito, A., Scarpa, M., and Telek, M. (2000). A discrete time approach to the analysis of non-markovian stochastic petri. In *Proc. of the 11th Int. Conf. on Computer Performance Evaluation. Modelling Techniques and Tools (TOOLS'00)*, number 1786 in LNCS, pages 171–187, Schaumburg, IL, USA. Springer–Verlag.

[Jones and Ciardo, 2001] Jones, R. L. and Ciardo, G. (2001). On phased delay stochastic petri nets: Definition and an application. In *Proc. of the 9th Int. Workshop on Petri nets and performance models (PNPM01)*, pages 165–174, Aachen, Germany. IEEE Comp. Soc. Press.

[Lindemann et al., 1999] Lindemann, C., Reuys, A., and Thümmler, A. (1999). DSPNexpress 2.000 performance and dependability modeling environment. In *Proc. of the 29th Int. Symp. on Fault Tolerant Computing*, Madison, Wisconsin.

[Lindemann and Schedler, 1996] Lindemann, C. and Schedler, G. (1996). Numerical analysis of deterministic and stochastic Petri nets with concurrent deterministic transitions. *Performance Evaluation*, 27–28:576–582. special issue: Proc. of PERFORMANCE'96.

[Puliafito et al., 1998] Puliafito, A., Scarpa, M., and Trivedi, K. (1998). K-simultaneously enable generally distributed timed transitions. *Performance Evaluation*, 32(1):1–34.

[Python team, 2004] Python team (2004). Python home page: `http://www.python.org`.

[Sidje and Stewart, 1999] Sidje, R. and Stewart, W. (1999). A survey of methods for computing large sparse matrix exponentials arising in Markov chains. *Computational Statistics and Data Analysis*, 29:345–368.

[Stewart, 1994] Stewart, W. J. (1994). *Introduction to the numerical solution of Markov chains*. Princeton University Press, USA.