Performance Evaluation of Fischer's Protocol through Steady-State Analysis of Markov Regenerative Processes

Stefano Martina, Marco Paolieri, Tommaso Papini, and Enrico Vicario
Department of Information Engineering, University of Florence, Italy
stefano.martina@stud.unifi.it, {marco.paolieri,tommaso.papini,enrico.vicario}@unifi.it

***Abstract*—Fischer's protocol is a well-known timed mechanism through which a set of processes can synchronize access to a critical section without relying on atomic test-and-set operations, as might occur in a distributed environment or on a low-level computing platform. The protocol is based on a deterministic waiting time that can be defined so as to guarantee that possible interference due to concurrent accesses with random bounded delays be resolved with certainty.**

While protocol correctness descends from firm lower and upper bounds on waiting times and random delays, performance attained in synchronization also depends on continuous distributions of delays. Performance evaluation of a correct implementation thus requires the solution of a non-Markovian model whose underlying stochastic process falls in the class of Markov regenerative processes (MRPs) with multiple concurrent delays with non-exponential duration. Numerical solution of this class of models is to a large extent still an open problem.

We provide a twofold contribution. We first introduce a novel method for the steady-state analysis of MRPs where regenerations are reached in a bounded number of discrete events, which enlarges the class amenable to numerical solution by allowing multiple concurrent timers with non-exponential distributions. The proposed technique is then applied to Fischer's protocol by characterizing the latency overhead due to synchronization, which comprises the first case where performance of the protocol is quantitatively assessed by jointly accounting for firm bounds and continuous distributions of delays.

***Index Terms*—Fischer's protocol, mutual exclusion, Markov Regenerative Process, steady-state, non-Markovian analysis.**

I. Introduction

Fischer's protocol [1] guarantees mutual exclusion of a number of processes competing for the access to a critical section. The protocol assumes the use of a shared memory where only atomic read and write operations are available: mutual exclusion is achieved by imposing firm bounds on the duration of write operations and test delays. The protocol finds application in systems where test-and-set operations are not available, such as low-cost processors of embedded devices [2]. Several qualitative properties of Fischer's protocol (such as mutual exclusion and deadlock freedom) have been verified either through mathematical proofs [3], [4] or real-time model checking algorithms [5].

While the protocol has become a well-known benchmark for the verification of qualitative timing properties using real-time model checkers [6], [5], [7], results on its quantitative analysis are still limited. The work of [8] analyzes a randomized version of the protocol where read and write operations are modeled with exponential or Erlang distributions, and it derives the expected number of processes that proceed along the phases of the protocol, together with analytical bounds on their outcome probability. As noted in [8], the unbounded supports of exponential or Erlang probability density functions (PDFs) do not guarantee mutual exclusion for this randomized version of Fischer's protocol; on the other hand, analytical results are difficult to extend to PDFs with finite supports, and simulation appears as the only alternative.

In fact, most numerical approaches only address the analysis of models where concurrent events are associated with *exponentially distributed* (EXP) timers: in this case, the underlying stochastic process is a continuous-time Markov chain (CTMC), and efficient numerical algorithms that leverage the memoryless property of EXP timers are available [9]. When the model includes *generally distributed* (GEN) timers, i.e., timers with non-exponential PDFs (such as those with bounded supports required to guarantee mutual exclusion in Fischer's protocol), the future evolution depends not only on the current state (as in CTMCs), but also on previous sojourn times.

In this case, the class of the underlying stochastic process is determined by the persistence of GEN timers after discrete events [10]. If GEN timers cannot persist across discrete events, the underlying stochastic process is semi-Markov and the system has only "memory" of sojourn time elapsed in the current state; the *Markov property* [9] is satisfied immediately after each sojourn, when the next state is reached. In Markov renewal theory [9], the corresponding time instants are called *regeneration points*, since the model "loses memory" and probabilistically restarts. When GEN timers can persist to discrete events, the underlying stochastic process is called Markov regenerative (MRP) if regeneration points are reached infinitely often with probability 1.

While transient and steady-state analysis of MRPs have been fully investigated [9], the computation of the *global* and *local kernels* on which their solution relies is more recent and targets only subclasses of MRPs. The work of [11], [12], [13], [14] focused on MRPs where at most one GEN timer is enabled in each state (usually known as "enabling restriction"); in [15], this requirement was relaxed, allowing the numerical computation of MRP kernels for models with multiple GEN timers. This technique is based on the enumeration of *stochastic state classes* [16], which provide the (analytical) joint PDF of GEN

timers after each discrete event; [15] applied the technique to *transient* analysis, while [17] proposed a solution for probabilistic model checking, analyzing quantitative properties for the transient regime of Fischer's protocol. No results have been reported for the application of Markov renewal theory to steady-state analysis of MRP models with multiple concurrent GEN timers (except for the case of deterministic timers [18]).

In this work, we provide a twofold contribution. We first introduce a novel method for steady-state analysis of models with multiple concurrent GEN timers that reach regenerations within a bounded number of discrete events (Sect. III); in contrast with existing results [11], [12], [13], [14], the method can be applied to models where GEN timers overlap their activity periods. We then apply the method to evaluate the performance of a set of processes that synchronize the access to a critical section using Fischer's protocol (Sect. IV); the protocol is formulated with stochastic time Petri nets (STPNs) and combines deterministic waiting times with upper-bounded stochastic delays for write operations, as required for a correct implementation (Sect. II). Conclusions are drawn in Sect. V.

II. A Stochastic Time Petri Net Model of Fischer's Protocol

A. Stochastic time Petri nets

An STPN [16] is defined by a set of *transitions*, representing activities with stochastic duration, and a set of *places*; a *marking* assigns a natural number of *tokens* to each place. Places can serve as *input* or *output* places of a transition: when the marking assigns at least one token to each input place, the transition is enabled; after its firing, one token is removed from each input place and one token is added to each output place. A transition samples a *time to fire* when it becomes enabled; as in discrete event systems, the transition with minimum time to fire is the next event and its firing enables, disables, or restarts other events by removing tokens from input places and adding tokens to output places. In addition, arbitrary constraints on token counts (e.g., $?\texttt{place}_1 > \texttt{place}_2$) can limit the enabling of a transition, and *update functions* of the form "$\texttt{place} \leftarrow$ *expression*" can specify additional updates of token counts after its firing.

Definition 1 (Syntax)**.** A stochastic time Petri net is a tuple $\langle P, T, A^-, A^+, B, U, EFT, LFT, F, W\rangle$ where: P and T are disjoint sets of places and transitions; $A^- \subseteq P \times T$ and $A^+ \subseteq T \times P$ are the precondition and post-condition relations; B and U associate each transition t with an enabling function $B(t)\colon \mathbb{N}^P \to \{\mathit{true}, \mathit{false}\}$ and with an update function $U(t)\colon \mathbb{N}^P \to \mathbb{N}^P$. For each transition $t \in T$, the STPN also specifies an earliest firing time $EFT(t) \in \mathbb{Q}_{\geqslant 0}$, a latest firing time $LFT(t) \in \mathbb{Q}_{\geqslant 0} \cup \{\infty\}$ such that $EFT(t) \leq LFT(t)$, a distribution F_t such that $x < EFT(t) \Rightarrow F_t(x) = 0$ and $x > LFT(t) \Rightarrow F_t(x) = 1$, and a weight $W(t) \in \mathbb{R}_{>0}$.

A place p is said to be an *input* or *output* place for a transition t if $(p,t) \in A^-$ or $(t,p) \in A^+$, respectively. Following the usual terminology of stochastic Petri nets, a transition t is called *immediate* (IMM) if $EFT(t) = LFT(t) = 0$ and *timed* otherwise; a timed transition is called *exponential* (EXP) if $F_t(x) = 1 - e^{-\lambda x}$ for some rate $\lambda \in \mathbb{R}_{>0}$, or *general* (GEN) if its time to fire is distributed according to a non-exponential distribution; as a special case, a GEN transition t is *deterministic* (DET) if $EFT(t) = LFT(t) > 0$. For each transition t, we assume that F_t can be expressed as the integral function of a PDF f_t, i.e., $F_t(x) = \int_0^x f_t(y)\, dy$.

Given an STPN, a *marking* $m \in \mathbb{N}^P$ assigns a natural number of tokens to each place of the net. A transition t is *enabled* by m if m assigns at least one token to each of its input places and the enabling function $B(t)(m)$ evaluates to *true*; the set of transitions enabled by m is denoted as $E(m)$.

Definition 2 (State)**.** The state of an STPN is a pair $\langle m, \vec{\tau}\rangle$ where $m \in \mathbb{N}^P$ is a marking and $\vec{\tau} \in \mathbb{R}_{\geqslant 0}^{E(m)}$ is a real-valued vector assigning a *time to fire* $\vec{\tau}(t) \in \mathbb{R}_{\geqslant 0}$ to each enabled transition $t \in E(m)$.

Definition 3 (Semantics)**.** Given an initial marking m_0, an *execution* of the STPN is represented by a (finite or infinite) path $\omega = s_0 \xrightarrow{\gamma_1} s_1 \xrightarrow{\gamma_2} s_2 \xrightarrow{\gamma_3} \cdots$ such that: $s_0 = \langle m_0, \vec{\tau}_0\rangle$ is the initial state, where the time to fire $\vec{\tau}_0(t)$ of each enabled transition $t \in E(m_0)$ is sampled according to the distribution F_t; $\gamma_i \in T$ is the ith transition fired along ω and $s_i = \langle m_i, \vec{\tau}_i\rangle$ is the state after its firing. In each state s_i:

- The next transition γ_{i+1} is selected from the set of enabled transitions with minimum time to fire according to a discrete distribution given by weights: if $E_{\text{min}} = \arg\min_{t \in E(m_i)} \vec{\tau}_i(t)$, then $t \in E_{\text{min}}$ is selected with probability $p_t = W(t)/(\sum_{u \in E_{\text{min}}} W(u))$.
- After the firing of γ_{i+1}, the new marking m_{i+1} is derived by (1) removing a token from each input place of γ_{i+1}, (2) adding a token to each output place of γ_{i+1}, and (3) applying the update function $U(\gamma_{i+1})$. A transition t enabled by m_{i+1} is said to be *persistent* if it is distinct from γ_{i+1} and it is enabled also by m_i and by the intermediate markings after steps (1) and (2); otherwise, t is said to be *newly enabled*.
- For each newly enabled transition t, the time to fire $\vec{\tau}_{i+1}(t)$ is sampled according to the distribution F_t; for each persistent transition t, the time to fire in s_i is reduced by the sojourn time in the previous marking, i.e., $\vec{\tau}_{i+1}(t) = \vec{\tau}_i(t) - \vec{\tau}_i(\gamma_{i+1})$.

B. A stochastic model of Fischer's protocol

Fig. 1 illustrates an STPN model of three processes P_1, P_2, P_3 (the same scheme can be extended to any number of processes) that synchronize their access to a mutually-exclusive critical section using Fischer's protocol. Each horizontal section represents a process, and place `id` encodes a shared variable *id* that can take the values $\{0, 1, 2, 3\}$. In the protocol specification [1], when $\mathit{id} = 0$, each process P_i can attempt to access the critical section by (1) writing its identifier i to *id*, (2) waiting for a time greater than the maximum write time of any other process, and (3) reading *id* again to ensure that $\mathit{id} = i$. If $\mathit{id} \neq i$, the process must wait until $\mathit{id} = 0$ before a new attempt. In the model, each process P_i eventually leaves

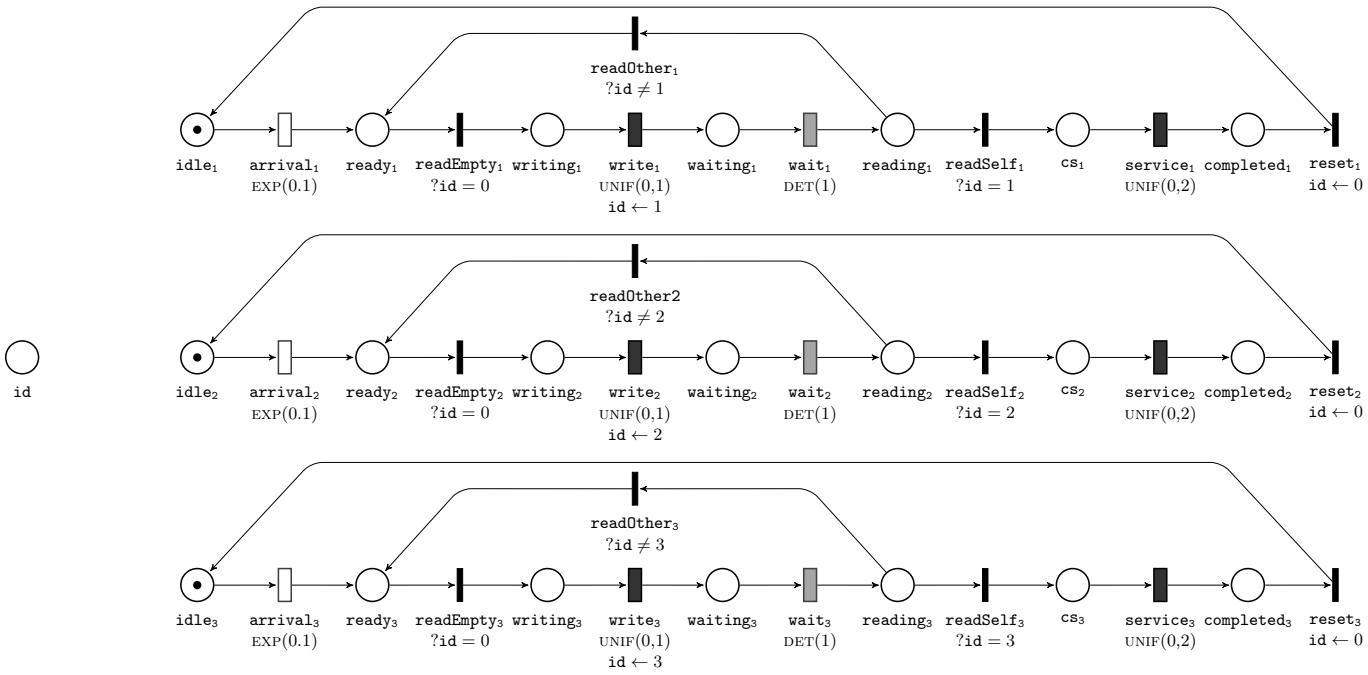

Fig. 1: STPN model of three processes accessing a critical section with Fischer's mutual exclusion protocol.

its idle state through transition $\texttt{arrival}_i$ (EXP with rate 0.1) and enters the contention by reaching place $\texttt{writing}_i$ as soon as $\texttt{id} = 0$ (IMM transition $\texttt{readEmpty}_i$ with enabling function $?\texttt{id} = 0$); it then sets the shared variable to its own identifier (update function $\texttt{id} \leftarrow i$) at the end of a write operation (transition $\texttt{write}_i$, with duration uniformly distributed over $[0, 1]$), and sojourns in a waiting state (place $\texttt{waiting}_i$) for a time greater than the maximum time that any process can spend writing to id (transition $\texttt{wait}_i$, DET equal to 1). When the wait completes, process P_i reads id again to ensure that its write has finished last (place $\texttt{reading}_i$): if $\texttt{id} \neq i$, the control goes back to the initial state of contention $\texttt{ready}_i$ (IMM transition $\texttt{readOther}_i$); whereas, if the shared variable is still equal to the process identifier (i.e., $\texttt{id} = i$), P_i enters the critical section $\texttt{cs}_i$ (IMM transition $\texttt{readSelf}_i$), performs its service (transition $\texttt{service}_i$, uniform over $[0, 2]$), and then resets id (IMM transition $\texttt{reset}_i$), returning idle.

Mutual exclusion is enforced through the use of concurrent GEN timers with bounded supports: the time spent in the waiting phase by each process is greater than the maximum writing time of any other process [3].

III. Steady-State Analysis of MRPs with Stochastic State Classes

A. Steady-state analysis of MRPs

An MRP [9] is a continuous-time stochastic process embedding a Markov renewal sequence that defines a sequence of *regeneration points* where future behavior depends on past history only through the current regeneration state.

Definition 4 (Markov renewal sequence)**.** Given a probability space $(\Omega, \mathcal{A}, P)$ and a finite set $\mathcal{R}$, the sequence of random variables $\{(X_n, T_n),\ n \in \mathbb{N}\}$ with $X_n \in \mathcal{R}$, $T_n \in \mathbb{R}_{\geqslant 0}$, and $0 = T_0 \leq T_1 \leq \cdots \leq T_n$, is a time-homogeneous *Markov renewal sequence* with state space $\mathcal{R}$ and kernel $G_{ik}(t)$ if

$$P\{X_{n+1} = k,\, T_{n+1} - T_n \leq t \mid X_0, \ldots, X_n, T_1, \ldots, T_n\} = P\{X_1 = k,\, T_1 \leq t \mid X_0 = i\} := G_{ik}(t)$$

for all $n \in \mathbb{N}$, $i, k \in \mathcal{R}$, and $t \in \mathbb{R}_{\geqslant 0}$.

Definition 5 (Markov regenerative process)**.** A stochastic process $\{M(t),\ t \geq 0\}$ defined on the probability space $(\Omega, \mathcal{A}, P)$ and taking values in $\mathcal{M}$ is said to be *Markov regenerative* if there exists a Markov renewal sequence $\{(X_n, T_n),\ n \in \mathbb{N}\}$ with finite state space $\mathcal{R}$ such that

- for each $n \in \mathbb{N}$, T_n is a stopping time for $\{M(t),\ t \geq 0\}$ and $X_n \in \mathcal{R}$ is determined by $\{M(u),\, 0 \leq u \leq T_n\}$;
- for each $n \in \mathbb{N}$, $m \geq 1$, $0 \leq t_1 \leq \cdots \leq t_m$ and positive function f defined on $\mathcal{M}^m$,

$$E\{f(M(T_n + t_1), \ldots, M(T_n + t_m)) \mid X_n, M(u)\, \forall u \leq T_n\} = E\{f(M(t_1), \ldots, M(t_m)) \mid X_n\}\,.$$

Steady-state behavior of an MRP can be analyzed through its global kernel $G_{ik}(t) := P\{X_1 = k, T_1 \leq t \mid X_0 = i\}$ and local kernel $L_{ij}(t) := P\{M(t) = j, T_1 > t \mid X_0 = i\}$ for all regeneration states $i, k \in \mathcal{R}$, and for all states $j \in \mathcal{M}$. The global kernel $G_{ik}(t)$ characterizes the occurrence of regenerations, while the local kernel $L_{ij}(t)$ characterizes the behavior between subsequent regenerations: if $i \in \mathcal{R}$ is the initial regeneration state, $G_{ik}(t)$ gives the probability that the next regeneration is reached within time t on $k \in \mathcal{R}$, while $L_{ij}(t)$ gives the probability that the next regeneration has not been reached at time t and the current state is $j \in \mathcal{M}$.

From the local and global kernels, steady-state probabilities of an MRP can be derived as follows [9], [19].

Theorem 1. *Let $\{M(t),\ t \geq 0\}$ be an MRP with kernels $G_{ik}(t)$ and $L_{ij}(t)$ for all $i,k \in \mathcal{R}$, $j \in \mathcal{M}$, $t \in \mathbb{R}_{\geqslant 0}$ and aperiodic, irreducible and recurrent non-null Markov renewal sequence. Then, for all $j \in \mathcal{M}$,*

$$\lim_{t\to\infty} P\{M(t) = j\} = \frac{\sum_{i\in\mathcal{R}} \pi_i \alpha_{ij}}{\sum_{i\in\mathcal{R}, m\in\mathcal{M}} \pi_i \alpha_{im}}$$

where: $\alpha_{ij} := \int_0^\infty L_{ij}(t)\,dt$ is the expected time spent in $j \in \mathcal{M}$ before a new regeneration after $i \in \mathcal{R}$; $\vec{\pi}$ is the steady-state solution of the DTMC embedded at regeneration points, i.e., $\sum_{i\in\mathcal{R}} \pi_i = 1$ and $\vec{\pi} = \vec{\pi}\,\mathbf{G}$, with $\mathbf{G} := \lim_{t\to\infty} \mathbf{G}(t)$.

B. Derivation of α_{ij} and G_{ik} using stochastic state classes

Using Theorem 1, steady-state probabilities of a model can be derived from the values of α_{ij} and G_{ik} for the underlying MRP. To compute these quantities, we leverage *stochastic state classes* [16], which have been previously applied to the evaluation of local and global kernels for transient analysis of MRPs with multiple concurrent GEN timers [15]. Each class characterizes the state of an STPN after a discrete event, encoding the marking and the support and joint distribution of remaining times to fire of GEN transitions.

Definition 6 (Stochastic state class)**.** A *stochastic state class* is a tuple $\Sigma = \langle m, D_{\vec{\tau}}, f_{\vec{\tau}} \rangle$ where: $m \in \mathbb{N}^P$ is a marking; $f_{\vec{\tau}}$ is the PDF (immediately after the previous firing) of the random vector $\vec{\tau} = (\tau_1, \ldots, \tau_n)$ of remaining times to fire of transitions enabled by m; $D_{\vec{\tau}} \subseteq \mathbb{R}^n$ is the support of $f_{\vec{\tau}}$.

Given a stochastic state class Σ, the state PDF conditioned on the firing of a transition γ is given by the *successor* of Σ.

Definition 7 (Succession relation)**.** We say that, with probability μ, $\Sigma' = \langle m', D_{\vec{\tau}'}, f_{\vec{\tau}'} \rangle$ is the *successor* of $\Sigma = \langle m, D_{\vec{\tau}}, f_{\vec{\tau}} \rangle$ through $\gamma \in E(m)$, and we write $\Sigma \overset{\gamma,\mu}{\Longrightarrow} \Sigma'$, if, given that the marking of the STPN is m and $\vec{\tau}$ is a random vector distributed over $D_{\vec{\tau}}$ according to $f_{\vec{\tau}}$, then: transition γ has nonzero probability μ of firing; if γ fires, its firing yields marking m' and, conditioned on this event, the new vector of times to fire $\vec{\tau}'$ is distributed over $D_{\vec{\tau}'}$ according to $f_{\vec{\tau}'}$.

If all transitions in the model are associated with a deterministic or expolynomial distribution [11] with possibly bounded support, the relation $\overset{\gamma,\mu}{\Longrightarrow}$ can be enumerated through a symbolic calculus that integrates the derivation of DBM supports and of the analytical form of joint PDFs [16].

We enumerate stochastic state classes from initial classes corresponding to all reachable regeneration states, which specify the marking and a vector of (deterministic) elapsed times for GEN timers [17]. The enumeration proceeds along any sequence of transition firings with nonzero probability until a new regeneration is detected (i.e., GEN timers are newly enabled or enabled since a deterministic time), resulting in a tree structure where nodes are associated with classes, edges with firing probabilities, and leaves with regenerations.

To compute G_{ik} for all $k \in \mathcal{R}$, we enumerate the tree of stochastic state classes from regeneration $i \in \mathcal{R}$ until a new regeneration is reached on every leaf. Each path $\Sigma_0 \overset{\gamma_1,\mu_1}{\Longrightarrow} \Sigma_1 \overset{\gamma_2,\mu_2}{\Longrightarrow} \cdots \overset{\gamma_n,\mu_n}{\Longrightarrow} \Sigma_n$ from the root Σ_0 to a leaf node Σ_n provides the probability $p_{reach}(\Sigma_n) = \prod_{i=1}^n \mu_i$ that Σ_n is reached from Σ_0. We compute $G_{ik} := \lim_{t\to\infty} G_{ik}(t) = P\{X_1 = k \mid X_0 = i\}$ for all $k \in \mathcal{R}$ by summing up reaching probabilities of leaves that correspond to regeneration k:

$$G_{ik} = \sum_{\substack{\Sigma\in\textsc{Leaves}(i) \text{ s.t.} \\ \Sigma \text{ has regeneration } k}} p_{reach}(\Sigma)\,.$$

The expected time α_{ij} spent in state $j \in \mathcal{M}$ after each occurrence of regeneration $i \in \mathcal{R}$, and before a new regeneration, is equal to the expected time spent in any inner node with marking j in the tree enumerated from i:

$$\alpha_{ij} = \sum_{\substack{\Sigma\in\textsc{Inner}(i) \text{ s.t.} \\ \Sigma \text{ has marking } j}} p_{reach}(\Sigma)\, SJ(\Sigma)$$

where $\textsc{Inner}(i)$ is the set of inner nodes in the tree and $SJ(\Sigma)$ is the expected sojourn time of Σ. In turn, according to Def. 6, if $\Sigma := \langle m, D, f \rangle$, the expected sojourn time $SJ(\Sigma)$ can be computed by partitioning the support D of f into a finite set of sub-zones $D_t := \{\vec{\tau} \in D \mid \vec{\tau}(t) \leq \vec{\tau}(u)\ \forall u \in E(m)\}$, one for each transition $t \in E(m)$ enabled by m, such that the time to fire $\vec{\tau}(t)$ of t is minimum in D_t and thus, according to the race semantics of STPNs, equal to the sojourn time in Σ. Hence, $SJ(\Sigma) = \sum_{t\in E(m)} \int_{D_t} x\, f(x)\, dx$.

IV. Evaluation

The solution technique presented in Sect. III allows us to compute steady-state probabilities for the STPN model of Fischer's protocol introduced in Sect. II. The solution was implemented using functions provided by the ORIS API and made available both as an API primitive and as a new analysis engine in the ORIS Tool (`www.oris-tool.org`) [20].

The new feature was applied to characterize the *latency* experienced by a process in the access to the critical section. Latency for the ith process is computed as the steady-state probability that the process is in any of the states visited after leaving the idle state and before reaching the critical section, i.e., $Latency_i := P\{\texttt{ready}_i + \texttt{writing}_i + \texttt{waiting}_i + \texttt{reading}_i = 1\}$. Model parameters were varied (uniformly for all processes, or only for processes other than i) so as to assess sensitivity with respect to: the arrival rate λ, which determines the sojourn time in the idle state; the number n of concurrent processes; the maximum duration st of the sojourn time in the critical section; the maximum writing time wt, which is assumed to be equal to the waiting time after a write operation. For example, $\lambda = 0.1$, $n = 3$, $st = 2$, $wt = 1$ in Fig. 1.

All the experiments were carried out on a machine with two Intel Xeon E5640 processors (2.66 GHz, 12 MB cache, 4 cores) and 32 GB of RAM; computation times were almost insensitive to any parameter variation, except for the number of processes n: on average, each experiment takes 0.26 s to complete with $n = 3$, 13 s with $n = 4$, and 930 s with $n = 5$.

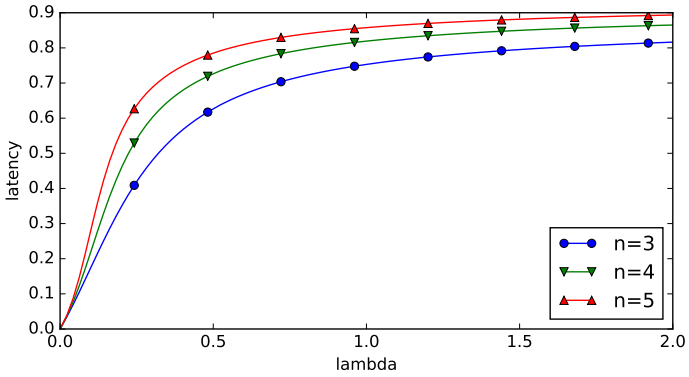

Fig. 2: $Latency_1$ as a function of the arrival rate λ for different values of the number n of processes.

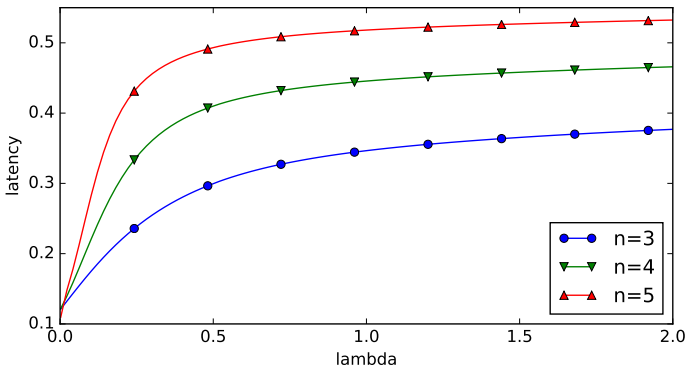

Fig. 3: $Latency_1$ as a function of the arrival rate λ of other processes, for constant arrival rate $\lambda_1 = 0.1$ and varying number n of processes.

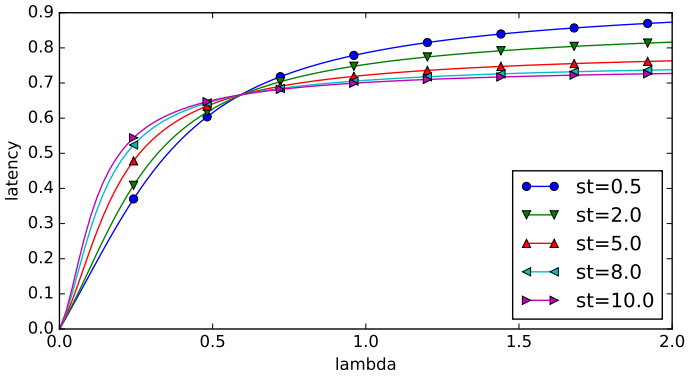

Fig. 4: Access latency as a function of λ for $n = 3$ processes, for varying values of maximum sojourn time in critical section.

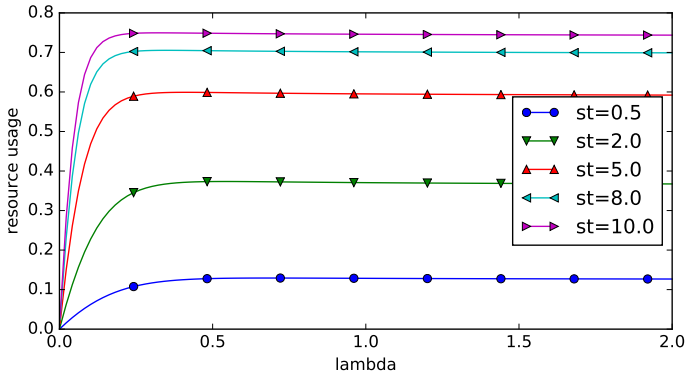

Fig. 5: Probability that the critical section is occupied, as a function of the arrival rate λ, for $n = 3$ processes and different values of the maximum sojourn time.

A. *Latency with respect to λ and n*

Fig. 2 plots latency of process P_1 as a function of the arrival rate λ (equal for all processes) and number of processes n. Latency increases with λ due to a twofold mechanism: on the one hand, an increase of λ reduces the time spent by P_1 in the idle state, thus increasing the number of times that the process engages the access protocol; on the other hand, it also increases the contention with other processes, resulting in a larger time spent by P_1 in latency states for each access. The latter mechanism also explains the increase of latency observed when the number of concurrent processes increases.

Fig. 3 plots the latency of process P_1 when the arrival rate λ_1 of P_1 is kept equal to 0.1 and the rate λ of other processes is increased, for different values of n. This setting isolates the effect on latency induced by increased concurrency, abstracting from the idle time of P_1. For this reason, the increase in latency with respect to λ is lower than in Fig. 2.

B. *Latency with respect to service times*

Fig. 4 plots latency of process P_1 as a function of the arrival rate λ (equal for all processes), for different values of the maximum sojourn time st in the critical section. For low values of λ, the degree of concurrency is low and latency grows almost linearly; in contrast, the use of the resource saturates for high values of λ.

For low arrival rates λ, higher service times st result in higher latency: when arrivals are rare, interference and blocking will arise only in the presence of long sojourn times, which will increase both the probability that the critical section is already in use when process P_1 becomes ready, and the expected number of contending processes. Whereas, for higher values of λ, an increase of the sojourn time st in the critical section will result in lower latency: when λ grows, the usage of the resource is saturated, and contentions occur at each access; a higher duration of the sojourn time st then reduces the proportion of time spent in the access to the critical section.

To support this interpretation, Fig. 5 plots the resource usage, calculated as the probability that at least one process is in the critical section (due to symmetry, cumulative resource usage is n times that of each process). Increased sojourn times in the critical section tend to anticipate saturation of the resource and result in higher usage. In fact, under a high degree of concurrency, the time required to complete a contention will be close to the worst-case writing time plus the waiting time; asymptotic resource usage can thus be approximated as $E[ST]/(E[ST] + 2\,wt)$ where $E[ST] = st/2$ is the expected value of the sojourn time.

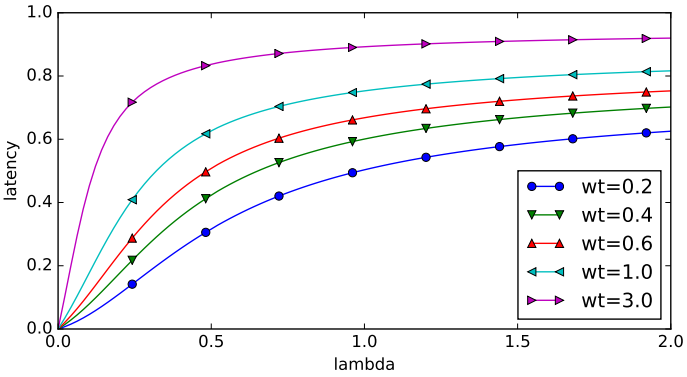

Fig. 6: Latency of process P_1 as a function of the arrival rate λ for different values of the maximum writing time.

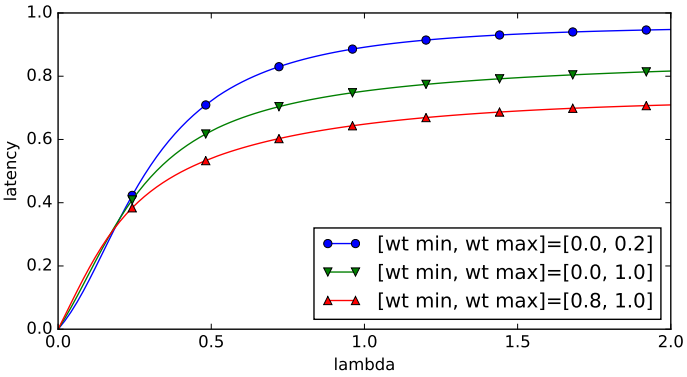

Fig. 7: Access latency as a function of the arrival rate λ for $n = 3$ processes in a non-homogeneous setting where only the writing time distribution of the first process varies.

C. Latency with respect to writing times

Fig. 6 plots latency of process P_1 as a function of the arrival rate λ (equal for all processes), for different values of the maximum writing time wt (also equal to the waiting time). As intuitive, higher writing times result in higher latency.

In contrast, Fig. 7 analyzes the latency of P_1 when only its writing time is varied and $wt = 1$ for other processes. The results highlight a subtle mechanism of the Fischer's protocol, where the first write operation excludes other processes from the contention, but access to the critical section is granted to the *last* process that completes its write operation. Should a process write quickly, trying to avoid contentions, or slowly, trying to finish last? We analyzed the impact of a lower expected value (uniform distribution over $[0, 0.2]$) or a higher one (uniform distribution over $[0.8, 1]$) for the writing time of P_1, with respect to the expected value of other processes (uniform distribution over $[0, 1]$). Results show that, for low arrival rates, fast write operations yield a small gain in latency, as contentions are rare and it is more advantageous to exclude other processes; however, as the arrival rate grows, contentions occur more frequently and slow write operations become more advantageous, since they result in higher probability of winning contentions.

V. Conclusions

In this paper, we provided a twofold contribution. On the one hand, the solution technique developed in Sect. III advances the state of the art of non-Markovian analysis by allowing steady-state analysis of models with multiple concurrent timers with non-exponential durations. On the other hand, the results of Sect. IV demonstrate the applicability of the proposed technique and provide a quantitative evaluation of performance for Fischer's mutual exclusion protocol. As a major element of novelty, this evaluation considers delays distributed over bounded supports, as needed for a correct implementation of the protocol.

References

[1] M. Fischer, "Re: Where are you?" Email sent to Leslie Lamport, ARPANET message number 8506252257.AA07636@YALE-BULLDOG.YALE.ARPA (47 lines), June 25, 1985 18:56:29 EDT.

[2] P. A. Buhr, D. Dice, and W. H. Hesselink, "High-performance N-thread software solutions for mutual exclusion," *Concurrency and Computation: Practice and Experience*, vol. 27, no. 3, pp. 651–701, 2015.

[3] L. Lamport, "A Fast Mutual Exclusion Algorithm," *ACM Trans. Comput. Syst.*, vol. 5, no. 1, pp. 1–11, Jan. 1987.

[4] N. Lynch and N. Shavit, "Timing-based mutual exclusion," in *Real-Time Systems Symposium, 1992*. IEEE, 1992, pp. 2–11.

[5] G. Behrmann, A. David, K. G. Larsen, J. Håkansson, P. Pettersson, W. Yi, and M. Hendriks, "UPPAAL 4.0," in *Int. Conf. on the Quantitative Evaluation of Systems (QEST'06)*, 2006, pp. 125–126.

[6] M. Bozga, C. Daws, O. Maler, A. Olivero, S. Tripakis, and S. Yovine, "Kronos: A Model-Checking Tool for Real-Time Systems," in *Int. Conf. on Computer Aided Verification (CAV'98)*, 1998, pp. 546–550.

[7] R. Kindermann, T. A. Junttila, and I. Niemelä, "SMT-Based Induction Methods for Timed Systems," in *Int. Conf. on Formal Modeling and Analysis of Timed Systems (FORMATS'12)*, 2012, pp. 171–187.

[8] E. Gafni and M. Mitzenmacher, "Analysis of timing-based mutual exclusion with random times," *SIAM Journal on Computing*, vol. 31, no. 3, pp. 816–837, 2001.

[9] V. Kulkarni, *Modeling and analysis of stochastic systems*. Chapman & Hall, 1995.

[10] G. Ciardo, R. German, and C. Lindemann, "A characterization of the stochastic process underlying a stochastic Petri net," *IEEE Trans. Softw. Eng.*, vol. 20, no. 7, pp. 506–515, Jul. 1994.

[11] K. S. Trivedi and R. Sahner, "SHARPE at the Age of Twenty Two," *SIGMETRICS Perform. Eval. Rev.*, vol. 36, no. 4, pp. 52–57, Mar. 2009.

[12] A. Zimmermann, "Modeling and evaluation of stochastic Petri nets with TimeNET 4.1," in *International Conference on Performance Evaluation Methodologies and Tools (VALUETOOLS)*, 2012, pp. 54–63.

[13] R. German, "Iterative analysis of Markov regenerative models," *Perform. Eval.*, vol. 44, no. 1-4, pp. 51–72, 2001.

[14] E. G. Amparore and S. Donatelli, "A component-based solution for reducible Markov regenerative processes," *Perform. Eval.*, vol. 70, no. 6, pp. 400–422, 2013.

[15] A. Horváth, M. Paolieri, L. Ridi, and E. Vicario, "Transient analysis of non-Markovian models using stochastic state classes," *Perform. Eval.*, vol. 69, no. 7-8, pp. 315–335, Jul. 2012.

[16] E. Vicario, L. Sassoli, and L. Carnevali, "Using stochastic state classes in quantitative evaluation of dense-time reactive systems," *IEEE Trans. Softw. Eng.*, vol. 35, no. 5, pp. 703–719, Sep./Oct. 2009.

[17] M. Paolieri, A. Horváth, and E. Vicario, "Probabilistic Model Checking of Regenerative Concurrent Systems," *IEEE Trans. Softw. Eng.*, vol. 42, no. 2, pp. 153–169, Feb 2016.

[18] C. Lindemann and G. S. Shedler, "Numerical Analysis of Deterministic and Stochastic Petri Nets with Concurrent Deterministic Transitions," *Perform. Eval.*, vol. 27/28, no. 4, pp. 565–582, 1996.

[19] D. Logothetis, K. S. Trivedi, and A. Puliafito, "Markov Regenerative Models," in *Computer Performance and Dependability Symposium*, 1995, pp. 134–142.

[20] G. Bucci, L. Carnevali, L. Ridi, and E. Vicario, "Oris: a tool for modeling, verification and evaluation of real-time systems," *Int. J. on Softw. Tools for Techn. Transfer*, vol. 12, no. 5, pp. 391–403, Sep. 2010.