A Hierarchy of Scheduler Classes
for Stochastic Automata

Pedro R. D'Argenio[1,2,3], Marcus Gerhold[4], Arnd Hartmanns[4(✉)],
and Sean Sedwards[5]

[1] Universidad Nacional de Córdoba, Córdoba, Argentina
dargenio@famaf.unc.edu.ar
[2] CONICET, Córdoba, Argentina
[3] Saarland University, Saarbrücken, Germany
[4] University of Twente, Enschede, The Netherlands
{m.gerhold,a.hartmanns}@utwente.nl
[5] University of Waterloo, Waterloo, Canada
sean.sedwards@uwaterloo.ca

Abstract. Stochastic automata are a formal compositional model for concurrent stochastic timed systems, with general distributions and non-deterministic choices. Measures of interest are defined over *schedulers* that resolve the nondeterminism. In this paper we investigate the power of various theoretically and practically motivated classes of schedulers, considering the classic complete-information view and a restriction to non-prophetic schedulers. We prove a hierarchy of scheduler classes w.r.t. unbounded probabilistic reachability. We find that, unlike Markovian formalisms, stochastic automata distinguish most classes even in this basic setting. Verification and strategy synthesis methods thus face a tradeoff between powerful and efficient classes. Using lightweight scheduler sampling, we explore this tradeoff and demonstrate the concept of a useful approximative verification technique for stochastic automata.

1 Introduction

The need to analyse continuous-time stochastic models arises in many practical contexts, including critical infrastructures [4], railway engineering [36], space mission planning [7], and security [28]. This has led to a number of discrete event simulation tools, such as those for networking [34,35,42], whose probabilistic semantics is founded on generalised semi-Markov processes (GSMP [21,33]). Nondeterminism arises through inherent concurrency of independent processes [11], but may also be deliberate underspecification. Modelling such uncertainty with probability is convenient for simulation, but not always adequate [3,29]. Various models and formalisms have thus been proposed to extend continuous-time

stochastic processes with nondeterminism [8,10,19,23,27,38]. It is then possible to *verify* such systems by considering the extremal probabilities of a property. These are the supremum and infimum of the probabilities of the property in the purely stochastic systems induced by classes of *schedulers* (also called *strategies*, *policies* or *adversaries*) that resolve all nondeterminism. If the nondeterminism is considered controllable, one may alternatively be interested in the *planning* problem of synthesising a scheduler that satisfies certain probability bounds.

We consider closed systems of stochastic automata (SA [16]), which extend GSMP and feature both generally distributed stochastic delays as well as discrete nondeterministic choices. The latter may arise from non-continuous distributions (e.g. deterministic delays), urgent edges, and edges waiting on multiple clocks. Numerical verification algorithms exist for very limited subclasses of SA only: Buchholz et al. [13] restrict to phase-type or matrix-exponential distributions, such that nondeterminism cannot arise (as each edge is guarded by a single clock). Bryans et al. [12] propose two algorithms that require an a priori fixed scheduler, continuous bounded distributions, and that all active clocks be reset when a location is entered. The latter forces regeneration on every edge, making it impossible to use clocks as memory between locations. Regeneration is central to the work of Ballarini et al. [6], however they again exclude nondeterminism. The only approach that handles nondeterminism is the region-based approximation scheme of Kwiatkowska et al. [30] for a model closely related to SA, but restricted to bounded continuous distributions. Without that restriction [22], error bounds and convergence guarantees are lost.

Evidently, the combination of nondeterminism and continuous probability distributions is a particularly challenging one. With this paper, we take on the underlying problem from a fundamental perspective: we investigate the power of, and relationships between, different classes of schedulers for SA. Our motivation is, on the one hand, that a clear understanding of scheduler classes is crucial to design verification algorithms. For example, Markov decision process (MDP) model checking works well because memoryless schedulers suffice for reachability, and the efficient time-bounded analysis of continuous-time MDP (CTMDP) exploits a relationship between two scheduler classes that are sufficiently simple, but on their own do not realise the desired extremal probabilities [14]. When it comes to planning problems, on the other hand, practitioners desire *simple* solutions, i.e. schedulers that need little information and limited memory, so as to be explainable and suitable for implementation on e.g. resource-constrained embedded systems. Understanding the capabilities of scheduler classes helps decide on the tradeoff between simplicity and the ability to attain optimal results.

We use two perspectives on schedulers from the literature: the classic complete-information *residual lifetimes* semantics [9], where optimality is defined via history-dependent schedulers that see the entire current state, and *non-prophetic* schedulers [25] that cannot observe the timing of *future* events. Within each perspective, we define classes of schedulers whose views of the state and history are variously restricted (Sect. 3). We prove their relative ordering w.r.t. achieving optimal reachability probabilities (Sect. 4). We find that SA distinguish most classes. In particular, memoryless schedulers suffice in the complete-information setting (as is implicit in the method of Kwiatkowska et al. [30]), but

turn out to be suboptimal in the more realistic non-prophetic case. Considering only the relative order of clock expiration times, as suggested by the first algorithm of Bryans et al. [12], surprisingly leads to partly suboptimal, partly incomparable classes. Our distinguishing SA are small and employ a common nondeterministic gadget. They precisely pinpoint the crucial differences and how schedulers interact with the various features of SA, providing deep insights into the formalism itself.

Our study furthermore forms the basis for the application of *lightweight scheduler sampling* (LSS) to SA. LSS is a technique to use Monte Carlo simulation/statistical model checking with nondeterministic models. On every LSS simulation step, a pseudo-random number generator (PRNG) is re-seeded with a hash of the identifier of the current scheduler and the (restricted) information about the current state (and previous states, for history-dependent schedulers) that the scheduler's class may observe. The PRNG's first iterate then determines the scheduler's action deterministically. LSS has been successfully applied to MDP [18,31,32] and probabilistic timed automata [15,26]. Using only constant memory, LSS samples schedulers uniformly from a selected scheduler class to find "near-optimal" schedulers that conservatively approximate the true extremal probabilities. Its principal advantage is that it is largely indifferent to the size of the state space and of the scheduler space; in general, sampling efficiency depends only on the likelihood of selecting near-optimal schedulers. However, the mass of *near*-optimal schedulers in a scheduler class that also includes the optimal scheduler may be *less* than the mass in a class that does *not* include it. Given that the mass of optimal schedulers may be vanishingly small, it may be advantageous to sample from a class of less powerful schedulers. We explore these tradeoffs and demonstrate the concept of LSS for SA in Sect. 5.

Other Related Work. Alur et al. first mention nondeterministic stochastic systems similar to SA in [2]. Markov automata (MA [19]), interactive Markov chains (IMC [27]) and CTMDP are special cases of SA restricted to exponential distributions. Song et al. [37] look into partial information distributed schedulers for MA, combining earlier works of de Alfaro [1] and Giro and D'Argenio [20] for MDP. Their focus is on information flow and hiding in parallel specifications. Wolf et al. [39] investigate the power of classic (time-abstract, deterministic and memoryless) scheduler classes for IMC. They establish (non-strict) subset relationships for almost all classes w.r.t. trace distribution equivalence, a very strong measure. Wolovick and Johr [41] show that the class of measurable schedulers for CTMDP is complete and sufficient for reachability problems.

2   Preliminaries

For a given set S, its power set is $\mathcal{P}(S)$. We denote by $\mathbb{R}$, $\mathbb{R}^+$, and $\mathbb{R}_0^+$ the sets of real numbers, positive real numbers and non-negative real numbers, respectively. A (discrete) *probability distribution* over a set Ω is a function $\mu\colon \Omega \to [0,1]$, such that $\mathrm{support}(\mu) \stackrel{\text{def}}{=} \{\,\omega \in \Omega \mid \mu(\omega) > 0\,\}$ is countable and $\sum_{\omega \in \mathrm{support}(\mu)} \mu(\omega) = 1$. $\mathrm{Dist}(\Omega)$ is the set of probability distributions over Ω. We write $\mathcal{D}(\omega)$ for the *Dirac*

distribution for ω, defined by $\mathcal{D}(\omega)(\omega) = 1$. Ω is *measurable* if it is endowed with a σ-algebra $\sigma(\Omega)$: a collection of *measurable* subsets of Ω. A (continuous) *probability measure* over Ω is a function $\mu\colon \sigma(\Omega) \to [0, 1]$, such that $\mu(\Omega) = 1$ and $\mu(\cup_{i \in I} B_i) = \sum_{i \in I} \mu(B_i)$ for any countable index set I and pairwise disjoint measurable sets $B_i \subseteq \Omega$. $\mathrm{Prob}(\Omega)$ is the set of probability measures over Ω. Each $\mu \in \mathrm{Dist}(\Omega)$ induces a probability measure. Given probability measures μ_1 and μ_2, we denote by $\mu_1 \otimes \mu_2$ the *product measure*: the unique probability measure such that $(\mu_1 \otimes \mu_2)(B_1 \times B_2) = \mu_1(B_1) \cdot \mu_2(B_2)$, for all measurable B_1 and B_2. For a collection of measures $(\mu_i)_{i \in I}$, we analogously denote the product measure by $\bigotimes_{i \in I} \mu_i$. Let $\mathit{Val} \overset{\text{def}}{=} V \to \mathbb{R}_0^+$ be the set of valuations for an (implicit) set V of (non-negative real-valued) variables. $\mathbf{0} \in \mathit{Val}$ assigns value zero to all variables. Given $X \subseteq V$ and $v \in \mathit{Val}$, we write $v[X]$ for the valuation defined by $v[X](x) = 0$ if $x \in X$ and $v[X](y) = v(y)$ otherwise. For $t \in \mathbb{R}_0^+$, $v + t$ is the valuation defined by $(v + t)(x) = v(x) + t$ for all $x \in V$.

Stochastic Automata [16] extend labelled transition systems with stochastic *clocks*: real-valued variables that increase synchronously with rate 1 over time and expire some random amount of time after having been *restarted*. Formally:

Definition 1. *A* stochastic automaton *(SA) is a tuple* $\langle \mathit{Loc}, \mathcal{C}, A, E, F, \ell_{init} \rangle$, *where Loc is a countable set of* locations*, $\mathcal{C}$ is a finite set of* clocks*, A is the* finite *action alphabet, and* $E\colon \mathit{Loc} \to \mathcal{P}(\mathcal{P}(\mathcal{C}) \times A \times \mathcal{P}(\mathcal{C}) \times \mathrm{Dist}(\mathit{Loc}))$ *is the* edge function*, which maps each location to a finite set of edges that in turn consist of a* guard set *of clocks, a* label*, a* restart set *of clocks and a distribution over target locations. $F\colon \mathcal{C} \to \mathrm{Prob}(\mathbb{R}_0^+)$ is the* delay measure function *that maps each clock to a probability measure, and* $\ell_{init} \in \mathit{Loc}$ *is the* initial location.

We also write $\ell \xrightarrow{G,a,R}_E \mu$ for $\langle G, a, R, \mu \rangle \in E(\ell)$. W.l.o.g. we restrict to SA where edges are fully characterised by source state and action label, i.e. whenever $\ell \xrightarrow{G_1,a,R_1}_E \mu_1$ and $\ell \xrightarrow{G_2,a,R_2}_E \mu_2$, then $G_1 = G_2$, $R_1 = R_2$ and $\mu_1 = \mu_2$.

Intuitively, an SA starts in ℓ_{init} with all clocks expired. An edge $\ell \xrightarrow{G,a,R}_E \mu$ may be taken only if all clocks in G are expired. If any edge is enabled, some edge must be taken (i.e. all actions are *urgent* and thus the SA is *closed*). When an edge is taken, its action is a, all clocks in R are restarted, other expired clocks remain expired, and we move to successor location ℓ' with probability $\mu(\ell')$. There, another edge may be taken immediately or we may need to wait until some further clocks expire, and so on. When a clock c is restarted, the time until it expires is chosen randomly according to the probability measure $F(c)$.

Example 1. We show an example SA, M_0, in Fig. 1. Its initial location is ℓ_0. It has two clocks, x and y, with $F(x)$ and $F(y)$ both being the continuous uniform distribution over the interval $[0, 1]$. No time can pass in locations ℓ_0 and ℓ_1, since they have outgoing edges with empty guard sets. We omit action labels and assume every edge to have a unique label. On entering ℓ_1, both clocks are restarted. The choice of going to either ℓ_2 or ℓ_3 from ℓ_1 is nondeterministic, since
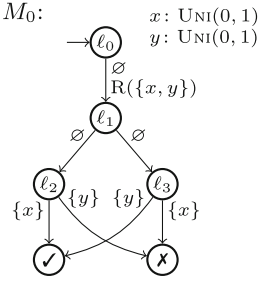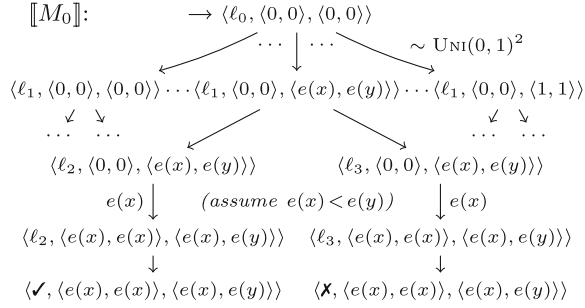
Fig. 1. Example SA M_0 **Fig. 2.** Excerpt of the TPTS semantics of M_0

the two edges are always enabled at the same time. In ℓ_2, we have to wait until the first of the two clocks expires. If that is x, we have to move to location ✓; if it is y, we have to move to ✗. The probability that both expire at the same time is zero. Location ℓ_3 behaves analogously, but with the target states interchanged.

Timed Probabilistic Transition Systems form the semantics of SA. They are finitely-nondeterministic uncountable-state transition systems:

Definition 2. *A (finitely nondeterministic) timed probabilistic transition system (TPTS) is a tuple $\langle S, A', T, s_{init} \rangle$. S is a measurable set of states. $A' = \mathbb{R}^+ \uplus A$ is the alphabet, partitioned into delays in $\mathbb{R}^+$ and jumps in A. $T : S \to \mathcal{P}(A' \times \mathrm{Prob}(S))$ is the transition function, which maps each state to a finite set of transitions, each consisting of a label in A' and a measure over target states. The initial state is $s_{init} \in S$. For all $s \in S$, we require $|T(s)| = 1$ if $\exists \langle t, \mu \rangle \in T(s) : t \in \mathbb{R}^+$, i.e. states admitting delays are deterministic.*

We also write $s \xrightarrow{a}_T \mu$ for $\langle a, \mu \rangle \in T(s)$. A *run* is an infinite alternating sequence $s_0 a_0 s_1 a_1 \ldots \in (S \times A')^\omega$, with $s_0 = s_{init}$. A *history* is a finite prefix of a run ending in a state, i.e. an element of $(S \times A')^* \times S$. Runs resolve all nondeterministic and probabilistic choices. A *scheduler* resolves only the nondeterminism:

Definition 3. *A measurable function $\mathfrak{s} : (S \times A')^* \times S \to \mathrm{Dist}(A' \times \mathrm{Prob}(S))$ is a scheduler if, for all histories $h \in (S \times A')^* \times S$, $\langle a, \mu \rangle \in \mathrm{support}(\mathfrak{s}(h))$ implies $lst_h \xrightarrow{a}_T \mu$, where lst_h is the last state of h.*

Once a scheduler has chosen $s_i \xrightarrow{a}_T \mu$, the successor state s_{i+1} is picked randomly according to μ. Every scheduler $\mathfrak{s}$ defines a probability measure $\mathbb{P}_\mathfrak{s}$ on the space of all runs. For a formal definition, see [40]. As is usual, we restrict to *non-Zeno* schedulers that make time diverge with probability one: we require $\mathbb{P}_\mathfrak{s}(\Pi_\infty) = 1$, where Π_∞ is the set of runs where the sum of delays is ∞. In the remainder of this paper we consider extremal probabilities of reaching a set of goal locations G:

Definition 4. *For $G \subseteq Loc$, let $J_G \overset{\text{def}}{=} \{ \langle \ell, v, e \rangle \in S \mid \ell \in G \}$. Let $\mathfrak{S}$ be a class of schedulers. Then $\mathrm{P}^\mathfrak{S}_{\min}(G)$ and $\mathrm{P}^\mathfrak{S}_{\max}(G)$ are the minimum and maximum reachability probabilities for G under $\mathfrak{S}$, defined as $\mathrm{P}^\mathfrak{S}_{\min}(G) = \inf_{\mathfrak{s} \in \mathfrak{S}} \mathbb{P}_\mathfrak{s}(\Pi_{J_G})$ and $\mathrm{P}^\mathfrak{S}_{\max}(G) = \sup_{\mathfrak{s} \in \mathfrak{S}} \mathbb{P}_\mathfrak{s}(\Pi_{J_G})$, respectively.*

Semantics of Stochastic Automata. We present here the residual lifetimes semantics of [9], simplified for closed SA: any delay step must be of the minimum delay that makes some edge become enabled.

Definition 5. *The semantics of an SA $M = \langle Loc, \mathcal{C}, A, E, F, \ell_{init} \rangle$ is the TPTS*

$$[\![M]\!] = \langle Loc \times Val \times Val, A \uplus \mathbb{R}^+, T_M, \langle \ell_{init}, \mathbf{0}, \mathbf{0} \rangle \rangle$$

where the states are triples $\langle \ell, v, e \rangle$ of the current location ℓ, a valuation v assigning to each clock its current value, and a valuation e keeping track of all clocks' expiration times. T_M is the smallest transition function satisfying inference rules

$$\frac{\ell \xrightarrow{G,a,R}_E \mu \quad \mathrm{En}(G,v,e)}{\langle \ell, v, e \rangle \xrightarrow{a}_{T_M} \mu \otimes \mathcal{D}(v[R]) \otimes \mathrm{Sample}_e^R}$$

$$\frac{t \in \mathbb{R}^+ \quad \exists \ell \xrightarrow{G,a,R}_E \mu\colon \mathrm{En}(G, v+t, e) \quad \forall\, t' \in [0,t), \ell \xrightarrow{G,a,R}_E \mu\colon \neg\, \mathrm{En}(G, v+t', e)}{\langle \ell, v, e \rangle \xrightarrow{t}_{T_M} \mathcal{D}(\langle \ell, v+t, e \rangle)}$$

with $\mathrm{En}(G,v,e) \overset{\mathrm{def}}{=} \forall\, x \in G\colon v(x) \geq e(x)$ characterising the enabled edges and

$$\mathrm{Sample}_e^R \overset{\mathrm{def}}{=} \bigotimes_{c \in \mathcal{C}} \begin{cases} F(c) & if\, c \in R \\ \mathcal{D}(e(c)) & if\, c \notin R. \end{cases}$$

The second rule creates *delay* steps of t time units if no edge is enabled from now until just before t time units have elapsed (third premise) but then, after exactly t time units, some edge becomes enabled (second premise). The first rule applies if an edge $\ell \xrightarrow{G,a,R}_E \mu$ is enabled: a transition is taken with the edge's label, the successor state's location is chosen by μ, v is updated by resetting the clocks in R to zero, and the expiration times for the restarted clocks are resampled. All other expiration times remain unchanged. Notice that $[\![M]\!]$ is also a nondeterministic labelled Markov process [40] (a proof can be found in [17]).

Example 2. Figure 2 outlines the semantics of M_0. The first step from ℓ_0 to all the states in ℓ_1 is a single transition. Its probability measure is the product of $F(x)$ and $F(y)$, sampling the expiration times of the two clocks. We exemplify the behaviour of all of these states by showing it for the case of expiration times $e(x)$ and $e(y)$, with $e(x) < e(y)$. In this case, to maximise the probability of reaching ✓, we should take the transition to the state in ℓ_2. If a scheduler $\mathfrak{s}$ can see the expiration times, noting that only their order matters here, it can always make the optimal choice and achieve $\mathrm{P}_{\max}^{\{\mathfrak{s}\}}(\{\checkmark\}) = 1$.

3 Classes of Schedulers

We now define classes of schedulers for SA with restricted information, hiding in various combinations the history and parts of states such as clock values and expiration times. All definitions consider TPTS as in Definition 5 with states $\langle \ell, v, e \rangle$ and we require for all $\mathfrak{s}$ that $\langle a, \mu \rangle \in \mathrm{support}(\mathfrak{s}(h)) \Rightarrow lst_h \xrightarrow{a}_T \mu$, as in Definition 3.

3.1   Classic Schedulers

We first consider the "classic" complete-information setting where schedulers can in particular see expiration times. We start with restricted classes of history-dependent schedulers. Our first restriction hides the values of all clocks, only revealing the total time since the start of the history. This is inspired by the step-counting or time-tracking schedulers needed to obtain optimal step-bounded or time-bounded reachability probabilities on MDP or Markov automata:

Definition 6. *A classic history-dependent* global-time *scheduler is a measurable function* $\mathfrak{s} \colon (S|_{\ell,t,e} \times A')^* \times S|_{\ell,t,e} \to \mathrm{Dist}(A' \times \mathrm{Prob}(S))$, *where* $S|_{\ell,t,e} \stackrel{\text{def}}{=} Loc \times \mathbb{R}_0^+ \times Val$ *with the second component being the total time t elapsed since the start of the history. We write* $\mathfrak{S}_{\ell,t,e}^{hist}$ *for the set of all such schedulers.*

We next hide the values of all clocks, revealing only their expiration times:

Definition 7. *A classic history-dependent* location-based *scheduler is a measurable function* $\mathfrak{s} \colon (S|_{\ell,e} \times A')^* \times S|_{\ell,e} \to \mathrm{Dist}(A' \times \mathrm{Prob}(S))$, *where* $S|_{\ell,e} \stackrel{\text{def}}{=} Loc \times Val$, *with the second component being the clock expiration times e. We write* $\mathfrak{S}_{\ell,e}^{hist}$ *for the set of all such schedulers.*

Having defined three classes of classic history-dependent schedulers, $\mathfrak{S}_{\ell,v,e}^{hist}$, $\mathfrak{S}_{\ell,t,e}^{hist}$ and $\mathfrak{S}_{\ell,e}^{hist}$, noting that $\mathfrak{S}_{\ell,v,e}^{hist}$ denotes all schedulers of Definition 3, we also consider them with the restriction that they only see the relative order of clock expiration, instead of the exact expiration times: for each pair of clocks c_1, c_2, these schedulers see the relation $\sim \in \{<,=,>\}$ in $e(c_1) - v(c_1) \sim e(c_2) - v(c_2)$. E.g. in ℓ_1 of Example 2, the scheduler would not see $e(x)$ and $e(y)$, but only whether $e(x) < e(y)$ or vice-versa (since $v(x) = v(y) = 0$, and equality has probability 0 here). We consider this case because the expiration order is sufficient for the first algorithm of Bryans et al. [12], and would allow optimal decisions in M_0 of Fig. 1. We denote the relative order information by o, and the corresponding scheduler classes by $\mathfrak{S}_{\ell,v,o}^{hist}$, $\mathfrak{S}_{\ell,t,o}^{hist}$ and $\mathfrak{S}_{\ell,o}^{hist}$. We now define memoryless schedulers, which only see the current state and are at the core of e.g. MDP model checking. On most formalisms, they suffice to obtain optimal reachability probabilities.

Definition 8. *A classic* memoryless *scheduler is a measurable function* $\mathfrak{s} \colon S \to \mathrm{Dist}(A' \times \mathrm{Prob}(S))$. *We write* $\mathfrak{S}_{\ell,v,e}^{ml}$ *for the set of all such schedulers.*

We apply the same restrictions as for history-dependent schedulers:

Definition 9. *A classic memoryless global-time scheduler is a measurable function* $\mathfrak{s} \colon S|_{\ell,t,e} \to \mathrm{Dist}(A' \times \mathrm{Prob}(S))$, *with* $S|_{\ell,t,e}$ *as in Definition 6. We write* $\mathfrak{S}_{\ell,t,e}^{ml}$ *for the set of all such schedulers.*

Definition 10. *A classic memoryless location-based scheduler is a measurable function* $\mathfrak{s} \colon S|_{\ell,e} \to \mathrm{Dist}(A' \times \mathrm{Prob}(S))$, *with* $S|_{\ell,e}$ *as in Definition 7. We write* $\mathfrak{S}_{\ell,e}^{ml}$ *for the set of all such schedulers.*

Again, we also consider memoryless schedulers that only see the expiration order, so we have memoryless scheduler classes $\mathfrak{S}_{\ell,v,e}^{ml}$, $\mathfrak{S}_{\ell,t,e}^{ml}$, $\mathfrak{S}_{\ell,e}^{ml}$, $\mathfrak{S}_{\ell,v,o}^{ml}$, $\mathfrak{S}_{\ell,t,o}^{ml}$ and $\mathfrak{S}_{\ell,o}^{ml}$. Class $\mathfrak{S}_{\ell,o}^{ml}$ is particularly attractive because it has a compact finite domain.

3.2   Non-prophetic Schedulers

Consider the SA M_0 in Fig. 1. No matter which of the previously defined scheduler classes we choose, we always find a scheduler that achieves probability 1 to reach ✓, and a scheduler that achieves probability 0. This is because they can all see the expiration times or expiration order of x and y when in ℓ_1. When in ℓ_1, x and y have not yet expired—this will only happen later, in ℓ_2 or ℓ_3—yet the schedulers already know which clock will "win". The classic schedulers can thus be seen to make decisions based on the timing of *future* events. This *prophetic* scheduling has already been observed in [9], where a "fix" in the form of the *spent lifetimes* semantics was proposed. Hartmanns et al. [25] have shown that this not only still permits prophetic scheduling, but even admits *divine* scheduling, where a scheduler can *change* the future. The authors propose a complex *non-prophetic* semantics that provably removes all prophetic and divine behaviour.

Much of the complication of the non-prophetic semantics of [25] is due to it being specified for open SA that include delayable actions. For the closed SA setting of this paper, prophetic scheduling can be more easily excluded by hiding from the schedulers all information about what will happen in the future of the system's evolution. This information is only contained in the expiration times e or the expiration order o. We can thus keep the semantics of Sect. 2 and modify the definition of schedulers to exclude prophetic behaviour by construction.

In what follows, we thus also consider all scheduler classes of Sect. 3.1 with the added constraint that the expiration times, resp. the expiration order, are not visible, resulting in the *non-prophetic* classes $\mathfrak{S}^{hist}_{\ell,v}$, $\mathfrak{S}^{hist}_{\ell,t}$, $\mathfrak{S}^{hist}_{\ell}$, $\mathfrak{S}^{ml}_{\ell,v}$, $\mathfrak{S}^{ml}_{\ell,t}$ and $\mathfrak{S}^{ml}_{\ell}$. Any non-prophetic scheduler can only reach ✓ of M_0 with probability $\frac{1}{2}$.

4   The Power of Schedulers

Now that we have defined a number of classes of schedulers, we need to determine what the effect of the restrictions is on our ability to optimally control an SA. We thus evaluate the power of scheduler classes w.r.t. unbounded reachability probabilities (Definition 4) on the semantics of SA. We will see that this simple setting already suffices to reveal interesting differences between scheduler classes.

For two scheduler classes $\mathfrak{S}_1$ and $\mathfrak{S}_2$, we write $\mathfrak{S}_1 \succcurlyeq \mathfrak{S}_2$ if, for all SA and all sets of goal locations G, $\mathrm{P}^{\mathfrak{S}_1}_{\min}(G) \leq \mathrm{P}^{\mathfrak{S}_2}_{\min}(G)$ and $\mathrm{P}^{\mathfrak{S}_1}_{\max}(G) \geq \mathrm{P}^{\mathfrak{S}_2}_{\max}(G)$. We write $\mathfrak{S}_1 \succ \mathfrak{S}_2$ if additionally there exists at least one SA and set G' where $\mathrm{P}^{\mathfrak{S}_1}_{\min}(G') < \mathrm{P}^{\mathfrak{S}_2}_{\min}(G')$ or $\mathrm{P}^{\mathfrak{S}_1}_{\max}(G') > \mathrm{P}^{\mathfrak{S}_2}_{\max}(G')$. Finally, we write $\mathfrak{S}_1 \approx \mathfrak{S}_2$ for $\mathfrak{S}_1 \succcurlyeq \mathfrak{S}_2 \wedge \mathfrak{S}_2 \succcurlyeq \mathfrak{S}_1$, and $\mathfrak{S}_1 \not\approx \mathfrak{S}_2$, i.e. the classes are incomparable, for $\mathfrak{S}_1 \not\succcurlyeq \mathfrak{S}_2 \wedge \mathfrak{S}_2 \not\succcurlyeq \mathfrak{S}_1$. Unless noted otherwise, we omit proofs for $\mathfrak{S}_1 \succcurlyeq \mathfrak{S}_2$ when it is obvious that the information available to $\mathfrak{S}_1$ includes the information available to $\mathfrak{S}_2$. All our distinguishing examples are based on the resolution of a single nondeterministic choice between two actions to eventually reach one of two locations. We therefore prove only w.r.t. the maximum probability, $p_{\max}$, for these examples since the minimum probability is given by $1 - p_{\max}$ and an analogous proof for $p_{\min}$ can be made by relabelling locations. We may write $\mathrm{P}_{\max}(\mathfrak{S}^y_x)$ for $\mathrm{P}^{\mathfrak{S}^y_x}_{\max}(\{✓\})$ to improve readability.

$$\begin{array}{ccccccc}
\mathfrak{S}_{\ell,o}^{ml} & \prec & \mathfrak{S}_{\ell,t,o}^{ml} & \prec & \mathfrak{S}_{\ell,v,o}^{ml} & \prec & \mathfrak{S}_{\ell,o}^{hist} \\
\rotatebox{90}{\(\not\approx\)} & & \rotatebox{90}{\(\not\approx\)} & & \curlywedge & & \rotatebox{90}{\(\approx\)} \\
\mathfrak{S}_{\ell,e}^{ml} & \prec & \mathfrak{S}_{\ell,t,e}^{ml} & \prec & \mathfrak{S}_{\ell,v,e}^{ml} & \succ & \mathfrak{S}_{\ell,t,o}^{hist} \\
\curlywedge & & \curlywedge & & \rotatebox{90}{\(\approx\)} & & \rotatebox{90}{\(\approx\)} \\
\mathfrak{S}_{\ell,e}^{hist} & \approx & \mathfrak{S}_{\ell,t,e}^{hist} & \approx & \mathfrak{S}_{\ell,v,e}^{hist} & \succ & \mathfrak{S}_{\ell,v,o}^{hist}
\end{array}$$

$$\begin{array}{ccccc}
\mathfrak{S}_{\ell}^{ml} & \prec & \mathfrak{S}_{\ell,t}^{ml} & \prec & \mathfrak{S}_{\ell,v}^{ml} \\
\curlywedge & & \curlywedge & & \curlywedge \\
\mathfrak{S}_{\ell}^{hist} & \approx & \mathfrak{S}_{\ell,t}^{hist} & \approx & \mathfrak{S}_{\ell,v}^{hist}
\end{array}$$
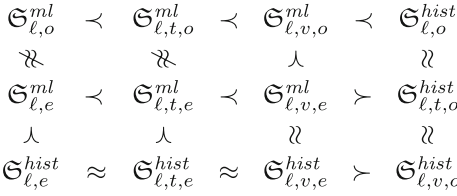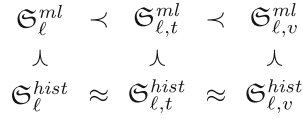
Fig. 3. Hierarchy of classic scheduler classes   **Fig. 4.** Non-prophetic classes

4.1   The Classic Hierarchy

We first establish that all classic history-dependent scheduler classes are equivalent:

Proposition 1. $\mathfrak{S}_{\ell,v,e}^{hist} \approx \mathfrak{S}_{\ell,t,e}^{hist} \approx \mathfrak{S}_{\ell,e}^{hist}$.

Proof. From the transition labels in $A' = A \uplus \mathbb{R}^+$ in the history $(S' \times A')^*$, with $S' \in \{ S, S|_{\ell,t,e}, S|_{\ell,e} \}$ depending on the scheduler class, we can reconstruct the total elapsed time as well as the values of all clocks: to obtain the total elapsed time, sum the labels in $\mathbb{R}^+$ up to each state; to obtain the values of all clocks, do the same per clock and perform the resets of the edges identified by the actions.

The same argument applies among the expiration-order history-dependent classes:

Proposition 2. $\mathfrak{S}_{\ell,v,o}^{hist} \approx \mathfrak{S}_{\ell,t,o}^{hist} \approx \mathfrak{S}_{\ell,o}^{hist}$.

However, the expiration-order history-dependent schedulers are strictly less powerful than the classic history-dependent ones:

Proposition 3. $\mathfrak{S}_{\ell,v,e}^{hist} \succ \mathfrak{S}_{\ell,v,o}^{hist}$.

Proof. Consider the SA M_1 in Fig. 5. Note that the history does not provide any information for making the choice in ℓ_1: we always arrive after having spent zero time in ℓ_0 and then having taken the single edge to ℓ_1. We can analytically determine that $P_{\max}(\mathfrak{S}_{\ell,v,e}^{hist}) = \frac{3}{4}$ by going from ℓ_1 to ℓ_2 if $e(x) \leq \frac{1}{2}$ and to ℓ_3 otherwise. We would obtain a probability equal to $\frac{1}{2}$ by always going to either ℓ_2 or ℓ_3 or by picking either edge with equal probability. This is the best we can do if e is not visible, and thus $P_{\max}(\mathfrak{S}_{\ell,v,o}^{hist}) = \frac{1}{2}$: in ℓ_1, $v(x) = v(y) = 0$ and the expiration order is always "y before x" because y has not yet been started.

Just like for MDP and unbounded reachability probabilities, the classic history-dependent and memoryless schedulers with complete information are equivalent:

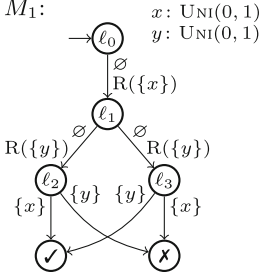**Proposition 4.** $\mathfrak{S}_{\ell,v,e}^{hist} \approx \mathfrak{S}_{\ell,v,e}^{ml}$.
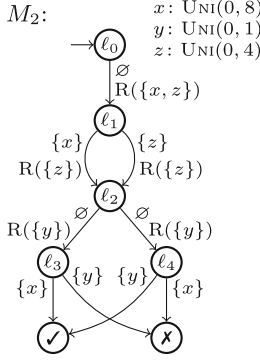
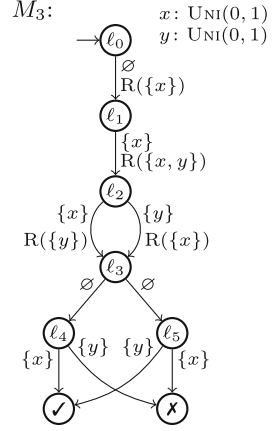**Fig. 5.** SA M_1 **Fig. 6.** SA M_2 **Fig. 7.** SA M_3

Proof sketch. Our definition of TPTS only allows finite nondeterministic choices, i.e. we have a very restricted form of continuous-space MDP. We can thus adapt the argument of the corresponding proof for MDP [5, Lemma 10.102]: For each state (of possibly countably many), we construct a notional optimal memoryless (and deterministic) scheduler in the same way, replacing the summation by an integration for the continuous measures in the transition function. It remains to show that this scheduler is indeed measurable. For TPTS that are the semantics of SA, this follows from the way clock values are used in the guard sets so that optimal decisions are constant over intervals of clock values and expiration times (see e.g. the arguments in [12] or [30]).

On the other hand, when restricting schedulers to see the expiration order only, history-dependent and memoryless schedulers are no longer equivalent:

Proposition 5. $\mathfrak{S}^{hist}_{\ell,v,o} \succ \mathfrak{S}^{ml}_{\ell,v,o}$.

Proof. Consider the SA M_2 in Fig. 6. Let $\mathfrak{s}^{opt}_{ml(l,v,o)}$ be the (unknown) optimal scheduler in $\mathfrak{S}^{ml}_{\ell,v,o}$ w.r.t. the max. probability of reaching ✓. Define $\mathfrak{s}^{better}_{hist(l,v,o)} \in \mathfrak{S}^{hist}_{\ell,v,o}$ as: when in ℓ_2 and the last edge in the history is the left one (i.e. x is expired), go to ℓ_3; otherwise, behave like $\mathfrak{s}^{opt}_{ml(l,v,o)}$. This scheduler distinguishes $\mathfrak{S}^{hist}_{\ell,v,o}$ and $\mathfrak{S}^{ml}_{\ell,v,o}$ (by achieving a strictly higher max. probability than $\mathfrak{s}^{opt}_{ml(l,v,o)}$) if and only if there are some combinations of clock values (aspect v) and expiration orders (aspect o) in ℓ_2 that can be reached with positive probability via the left edge into ℓ_2, for which $\mathfrak{s}^{opt}_{ml(l,v,o)}$ must nevertheless decide to go to ℓ_4.

All possible clock valuations in ℓ_2 can be achieved via either the left or the right edge, but taking the left edge implies that x expires before z in ℓ_2. It is thus sufficient to show that $\mathfrak{s}^{opt}_{ml(l,v,o)}$ must go to ℓ_4 in *some* cases where x

expires before z. The general form of schedulers in $\mathfrak{S}^{ml}_{\ell,v,o}$ in ℓ_2 is "go to ℓ_3 iff (a) x expires before z and $v(x) \in S_1$ or (b) z expires before x and $v(x) \in S_2$" where the S_i are measurable subsets of $[0,8]$. S_2 is in fact *irrelevant*: whatever $\mathfrak{s}^{opt}_{ml(l,v,o)}$ does when (b) is satisfied will be mimicked by $\mathfrak{s}^{better}_{hist(l,v,o)}$ because z can only expire before x when coming via the right edge into ℓ_2. Conditions (a) and (b) are independent.

With $S_1 = [0,8]$, the max. probability is $\frac{77}{96} = 0.80208\overline{3}$. Since this is the only scheduler in $\mathfrak{S}^{ml}_{\ell,v,o}$ that is *relevant* for our proof and never goes to l_4 when x expires before z, it remains to show that the max. probability under $\mathfrak{s}^{opt}_{ml(l,v,o)}$ is $> \frac{77}{96}$. With $S_1 = [0, \frac{35}{12})$, we have a max. probability of $\frac{7561}{9216} \approx 0.820421$. Thus $\mathfrak{s}^{opt}_{ml(l,v,o)}$ must sometimes go to l_4 even when the left edge was taken, so $\mathfrak{s}^{better}_{hist(l,v,o)}$ achieves a higher probability and thus distinguishes the classes.

Knowing only the global elapsed time is less powerful than knowing the full history or the values of all clocks:

Proposition 6. $\mathfrak{S}^{hist}_{\ell,t,e} \succ \mathfrak{S}^{ml}_{\ell,t,e}$ and $\mathfrak{S}^{ml}_{\ell,v,e} \succ \mathfrak{S}^{ml}_{\ell,t,e}$.

Proof sketch. Consider the SA M_3 in Fig. 7. We have $P_{\max}(\mathfrak{S}^{hist}_{\ell,t,e}) = 1$: when in ℓ_3, the scheduler sees from the history which of the two incoming edges was used, and thus knows whether x or y is already expired. It can then make the optimal choice: go to ℓ_4 if x is already expired, or to ℓ_5 otherwise. We also have $P_{\max}(\mathfrak{S}^{ml}_{\ell,v,e}) = 1$: the scheduler sees that either $v(x) = 0$ or $v(y) = 0$, which implies that the other clock is already expired, and the argument above applies. However, $P_{\max}(\mathfrak{S}^{ml}_{\ell,t,e}) < 1$: the distribution of elapsed time t on entering ℓ_3 is itself independent of which edge is taken. With probability $\frac{1}{4}$, exactly one of $e(x)$ and $e(y)$ is below t in ℓ_3, which implies that that clock has just expired and thus the scheduler can decide optimally. Yet with probability $\frac{3}{4}$, the expiration times are not useful: they are both positive and drawn from the same distribution, but one unknown clock is expired. The wait for x in ℓ_1 ensures that comparing t with the expiration times in e does not reveal further information in this case.

In the case of MDP, knowing the total elapsed time (i.e. steps) does not make a difference for unbounded reachability. Only for step-bounded properties is that extra knowledge necessary to achieve optimal probabilities. With SA, however, it makes a difference even in the unbounded case:

Proposition 7. $\mathfrak{S}^{ml}_{\ell,t,e} \succ \mathfrak{S}^{ml}_{\ell,e}$.

Proof. Consider SA M_4 in Fig. 8. We have $P_{\max}(\mathfrak{S}^{ml}_{\ell,t,e}) = 1$: in ℓ_2, the remaining time until y expires is $e(y)$ and the remaining time until x expires is $e(x) - t$ for the global time value t as ℓ_2 is entered. The scheduler can observe all of these quantities and thus optimally go to ℓ_3 if x will expire first, or to ℓ_4 otherwise. However, $P_{\max}(\mathfrak{S}^{ml}_{\ell,e}) < 1$: $e(x)$ only contains the absolute expiration time of x, but without knowing t or the expiration time of z in ℓ_1, and thus the current value $v(x)$, this scheduler cannot know with certainty which of the clocks will expire first and is therefore unable to make an optimal choice in ℓ_2.
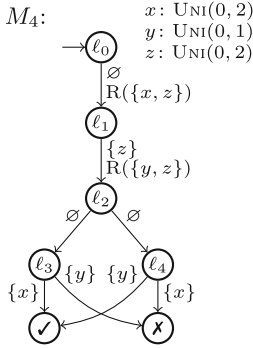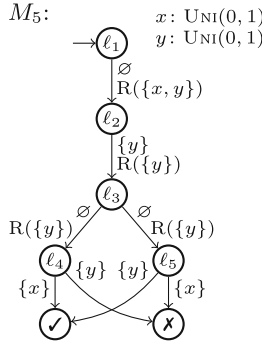
Fig. 8. SA M_4
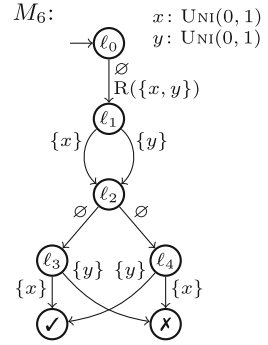

Fig. 9. SA M_5

Fig. 10. SA M_6

Finally, we need to compare the memoryless schedulers that see the clock expiration times with memoryless schedulers that see the expiration order. As noted in Sect. 3.1, these two views of the current state are incomparable unless we also see the clock values:

Proposition 8. $\mathfrak{S}^{ml}_{\ell,v,e} \succ \mathfrak{S}^{ml}_{\ell,v,o}$.

Proof. $\mathfrak{S}^{ml}_{\ell,v,e} \not\preccurlyeq \mathfrak{S}^{ml}_{\ell,v,o}$ follows from the same argument as in the proof of Proposition 3. $\mathfrak{S}^{ml}_{\ell,v,e} \succcurlyeq \mathfrak{S}^{ml}_{\ell,v,o}$ is because knowing the current clock values v and the expiration times e is equivalent to knowing the expiration order, since that is precisely the order of the differences $e(c) - v(c)$ for all clocks c.

Proposition 9. $\mathfrak{S}^{ml}_{\ell,t,e} \not\approx \mathfrak{S}^{ml}_{\ell,t,o}$.

Proof. $\mathfrak{S}^{ml}_{\ell,t,e} \not\preccurlyeq \mathfrak{S}^{ml}_{\ell,t,o}$ follows from the same argument as in the proof of Proposition 3. For $\mathfrak{S}^{ml}_{\ell,t,e} \not\succcurlyeq \mathfrak{S}^{ml}_{\ell,t,o}$, consider the SA M_3 of Fig. 7. We know from the proof of Proposition 6 that $P_{\max}(\mathfrak{S}^{ml}_{\ell,t,e}) < 1$. However, if the scheduler knows the order in which the clocks will expire, it knows which one has already expired (the first one in the order), and can thus make the optimal choice in ℓ_3 to achieve $P_{\max}(\mathfrak{S}^{ml}_{\ell,t,o}) = 1$.

Proposition 10. $\mathfrak{S}^{ml}_{\ell,e} \not\approx \mathfrak{S}^{ml}_{\ell,o}$.

Proof. The argument of Proposition 9 applies by observing that, in M_3 of Fig. 7, we also have $P_{\max}(\mathfrak{S}^{ml}_{\ell,e}) < 1$ via the same argument as for $\mathfrak{S}^{ml}_{\ell,t,e}$ in the proof of Proposition 6.

Among the expiration-order schedulers, the hierarchy is as expected:

Proposition 11. $\mathfrak{S}^{ml}_{\ell,v,o} \succ \mathfrak{S}^{ml}_{\ell,t,o} \succ \mathfrak{S}^{ml}_{\ell,o}$.

Proof sketch. Consider M_5 of Fig. 9. To maximise the probability, in ℓ_3 we should go to ℓ_4 whenever x is already expired or close to expiring, for which the amount of time spent in ℓ_2 is an indicator. $\mathfrak{S}_{\ell,o}^{ml}$ only knows that x may have expired when the expiration order is "x before y", but definitely has not expired when it is "y before x". Schedulers in $\mathfrak{S}_{\ell,t,o}^{ml}$ can do better: They also see the amount of time spent in ℓ_2. Thus $\mathfrak{S}_{\ell,t,o}^{ml} \succ \mathfrak{S}_{\ell,o}^{ml}$. If we modify M_5 by adding an initial delay on x from a new ℓ_0 to ℓ_1 as in M_3, then the same argument can be used to prove $\mathfrak{S}_{\ell,v,o}^{ml} \succ \mathfrak{S}_{\ell,t,o}^{ml}$: the extra delay makes knowing the elapsed time t useless with positive probability, but the exact time spent in l_2 is visible to $\mathfrak{S}_{\ell,v,o}^{ml}$ as $v(x)$.

We have thus established the hierarchy of classic schedulers shown in Fig. 3, noting that some of the relationships follow from the propositions by transitivity.

4.2   The Non-prophetic Hierarchy

Each non-prophetic scheduler class is clearly dominated by the classic and expiration-order scheduler classes that otherwise have the same information, for example $\mathfrak{S}_{\ell,v,e}^{hist} \succ \mathfrak{S}_{\ell,v}^{hist}$ (with very simple distinguishing SA). We show that the non-prophetic hierarchy follows the shape of the classic case, including the difference between global-time and pure memoryless schedulers, with the notable exception of memoryless schedulers being weaker than history-dependent ones.

Proposition 12. $\mathfrak{S}_{\ell,v}^{hist} \approx \mathfrak{S}_{\ell,t}^{hist} \approx \mathfrak{S}_{\ell}^{hist}$.

Proof. This follows from the argument of Proposition 1.

Proposition 13. $\mathfrak{S}_{\ell,v}^{hist} \succ \mathfrak{S}_{\ell,v}^{ml}$.

Proof. Consider the SA M_6 in Fig. 10. It is similar to M_4 of Fig. 8, and our arguments are thus similar to the proof of Proposition 7. On M_6, we have $P_{\max}(\mathfrak{S}_{\ell,v}^{hist}) = 1$: in ℓ_2, the history reveals which of the two incoming edges was used, i.e. which clock is already expired, thus the scheduler can make the optimal choice. However, if neither the history nor e is available, we get $P_{\max}(\mathfrak{S}_{\ell,v}^{ml}) = \frac{1}{2}$: the only information that can be used in ℓ_2 are the values of the clocks, but $v(x) = v(y)$, so there is no basis for an informed choice.

Proposition 14. $\mathfrak{S}_{\ell,t}^{hist} \succ \mathfrak{S}_{\ell,t}^{ml}$ *and* $\mathfrak{S}_{\ell,v}^{ml} \succ \mathfrak{S}_{\ell,t}^{ml}$.

Proof. Consider the SA M_3 in Fig. 7. We have $P_{\max}(\mathfrak{S}_{\ell,t}^{hist}) = P_{\max}(\mathfrak{S}_{\ell,v}^{ml}) = 1$, but $P_{\max}(\mathfrak{S}_{\ell,t}^{ml}) = \frac{1}{2}$ by the same arguments as in the proof of Proposition 6.

Proposition 15. $\mathfrak{S}_{\ell,t}^{ml} \succ \mathfrak{S}_{\ell}^{ml}$.

Proof. Consider the SA M_4 in Fig. 8. The schedulers in $\mathfrak{S}_{\ell}^{ml}$ have no information but the current location, so they cannot make an informed choice in ℓ_2. This and the simple loop-free structure of M_4 make it possible to analytically calculate the resulting probability: $P_{\max}(\mathfrak{S}_{\ell}^{ml}) = \frac{17}{24} = 0.708\overline{3}$. If information about the global elapsed time t in ℓ_2 is available, however, the value of x is revealed. This allows making a better choice, e.g. going to ℓ_3 when $t \leq \frac{1}{2}$ and to ℓ_4 otherwise, resulting in $P_{\max}(\mathfrak{S}_{\ell,t}^{ml}) \approx 0.771$ (statistically estimated with high confidence).

We have thus established the hierarchy of non-prophetic schedulers shown in Fig. 4, where some relationships follow from the propositions by transitivity.

5 Experiments

We have built a prototype implementation of lightweight scheduler sampling for SA by extending the MODEST TOOLSET's [24] MODES simulator, which already supports deterministic stochastic timed automata (STA [8]). With some care, SA can be encoded into STA. Using the original algorithm for MDP of [18], our prototype works by providing to the schedulers a discretised view of the continuous components of the SA's semantics, which, we recall, is a continuous-space MDP. The currently implemented discretisation is simple: for each real-valued quantity (the value $v(c)$ of clock c, its expiration time $e(c)$, and the global elapsed time t), it identifies all values that lie within the same interval $[\frac{i}{n}, \frac{i+1}{n})$, for integers i, n. We note that better static discretisations are almost certainly possible, e.g. a region construction for the clock values as in [30].

We have modelled M_1 through M_6 as STA in MODEST. For each scheduler class and model in the proof of a proposition, and discretisation factors $n \in \{1, 2, 4\}$, we sampled 10 000 schedulers and performed statistical model checking for each of them in the lightweight manner. In Fig. 11 we report the min. and max. estimates, $(\hat{p}_{\min}, \hat{p}_{\max})_{...}$, over all sampled schedulers. Where different discretisations lead to different estimates, we report the most extremal values. The subscript denotes the discretisation factors that achieved the reported estimates. The analysis for each sampled scheduler was performed with a number of simulation runs sufficient for the overall max./min. estimates to be within ± 0.01 of the true maxima/minima of the *sampled* set of schedulers with probability ≥ 0.95 [18]. Note that $\hat{p}_{\min}$ is an upper bound on the true minimum probability and $\hat{p}_{\max}$ is a lower bound on the true maximum probability.

Increasing the discretisation factor or increasing the scheduler power generally increases the number of decisions the schedulers *can* make. This may also increase the number of *critical* decisions a scheduler *must* make to achieve the extremal probability. Hence, the sets of discretisation factors associated to specific experiments may be informally interpreted in the following way:

- $\{1, 2, 4\}$: Fine discretisation is not important for optimality and optimal schedulers are not rare.
- $\{1, 2\}$: Fine discretisation is not important for optimality, but increases rarity of optimal schedulers.
- $\{2, 4\}$: Fine discretisation is important for optimality, optimal schedulers are not rare.
- $\{1\}$: Optimal schedulers are very rare.
- $\{2\}$: Fine discretisation is important for optimality, but increases rarity of schedulers.
- $\{4\}$: Fine discretisation is important for optimality and optimal schedulers are not rare.

M_1:

Proposition 3:
$\mathfrak{S}^{hist}_{\ell,v,e}$: $(0.24, 0.76)_{2,4}$
$\mathfrak{S}^{hist}_{\ell,v,o}$: $(0.49, 0.51)_{1,2,4}$

Proposition 8:
$\mathfrak{S}^{ml}_{\ell,v,e}$: $(0.24, 0.76)_{2,4}$
$\mathfrak{S}^{ml}_{\ell,v,o}$: $(0.49, 0.51)_{1,2,4}$

Proposition 9:
$\mathfrak{S}^{ml}_{\ell,t,e}$: $(0.24, 0.76)_{2,4}$
$\mathfrak{S}^{ml}_{\ell,t,o}$: $(0.49, 0.51)_{1,2,4}$

Proposition 10:
$\mathfrak{S}^{ml}_{\ell,e}$: $(0.24, 0.76)_{2,4}$
$\mathfrak{S}^{ml}_{\ell,o}$: $(0.49, 0.51)_{1,2,4}$

M_3:

Proposition 6:
$\mathfrak{S}^{hist}_{\ell,t,e}$: $(0.00, 1.00)_1$
$\mathfrak{S}^{ml}_{\ell,v,e}$: $(0.12, 0.86)_4$
$\mathfrak{S}^{ml}_{\ell,t,e}$: $(0.37, 0.65)_2$

Proposition 9:
$\mathfrak{S}^{ml}_{\ell,t,e}$: $(0.37, 0.65)_2$
$\mathfrak{S}^{ml}_{\ell,t,o}$: $(0.00, 1.00)_{1,2}$

Proposition 10:
$\mathfrak{S}^{ml}_{\ell,e}$: $(0.34, 0.67)_4$
$\mathfrak{S}^{ml}_{\ell,o}$: $(0.00, 1.00)_{1,2,4}$

Proposition 14:
$\mathfrak{S}^{hist}_{\ell,t}$: $(0.00, 1.00)_{1,2}$
$\mathfrak{S}^{ml}_{\ell,v}$: $(0.21, 0.78)_4$
$\mathfrak{S}^{ml}_{\ell,t}$: $(0.49, 0.51)_{1,2,4}$

M_2:

Proposition 5:
$\mathfrak{S}^{hist}_{\ell,o}$: $(0.06, 0.94)_{1,2,4}$
$\mathfrak{S}^{ml}_{\ell,v,o}$: $(0.18, 0.83)_1$

M_4:

Proposition 7:
$\mathfrak{S}^{ml}_{\ell,t,e}$: $(0.25, 0.79)_1$
$\mathfrak{S}^{ml}_{\ell,e}$: $(0.29, 0.71)_1$

Proposition 15:
$\mathfrak{S}^{ml}_{\ell,t}$: $(0.22, 0.78)_{2,4}$
$\mathfrak{S}^{ml}_{\ell}$: $(0.28, 0.72)_{1,2,4}$

M_5:

Proposition 11:
$\mathfrak{S}^{ml}_{\ell,t,o}$: $(0.15, 0.86)_4$
$\mathfrak{S}^{ml}_{\ell,o}$: $(0.16, 0.84)_{1,2,4}$

M_6:

Proposition 13:
$\mathfrak{S}^{hist}_{\ell,v}$: $(0.00, 1.00)_{1,2}$
$\mathfrak{S}^{ml}_{\ell,v}$: $(0.49, 0.51)_{1,2,4}$

Fig. 11. Results from the prototype of lightweight scheduler sampling for SA

The results in Fig. 11 respect and differentiate our hierarchy. In most cases, we found schedulers whose estimates were within the statistical error of calculated optima or of high confidence estimates achieved by alternative statistical techniques. The exceptions involve M_3 and M_4. We note that M_4 makes use of an additional clock, increasing the dimensionality of the problem and potentially making near-optimal schedulers rarer. The best result for M_3 and class $\mathfrak{S}^{ml}_{l,t,e}$ was obtained using discretisation factor $n = 2$: a compromise between nearness to optimality and rarity. A greater compromise was necessary for M_4 and classes $\mathfrak{S}^{ml}_{l,t,e}, \mathfrak{S}^{ml}_{l,e}$, where we found near-optimal schedulers to be very rare and achieved best results using discretisation factor $n = 1$.

The experiments demonstrate that lightweight scheduler sampling can produce useful and informative results with SA. The present theoretical results will allow us to develop better abstractions for SA and thus to construct a refinement algorithm for efficient lightweight verification of SA that will be applicable to realistically sized case studies. As is, they already demonstrate the importance of selecting a proper scheduler class for efficient verification, and that restricted classes are useful in planning scenarios.

6   Conclusion

We have shown that the various notions of information available to a scheduler class, such as history, clock order, expiration times or overall elapsed time, almost all make distinct contributions to the power of the class in SA. Our choice of notions was based on classic scheduler classes relevant for other stochastic models, previous literature on the character of nondeterminism in and verification of SA, and the need to synthesise simple schedulers in planning. Our distinguishing examples clearly expose how to exploit each notion to improve the probability

of reaching a goal. For verification of SA, we have demonstrated the feasibility of lightweight scheduler sampling, where the different notions may be used to finely control the power of the lightweight schedulers. To solve stochastic timed planning problems defined via SA, our analysis helps in the case-by-case selection of an appropriate scheduler class that achieves the desired tradeoff between optimal probabilities and ease of implementation of the resulting plan.

We expect the arguments of this paper to extend to steady-state/frequency measures (by adding loops back from absorbing to initial states in our examples), and that our results for classic schedulers transfer to SA with delayable actions. We propose to use the results to develop better abstractions for SA, the next goal being a refinement algorithm for efficient lightweight verification of SA.

References

1. de Alfaro, L.: The verification of probabilistic systems under memoryless partial-information policies is hard. Technical report, DTIC Document (1999)
2. Alur, R., Courcoubetis, C., Dill, D.: Model-checking for probabilistic real-time systems. In: Albert, J.L., Monien, B., Artalejo, M.R. (eds.) ICALP 1991. LNCS, vol. 510, pp. 115–126. Springer, Heidelberg (1991). https://doi.org/10.1007/3-540-54233-7_128
3. Andel, T.R., Yasinsac, A.: On the credibility of MANET simulations. IEEE Comput. **39**(7), 48–54 (2006)
4. Avritzer, A., Carnevali, L., Ghasemieh, H., Happe, L., Haverkort, B.R., Koziolek, A., Menaschè, D.S., Remke, A., Sarvestani, S.S., Vicario, E.: Survivability evaluation of gas, water and electricity infrastructures. Electr. Notes Theor. Comput. Sci. **310**, 5–25 (2015)
5. Baier, C., Katoen, J.P.: Principles of Model Checking. MIT Press, Cambridge (2008)
6. Ballarini, P., Bertrand, N., Horváth, A., Paolieri, M., Vicario, E.: Transient analysis of networks of stochastic timed automata using stochastic state classes. In: Joshi, K., Siegle, M., Stoelinga, M., D'Argenio, P.R. (eds.) QEST 2013. LNCS, vol. 8054, pp. 355–371. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-40196-1_30
7. Bisgaard, M., Gerhardt, D., Hermanns, H., Krčál, J., Nies, G., Stenger, M.: Battery-aware scheduling in low orbit: the GoMX–3 case. In: Fitzgerald, J., Heitmeyer, C., Gnesi, S., Philippou, A. (eds.) FM 2016. LNCS, vol. 9995, pp. 559–576. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-48989-6_34
8. Bohnenkamp, H.C., D'Argenio, P.R., Hermanns, H., Katoen, J.P.: MoDeST: a compositional modeling formalism for hard and softly timed systems. IEEE Trans. Softw. Eng. **32**(10), 812–830 (2006)
9. Bravetti, M., D'Argenio, P.R.: Tutte le algebre insieme: concepts, discussions and relations of stochastic process algebras with general distributions. In: Baier, C., Haverkort, B.R., Hermanns, H., Katoen, J.-P., Siegle, M. (eds.) Validation of Stochastic Systems. LNCS, vol. 2925, pp. 44–88. Springer, Heidelberg (2004). https://doi.org/10.1007/978-3-540-24611-4_2
10. Bravetti, M., Gorrieri, R.: The theory of interactive generalized semi-Markov processes. Theor. Comput. Sci. **282**(1), 5–32 (2002)

11. Brázdil, T., Krčál, J., Křetínský, J., Řehák, V.: Fixed-delay events in generalized semi-Markov processes revisited. In: Katoen, J.-P., König, B. (eds.) CONCUR 2011. LNCS, vol. 6901, pp. 140–155. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-23217-6_10

12. Bryans, J., Bowman, H., Derrick, J.: Model checking stochastic automata. ACM Trans. Comput. Log. **4**(4), 452–492 (2003)

13. Buchholz, P., Kriege, J., Scheftelowitsch, D.: Model checking stochastic automata for dependability and performance measures. In: DSN, pp. 503–514. IEEE Computer Society (2014)

14. Butkova, Y., Hatefi, H., Hermanns, H., Krčál, J.: Optimal continuous time Markov decisions. In: Finkbeiner, B., Pu, G., Zhang, L. (eds.) ATVA 2015. LNCS, vol. 9364, pp. 166–182. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-24953-7_12

15. D'Argenio, P.R., Hartmanns, A., Legay, A., Sedwards, S.: Statistical approximation of optimal schedulers for probabilistic timed automata. In: Ábrahám, E., Huisman, M. (eds.) IFM 2016. LNCS, vol. 9681, pp. 99–114. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-33693-0_7

16. D'Argenio, P.R., Katoen, J.P.: A theory of stochastic systems part I: stochastic automata. Inf. Comput. **203**(1), 1–38 (2005)

17. D'Argenio, P.R., Lee, M.D., Monti, R.E.: Input/output stochastic automata. In: Fränzle, M., Markey, N. (eds.) FORMATS 2016. LNCS, vol. 9884, pp. 53–68. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-44878-7_4

18. D'Argenio, P.R., Legay, A., Sedwards, S., Traonouez, L.M.: Smart sampling for lightweight verification of Markov decision processes. STTT **17**(4), 469–484 (2015)

19. Eisentraut, C., Hermanns, H., Zhang, L.: On probabilistic automata in continuous time. In: LICS, pp. 342–351. IEEE Computer Society (2010)

20. Giro, S., D'Argenio, P.R.: Quantitative model checking revisited: neither decidable nor approximable. In: Raskin, J.-F., Thiagarajan, P.S. (eds.) FORMATS 2007. LNCS, vol. 4763, pp. 179–194. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-75454-1_14

21. Haas, P.J., Shedler, G.S.: Regenerative generalized semi-Markov processes. commun. stat. Stochast. Models **3**(3), 409–438 (1987)

22. Hahn, E.M., Hartmanns, A., Hermanns, H.: Reachability and reward checking for stochastic timed automata. In: Electronic Communications of the EASST, AVoCS 2014, vol. 70 (2014)

23. Harrison, P.G., Strulo, B.: SPADES - a process algebra for discrete event simulation. J. Log. Comput. **10**(1), 3–42 (2000)

24. Hartmanns, A., Hermanns, H.: The Modest Toolset: an integrated environment for quantitative modelling and verification. In: Ábrahám, E., Havelund, K. (eds.) TACAS 2014. LNCS, vol. 8413, pp. 593–598. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-642-54862-8_51

25. Hartmanns, A., Hermanns, H., Krčál, J.: Schedulers are no Prophets. In: Probst, C.W., Hankin, C., Hansen, R.R. (eds.) Semantics, Logics, and Calculi. LNCS, vol. 9560, pp. 214–235. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-27810-0_11

26. Hartmanns, A., Sedwards, S., D'Argenio, P.: Efficient simulation-based verification of probabilistic timed automata. In: WSC. IEEE (2017). https://doi.org/10.1109/WSC.2017.8247885

27. Hermanns, H.: Interactive Markov Chains: The Quest for Quantified Quality. LNCS, vol. 2428. Springer, Heidelberg (2002). https://doi.org/10.1007/3-540-45804-2

28. Hermanns, H., Krämer, J., Krčál, J., Stoelinga, M.: The value of attack-defence diagrams. In: Piessens, F., Viganò, L. (eds.) POST 2016. LNCS, vol. 9635, pp. 163–185. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-49635-0_9

29. Kurkowski, S., Camp, T., Colagrosso, M.: MANET simulation studies: the incredibles. Mob. Comput. Commun. Rev. **9**(4), 50–61 (2005)

30. Kwiatkowska, M., Norman, G., Segala, R., Sproston, J.: Verifying quantitative properties of continuous probabilistic timed automata. In: Palamidessi, C. (ed.) CONCUR 2000. LNCS, vol. 1877, pp. 123–137. Springer, Heidelberg (2000). https://doi.org/10.1007/3-540-44618-4_11

31. Legay, A., Sedwards, S., Traonouez, L.M.: Estimating rewards & rare events in nondeterministic systems. In: Electronic Communications of the EASST, AVoCS 2015, vol. 72 (2015)

32. Legay, A., Sedwards, S., Traonouez, L.-M.: Scalable verification of Markov decision processes. In: Canal, C., Idani, A. (eds.) SEFM 2014. LNCS, vol. 8938, pp. 350–362. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-15201-1_23

33. Matthes, K.: Zur Theorie der Bedienungsprozesse. In: 3rd Prague Conference on Information Theory, Stat. Dec. Fns. and Random Processes, pp. 513–528 (1962)

34. NS-3 Consortium: ns-3: A Discrete-event Network Simulator for Internet Systems. https://www.nsnam.org/

35. Pongor, G.: OMNeT: objective modular network testbed. In: MASCOTS, pp. 323–326. The Society for Computer Simulation (1993)

36. Ruijters, E., Stoelinga, M.: Better railway engineering through statistical model checking. In: Margaria, T., Steffen, B. (eds.) ISoLA 2016. LNCS, vol. 9952, pp. 151–165. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-47166-2_10

37. Song, L., Zhang, L., Godskesen, J.C.: Late weak bisimulation for Markov automata. CoRR abs/1202.4116 (2012)

38. Strulo, B.: Process algebra for discrete event simulation. Ph.D. thesis, Imperial College of Science, Technology and Medicine. University of London, October 1993

39. Wolf, V., Baier, C., Majster-Cederbaum, M.E.: Trace semantics for stochastic systems with nondeterminism. Electr. Notes Theor. Comput. Sci. **164**(3), 187–204 (2006)

40. Wolovick, N.: Continuous probability and nondeterminism in labeled transition systems. Ph.D. thesis, Universidad Nacional de Córdoba, Córdoba, Argentina (2012)

41. Wolovick, N., Johr, S.: A characterization of meaningful schedulers for continuous-time Markov decision processes. In: Asarin, E., Bouyer, P. (eds.) FORMATS 2006. LNCS, vol. 4202, pp. 352–367. Springer, Heidelberg (2006). https://doi.org/10.1007/11867340_25

42. Zeng, X., Bagrodia, R.L., Gerla, M.: Glomosim: a library for parallel simulation of large-scale wireless networks. In: PADS, pp. 154–161. IEEE Computer Society (1998)