Solving Generalized Semi-Markov Processes using Continuous Phase-Type Distributions

Håkan L. S. Younes; Reid G. Simmons

In *Proceedings of the Nineteenth National Conference on Artificial Intelligence*, 742–747, San Jose, California. AAAI Press.

Author Annotations

A better policy-execution technique has been developed by Younes (2005). Instead of simulating phase transitions, as is done in this paper, the improved technique uses transient analysis to maintain a probability distribution over phase configurations.

References

Younes, Håkan L. S. 2005. Planning and execution with phase transitions. In *Proceedings of the Twentieth National Conference on Artificial Intelligence*, 1030–1035. AAAI Press.

Solving Generalized Semi-Markov Decision Processes using Continuous Phase-Type Distributions

Håkan L. S. Younes and **Reid G. Simmons**
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213, USA
{lorens, reids}@cs.cmu.edu

Abstract

We introduce the generalized semi-Markov decision process (GSMDP) as an extension of continuous-time MDPs and semi-Markov decision processes (SMDPs) for modeling stochastic decision processes with asynchronous events and actions. Using phase-type distributions and uniformization, we show how an arbitrary GSMDP can be approximated by a discrete-time MDP, which can then be solved using existing MDP techniques. The techniques we present can also be seen as an alternative approach for solving SMDPs, and we demonstrate that the introduction of phases allows us to generate higher quality policies than those obtained by standard SMDP solution techniques.

Introduction

We are interested in modeling and solving stochastic decision processes with asynchronous events and actions, where the delay from when an event or action is enabled until it triggers can be governed by an arbitrary positive distribution. While each of the aspects of stochastic decision processes listed above have been individually addressed in research on decision theoretic planning, no existing approach deals with all aspects simultaneously.

Guestrin, Koller, & Parr (2002) show how factored MDPs can be used to handle concurrent actions, but only for sets of actions executed in synchrony. Continuous-time MDPs (Howard 1960) can be used to model asynchronous systems, but are restricted to events and actions with exponential delay distributions. Semi-Markov decision processes (SMDPs) (Howard 1971) lift the restriction on delay distributions, but cannot model asynchrony.

We therefore introduce the *generalized* semi-Markov decision process (GSMDP), based on the GSMP model of discrete event systems (Glynn 1989), as a model for asynchronous stochastic decision processes. A GSMDP, unlike an SMDP, can remember if an event enabled in the current state has been continuously enabled in previous states without triggering. This is key in modeling asynchronous processes, which typically involve events that race to trigger first in a state, but the event that triggers first does not necessarily disable the competing events. For example, if we boil water for tea while simultaneously toasting bread, the fact that the bread pops out of the toaster first does not mean that we have to start over with the tea. Intuitively, a GSMDP can be viewed as the composition of asynchronous SMDPs.

In order to solve a GSMDP, we propose an approximation scheme that transforms an arbitrary GSMDP into a continuous-time MDP. Each non-exponential delay distribution in the GSMDP is approximated by a continuous *phase-type distribution* (Neuts 1981). This yields a continuous-time MDP, which in turn can be transformed into a discrete-time MDP using a technique called *uniformization* (Jensen 1953). Consequently, by using our proposed approximation scheme, we can approximately solve any GSMDP using standard MDP solution techniques. We will show that phase-type distributions are a practical tool for solving GSMDPs and SMDPs, and that our approach can produce policies of higher quality for SMDPs than standard techniques, because the introduction of phases indirectly allows us to take into account the time spent in a state.

Generalized Semi-Markov Decision Processes

The generalized semi-Markov process (GSMP), first introduced by Matthes (1962), is an established formalism in queuing theory for modeling continuous-time stochastic discrete event systems (Glynn 1989). We add a decision dimension to the formalism by distinguishing a subset of the events as controllable and adding rewards, thereby obtaining the generalized semi-Markov *decision* process (GSMDP).

A Model of Stochastic Discrete Event Systems

Consider a computer that can be in one of two states labeled up and $\neg\mathrm{up}$. Imagine an event, crash, that can cause a transition at a random point in time from the up state to the $\neg\mathrm{up}$ state, with the delay being governed by some positive distribution G. For example, if G is the uniform distribution $U(0,1)$, then the time a computer remains up before crashing is uniformly distributed in the interval $(0,1)$. A system consisting of a single such computer can be modeled as a semi-Markov process. Figure 1(a) shows the state transition diagrams for two identical processes modeling two computers running asynchronously.

Now, assume the two computers are connected to each other in a network. If we view the network as a single

stochastic process, we get the state transition diagram in Figure 1(b). Note that for continuous delay distributions, the probability is zero that two events trigger at exactly the same time, so there is no direct transition from state $\mathrm{up}_1 \wedge \mathrm{up}_2$ to state $\neg\mathrm{up}_1 \wedge \neg\mathrm{up}_2$. The resulting process is, in general, no longer a semi-Markov process. To see why, consider the scenario in which all events have a $U(0,1)$ delay distribution and after 0.2 time units the crash_1 event triggers in state $\mathrm{up}_1 \wedge \mathrm{up}_2$, causing a transition to state $\neg\mathrm{up}_1 \wedge \mathrm{up}_2$. Note that crash_2, which is enabled in the new state, has already been enabled for 0.2 time units, as it was enabled in the previous state without triggering. Consequently, the delay distribution for up_2 is not $U(0,1)$ at this point, but $U(0,0.8)$. In general, the delay distribution of an event at a particular point in time can depend on the entire execution history, and this history dependence cannot be captured by a semi-Markov process without augmenting the state space. However, the composite process is, without modifications to the state space, a *generalized* semi-Markov process (GSMP).

A GSMP consists of a set of states S and a set of events E. At any point in time, the process occupies a state $s \in S$ in which a subset E_s of the events are enabled. With each event e is associated a distribution G_e, governing the time until e triggers if it remains enabled, and a next-state probability distribution $p_e(s'|s)$. The enabled events race to trigger first, and the event that triggers causes a transition to a state $s' \in S$ according to the next-state probability distribution for the triggering event. A GSMP is equivalent to a continuous-time Markov chain if all delay distributions are exponential.

To formally define the semantics of a GSMP, we associate a real-valued clock c_e with each event that indicates the time remaining until e is scheduled to occur. The process starts in some initial state s with events E_s enabled. For each enabled event $e \in E_s$, we sample a trigger time according to the distribution G_e and set c_e to the sampled value. Let e^* be the event in E_s with the smallest clock value, i.e. $e^* = \arg\min_{e \in E_s} c_e$. The event e^* becomes the triggering event in s. When e^* triggers after c_{e^*} time units in s, we sample a next state s' according to $p_{e^*}(s'|s)$ and update each clock c_e for $e \in E_{s'}$ as follows: $c'_e = c_e - c_{e^*}$ if $e \in E_s \setminus \{e^*\}$, otherwise c'_e is sampled from G_e. In other words, events that remain enabled without triggering are not rescheduled in s', while the triggering event and newly enabled events are. The specified procedure is repeated with $s = s'$ and $c_e = c'_e$ so long as $E_s \neq \emptyset$.

Before we discuss actions, policies, and rewards for GSMDPs, we need to introduce the concept of an *execution path*. An execution path σ for a GSMP is a sequence

$$\sigma = s_0 \xrightarrow{t_0,e_0} s_1 \xrightarrow{t_1,e_1} s_2 \xrightarrow{t_2,e_2} \ldots$$

with $s_i \in S$, $e_i \in E_{s_i}$, and $t_i > 0$ being the sojourn time in state s_i before event e_i triggers a transition to state s_{i+1}. An execution path is finite if a state s_i along the path is absorbing (i.e. $E_{s_i} = \emptyset$), in which case we set $t_i = \infty$ (e_i is undefined). The length of an execution path equals the number of state transitions along the path, which can be infinite. An execution path for a GSMP up to state s_i contains sufficient information to characterize the future execution behavior of the process.

Actions, Policies, and Rewards

Given a GSMP with event set E, we identify a set $A \subset E$ of controllable events, or *actions*. The remaining events are called *exogenous events*. Actions differ from exogenous events in that they can be disabled at will in a state, while an exogenous event e always remains enabled in a state s if $e \in E_s$. For a given state s, a *policy* π picks the action in the set of potentially enabled actions $A \cap E_s$ to be enabled in s, or a special action a_∞ representing idleness and meaning that only exogenous events should be enabled in s. We allow the action choice to depend not only on the current state, but on the entire execution history up to the current choice point including the time spent in the current state. Thus, a policy is a mapping from partial execution paths and non-negative real numbers to actions.

In addition to actions, we need rewards to fully specify a GSMDP. We assume a traditional reward structure with a lump sum reward $k(s,e,s')$ associated with the transition from state s to s' caused by the triggering of event e, and a continuous reward rate $c(s,a)$ associated with action a being enabled in s. Since policies can switch actions at arbitrary time points, reward rates can vary over time in a given state.

For a fixed policy π, we define the discounted value of an execution path σ of length n (possibly infinite) by

$$v^\pi_\alpha(\sigma) \equiv \sum_{i=0}^{n} e^{-\alpha T_i}\left(e^{-\alpha t_i}k(s_i,e_i,s_{i+1}) + \int_0^{t_i} e^{-\alpha t} c(s_i,\pi(\sigma_{\leq i},t))\,dt\right),$$

where $T_0 = 0$ and $T_i = \sum_{j=0}^{i-1} t_j$ for $i > 0$, and $\sigma_{\leq i}$ is the prefix of σ up to and including state s_i. The parameter α is the *discount rate*, and can be interpreted as the rate of a termination event with exponential delay distribution (Howard 1960). We now define the expected infinite-horizon discounted reward for π, given that execution starts in state s at time 0, by

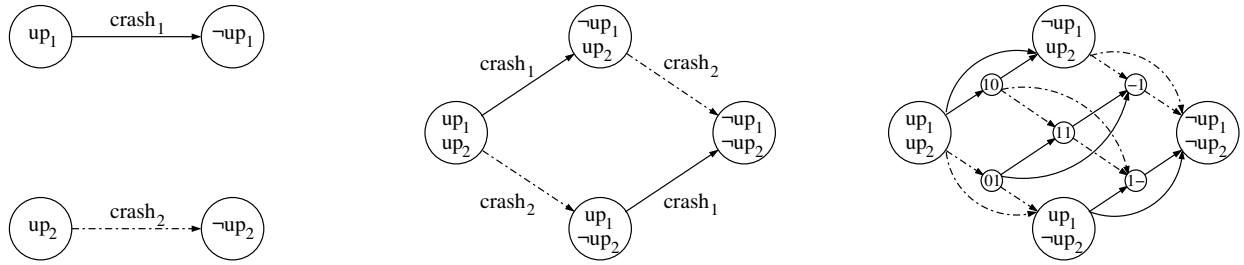
$$v^\pi_\alpha(s) \equiv E^\pi_s[v^\pi_\alpha(\sigma)]. \quad (1)$$

Here, $E^\pi_s[\cdot]$ denotes the expectation with respect to the probability measure induced by policy π over the set of execution paths starting in state s. The objective is to find a policy π maximizing the expectation in (1). We propose to do so by first approximating the GSMDP with a continuous-time MDP using phase-type distributions.

Continuous Phase-Type Distributions

The memoryless property of the exponential distribution plays a key role in the analytical tractability of continuous-time Markov chains and MDPs. Erlang (1917) was the first to consider a generalization of the exponential distribution that preserves much of its analytic tractability. The Erlang distribution introduces n sequential *phases*, with the time spent in phase i being exponentially distributed with rate λ. A stochastic process with Erlang distributed sojourn times is Markovian if phase is included in state descriptions.

The Erlang distribution is a special case of the class of continuous *phase-type distributions* (Neuts 1981). A ran-



(a) Two asynchronous processes.

(b) Composite system.

(c) Composite system with phase transitions.

Figure 1: State transition diagrams for stochastic processes modeling a network of computers. The rightmost diagram includes additional states to record the phase of the events. The first digit represents the phase of crash_1 and the second digit the phase of crash_2. A dash ("–") means that the event associated with the phase is disabled in the state.

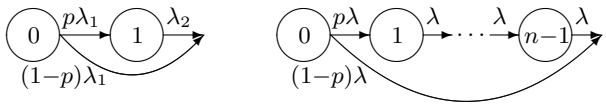
dom variable associated with a continuous phase-type distribution of order n represents the time to absorption in an n-state Markov chain and has the cumulative distribution function $1-\boldsymbol{\alpha}e^{\boldsymbol{T}t}\boldsymbol{e}$, where the n-dimensional row vector $\boldsymbol{\alpha}$ is the initial distribution of the Markov chain, $\boldsymbol{T}$ is its $n \times n$ infinitesimal generator matrix, and $\boldsymbol{e}$ is an n-dimensional column vector of ones.

There exist numerous techniques for approximating a general distribution G with a continuous phase-type distribution PH. The most straightforward technique is the *method of moments*, where the objective is to find $\boldsymbol{\alpha}$ and $\boldsymbol{T}$ such that the first k moments of G and PH match. The ith moment of a distribution G is defined as $\mu_i \equiv E[X^i]$, where X is a random variable with distribution G. Two useful concepts are the *mean* of a distribution (μ_1) and the *coefficient of variation* ($cv^2 \equiv \mu_2/\mu_1^2 - 1$). Note that $cv^2 = 1$ for the exponential distribution.

When using the method of moments, it is desirable to match as many moments of G as possible, but we will typically need more phases to match more moments. More phases means a larger state space, so there is a tradeoff between accuracy and complexity of the approximate model.

We can match a single moment using an exponential distribution with rate $1/\mu_1$ but this typically yields a poor approximation of G. A better approximation is obtained by matching the second moment as well. For G with $cv^2 \geq 1/2$, this can be accomplished by using a two-phase Coxian distribution (Figure 2(a)) with parameters $\lambda_1 = 2/\mu_1$, $\lambda_2 = 1/(\mu_1 \cdot cv^2)$, and $p = 1/(2 \cdot cv^2)$ (Marie 1980). For example, a Weibull distribution $W(1, 1/2)$ with scale parameter 1 and shape parameter $1/2$ ($\mu_1 = 2$ and $cv^2 = 5$) can be approximated by a two-phase Coxian distribution ($\lambda_1 = 1$, $\lambda_2 = 1/10$, and $p = 1/10$). If $W(1, 1/2)$ is the delay distribution for both crash_1 and crash_2 in Figure 1(b), then Figure 1(c) is the Markov chain approximating the original GSMP obtained by approximating each delay distribution with a phase-type distribution.

If $cv^2 < 1/2$, we match the first two moments by using a generalized Erlang distribution (Figure 2(b)) with parameters $n = \lceil 1/cv^2 \rceil$, $p = 1 - (2n \cdot cv^2 + n - 2 - \sqrt{n^2 + 4 - 4n \cdot cv^2})/(2(n-1)(cv^2+1))$, and $\lambda = (1-p+np)/\mu_1$ (Sauer & Chandy 1975; Marie 1980). For example, a uniform distribution $U(0, 1)$ ($\mu_1 = 1/2$ and $cv^2 = 1/3$) can be approximated by a three-phase generalized Erlang distribution with $p = 1$ and $\lambda = 6$.



(a) (b)

Figure 2: (a) Two-phase Coxian distribution; (b) n-phase generalized Erlang distribution.

More recently, Osogami & Harchol-Balter (2003) have presented a closed-form solution for matching the first three moments of any positive distribution using an Erlang distribution followed by a two-phase Coxian distribution. Later in this paper, we compare the quality of policies obtained by matching one, two, and three moments of each non-exponential distribution in a model. Better policies are generally obtained by increasing the number of matching moments, but at the cost of larger state spaces.

Solution Method

To solve a GSMDP, we first convert it into a discrete-time MDP, which can be solved using standard techniques such as value iteration. The resulting policy is then translated into a policy for the GSMDP.

From GSMDP to Discrete-Time MDP

We note that if all events of a GSMDP are Markovian (i.e. have exponential delay distribution) then the GSMDP is simply a continuous-time MDP. By using phase-type distributions, we can replace each non-Markovian event in the GSMDP with one or more Markovian events, thereby obtaining a continuous-time MDP that approximates the original GSMDP. The continuous-time MDP can be turned into

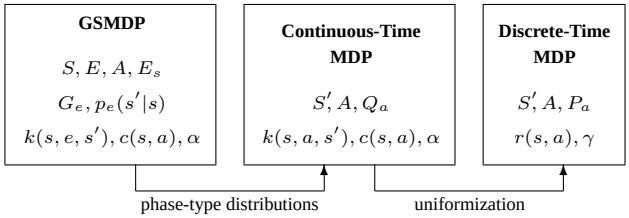


Figure 3: From GSMDP to discrete-time MDP.

a discrete-time MDP using uniformization, described below. The complete process of transforming a GSMDP into a discrete-time MDP, detailed below, is outlined in Figure 3.

We assume a factored representation of the state space with state variables V, and associate an enabling condition ϕ_e with each event $e \in E$ such that $s \models \phi_e$ iff $e \in E_s$ for all $s \in S$. We also assume that $p_e(s'|s)$ is implicitly represented by an effect formula eff_e for each $e \in E$, for example using the effect formalism described by Rintanen (2003). Thus, a GSMDP event e is represented by a triple $\langle \phi_e, G_e, \mathit{eff}_e \rangle$.

For each non-Markovian event e with delay distribution G_e, we find a phase-type distribution of order n_e approximating G_e. We add a phase variable s_e to V for each event e with $n_e > 1$ and replace e with one or more Markovian events. For example, if we match two moments, we get a phase-type distribution defined in terms of n_e, p_e, λ_1^e, and λ_2^e ($\lambda_2^e = \lambda_1^e$ if $n_e > 2$). If $n_e > 2$ we add the event $\langle \phi_e \wedge s_e \in [1, n-2], \mathit{Exp}(\lambda_1), s_e \leftarrow s_e + 1 \rangle$. The following events are added for all values of n_e:

$$\langle \phi_e \wedge s_e = 0, \mathit{Exp}(p_e \lambda_1^e), s_e \leftarrow 1 \rangle$$
$$\langle \phi_e \wedge s_e = 0, \mathit{Exp}((1-p_e)\lambda_1^e), \mathit{eff}_e \wedge s_e \leftarrow 0 \rangle$$
$$\langle \phi_e \wedge s_e = n_e - 1, \mathit{Exp}(\lambda_2^e), \mathit{eff}_e \wedge s_e \leftarrow 0 \rangle$$

We associate the transition reward $k(s, e, s')$ with the last two events and zero transition reward with the other events, which represent pure phase transitions.

An event $e = \langle \phi_e, \mathit{Exp}(\lambda_e), \mathit{eff}_e \rangle$ by itself represents a continuous-time Markov chain having an infinitesimal generator matrix Q_e with elements

$$q_{ij} = \begin{cases} 0 & \text{if } i \not\models \phi_e \\ p_e(j|i)\lambda_e & \text{if } i \models \phi_e \text{ and } i \neq j \\ -(1 - p_e(i|i))\lambda_e & \text{if } i \models \phi_e \text{ and } i = j \end{cases},$$

where $p_e(s'|s)$ is the next-state probability distribution induced by eff_e. A set of asynchronous events E represents a continuous-time Markov chain with $Q_E = \sum_{e \in E} Q_e$, and we call $\sum_{e \in E} \lambda_e$ the *exit rate* of E.

Let E' be the set of events obtained from the original set of events E for the GSMDP through the translation process described above, and let $E'_e \subset E'$ be the events derived from the original GSMDP event e. The set $A' = \{e' \in E'_e | e \in A\}$ represents all Markovian events associated with GSMDP actions. We compute infinitesimal generator matrices $Q_\infty = \sum_{e \in E' \setminus A'} Q_e$ for the idle action a_∞ and $Q_a = Q_\infty + \sum_{e \in E'_a} Q_e$ for each action $a \in A$, and we define the exit rate λ_a of an action as the sum of exit rates for the events factored into Q_a. The matrices Q_∞ and Q_a for each $a \in A$, together with the expected transition rewards $k(s, a, s') = \sum_{e \in (E' \setminus A') \cup E'_a} k(s, e, s') \lambda_e / \lambda_a$ and reward rates $c(s, a)$ (reward rates are independent of phase and remain unmodified), constitute a continuous-time MDP with action space $A \cup \{a_\infty\}$ and state space S' defined by the state variables V and phase variables s_e for each non-Markovian GSMDP event $e \in E$.

The obtained continuous-time MDP approximates the original GSMDP, and can be solved by first transforming it into a discrete-time MDP using *uniformization*. This technique, introduced by Jensen (1953) as a technique for analyzing continuous-time Markov chains and implicitly used by Howard (1960) for solving continuous-time MDPs, allows us to use regular solution methods for discrete-time MDPs, such as value iteration, to solve continuous-time MDPs. Puterman (1994) gives an excellent description of the uniformization process, to which we refer the reader for details. In brief, we choose a uniformization constant $q \geq \max_{a \in A \cup \{a_\infty\}, i \in S'} -q_{ii}^a$. We derive transition probability matrices $P_a = I + Q_a/q$ for the discrete-time MDP, where I is the identity matrix of the same dimension as Q_a. The expected discounted reward between decision epochs for the discrete-time MDP becomes $r(s, a) = \sum_{s' \in S'} p_a(s'|s) k(s, a, s') + c(s, a)/(\alpha + \lambda_a)$, where $p_a(s'|s)$ is an entry in P_a and λ_a is the exit rate for a as defined above. Finally, we derive a discount factor $\gamma = q/(q + \alpha)$.

Policy Execution

The execution history of a GSMDP can be represented by a set of real-valued variables, one for each event $e \in E$ representing the time e has been continuously enabled without triggering. The phases we introduce when approximating a GSMDP with a continuous-time MDP can be thought of as a randomized discretization of the time events have remained enabled. For example, approximating G with an n-phase Erlang distribution with parameters p and λ represents a discretization of the time G has been enabled into n random-length intervals. The length of each interval is a random variable with distribution $\mathit{Exp}(\lambda)$. A policy for the continuous-time MDP with phase transitions is therefore approximately a mapping from states and the times events have been enabled to actions for the original GSMDP.

Phase transitions are not part of the original model, so we have to simulate them when executing the policy obtained for the approximate model. When a GSMDP event or action e becomes enabled during execution, we sample a first phase transition time t_1 for the phase-type distribution used to approximate G_e. If e remains enabled for t_1 time units without triggering, we increment the phase associated with e and sample a second phase transition time t_2. This continues until e triggers or is disabled, in which case the phase is reset to zero, or we reach the last phase, in which case the phase does not change until e triggers or is disabled. The action choice can change every time a simulated phase transition occurs, although phase transitions do not change the actual state of the process. This allows us to take the time we have spent in a state into account when selecting which

action to enable. We will see in the next section that this will permit us to obtain better policies than if we switched the enabled action only at actual state transitions.

Empirical Evaluation

We have implemented a basic GSMDP planner based on the transformation procedure outlined in Figure 3. Our implementation uses multi-terminal BDDs (MTBDDs, also known as ADDs) (Fujita, McGeer, & Yang 1997) to represent matrices and vectors. The discrete-time MDPs that result from the transformation are solved using value iteration, essentially using the same approach as Hoey *et al.* (1999).

Our first test case is a variation of Howard's "the Foreman's Dilemma" (Howard 1960), where we have a machine that can be working (s_0), failed (s_1), or serviced (s_2). A failure event with delay distribution G is enabled in s_0 and causes a transition to s_1. Once in s_1, the repair time for the machine has distribution $Exp(1/100)$. The foreman can at any time in s_0 choose to enable a service action with delay distribution $Exp(10)$. If this action triggers before the failure event, the system enters s_2 where the machine is being serviced with service time distribution $Exp(1)$. Given reward rates $c(s_0) = 1$, $c(s_1) = 0$, and $c(s_2) = 1/2$, independent of action choice, and no transition rewards, the problem is to produce a service policy that maximizes expected infinite-horizon discounted reward in s_0.

We can model this problem as an SMDP, noting that the probability of the service action triggering before the failure event is $p_{02} = 1 - \int_{t_0}^{\infty} 10e^{-10(t-t_0)}F(t)\,dt$ (where $F(t)$ is the cumulative distribution function for G) if we enable the service action after t_0 time units in s_0. We can solve the SMDP using the techniques described by Howard (1971), but then we can choose to enable the action in s_0 only immediately ($t_0 = 0$) or not at all ($t_0 = \infty$). Alternatively, we can express the expected reward in s_0 as a function of t_0 and use numerical solution techniques to find the value for t_0 that maximizes the expected reward. Depending on the shape of $F(t)$, both of these approaches may require numerical integration over semi-infinite intervals.

Figure 4 plots the expected reward as a percentage of the optimal expected reward for policies obtained using standard SMDP solution techniques as well as our proposed technique for approximating a (G)SMDP with an MDP using phase-type distributions. The optimal value and the value for the SMDP solution were computed using MATLAB, while the other values were computed by simulating execution of the policy generated by our GSMDP planner and taking the average value over 5000 samples. We used $\alpha = -\log 0.95$ as the discount factor. The two graphs are for different choices of the failure time distribution G ($U(5, x)$ for left graph and $W(1.6x, 4.5)$ for right).

We can see that the SMDP solution is well below the optimal solution because it has to enable the action either immediately, or not at all, in s_0. For small values of x, the optimal SMDP policy is to enable the action in s_0, but as x increases so does the expected failure time, so for larger x it is better not to enable the action because it allows us to spend more time in s_0 where the reward rate is highest. The performance of the policy obtained by matching a single moment of G is almost identical to that of the SMDP solution. This policy is also restricted to either enabling the action immediately or nor at all in s_0. Due to the approximation of G, the performance is slightly worse around the point where the optimal SMDP policy changes. We can see that by increasing the number of moments that are matched, we can increase the quality of the policy. Note the number of phases varies with x for $U(5, x)$, but is constant for $W(1.6x, 4.5)$, which explains why the curves in the right graph are smoother than in the left. The only exception is for lower values of x in the right graph. In this case, the phase-type distribution we obtain underestimates the probability that failure occurs at a very early stage, so the enabling of the action comes later than needed to perform well. In general, however, matching more moments gives us better policies, mainly because the phases that are introduced allow us to take into account the fact that G is not memoryless, and so delay the enabling of the action in s_0 accordingly.

Our second test case is a system administration problem based on the computer network system introduced earlier in this paper (cf. Guestrin, Koller, & Parr 2002). To make it a decision problem, we add a reboot action for each machine m_i, which can be enabled if m_i is not up. The reward rate for a state is equal to the number of machines that are up. The delay distributions for the reboot actions are all set to $U(0, 1)$, while $Exp(1)$ was chosen for the crash events.

Figure 5 plots the expected discounted value of the policy ($\alpha = -\log 0.95$) obtained by our GSMDP planner when matching between one and three moments of non-exponential delay distributions. We see that by matching two or three moments, we can obtain a policy that performs noticeably better than the policy obtain by matching only a single moment. By matching a single moment, we can select to enable a reboot action based only on which machines are currently not up, and the resulting policy reboots m_i before m_j if $i < j$. In contrast, the policy obtained when matching two or three moments keeps a reboot action enabled if it is in a phase other than zero, because it is then expected to trigger soon. While the first test case illustrated that phases can result in better policies by delaying the enabling of an action in a state, this test case illustrates that phases can help by keeping an action enabled if it has already been enabled for some time.

By matching more moments, we increase the accuracy of the approximation, but we also increase the state space. In terms of planning time, a larger state space means a solution will take longer to obtain. Matching more moments typically also results in a larger uniformization constant, which translates to a slower convergence for value iteration. Thus, as one can expect, better policies are obtained at the price of longer solution times. For the system administration domain with n machines, $|S|$ is 2^n, $(n+1)2^n$, and $(1.5n+1)2^n$ when matching one, two, and three moments, respectively. The uniformization constant is $n + 1$, $n + 5$, and approximately $n+17$, respectively. With our implementation, it takes 4–12 times longer to solve the model obtained by matching two moments than the model obtained by matching only a single moment.

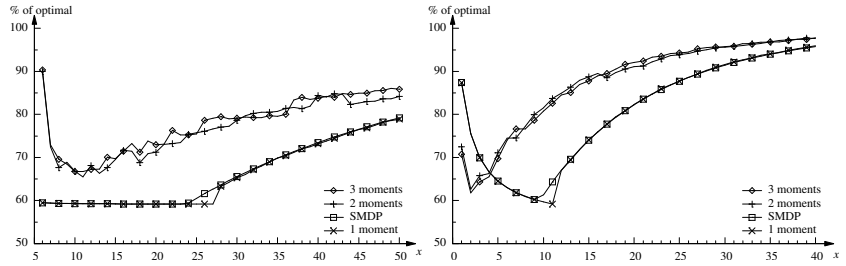


Figure 4: Policy value, as a percentage of the optimal value, obtained using different solution techniques for a variation of "the Foreman's Dilemma" with the failure time distribution G being $U(5, x)$ (left) and $W(1.6x, 4.5)$ (right).

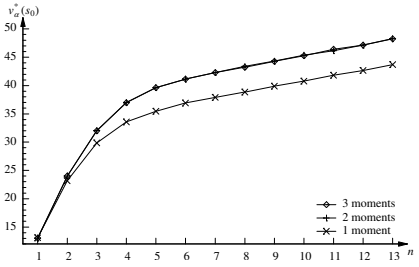


Figure 5: Expected discounted reward for the system administration problem with n machines using phase-type distributions.

Discussion

The generalized semi-Markov decision process is a natural model for stochastic systems with asynchronous events and actions. The main contribution of this paper has been a general technique for approximating a GSMDP with an MDP by the use of phase-type distributions. We have shown that the approximation technique is useful not only for solving GSMDPs, but also for generating policies for SMDPs. Phases allow us to take into account the time spent in a state when selecting actions, which can result in better policies than by using standard SMDP techniques. In addition, the approximation technique is not tied to any specific solution technique for MDPs. This makes the approach very general and allows one to easily take advantage of novel MDP solution techniques when solving GSMDP, including approximate solution methods.

Acknowledgments. Thanks to Vu Ha and David Musliner for insightful comments on an early draft.

This paper is based upon work supported by DARPA and ARO under contract no. DAAD19–01–1–0485, and a grant from the Royal Swedish Academy of Engineering Sciences (IVA). The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright annotation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the sponsors.

References

Erlang, A. K. 1917. Solution of some problems in the theory of probabilities of significance in automatic telephone exchanges. *The Post Office Electrical Engineers' Journal* 10:189–197.

Fujita, M.; McGeer, P. C.; and Yang, J. C.-Y. 1997. Multiterminal binary decision diagrams: An efficient data structure for matrix representation. *Formal Methods in System Design* 10(2/3):149–169.

Glynn, P. W. 1989. A GSMP formalism for discrete event systems. *Proceedings of the IEEE* 77(1):14–23.

Guestrin, C.; Koller, D.; and Parr, R. 2002. Multiagent planning with factored MDPs. In *Advances in Neural Information Processing Systems 14: Proc. 2001 Conference*. Cambridge, MA: The MIT Press. 1523–1530.

Hoey, J.; St-Aubin, R.; Hu, A.; and Boutilier, C. 1999. SPUDD: Stochastic planning using decision diagrams. In *Proc. Fifteenth Conference on Uncertainty in Artificial Intelligence*, 279–288. Morgan Kaufmann Publishers.

Howard, R. A. 1960. *Dynamic Programming and Markov Processes*. New York, NY: John Wiley & Sons.

Howard, R. A. 1971. *Dynamic Probabilistic Systems*, volume II. New York, NY: John Wiley & Sons.

Jensen, A. 1953. Markoff chains as an aid in the study of Markoff processes. *Skandinavisk Aktuarietidskrift* 36:87–91.

Marie, R. 1980. Calculating equilibrium probabilities for $\lambda(n)/C_k/1/N$ queues. In *Proc. Performance '80*, 117–125. ACM Press.

Matthes, K. 1962. Zur Theorie der Bedienungsprozesse. In *Trans. Third Prague Conference on Information Theory, Statistical Decision Functions, Random Processes*, 513–528. Publishing House of the Czechoslovak Academy of Sciences.

Neuts, M. F. 1981. *Matrix-Geometric Solutions in Stochastic Models: An Algorithmic Approach*. Baltimore, MD: Johns Hopkins University Press.

Osogami, T., and Harchol-Balter, M. 2003. A closed-form solution for mapping general distributions to minimal PH distributions. In *Proc. 13th International Conference on Modelling Techniques and Tools for Computer Performance Evaluation*, volume 2794 of *LNCS*, 200–217. Springer.

Puterman, M. L. 1994. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. New York, NY: John Wiley & Sons.

Rintanen, J. 2003. Expressive equivalence of formalism for planning with sensing. In *Proc. Thirteenth International Conference on Automated Planning and Scheduling*, 185–194. AAAI Press.

Sauer, C. H., and Chandy, K. M. 1975. Approximate analysis of central server models. *IBM Journal of Research and Development* 19(3):301–313.