Special subclass of Generalized Semi-Markov Decision Processes with discrete time

Alexander Frank

January 8, 2020

Special subclass of Generalized Semi-Markov Decision Processes with discrete time

Alexander Frank

1 Introduction

Many planning problems with stochastic uncertainty can be modelled as Markovian Decision Processes. Because of their condition to be memoryless the resulting agent assigns an optimal action in a given state and released time by paying attention to the gaining rewards and the future states. Those processes are used in stochastic games, network planning, robotics and further more. Discrete- and continuous-time Markov Decision Processes (MDPs and CTMDPs) can be solved efficiently with policy iteration or linear programming, [6].
One more universal class of decision problems is given by Generalized Semi-Markov Decision Processes (GSMDPs). The formalism in this article is similar to the definition by [7], based on previous definitions of GSMPs by [3]. In this class of problems we have several events, which can be triggered. Those events cause transitions from state to state and achieve some rewards. It is possible that for a period of time no event is triggered and the agent only knows the progressing clocks, so there are different sojourn times. By adding a choice of actions affecting the active set of clocks the agent has to make a decision for the underlying problem.
There are only a few articles about continuous time GSMDPs. [2] examine a generalized model of Stochastic Automate (SA) with clocks, which are triggered asynchronously, activating transitions in the SA. By the Kronecker product clock states are combined to handle their interaction. Similarly [7], defined asynchronous events by continuous phase-type distributions (PHDs). Events are triggered and affect the underlying Markovian problem. They bring all active events in relationship and calculate their coherent probabilities to trigger one of them without losing the current progress of the others. Based on that article and a previous of [5], an approximative planner for solving deliberation scheduling problems was build using results for GSMDPs in [4].

Alexander Frank
TU Dortmund, 44227 Dortmund, e-mail: alexander.frank@tu-dortmund.de

To the best of my knowledge, there are no research articles written up to now about discrete time GSMDPs. The fact that there are discrete time steps, in which several events can be released, leads to a high dimensional problem. Even if the events have a certain order to be worked off, the agent has to consider over an exponential number of possible event combinations. This paper is focused on a special subclass of discrete time GSMDPs. The first limitation is that once an action is chosen in a state it is fixed until at least one event is triggered. The second limitation is that all progress for all events is lost if at least one single event is triggered.
The complexity is still PSPACE hard, but in this paper two randomized algorithms in polynomial runtime are introduced and analysed. Some backgrounds and a completed formulation for discrete time GSMDPs are given. After that, the two randomized algorithms are explained and in the last section the results are discussed.

2 Background and Definitions

In this section, basic definitions and problem formulations are introduced. In general $\mathbb{P}(X)$ is the probability of X and $\mathbb{E}(X)$ is the expected reward. Bold letters are for linear functions (like $\mathbf{P}$) and calligraphic letters (like $\mathscr{S}$) are used for sets. $\mathbf{1}$ and $\mathbf{0}$ are vectors only consisting of 0 or rather 1 (where i have to say that the dimension is always logically conceivable).

2.1 Markov Decision Process

A tuple of $(\mathscr{S},\mathscr{A},\mathbf{P},\mathbf{R},\mathbf{p}_0)$ defines a discrete Markov Decision Process (MDP) where $\mathscr{S}$ is a finite set of states, $\mathscr{A}$ is a finite set of actions, $\mathbf{P}:\mathscr{S}\times\mathscr{A}\times\mathscr{S}\to[0,1]$ is the transition function for moving from s to s' choosing action a and is used as a set of stochastic matrices $\mathbf{P}(s,a,s')\equiv\mathbf{P}^a(s,s')$, so that the condition $\sum_{s'\in\mathscr{S}}\mathbf{P}^a(s,s')=1$ is fulfilled for all $s\in\mathscr{S}$. Furthermore $\mathbf{R}:\mathscr{S}\times\mathscr{A}\times\mathscr{S}\to\mathbb{R}$ is the reward function and is also used as a set of matrices $\mathbf{R}(s,a,s')\equiv\mathbf{R}^a(s,s')$. At least $\mathbf{p}_0\in\mathbb{R}^{1\times|\mathscr{S}|}$ is the initial distribution over all states.
By mapping actions to states the agent produces a policy $\pi(s,t)=a$, depending also on the past time $t\in[0,T]$, or a pure policy $\pi:\mathscr{S}\to\mathscr{A}$, if the time has no relevance. Let Π be the set of all policies. For an explicit policy π several paths $\sigma=<s_0,s_1,s_2,...s_T>$ in the MDP are enabled, with $\mathbf{p}_0(s_0)>0$ and to every pair of following states $<s_i,s_{i+1}>$ applies $\mathbf{P}^{\pi(s_i,i)}(s_i,s_{i+1})>0$. The set over all possible paths (for a given policy) is called Σ^π. A discounting factor $\beta\in(0,1]$ may be added to reduce the weights of future decisions. For an infinite horizon β is necessary.
The optimization criteria for the agent is to find an optimal policy π maximizing the discounted rewards over all possible paths in infinite horizon (there might be other optimization criteria, but they are not of interest in this paper). Intentionally the formalism for the optimization criteria depends on the set of possible paths Σ^π,

because a similar formalism for zero-steps will be introduced later.

$$\max_{\pi\in\Pi} \sum_{\sigma\in\Sigma^{\pi}} \mathbb{P}(\sigma|\pi) \sum_{i=0}^{T-1} \beta^{i}\mathbf{R}^{\pi(s_i,i)}(s_i,s_{i+1}). \tag{1}$$

There are some options to solve MDPs like policy iteration and linear programming. These methods are exact and solve MDPs in a polynomial time. Much more information about MDPs can be found in [6].

2.2 Generally Semi-Markov Decision Process

GSMDPs are defined as a tuple of $(\mathscr{S},\mathscr{A},\mathscr{E},\mathbf{C},\mathbf{P},\mathbf{R},F)$. As in section 2.1 , $\mathscr{S}$ and $\mathscr{A}$ are sets of states and actions. $\mathscr{E}$ is an extension and a set of independent events which are triggered with a probability given by $F(t,e)$ for a discrete passed time $t\in\mathbb{N}$ since activation of the event. $\mathscr{E}_0$ includes the trivial event e_0, that nothing happens. The function $\mathbf{C}:\mathscr{S}\times\mathscr{A}\times\mathscr{E}\to\{0,1\}$ specifies if an event $e\in\mathscr{E}$ is active $\mathbf{C}(s,a,e)=1$ or inactive $\mathbf{C}(s,a,e)=0$ in a given state and a chosen action. The transition function $\mathbf{P}:\mathscr{S}\times\mathscr{E}\to\mathscr{S}$ declares the full known following state, if an event $e\in\mathscr{E}$ is triggered in $s\in\mathscr{S}$. Also the rewards depend on the occurred events $\mathbf{R}:\mathscr{S}\times\mathscr{E}_0\times\mathscr{S}\to\mathbb{R}$, however it is sufficient to know the current state and event.
The agent has to make decisions identifying the active events. Then discrete time steps are made until the first event is triggered. Furthermore, all other active events can be triggered in the same time step. For a given order (or rather with decreasing priority) the system is affected by the events so that the status of events can be changed, transitions switches the state and rewards are gained. An optimal decision earns optimal rewards heeding to the next status of the system. So the policy $\pi:\mathscr{S}\times\mathbb{N}^{|\mathscr{E}|}$ maps an action to the given state and the passed event times since an event gets active and has not been triggered before.

Figure 1 shows a small example for the transition graph with four states and three events. $\mathbf{C}(\cdot,a,\cdot)$ is visualized by the set of edges, so all events not belonging to an edge are blocked (like $\mathbf{C}(s_2,a,e_1)=0$). If the current state is s_1 and all events are triggered a zero-step path $\gamma=<s_1,e_1,s_2,e_2,s_3,e_3,s_1>$ is produced for the system transitions in actual a single time step. So the following state for the next decision is s_1, the same as when only e_3 is released with $\gamma=<s_1,\overline{e_1},s_1,\overline{e_2},s_1,e_3,s_1>$, but this path gains different rewards.
Those paths γ are paths in the zero-step-phase, which are handled between two decisions. All possible paths γ during the zero-step-phase are parts of the set of regular zero-steps Γ^*. Every regular path $\gamma=<s_0,x_1,s_1,x_1,...,x_{|\mathscr{E}|},s_{|\mathscr{E}|}>$ with $x_i\in\{e_i,\overline{e_i}\}$ is also well defined for a given initial state s_0 by the formula $\gamma=<x_1,x_2,...,x_{|\mathscr{E}|}>$.
The reward for a zero-step path γ is computed additively

$$\mathbf{R}(\gamma)=\sum_{i:\ x_i=e_i}\mathbf{R}(s_i,e_i,s_{i+1}) \quad\text{or}\quad \mathbf{R}(\gamma)=\mathbf{R}(s_0,e_0,s_0). \tag{2}$$

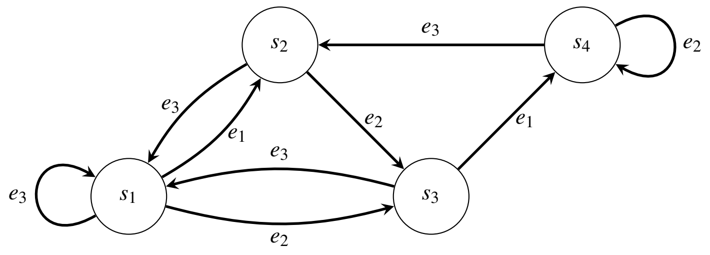


Fig. 1 Example: Transition graph for a fixed action, 4 states and 3 events

For the small example in Fig. 1 and s_3 as the initial state the set of possible paths is

$$\Gamma^*(s_3,a) = \{< e_1, \overline{e_2}, e_3 >, < e_1, \overline{e_2}, \overline{e_3} >, < \overline{e_1}, \overline{e_2}, e_3 >, < \overline{e_1}, \overline{e_2}, \overline{e_3} >\}.$$

So it is possible to transfer in every state $s \in \mathscr{S}$ before the next action has to be decided.

The optimization criteria is also defined with the set of all paths Σ and zero-step paths Γ^*. Therefore $\Gamma^*(s,a,s')$ is defined as the set of all regular zero-step paths starting in s and ending in s' by choosing action a. Also $\mathbf{t}$ means the actual progress in the current time step i.

$$\max_{\pi\in\Pi} \sum_{\sigma\in\Sigma^\pi} \mathbb{P}(\sigma|\pi) \sum_{i=0}^{T} \beta^i \sum_{\gamma\in\Gamma^*(s_i,\pi(s_i,\mathbf{t}),s_{i+1})} \mathbb{P}(\gamma|\pi(s_i,\mathbf{t}),\mathbf{t}) \cdot \mathbf{R}(\gamma) \tag{3}$$

2.3 Resetting discrete GSMDPs

Discrete GSMDPs are very difficult to solve, due to an exponential huge definition amount the agent has to handle. Even if the passed time has an upper bound for each event forcing it to be triggered, the system is too huge to be solved in polynomial time.
A resetting discrete GSMDP ($GSMDP_\mathbf{0}$) has the same definition as a GSMDP with two more restrictions:

a. If one or more events are triggered, before they are inactivated (per action or in transitions of zero-steps), all progress of each event is set to 0 after the zero-step-phase.
b. When entering a state after one or more events are released the agent has to make a decision for an action. This action is not able to be changed until the next regular event is triggered.

Due to these two restrictions the agent only has to find a policy $\pi : \mathscr{S} \to \mathscr{A}$. The vector for the progress time in GSMDPs can be seen as $\mathbf{t} := t \cdot \mathbf{C}(s, a, \cdot)$. Nevertheless the problem is further hard to solve, due to the evaluation of zero-steps.
At least every time the agent has to make a decision the progress vector $\mathbf{t} \in \mathbb{N}^{1\times|\mathscr{E}|}$ is $\mathbf{0}$. This criteria makes it possible to create an approximating model in polynomial time. In the conclusion some approaches for future algorithms are presented, solving discrete GSMDPs without specifications.

2.4 Discrete Phase-Type Distributions

Acyclic discrete Phase-Type distributions (ADPHs) are introduced in [1]. They are used to simulate the progress of the events. It is obvious that discrete distributions are suitable for a GSMDP, because of the discrete time steps, but not necessary. The focus of this paper is the zero-step-phase, so ADPHs are an adequate choice by increasing the probability to be released.
ADPHs guarantee that the probability to come in the absorbing phase s_{n+1} is monotonous increasing. Also every ADPH has a unique representation in a canonical form with an initial vector $\mathbf{q} \in [0,1]^{1\times n}$ and a probability matrix $\mathbf{Q}$:

$$\mathbf{Q} = \begin{pmatrix} \tilde{\mathbf{Q}} & \begin{pmatrix} \mathbf{0} \\ q_n \end{pmatrix} \\ \mathbf{0} & 1 \end{pmatrix}, \quad \tilde{\mathbf{Q}} = \begin{pmatrix} (1-q_1) & q_1 & 0 & 0 & \dots & 0 \\ 0 & (1-q_2) & q_2 & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots & & \vdots \\ 0 & 0 & 0 & 0 & \dots & q_{n-1} \\ 0 & 0 & 0 & 0 & \dots & (1-q_n) \end{pmatrix} \tag{4}$$

with $\sum_{i=1}^{n} \mathbf{q}(i) = 1$ and $0 \le q_1 \le q_2 \le \dots \le q_n \le 1$. In this paper ADPHs are used to describe the progress of the events. So the probability of triggering event e, if it is activated since t time steps, is

$$\mathbb{P}(e|t) = 1 - ||\mathbf{q}\tilde{\mathbf{Q}}^t||_1 = 1 - \sum_{i=1}^{n} \mathbf{q} \cdot \left(\tilde{\mathbf{Q}}^t(\cdot, i)\right). \tag{5}$$

.

3 Randomized Approaches

Now the basic definitions are explained and a closer look at the analysis of $GSMDP_0$s is possible. The first question is: What happens in a discrete time step of our model? The behaviour of the model in a discrete time step is defined as zero-step-phase. Nevertheless there is also the opportunity that no event is released in this time step. The set of all regular zero-steps is Γ^* with $|\Gamma^*| \le 2^{|\mathscr{E}|}$, on the other hand Γ is the

set of all paths, regular or not.
At least two randomized approaches to solve discrete *GSMDP*$_0$s approximate in polynomial time are given. Both solve every instance exact if their input value for the bounding capacity is unlimited.

3.1 Zero-Step-Phase

This phase is the main focus of this paper, because the zero-step-phases make GSM-PDs so difficult. For the analysis of the zero-step-phases the current state $s \in \mathscr{S}$, only a single available action $a \in \mathscr{A}$ is considered and the past time since no event has been triggered $t \in \mathbb{N}$ is known. Also for every event $e \in \mathscr{E}$ the ADPH is known as tuple $(\mathbf{q}_e, \mathbf{Q}_e)$, so the probability for a triggering event can be computed like in equation 5.

Not all events are active for a fixed combination of (s,a). The set of active events in s under a is defined as

$$\mathscr{E}_{act}(s,a) := \{e \in \mathscr{E} \mid \mathbf{C}(s,a,e) = 1.\} \tag{6}$$

The probability that no event is triggered for the progress time t is

$$\mathbb{P}(e_0|\mathbf{t}) = \prod_{e \in \mathscr{E}_{act}} ||\mathbf{q}_e \tilde{\mathbf{Q}}_e^{\mathbf{t}(e)}||_1 \tag{7}$$

In the other case one or more events are released. All in all there are $2^{|\mathscr{E}_{act}|}$ possibilities of combinations of events, which are triggered or stay in progress. For a given zero-step path $\gamma \in \Gamma$ with $\gamma =< s_0, x_1, s_1, x_2, ..., x_{|\mathscr{E}|}, s_{|\mathscr{E}|} >$ and $x_i \in \{e_i, \overline{e_i}\}$ the correctness has to be evaluated. Also the probability $\mathbb{P}(\gamma|a,\mathbf{t})$ and rewards $\mathbf{R}(\gamma)$ of a path can be computed. The special path $< e_0 >:=< s_0, \overline{e_1}, s_0, \overline{s_0}, ..., \overline{e_{|\mathscr{E}|}}, s_0 >$ is defined for no triggering event.

Definition 1. A path $\gamma \in \Gamma$ is regular for an action $a \in \mathscr{A}$ and a progress vector $\mathbf{t}$ (for a decreasing priority), if and only if

$$\forall i \in \{1, ..., |\mathscr{E}|\} : (x_i = e_i) \Rightarrow (\forall j < i : \mathbf{C}(s_j, a, e_i) = 1). \tag{8}$$

So it has to be verified that the event is not set inactive before the priority of this event is high. As an example you want to buy several things online in one session, but when you come to the fourth article, it is already sold out.

Definition 2. The probability of a regular zero-step path $\gamma \in \Gamma^*$ depends on the probability that e is triggered in γ and equation 5, so it is $\mathbb{P}(e_i|\gamma,a,\mathbf{t}) = \mathbb{P}(e_i|\mathbf{t}(e_i)) \cdot \prod_{j=0}^{i-1} \mathbf{C}(s_j, a, e_i)$. The probability for the path now is given by

$$\mathbb{P}(\gamma \mid a, \mathbf{t}) = \prod_{i:x_i=e_i} \mathbb{P}(e_i|\gamma,a,\mathbf{t}) \cdot \prod_{i:x_i=\overline{e_i}} (1 - \mathbb{P}(e_i|\gamma,a,\mathbf{t})) \tag{9}$$

Definition 3. The additively gained rewards are already defined in 2. For the planing of the agent it is important to calculate correctness, probabilities and rewards for all $\gamma \in \Gamma^*$. If multiple paths end in a state s' the probabilities can be summarized as

$$\mathbb{P}(s'|s,a,\mathbf{t}) = \sum_{\gamma:\ \gamma(s_{|\mathscr{E}|})=s'} \mathbb{P}(\gamma|a,\mathbf{t}). \tag{10}$$

On the other hand the rewards are summarized with weights in relation to their probabilities

$$\mathbf{R}(s,a,\mathbf{t},s') = \mathbb{P}(s'|s,a,\mathbf{t})^{-1} \cdot \sum_{\gamma:\ \gamma(s_{|\mathscr{E}|})=s'} \mathbb{P}(\gamma|a,\mathbf{t}) \cdot \mathbf{R}(\gamma). \tag{11}$$

Since all possibilities and rewards are computed, the agent has total knowledge about the future status of the system. With these information an optimal decision can be made to collect discounted rewards.

3.2 Randomized Γ-Method

The first approach to avoid an exponential number of zero-steps is to limit the set of active unset events like in algorithm 1. Generally there are $|\mathscr{E}_{act}|$ events which can be triggered or not, leading to a set Γ_{act} with $|\Gamma_{act}| = 2^{|\mathscr{E}_{act}|}$ different paths (also with irregular paths).
The main idea of the Γ-method 1 is to fix so many events randomly in step *randomize* of the algorithm depending on their probability, that the set of the other events $\mathscr{E}_{rest}$ fulfils $2^{|\mathscr{E}_{rest}|} \leq \Omega$. The more the probability of an event is near to 0 or 1, the more it is fixed randomly by the method. So for all active unset events e is the probability to be fixed $|\mathbb{P}(e) - 0.5|$ normed by the total of all active unset events. Also the fixed value is randomly chosen equal to the triggering probability.
So in the approximation step $\mathscr{E}_{act}$ is split into $\mathscr{E}_{rest}, \mathscr{E}_0$ and $\mathscr{E}_1$, where the last define sets of events specified to be 0 or 1. Now the main loop of the following algorithm has an upper bound of Ω.
The update step in algorithm 1 is similar to the equations 10 and 11. The probabilities are summarized and the rewards are summed with weights of their probabilities. This method has a running time in $\Theta(\Omega \cdot |\mathscr{E}|^2 + |\mathscr{S}|)$. Moreover $\Omega = 2^{|\mathscr{E}_{act}|}$ leads to the exact solution and calculates all possible paths in the zero steps.

3.3 Randomized $\mathscr{E}$-Method

The other algorithm 2 based on a totally different structure. This time all steps for a single event are evaluated and saved in a double sorting list. The higher priority is the actual state and the lower sorting priority is for blockings from $\mathbf{C}$.

```
algorithm: zero_steps_$\Gamma$
input : $(\mathscr{S}, \mathscr{E}_{act}, \mathbf{P}, \mathbf{C}, \mathbf{R}, F)$,   $(\tilde{s}, a, \mathbf{t}) \in \mathscr{S} \times \mathscr{A} \times \mathbb{N}_0^{|\mathscr{E}|}$,   $\Omega \in \mathbb{N}$
output: $L$ list of reachable $s$, probabilities and rewards
$L(s,\cdot,\cdot) \leftarrow [s,0,0]$;                       // $\forall s \in \mathscr{S}$ ;
$(\mathscr{E}_{rest}, \mathscr{E}_0, \mathscr{E}_1) \leftarrow randomize(\mathscr{E}_{act}, \tilde{s}, a, \mathbf{t}, \Omega)$        // explained in 3.2;
$\Gamma' \leftarrow \mathscr{P}(\mathscr{E}_{rest}) \backslash \{< e_0 >\}$;
for $\gamma \in \Gamma'$ do
    if $\gamma$ is regular (8) then
        $L$ is updated with results of $\gamma$          // explained in 3.2;
    end
end
```

Algorithm 1: zero-steps over randomized paths

So at the time point when event $e \in \mathscr{E}_{act}$ is evaluated all list entries become an update for triggering and for staying in progress. The list size will grow by factor 2 in every iteration, so again we limit the size to Ω.
In general two entries which are at the same state after an iteration step are not able to be combined, cause they walked different paths and passed different $\mathbf{C}(\cdot, a, \cdot)$. With an additional function to find and combine same acting entries for all future iterations the list is kept small. So if $\mathscr{S}$ has no exponential size and there is no upper bound Ω the list size will increase to $2^{|\mathscr{E}_{act}|/2}$ and not later than this point it will shrink in every iteration until there are not more than $|\mathscr{S}|$ entries. The combination is equal to the proceeding in 3.2
If the list size of Ω is exceeded (only possible by an single element) the algorithm discards randomly an entry depending on their current probabilities. As well as the Γ-method the algorithm solves the zero-steps exactly if Ω is great enough.
The last line in the algorithm is to correct the influence of $< e_0 >$. On the one hand it is possible to stay in the initial state $\tilde{s}$ on the other hand it is possible to join the state per a chain of transitions (zero-steps). But in the first case the progress is increased by 1 and otherwise $\mathbf{t}$ is reset to 0. So both cases has to be separated.

The running time of the $\mathscr{E}$-method 2 is defined in $\Theta(|\mathscr{E}| \cdot \Omega^3 + |\mathscr{S}|)$. However the expected accuracy of this method is better compared to the Γ-method for the same Ω, because there are no irregular paths consuming resources and combined entries imply that more regular paths can be evaluated.

3.4 Transformation to a MDP

The chance that no event triggers in a state s by choosing action a in $t \in \mathbb{N}$ time steps converges to 0, because of the ADPHs. For a negligible error of $\varepsilon > \mathbb{P}(e_0|\mathbf{t})$ the progress time t gets an upper bound of $\theta(s,a) \in \mathbb{N}$ for every tuple of state and action. The new set of states $\tilde{\mathscr{S}}$ consists of

```
algorithm: zero_steps_$\mathscr{E}$
input : $(\mathscr{S},\mathscr{E}_{act},\mathbf{P},\mathbf{C},\mathbf{R},F)$,   $(\tilde{s},a,\mathbf{t}) \in \mathscr{S}\times\mathscr{A}\times\mathbb{N}_0^{|\mathscr{E}|}$,   $\Omega \in \mathbb{N}$
output: $L$ list of reachable $s$, blockings, probabilities and rewards
$L1 \leftarrow [\tilde{s},\mathscr{E}_{act},0,0]$;;
for $e \in \mathscr{E}_{act}$ do
    for $l \in L$ and $e$ unevaluated for $l$ do
        if $e$ in $l(2)$ inactive then
            update $l$ with $e$ blocked;
        else
            $s_{new} \leftarrow \mathbf{P}(l(1),e)$;
            $neu \leftarrow [s_{new}, l(2)\cdot\mathbf{C}(s_{new},a,\cdot), l(3)\cdot\mathbb{P}(e), l(4)+\mathbf{R}(l(1),e,s_{new})]$;
            $l(3) \leftarrow l(3)\cdot(1-\mathbb{P}(e))$;
            $L \leftarrow$ insert$(L,neu,\Omega)$                // explained in 3.3;
        end
    end
    $L \leftarrow$ combine_entries(L)                         // explained in 3.3;
end
Correct the entry $l(1)=\tilde{s}$                            // explained in 3.3;
```

Algorithm 2: zero-steps over events

$$\tilde{\mathscr{S}} = \bigcup_{s\in\mathscr{S}} (s,0) \quad \cup \bigcup_{s\in\mathscr{S},a\in\mathscr{A},t\in\mathbb{N}_{\le\theta(s,a)}} (s,a,t).$$

For every state (s,a,t) the zero-step-phase has to be evaluated. With the results of the zero-step-phases a MDP $(\tilde{\mathscr{S}},\mathscr{A},\tilde{\mathbf{P}},\tilde{\mathbf{R}})$ can be built with the expected transition probabilities and rewards. The entries in the list describes the probabilities $\mathbb{P}(s'|s,a,t)=\tilde{\mathbf{P}}((s,a,t),a,(s',0))$ and the collected rewards $\tilde{\mathbf{R}}((s,a,t),a,(s',0))$. By adding expensive penalties for switching a chosen action a' in a given tuple (s,a,t) and no transition happens the second restriction is guaranteed. At least transitions from the states $(s,0)\to(s,a,1)$ and $(s,a,t-1)\to(s,a,t)$ have to be computed with $(t-1)$ and equation 7.
By using one of the presented methods (3.2 or 3.3) a MDP is build in a time based on the method and $\tilde{\mathscr{S}}$. Of course ADPHs can be built so that $\tilde{\mathscr{S}}$ is enormous, but in general the upper bound θ is reached super exponentially.

4 Experiments

In this section the results for both randomized methods are shown. Their solutions will be compared to each other and to the exact ones using the $\mathscr{E}$-method with $\Omega = 2^{|\mathscr{E}|}$.
The test instances are randomized in transitions and transition probabilities, the order of ADPHs is normally distributed with expectation 5 and variance 1. The entries of ADPHs are randomized exponentially. The rewards are equally distributed just about $[-|\mathscr{E}|,|\mathscr{E}|]$ and also the entries $\mathbf{C}(s,a,e)\in\{0,1\}$ have the same probability.

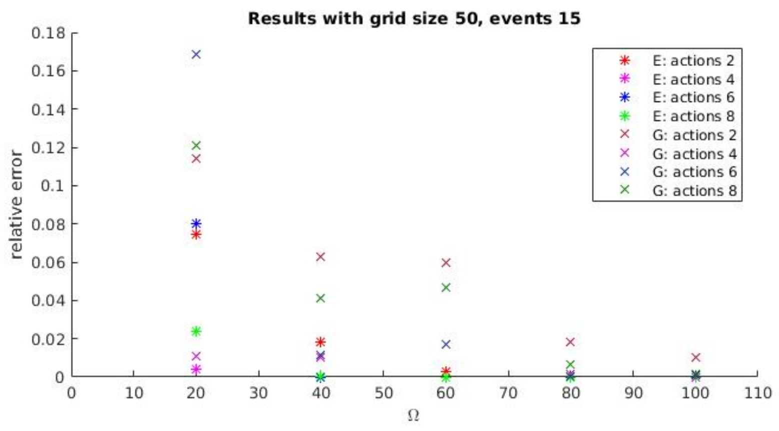
The instances are build for $|\mathscr{S}| = \{50, 100\}$, different number of actions $|\mathscr{A}| = \{2, 4, 6, 8\}$ and various events $|\mathscr{E}| \in \{15, 20, 25\}$ (for greater $\mathscr{E}$s the exact solution can not be computed with the used computers).
Both randomized approaches run 10 times for a single instance and for 10 different instances. Over all results for a fixed number of states, actions and events the average values are calculated and presented. Here the $\mathscr{E}$-method is shorten with E and the Γ-method is named G.

In figure 2 the results for different actions and different Ωs are presented, with a grid size ($|\mathscr{S}|$) of 50 and several numbers of events. There is no obvious pattern for a specific influence by the number of actions. But as suspected the $\mathscr{E}$-method has mostly a smaller relative error to the exact solution, which is also cut with an error below ε, than the Γ-method for an equal Ω. Also it is visible that the relative error raises for a greater set of $\mathscr{E}$. Of course there are more unnoticed zero-step-paths with a higher number of events. In general the quality of the solutions gets better for a greater Ω. Here it should be mentioned that for $\Omega = |\mathscr{S}| \cdot |\mathscr{E}|$ always all relative errors are lower than 10^{-5}.
The next figure 3 shows the average results for fixed sets of $\mathscr{S}$ and $\mathscr{A}$. It confirms the previous statements that the goodness of the algorithms for a fixed Ω is decreased. Several tests confirm, similar to the runtime of the algorithms, neither the size of $\mathscr{S}$ nor $\mathscr{A}$ is really relevant for the quality of both approaches.

Table 1 Relative run times for $|\mathscr{S}| = 100$ and $|\mathscr{A}| = 4$

method and $\lvert\mathscr{E}\rvert$	$\Omega = 20$	$\Omega = 40$	$\Omega = 60$	$\Omega = 80$	$\Omega = 100$
$\mathscr{E}$-method, $\lvert\mathscr{E}\rvert = 15$	0.985	0.973	0.981	0.985	1.004
Γ-method, $\lvert\mathscr{E}\rvert = 15$	0.791	1.379	1.385	2.543	2.531
$\mathscr{E}$-method, $\lvert\mathscr{E}\rvert = 20$	0.840	0.953	0.971	0.988	0.992
Γ-method, $\lvert\mathscr{E}\rvert = 20$	0.368	0.689	0.689	1.364	1.365
$\mathscr{E}$-method, $\lvert\mathscr{E}\rvert = 25$	0.607	0.863	0.929	0.946	0.957
Γ-method, $\lvert\mathscr{E}\rvert = 25$	0.204	0.303	0.305	0.613	0.610

The run times in Table 1 are relative to the time an exact model takes time to be created and solved. Because there is no force to combine list elements of the $\mathscr{E}$-method, it is possible that the polynomial approach takes a longer time as presented in the last column. It is shown that the Γ-method is really faster than the $\mathscr{E}$-method, if the upper bound Ω is far away from $2^{|\mathscr{E}|}$, otherwise if both algorithms compute almost the exact zero-steps, Γ-method takes much longer. This results in the fact, that the list for the $\mathscr{E}$-method is naturally limited by $\sqrt{2^{|\mathscr{E}|}}$ and the Γ-method has no smaller natural bound than $2^{|\mathscr{E}|}$.



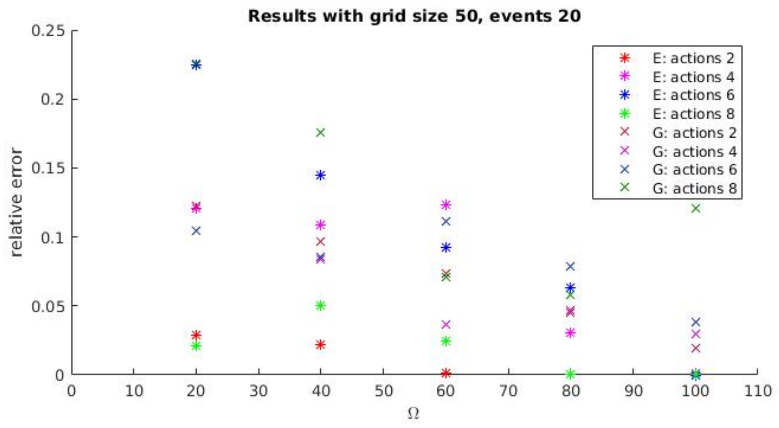


Fig. 2 Average results for tests with $|\mathscr{S}| = 50$

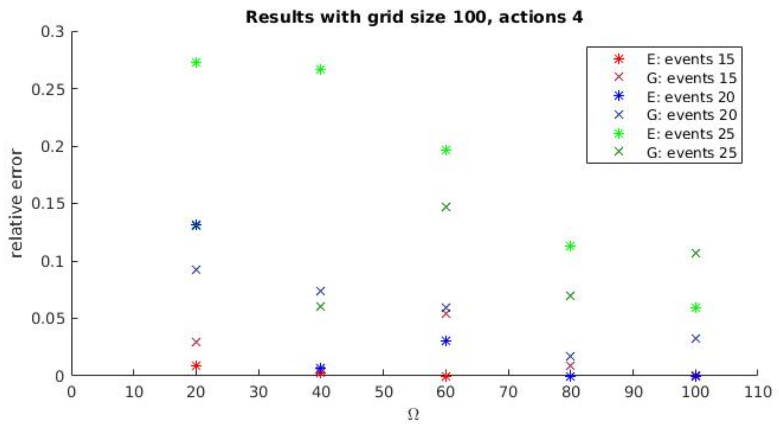


Fig. 3 Average results for tests with $|\mathscr{S}| = 100$ and $|\mathscr{A}| = 4$

5 Conclusion

The experiments show, that both approaches have their own advantages. Generally very small run times are possible with the Γ-method in exchange to the quality compared to the exact solution. On the other hand are the results of the $\mathscr{E}$-method for the same Ω closer to exact ones, but it takes more time. Both algorithms have a polynomial time to evaluate the zero-steps and can be used to transform a $GSMDP_0$ to an approximating MDP.
There are also polynomial methods to compute an approximative degree Δ of the problem instance, which gives more information about all regular paths Γ^*:

$$|\Gamma^*| = 2^{\Delta^*} \leq 2^{\Delta} \leq 2^{|\mathscr{E}|}. \tag{12}$$

This information can be used to set Ω in relation to the degree of an instance, because in all test instances with $|\mathscr{E}|$ up to 25 it unnecessary to set $\Omega = |\mathscr{S}| \cdot |\mathscr{E}|$.
Future work has to be focussed on the class of normal GSMDPs, which are more complex. There is an exponential number of combinations of states and different progress times. So new approaches will be needed to decrease the decision space by aggregating progress times or selecting representative states. If that will be successful, the algorithms in this paper with small additions, can be used to evaluate the zero-step-phase and to transform the problem in a MDP.

References

1. Bobbio, A., Horvth, A., Scarpa, M., Telek, M.: Acyclic discrete phase type distributions: properties and a parameter estimation algorithm. Performance Evaluation **54**(1), 1 – 32 (2003)
2. Buchholz, P., Kriege, J., Scheftelowitsch, D.: Model checking stochastic automata for dependability and performance measures. In: 2014 44th Annual IEEE/IFIP International Conference on Dependable Systems and Networks, pp. 503–514. IEEE (2014)
3. Glynn, P.W.: A gsmp formalism for discrete event systems. PROCEEDINGS OF THE IEEE **77**(1) (1989)
4. Krebsbach, K.D.: Deliberation scheduling using gsmdps in stochastic asynchronous domains. International Journal of Approximate Reasoning **50**(9), 1347 – 1359 (2009). Special Track on Uncertain Reasoning of the 19th International Florida Artificial Intelligence Research Symposium (FLAIRS 2006)
5. Musliner, D.J., Goldman, R.P., Krebsbach, K.D.: Deliberation scheduling strategies for adaptive mission planning in real-time environments. In: AAAI Spring Symposium: Metacognition in Computation (2005)
6. Puterman, M.L.: Markov decision processes: Discrete stochastic dynamic programming (1994)
7. Younes, H.L., Simmons, R.G.: Solving generalized semi-markov decision processes using continuous phase-type distributions. In: AAAI, vol. 4, p. 742 (2004)