Reinforcement Learning for MDPs with Constraints

Peter Geibel

Institute of Cognitive Science, AI Group, University of Osnabrück, Germany
pgeibel@uos.de
www.cs.tu-berlin.de\~geibel

Abstract. In this article, I will consider Markov Decision Processes with two criteria, each defined as the expected value of an infinite horizon cumulative return. The second criterion is either itself subject to an inequality constraint, or there is maximum allowable probability that the single returns violate the constraint. I describe and discuss three new reinforcement learning approaches for solving such control problems.

1 Introduction

Most approaches in reinforcement learning (RL, see e.g. [8]) consider only Markov decision processes (MDPs) with a single criterion, or with several criteria related to hierarchical dependencies between behaviors. On the other hand, in practical applications like robot control, there might exist several possibly *conflicting* objectives requiring a strategy that mediates between them. Problems with multiple, non-hierarchical objectives have hardly been considered in RL, although some articles from the field of dynamic programming (DP, [2]) can be found.

A typical example for a problem with constraints is the accomplishment of some task with a limited amount of energy or time expressed as a second criterion subject to a constraint. Imagine e.g. a robot equipped with a battery. The task of the robot is to collect as much items as possible, but it shouldn't run out of energy.

I will consider two different kinds of constraints. The first group of problems have a constraint on the second criterion function itself, i.e. on the *expected value* of the return. Such problems are typically called constrained MDPs `CMDPs` in the following (see also [1]). The second group contains problems in which we constrain the probability that the return, considered a random variable, violates a constraint. Such problems will be called MDPs with constrained probability of constraint violation (`CPMDP`). Examples are constraints on the probability of resource overutilization as discussed by Dolgov and Durfee in [4,3]. In `CPMDPs`, actually *two* constraints are involved.

Since applications with unequal discount factors are relatively rare and very difficult to solve [5,6,7], we will only consider MDPs with several criteria each based on its own reward function, but using a *common discount factor* γ. We will also focus on MDPs with two criteria only, where the first one is to be optimized, and the second one is subject to a constraint.

© Springer-Verlag Berlin Heidelberg 2006

The purpose of this paper is to undertake a description and comparison of different approaches for solving constrained problems (including some new ones), and to discuss their respective advantages and shortcomings. In section 2, unconstrained RL problems including Markov Decision Processes, policies, and value functions are introduced. In section 3, I consider MDPs with constraints, i.e. CMDPs and CPMDPs. I will present standard solutions methods as well as three new approaches for solving CMDPs and CPMDPs.

Each method has a parameter that allows to select different behaviours with respect to the first and second criterion function yielding a curve in a 2-dimensional space corresponding to the two criteria in the case of CMDPs I will base the experimental comparison in section 5 on this approach that can also be used for CPMDPs. A concluding discussion can be found in section 6.

2 Unconstrained MDPs

In RL and DP, one considers finite Markov decision processes (MDPs), that are characterized by a finite state set X, a finite action set U, and state transition probabilities $p_{x,u}(x')$ defined as the probabilty that x' is reached when u is executed in x. The value $r_{x,u}$ denotes the reward obtained when executing action u in state x.

A **policy** represents the action selection strategy of the agent. Stationary, deterministic policies are functions π mapping a state x to an action $\pi(x)$. Randomized policies are described using state dependent distributions $\pi(x,.)$ on possible actions.

The aim of the agent is to find a policy π for selecting actions that maximizes the cumulative reward, called the return. The return is defined as $R = \sum_{t=0}^{\infty} \gamma^t r_t$, where the random variable r_t denotes the reward occurring in the t-th time step when the agent uses policy π. Let $x_0, x_1, x_2, \ldots$ denote the corresponding probabilistic sequence of states, and u_i the sequence of actions chosen according to policy π.

The constant $\gamma \in [0,1]$ is a discount factor that allows to control the influence of future rewards. The expectation of the return $V^\pi(x) = \mathbb{E}\Big[R \,|\, x_0 = x\Big]$ is defined as the **value** of x with respect to π. It is well-known that there exist stationary, deterministic policies π^* for which $V^{\pi^*}(x)$ is optimal (maximal) for *every* state x simultaneously. The optimal values $V^*(x) := V^{\pi^*}(x)$ are the same for every optimal policy π^*.

In order to define optimal stationary policies, let D be an initial distribution on the possible starting states, e.g. the uniform distribution on the set X. We define the **value of a policy** as the expected value of the value function, i.e.

$$\mathcal{V}^\pi = \mathbb{E}_{x \sim D}\,\big[V^\pi(x)\big] = \sum_{x \in X} D(x) V^\pi(x)\,. \tag{1}$$

For a fixed distribution D, this value is maximized by any optimal stationary, deterministic policy.

While DP algorithms often assume a fully know model, RL algorithms like Q-Learning are able to learn in interaction with the real process. We suppose that the reader is familiar with basic RL techniques and leave out the defintion of the algorithm. If the model is known, then standard approaches can be applied. One approach consists in formulating a linear program and solve it with standard techniques, see [1,4].

3 Problems with Constraints

A constrained MDP has an additional **second reward function** $c_{x,u}$ that is used to define the constrained value function C^π, see below. In the case that $c_{x,u} \leq 0$ holds, these values can be considered costs for the actions, but positive values, i.e. rewards, might also occur.

We define constrained MDPs (`CMDPs`) as problems of the form

$$\begin{aligned} &\max \mathcal{V}^\pi \\ &s.t.\ \ \mathcal{C}^\pi \ \geq \ c \end{aligned}$$

where the threshold c is a real value, and $\mathcal{C}^\pi = \mathbb{E}\ C^\pi(x)$ with $C^\pi(x) = \mathbb{E} \sum_{t=0}^{\infty} \gamma^t c_{x_t,u_t}$. Problems with $\leq$ instead of $\geq$ can be normalized to yield the above form.

From a practical point of view, we often consider (A) *problems with maximum costs*, in which all $c_{x,u} \leq 0$ and $c \leq 0$, e.g. a robot task and risk-sensitive control as discussed by Geibel und Wysotzki in [7]; (B) *Problems with minimum gain,* in which all $c_{x,u} \geq 0$ and $c \geq 0$, e.g. the problem of Buridan's ass, see [6], where a minimum return must be achieved on average; (C) *Mixed Problems* where the $c_{x,u}$ might take on positive as well as negative values and c is arbitrary.

In the robot example, one can argue that the introduction of the expectation operator makes no sense, because we want the robot to *never* run out of energy, or only with a maximum allowable probability p_0, see [4]. Let C denote a random variable that denotes the $c_{x,u}$-based cumulative return occurred in a single run. Now we define MDPs with constrained probability of constraint violation (`CPMDPs`) as maximizing $\mathcal{V}^\pi$ under the condition

$$P(C \leq c') \leq p_0 \tag{2}$$

with c' being the threshold for the $c_{x,u}$-based return.

It should be noted, that for constrained problems it is no longer the case that stationary deterministic policies are optimal. First of all, we might need to consider randomized policies and a dependence on the initial distribution D. For `CMDPs` randomized optimal policies can be found by solving a modified linear program.

4 Solution Approaches for MDPs with Constraints

In the following, I will describe and discuss several approaches for solving MDPs with constraints. I will start with the DP approach because it constitutes a baseline for comparing the performance of the approaches.

4.1 `LinMDP`: Linear Programming

In order to solve a standard constrained MDP, a linear program describing optimal solutions of the unconstrained MDP is simply augmented by an additional constraint expressing $\mathcal{C}^\pi \geq c$ is required to hold.This augmented program can again be solved with standard linear programming methods. The method yields a randomized optimal policy dependent on the `CMDP` to be solved, and the initial distribution D. In the following we refer to this method as `LinMDP`.

`LinMDP` is tailored for problems with constraints on the expected costs. As described by Dolgov and Durfee in [4], it can be used for `CPDMPs` by mapping to a `CMDP` with constant $c = p_0 c'$ which is possible in the case of negative values of the $c_{x,u}$ (based on the Markov inequality).

Because `LinMDP` might be suboptimal for solving `CPMDPs`, we propose the following method: the policy π resulting from solving the linear `CMDP`-program has also a specific probability p_π for constraint violation. I.e. given a fixed c' (for the `CPMDP`), instead of setting $c = p_0 c'$ we can vary the c in the corresponding `CMDP`. We then pick that c which results in a feasible policy π for which $p_\pi \leq p_0$ holds and that has the highest $\mathcal{V}^\pi$-value. p_π, $\mathcal{C}^\pi$, and $\mathcal{V}^\pi$ can be estimated using several test runs. In the following, we will refer to this method for solving `CMDPs` as well as `CPMDPs` as `LinMDP`.

4.2 `WeiMDP`: A Weighted Approach

Geibel and Wysotzki in [7] considered the problem of finding policies that have a constrained risk of failing in an MDP with error states. We expressed the probability of entering an undesirable state as an (undiscounted) second value function. This resulted in a constrained MDP with possibly unequal discount factors, which was solved by introducing a weight parameter for risk and value. For solving CMDPs, we suggest to introduce a weight parameter $\xi \in [0, 1]$ and a derived weighted reward function defined as

$$w_{x,u} = \xi r_{x,u} + (1 - \xi) c_{x,u} \, .$$

For a fixed ξ, this new unconstrained MDP can be solved with standard methods, e.g. Q-Learning resulting in an online-method.

Similar to `LinMDP` parameterized with c, using different values of $\xi \in [0, 1]$ will result in different points in the $(\mathcal{V}, \mathcal{C})$-space (`CMDPs`) and $(\mathcal{V}, p_.)$-space (`CPMDPs`), respectively. This method will be called `WeiMDP` in the following.

Again, there is a parameter ξ for choosing a suitable policy dependent on c in the case of `CMDPs` and on `CPMDPs`. Unlike the approach proposed by Dolgov and Durfee, we make no prior assumption on the sign of $c_{x,u}$ and c, i.e. we can naturally treat mixed problems with mixed signs.

In contrast to `LinMDP`, our algorithm performs online learning of a stationary-determinstic optimal policy for the weighted criterion that is a feasible one for the `CMDP` (if the problem has a solution).

4.3 AugMDP: State Space Extension

The rewards $c_{x,u}$ correspond e.g. to costs like energy or time. It is RL folklore to include the status of the battery in the state description. Because we want to deal with finite state MDPs only, the possible values of the so far accumulated costs (e.g. consumed energy since $t = 0$) need to be discretized in an appropriate manner, e.g. by using intervals of equal length covering the possible range of values.

Since we are interested in the costs with respect to a starting state x_0, we need to keep track of the elapsed time if $\gamma < 1$. Otherwise the cost of the successor state cannot be computed correctly. If $i(0)$ is the interval corresponding to zero costs, the process starts in the state $(x_0, i(0), 0)$ where x_0 is a starting state of the original MDP. Given a current state $(x, i(C), t)$, a successor state obtained for action u might be $(x', i(C + \gamma^t c_{x,u}), t + 1)$ where $c_{x,u}$ is the cost incurred by the executed action u, and $i(C + \gamma^t c_{x,u})$ the new interval.

We don't have to consider the time if $\gamma = 1$ holds. But for $\gamma < 1$, the state space is possibly infinite. Assuming a maximum episode length of T and N intervals for discretizing the costs, we arrive at an MDP having $|X|NT$ states if $\gamma < 1$, and $|X|N$ states if $\gamma = 1$ holds.

In order to solve a CPMDP, we apply e.g. Q-learning using $r_{x,u}$ and the augmented state space. An **additional negative reward** $S \leq 0$ is given, when the process enters a state such that the accumlated costs are below c', i.e. when the cost constraint of the CPMDP is violated. Using high absolute values of S will prevent the process from entering such states at all, yielding a policy with a minimal p_0.

In order to deal with CPMDPs and also with CMDPs, we vary S in some sufficiently large interval. Again we have a parameter to "tune the behaviour" until we find a feasible policy for the CMDP or CPMDP, respectively. This way we have a new method method that will be called AugMDP in the following.

The method seems only to be applicable to problems with a constraint on the maximum cost, but not such with a constraint on the minimum profit. The latter start with initial states where the constraint is already violated (the initial gain is zero) resulting in a punishment S right from the start.

4.4 RecMDP: Recursive Reformulation of the Constraint

Gabor, Kalmar, and Szepesvari [6] developed an approach that is suited for dealing with problems of the type (B) described above, i.e. in which $c_{x,u} \geq 0$ and $c \geq 0$ hold.

Gabor, Kalmar, and Szepesvari give a recurrent reformulation of the constraint $C^\pi(x) \geq c$ based on the observation that the actual value of C^π is not really important as long it is above the threshold c (the minimum gain). Note that $\forall x\ C^\pi(x) \geq c$ implies $\mathcal{C}^\pi \geq c$ for every distribution D on the starting states, while the reverse is not true in general. That is, the method will generally arrive at a feasible, suboptimal solution.

Gabor et al. propose the recurrent formulation of a new value function defined by $\bar{C}^\pi(x) = \min\Big(\bar{c}, C^\pi(x)\Big)$ as

$$\bar{C}^\pi(x) = \min\Big(\bar{c}, c_{x,\pi(x)} + \min(\bar{c}, \gamma \sum_{x' \in X} \big[p_{x,\pi(x)}(x')\bar{C}^\pi(x')\big])\Big) \tag{3}$$

It holds $\bar{C}^\pi(x) \leq C^\pi(x)$ holds if we set $\bar{c} = c$. Therefore we might choose a value $\bar{c} \geq c$ in order to cover a larger range of feasible policies π.

Based on the recursive formulations of V^π and $\bar{C}^\pi$, we developed an online algorithm not requiring an initially known model. We leave out the details of the algorithms for reasons of space.

The approach produces necessarily suboptimal stationary, deterministic policies. We have the parameter $\bar{c}$ to adapt the result of the algorithm as in the previous approaches.

Problems with maximum costs can be solved by adding a large enough positive constant k to the values of the $c_{x,u}$ resulting in $c_{x,u} + k \geq 0$ for all x and u. Note that it is not obvious what constant should be added to the threshold $\bar{c}$. But since we adapt $\bar{c}$ anyway, a suitable value of $\bar{c}$ can be found via trial and error. This method will be called `RecMDP` in the following.

5 Experiments

In this section we describe the results of experiments with a series of randomly generated MDPs. We decided to focus on problems with maximum costs (i.e., $c_{x,u}, c \leq 0$) because such problems occur most often in RL applications (e.g., time, energy). In our first experiment, we focused on uniform distributions D. The MDPs were generated in the following manner:

– States: the number of states was selected randomly in the interval $[2, 50]$. With a probability of $\frac{1}{|X|}$, a state was turned into an absorbing state.
– Actions: the number of actions ranged between 2 and 4. We generated randomized actions such that for every state x and action u, $p_{x,u}(x') > 0$ holds for only approximately 25% of the possible successor states x'.
– Rewards: the rewards were selected in the interval $[0, 5]$ with uniform probability.
– Costs: $c_{x,u}$ was selected randomly in the interval $[-r_{x,u} - 1.0, -r_{x,u} + 1.0]$ to ensure that actions with a high reward also tend to have a high cost. Values larger than zero were set to 0.
– Discount factor: γ was selected randomly from the interval $[0, 1]$.

We decided to qualitatively compare the approaches by looking at the possible behaviours that can be generated using different parameter values ($c, \xi, S, \bar{c}$, resp.). We will focus on `CPDMPs` where the curves in the $(\mathcal{V}, p_.)$-space were depicted in Fig. 1. The results for `CMDPs` were quite similar. For reasons of space, we only depict the results for five MDPs being positively representative for all 20 runs performed. For the experiments, we chose c' as $-0.25\frac{6}{1-\gamma}$ where $\frac{6}{1-\gamma}$ is

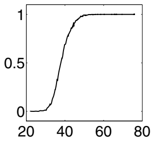
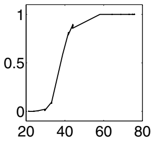
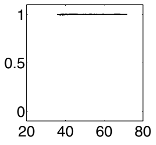
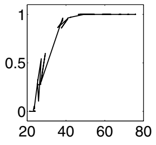
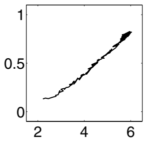
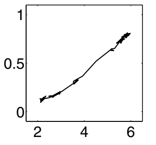
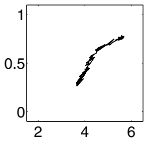
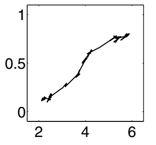
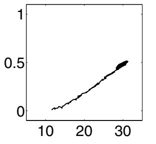
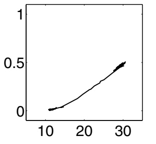
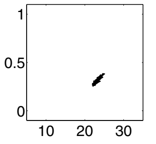
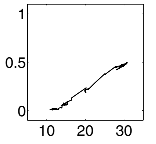
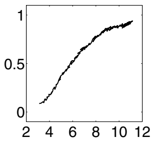
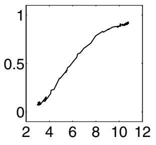
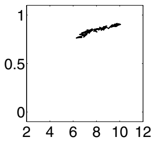
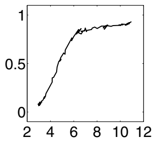
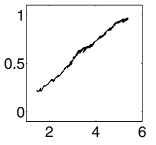
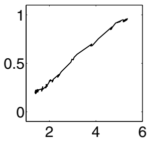
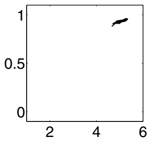
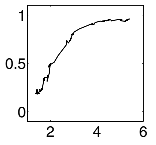
MDP	LinMDP	WeiMDP	AugMDP	RecMDP
$\lvert X\rvert = 12$ $\lvert U\rvert = 3$ $\gamma = 0.95$				
$\lvert X\rvert = 14$ $\lvert U\rvert = 2$ $\gamma = 0.61$				
$\lvert X\rvert = 31$ $\lvert U\rvert = 2$ $\gamma = 0.96$				
$\lvert X\rvert = 32$ $\lvert U\rvert = 4$ $\gamma = 0.69$				
$\lvert X\rvert = 19$ $\lvert U\rvert = 4$ $\gamma = 0.25$				

Fig. 1. CPMDP: curves in the $(\mathcal{V}, p_.)$-space

the theoretical upper bound for the accumulated costs given that the $c_{x,u}$ range in $[-6, 0]$ (see description of the MDPs above).

In Fig. 1, curves are to be considered better that cover a larger range of possible $\mathcal{V}$-values and p_π-values (corresponding to the projection onto the respective axis), and that attain better combinations of the two values, corresponding to curves that run more in the **lower right part** of the diagrams (i.e. with high returns and low probabilites of constraint violation). LinMDP is depicted in the first collumn. It can be seen that our weighted approach WeiMDP has a comparable performance. This is a very surprising result, because LinMDP can find randomized policies with a possibly better performance than the deterministic policies WeiMDP is restricted to. When looking at the policies computed by LinMDP, we found that randomization occurs very rarely which explain the small differences between WeiMDP and LinMDP.

AugMDP performed much worse especially with respect to the possible ranges of values, see Fig. 1. The reason is the very much enlarged state space that has to be considered. RecMDP performs quite well but seems to produce less stable results and worse combinations compared to WeiMDP and LinMDP.

6 Conclusion

All four presented methods have parameters that allow to switch between different behaviours. The parameters can be adapted to produce a feasible policy for the originally given constrained problem. We found that the method `LinMDP` performs best for `CMDP`s as well as `CPMDP`s, although it cannot be applied in an online learning manner. The weighted method `WeiMDP` performs quite well, too, and can be applied for problems with unknown model. Method `WeiMDP`, however, has a higher time complexity than `LinMDP`. Both methods can be extended for more than two criteria. Note that is possible to define handcrafted `CMDP`s, where `LinMDP` will outperform `WeiMDP`.

Encoding the costs in the state space (method `AugMDP`) yields the worst results, because of the much larger state space, and the approximation errors due to the necessary discretization of the possible cost values. We tried a series of different learning strategies with consistently bad results. The recursive method `RecMDP` produced acceptable results. Its performance on gain-constrained problems, for which it actually is designed, is still open and will be investigated in the future.

References

1. E. Altman. *Constrained Markov Decision Processes.* Chapman and Hall/CRC, 1999.
2. D. P. Bertsekas. *Dynamic Programming and Optimal Control.* Athena Scientific, Belmont, Massachusetts, 1995. Volumes 1 and 2.
3. D. A. Dolgov and E. H. Durfee. Constructing optimal policies for agents with constrained architectures. In *AAMAS*, pages 974–975, 2003.
4. D.A. Dolgov and E.H. Durfee. Approximating optimal policies for agents with limited execution resources. In *Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence*, pages 1107–1112. AAAI Press, 2004.
5. E.A. Feinberg and A. Shwartz. Constrained markov decision models with weighted discounted rewards. *Math. of Operations Research*, 20(4):302–320, 1995.
6. Zoltan Gabor, Zsolt Kalmar, and Csaba Szepesvari. Multi-criteria reinforcement learning. In *Proc. 15th International Conf. on Machine Learning*, pages 197–205. Morgan Kaufmann, San Francisco, CA, 1998.
7. P. Geibel and F. Wysotzki. Risk-sensitive reinforcement learning applied to chance constrained control. *JAIR*, 24:81–108, 2005.
8. R. S. Sutton and A. G. Barto. *Reinforcement Learning – An Introduction.* MIT Press, 1998.