

# Reinforcement Learning Approaches for Constrained MDPs

Peter Geibel

Institute of Cognitive Science, AI Group  
University of Osnabrück, Germany  
[pgeibel@uos.de](mailto:pgeibel@uos.de)

**Abstract:** Most reinforcement learning approaches consider Markov decision processes (MDPs) with a single criterion. In practical applications, however, we often have to deal with additional criteria, e.g. the energy consumed or the time spent during solving the main task. In this article, we will therefore consider Markov Decision Processes with two criteria. Each criterion is defined as the expected value of a cumulative return. The second criterion is subject to an inequality constraint. We will describe two new reinforcement learning approaches for solving such control problems, discuss their advantages and shortcomings, and present experimental results based on randomly generated MDPs. **Keywords:** Machine Learning, Reinforcement Learning, Dynamic Programming, Constraints

## I. Introduction

Most of the 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 by some robot such that the mean amount of time or energy needed does not exceed a given threshold. In this article, I will consider such constrained MDPs (CMDPs) only, although other, less common types of constraints, e.g. on the *probability* of constraint violation, can be considered as well, see e.g. [3].

Since applications with unequal discount factors [4, 5] are relatively rare and it is unclear how to solve them with model-free methods, we will only consider MDPs with several criteria each based on its own reward function, but using a *common discount factor*  $\gamma$ . 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.

The purpose of this paper is to undertake a description and comparison of different approaches for solving constrained problems (including two new ones), and to discuss their respective advantages and shortcomings. Model-free algorithms for solving CMDPs have not been described before.

The following solution approaches for MDPs with constraints will be discussed in section IV: `LinMDP` solves constrained MDPs using linear programming techniques [1, 3]; `WeiMDP` is a new, weighted approach for solving CMDPs based on an algorithm described by Geibel and Wysotzki in [6]; `RecMDP` is an online method based on a multi-criteria algorithm for CMDPs introduced in [5]

Each method finds suboptimal solutions, except method `LinMDP`. It turns out, however, that each method has a parameter that allows to select different behaviors with respect to the first and second criterion function yielding a curve in a 2-dimensional space, similar to the (profit, risk)-space introduced by Markovitz [7] for portfolio selection. We will show that it is computationally feasible to try out several parameter values and select that parameter that fits the original problem definition best.

The article is structured as follows. In section II, I describe the unconstrained problem including Markov Decision Processes, policies, and value functions. CMDPs are described in section III. In section IV, I will describe the approaches mentioned above. Section V gives an experimental comparison of these approaches using randomly generated MDPs. A concluding discussion can be found in section VI.

## II. 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 probability 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  $\pi$  mapping a state  $x$  to an action  $\pi(x)$ . Randomized policies are described using state dependent distributions  $\pi(x, \cdot)$  on possible actions.

The aim of the agent is to find a policy  $\pi$  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  $\pi$ . Let  $x_0, x_1, x_2, \dots$  denote the corresponding probabilistic sequence of states, and  $u_i$  the sequence of actions chosen according to policy  $\pi$ .

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

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

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

While DP algorithms often assume a fully known 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 definition of the Q-learning algorithm (e.g. [8]). If the model is known, then standard approaches can be applied. The approach we used consists in formulating a linear program and solve it with standard techniques, see [1, 3].

### III. Problems with Constraints

A constrained MDP (CMDP) consists of states  $X$ , actions  $U$ , transition probabilities  $p_{x,u}(x')$ , rewards  $r_{x,u}$ , a distribution on starting states  $D$ , and a discount factor  $\gamma$ , as before. There is an additional **second reward function**  $c_{x,u}$  used to define the constrained value function  $C^\pi$ , 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 Markov Decision Processes (CMDPs) as problems of the form

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

where the threshold  $c$  is a real value, and  $C^\pi = \mathbb{E} C^\pi(x)$  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 the following problems: For *problems with maximum costs* it holds

that  $c_{x,u} \leq 0$  and  $c \leq 0$ , e.g. the robot task and risk-sensitive control as discussed by Geibel and Wyszotzki in [6]. For *problems with minimum profit* we have  $c_{x,u} \geq 0$  and  $c \geq 0$ . In *Mixed Problems*, the  $c_{x,u}$  might take on positive as well as negative values and  $c$  is arbitrary. Think e.g. of a bank account with withdrawals and inpayments, where the account balance has to be kept above some balance  $c$ .

It should be noted that for constrained problems it is no longer the case that optimal stationary deterministic policies exist. We might need to consider randomized policies as well. Optimal randomized policies generally depend on the initial distribution  $D$ .

### IV. Solution Approaches

In the following we will describe and discuss several approaches for solving MDPs with constraints. An experimental comparison can be found in section V. We will start with the DP approach because it constitutes a baseline for comparing the performance of the other two RL approaches.

#### A. LinMDP: Linear Programming

In order to solve a standard constrained MDP, the standard linear program is extended with a constraint expressing  $C^\pi \geq c$ . This augmented program can again be solved with standard linear programming methods. The method yields an optimal randomized policy dependent on the CMDP to be solved (particularly on  $c$ ), and the initial distribution  $D$ . In the following we refer to this method as LinMDP.

#### B. WeiMDP: A Weighted Approach

Geibel and Wyszotzki in [6] considered the problem of finding policies that have a constrained risk of failing. We expressed the probability of entering an undesirable state as an (undiscounted) second value function. This resulted in a problem similar to a constrained MDP with possibly unequal discount factors, which was solved heuristically by introducing a weight parameter  $\xi$  for risk and value.

For solving real CMDPs, we suggest to also 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  $\xi$ , this new *unconstrained* MDP can be solved with standard methods, e.g. Q-Learning. For  $\xi = 0$  this results in an optimal policy for  $w_{x,u} = c_{x,u}$ . If the CMDP has a solution at all, this yields a *feasible policy* for the CMDP – feasible policies fulfill  $C^\pi \geq c$ . Larger values of  $\xi$  will result in policies with a possibly higher  $\mathcal{V}^\pi$  and lower  $C^\pi$  in case that the two objectives are in conflict.  $C^\pi$  and  $\mathcal{V}^\pi$  can be estimated using a series of test runs for  $\pi$ . Using different values of  $\xi \in [0, 1]$  will result in different points in the  $(\mathcal{V}, \mathcal{C})$ -space. We can select that  $\xi$  for which the correspond-

ing policy obeys  $C^\pi \geq c$  and  $\mathcal{V}^\pi$  is maximal. This method will be called `WeimMDP` in the following.

In contrast to `LinMDP`, our algorithm performs online learning of an optimal policy for the weighted criterion. It considers deterministic policies only, which means that the proposed solutions might be suboptimal. We will investigate the behavior of the algorithm in section V.

### C. `RecMDP`: Recursive Reformulation of the Constrained Value Function

Gabor, Kalmar, and Szepesvari [5] developed an approach that is suited for dealing with problems with minimum profit, where the agent has to maximize the first criterion function  $\mathcal{V}^\pi$ , while the second criterion function,  $C^\pi$ , is based on *positive rewards* ( $c_{x,u} \geq 0$ ). The threshold  $c$  is also required to be positive.

It is well-known, that the value function  $V^\pi$  can be expressed recurrently by the Bellman-equation stating that  $V^\pi(x) = \mathbb{E}[r_{x,u} + \gamma V^\pi(x')]$  holds. 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  $C^\pi$  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  $C^\pi \geq c$  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, but suboptimal solution.

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

$$\bar{C}^\pi(x) = \min(\bar{c}, c_{x,\pi(x)} + \quad (2)$$

$$\min(\bar{c}, \gamma \sum_{x' \in X} [p_{x,\pi(x)}(x') \bar{C}^\pi(x')]) \quad (3)$$

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

Based on the recursive formulations of  $V^\pi$  and  $\bar{C}^\pi$ , we developed an online algorithm based on state-action value functions. The algorithm finds a stationary policy that, in every state  $x$ , is *lexicographically optimal* with respect to  $(\bar{C}^\pi(x), V^\pi(x))$ . We leave out the details of the algorithms for reasons of space.

The approach produces necessarily suboptimal stationary, deterministic policies. As in `WeimMDP`, we have the parameter  $\bar{c}$  to adapt the result of the algorithm in order to find a policy with maximum  $\mathcal{V}^\pi$ , for which also  $C^\pi \geq c$  holds.

Note that the algorithm is designed for problems with minimum profit. 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  $\bar{c}$ . Since we, however, 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.

## V. Experiments

It is well-known in machine learning that it is almost always possible to find particular applications where some method “x” will outperform another method “y”. Instead of looking at specific application problems, we therefore decided to conduct experiments with a series of randomly generated MDPs. We focused on cost-constrained problems ( $c_{x,u}, c \leq 0$ ) 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:

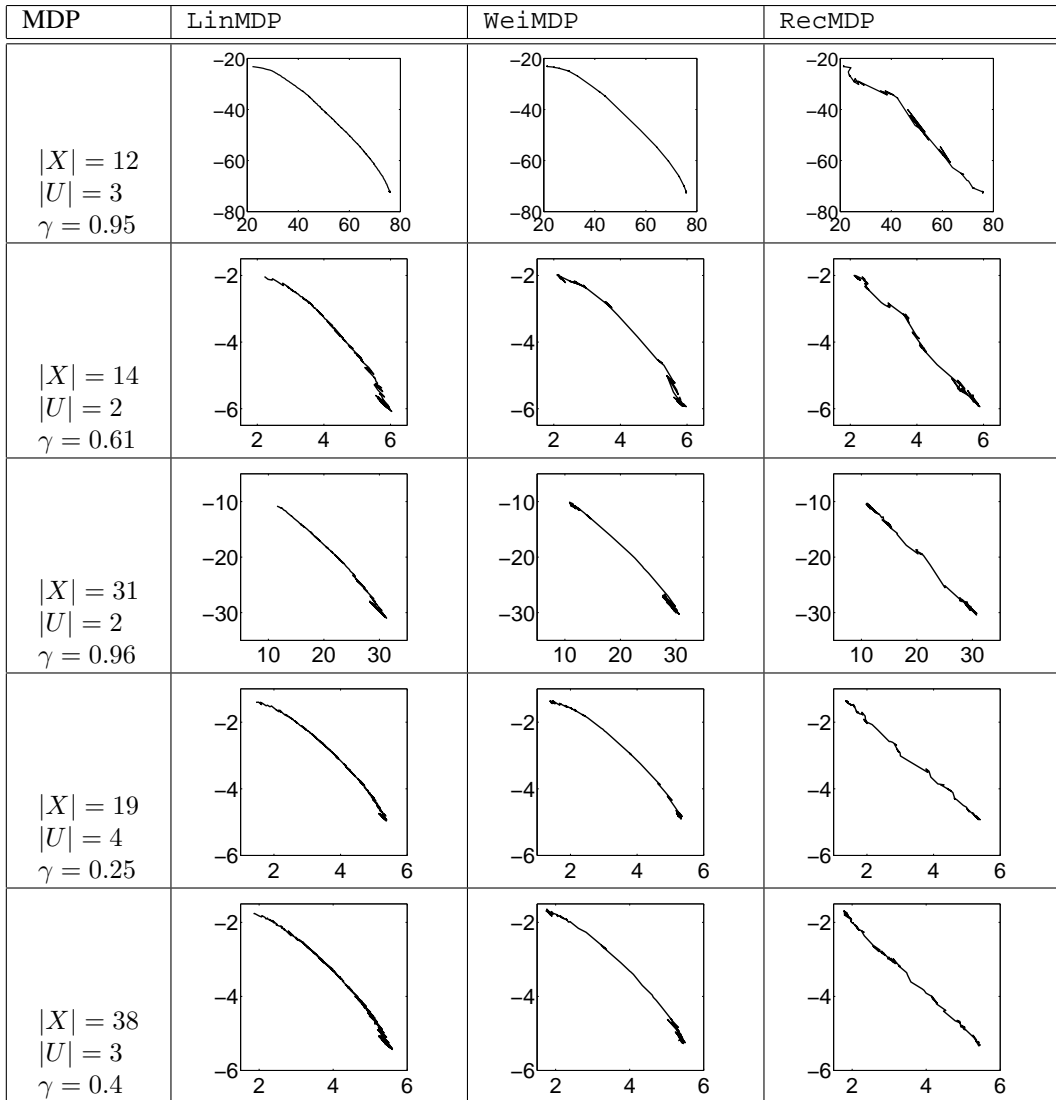
- States: the number of states was selected randomly in the interval  $[2, 50]$ . With a probability of  $\frac{1}{|X|}$ , a state was turned into an absorbing state.
- Actions: the number of actions ranged between 2 and 4. We generated randomized actions such that for every state  $x$  and action  $u$ ,  $p_{x,u}(x') > 0$  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  $[-r_{c,u} - 1.0, -r_{c,u} + 1.0]$  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:  $\gamma$  was selected randomly from the interval  $[0, 1]$ .

Each of the approaches defined in section IV has a parameter: the solution of `LinMDPs` depends on the constraint parameter  $c$ , for method `WeimMDP` several values of  $\xi$  can be investigated, and the result of `RecMDP` depends on  $\bar{c}$ . We measured the performance of each method with respect to the estimated mean return  $\mathcal{V}^\pi$  and the mean costs  $C^\pi$ .

We decided to compare the algorithms by looking at the possible behaviors that can be generated using different parameter values. We depicted the curves in the  $(\mathcal{V}, \mathcal{C})$ -space (Fig. 1) where the results for 5 randomly generated MDPs. We generated 20 problems all showing the same characteristics.

In Fig. 1, curves are to be considered better that cover a larger range of possible  $\mathcal{V}$ -values and  $\mathcal{C}$ -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 upper right part of the diagrams (i.e. with high returns and low costs). Because the three curves for an MDP (corresponding to a row) turned out to be relatively similar, we depicted them in separate diagrams with identical labellings of the axes.

Notice that although the curves look quite similar, each is based on different parameters with different ranges, and therefore also on different supporting points. Considering a feasible problem for a fixed value of  $c$ , each method  $m$  results in some policy  $\pi_m$  having two values  $\mathcal{V}^{\pi_m}$  and  $C^{\pi_m}$



**Figure 1:** Results for LinMDP, WeimDP, and RecMDP: curves in the  $(\mathcal{V}, \mathcal{C})$ -space.

(estimated using test runs) with  $\mathcal{C}^{\pi_m} \leq c$ . For LinMDP,  $\mathcal{V}^{\pi_{LinMDP}}$  is guaranteed to be optimal, whereas the results of WeimDP and RecMDP depend on the different values for  $\xi$  and  $\bar{c}$  considered, which together with the fact that the respective policy performance is characterized by *two* values  $\mathcal{V}^{\pi_m}$  and  $\mathcal{C}^{\pi_m}$  makes it difficult to apply standard statistical tests. We therefore compared the curves only qualitatively. For the considered MDPs, the computed policy  $\pi_m$  is more sensitive to the resp. parameter setting if it is selected so as to produce a policy with a relatively small  $\mathcal{C}^{\pi_m}$ . For e.g. WeimDP this is the case if  $\xi \in [0, 1]$  is close to zero.

LinMDP depicted in the first column denotes the theoretical optimum for CMDPs. 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 look-

ing at the policies computed by LinMDP, we found that randomization occurs very rarely which explain the small differences between WeimDP and LinMDP. RecMDP performs quite well but seems to produce less stable results and worse combinations compared to WeimDP and LinMDP.

We also ran experiments with other, less symmetrically constructed MDPs and other initial distributions  $D$ . Despite such less uniform setup that might have been the reason for the good performance of WeimDP, the behavior of the algorithms was quite similar to that given in Fig. 1.

## VI. Conclusion

All three presented methods have a parameter that allows to switch between different behaviors. For WeimDP and RecMDP, the respective parameter can be adapted to produce a feasible deterministic policy for the originally given

constrained problem with threshold  $c$ . We found that the method `LinMDP` performs best, but 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. The methods `WeimDP` and `RecMDP`, however, have a much higher time complexity than `LinMDP`, i.e. convergence is reached much more slowly, because they are “model-free” (cf. also [1, 2, 8]). Moreover, several parameter values have to be considered to arrive at a good result. Note that it is possible to define handcrafted MDPs, where `LinMDP` will outperform `WeimDP`. All three approaches can be adapted for dealing with more than one unconstrained value functions, and several constrained ones.

The recursive method `RecMDP` produced acceptable results. Its performance on problems with minimum profit, for which it actually was 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, MA, 1995.
- [3] D.A. Dolgov and E.H. Durfee. Approximating optimal policies for agents with limited execution resources. In *Proceedings of the Eighteenth IJCAI*, pages 1107–1112. AAAI Press, 2004.
- [4] E.A. Feinberg and A. Shwartz. Constrained markov decision models with weighted discounted rewards. *Math. of Operations Research*, 20(4):302–320, 1995.
- [5] 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.
- [6] P. Geibel and F. Wysotzki. Risk-sensitive reinforcement learning applied to chance constrained control. *JAIR*, 24:81–108, 2005.
- [7] H. M. Markowitz. *Portfolio Selection*. John Wiley and Sons, New York, 1959.
- [8] R. S. Sutton and A. G. Barto. *Reinforcement Learning – An Introduction*. MIT Press, 1998.