Konrad-Zuse-Zentrum
für Informationstechnik Berlin

Takustraße 7
D-14195 Berlin-Dahlem
Germany

STEFAN HEINZ[1] VOLKER KAIBEL[1,2]
MATTHIAS PEINHARDT[1] JÖRG RAMBAU[3]
ANDREAS TUCHSCHERER[1]

LP-Based Local Approximation for Markov Decision Problems

[1] Konrad-Zuse-Zentrum für Informationstechnik Berlin, Department Optimization, Takustr. 7, 14195 Berlin, Germany. E-mail: {heinz,kaibel,peinhardt,tuchscherer}@zib.de
[2] Technische Universität Berlin, MA 6–2, 10623 Berlin, Germany.
[3] Universität Bayreuth, Lehrstuhl für Wirtschaftsmathematik, 95440 Bayreuth, Germany. E-mail: Joerg.Rambau@uni-bayreuth.de

Supported by the DFG Research Center MATHEON *Mathematics for key technologies* in Berlin.

ZIB-Report 06-20 (April 2006)

LP-Based Local Approximation for Markov Decision Problems*

Stefan Heinz[1], Volker Kaibel[1,2], Matthias Peinhardt[1], Jörg Rambau[3], and Andreas Tuchscherer[1]

[1] Konrad-Zuse-Zentrum für Informationstechnik Berlin, Department Optimization, Takustr. 7, 14195 Berlin, Germany. E-mail: {heinz,kaibel,peinhardt,tuchscherer}@zib.de
[2] Technische Universität Berlin, MA 6–2, 10623 Berlin, Germany.
[3] Universität Bayreuth, Lehrstuhl für Wirtschaftsmathematik, 95440 Bayreuth, Germany. E-mail: Joerg.Rambau@uni-bayreuth.de

Abstract. The standard computational methods for computing the optimal value functions of Markov Decision Problems (MDP) require the exploration of the entire state space. This is practically infeasible for applications with huge numbers of states as they arise, e. g., from modeling the decisions in online optimization problems by MDPs. Exploiting column generation techniques, we propose and apply an LP-based method to determine an ε-approximation of the optimal value function at a given state by inspecting only states in a small neighborhood. In the context of online optimization problems, we use these methods in order to evaluate the quality of concrete policies with respect to given initial states. Moreover, the tools can also be used to obtain evidence of the impact of single decisions. This way, they can be utilized in the design of policies.

1 Introduction

Many real-world problems that call for planning under uncertainty over an infinite time horizon, e. g., online optimization problems, can be modeled as Markov Decision Problems (MDPs). See Section 6 for an example. Due to several drawbacks of other concepts of evaluation of online algorithms (like competitive analysis, see, e. g., [3]), MDPs provide an attractive alternative in this context whenever stochastic information about future requests is available. However, the sizes of the state spaces of MDPs arising this way usually are exponential in the original input parameters. The classical methods for computing an optimal policy are value iteration, policy iteration, and linear programming [2]. As the complexity of these methods depends at least linearly on the number of states, their use is impossible in MDPs for practical applications as mentioned above.

The aim of the approach presented in this paper is to evaluate given policies or just single controls w. r. t. an optimal policy in one state without exploring the entire state space. In order to obtain an estimation of the relative gap between the value function of a given policy and the optimal value function at a particular state, we must provide an upper bound for the first value and a lower bound for the second one. In fact, our algorithm computes lower and upper bounds for both values separately.

* Supported by the DFG Research Center MATHEON *Mathematics for key technologies* in Berlin.

Our methods are based on the classical linear programming formulation for computing the optimal value function in an infinite-horizon discounted MDP. We modify the formulation by taking into account only a (small) part $\hat{S}$ of the state space S. The paper aims at making two main contributions. On the theoretical side, we show in Section 3 that, for a given $\varepsilon > 0$ and an initial state $i_0 \in S$, there is always some part $\hat{S}$ of S such that the restricted LP yields an ε-approximation of the optimal value function at i_0, while the size of $\hat{S}$ does not depend on the total number of states. In fact, the size of $\hat{S}$ depends only on the local structure of S at i_0. On the practical side, in Section 5 we propose an algorithm in which the local part $\hat{S}$ of the state space is changed dynamically during the run by means of a column generation procedure. We report on experiments with a simplified version of an online target date assignment problem that arose from an industrial application. At least in this simplified setting, the results of our experiments are quite encouraging. The main reason for the practical efficiency of the algorithm seems to be the feature that the dynamic change of the set $\hat{S}$ is guided by the reduced cost structure of the current reduced LP.

The approach described in the literature that yields the results closest to our's is a sparse sampling algorithm for large-scale Markov Decision Problems [9]. The authors also give theoretical bounds on the necessary size of a subset of the state space that is needed by their approach in order to obtain an ε-approximation (see the remark at the end of Section 3). However, for the applications that we aim at their bounds are weaker than ours. Furthermore, and maybe more importantly, their method does not seem to allow a modification that in practice visits substantially fewer states than guaranteed by the theoretical analysis, *yet maintaining the approximation guarantee a-posteriori*. In our view, this is a main advantage of the method that we propose (see the results in Section 6). Other approaches to locally explore the state space include [6, 1], where in [6] policy iteration is used with a concept of locality that is similar to ours. However, their method does not provide any approximation guarantees either.

This paper is arranged as follows. In Section 2, we give the definition of an MDP together with related notations. Section 3 presents the proof of our theoretical local approximation theorem. A couple of useful applications of this result are given in Section 4. Our algorithmic approach for practical use is described in Section 5. Finally, Section 6 shows computational results of our algorithm applied to a simple MDP which is derived from an industrial problem.

2 Preliminaries

The basic data of a *stationary optimal control problem* in discrete time are a set S of *states*, together with a non-empty set $U(i)$ of *feasible controls* for every state $i \in S$. Applying in state $i \in S$ control $u \in U(i)$ moves the system into state $j \in S$ with *transition probability* $p_{ij}(u)$ (thus, we have $\sum_{j \in S} p_{ij}(u) = 1$). The function $g(i,u,j)$ denotes the *stage cost* of control $u \in U(i)$ in state $i \in S$ in the case when state $j \in S$ is the next state. A *policy* μ is a mapping that assigns a feasible control to each state. For $\alpha \in (0,1)$ the α-*discounted cost* of a policy μ for initial state $i \in S$ is the expected discounted total cost defined by

$$J_\mu(i) := \mathbb{E}_{j_1,j_2,\dots}\Big(\sum_{k=0}^{\infty} \alpha^k g(j_k,\mu(j_k),j_{k+1})\Big), \tag{1}$$

where $j_0 = i$ and j_k is a random variable specifying the state of the system at stage k. J_μ is called the *value function* of policy μ and the *optimal value function* is defined at state $i \in S$ by

$$J^*(i) := \min_\mu J_\mu(i).$$

Any policy μ^* with $J_{\mu^*} = J^*$ is an *optimal policy*. The goal is to find an optimal or close-to-optimal policy.

In the computer science literature such systems are usually called *Markov Decision Problems*, MDP, for short. For an introduction to the subject see, e. g., [11]. If J^* is known then—by Bellman's principle of optimality—any policy μ with

$$\mu(i) \in \operatorname*{argmin}_{u\in U(i)} \Big\{ g(i,u) + \alpha \sum_{j\in S} p_{ij}(u) J^*(j) \Big\} \quad \forall i \in S$$

is optimal, where $g(i,u) := \sum_{j\in S} p_{ij}(u) g(i,u,j)$ is the *expected stage cost* of using control $u \in U(i)$ at state $i \in S$.

Methods to compute J^* include *value iteration*, *policy iteration*, and *linear programming* [2, 10]. In all these methods, the complexity of finding an optimal policy depends at least linearly on the number of states, which itself in our context is usually exponential in the common input parameters.

According to the classical linear programming model, the optimal value function J^* can be derived as the unique optimal solution of the following linear program:

$$\max\Big\{\sum_{i\in S} J(i) \,\Big|\, J(i) \le g(i,u) + \alpha \sum_{j\in S} p_{ij}(u) J(j) \ \forall i \in S, \forall u \in U(i)\Big\} \tag{2}$$

This formulation has been the starting point for several approaches [4, 5]. Since we are only interested in $J^*(i_0)$ for a particular state $i_0 \in S$, we will consider the following LP with simplified objective function instead:

$$\max\Big\{J(i_0) \,\Big|\, J(i) \le g(i,u) + \alpha \sum_{j\in S} p_{ij}(u) J(j) \ \forall i \in S, \forall u \in U(i)\Big\} \tag{3}$$

The following relation is due to the fact that the componentwise maximum of two feasible solutions to the LPs is feasible as well.

Remark 1. Every optimal solution to (2) is an optimal solution to (3).

Since in our model there is only a finite number of states and controls the expected stage costs $g(i,u)$ are bounded from above by some constant G, i. e., $g(i,u) \le G$ for all states $i \in S$ and all controls $u \in U(i)$. This implies an upper bound on J: from (1) we easily get $J_\mu(i) \le C := G/(1-\alpha)$, for all policies μ and all $i \in S$. Since it will be convenient sometimes, we generally assume $G \ge 1$ (which easily can be obtained by scaling the expected stage costs).

3 The Structural Result

In the following we present our structural result which shows that an ε-approximation of the optimal value function at one given state can be obtained by taking into account only a small local part of the entire state space.

Theorem 1. *Let S be the state space of an MDP with the following properties, where $c,d \in \mathbb{N}$ with $cd \geq 2$ and $G \geq 1$:*

- *The number $|U(i)|$ of controls is bounded by c for each $i \in S$.*
- *For each $i \in S$ and $u \in U(i)$, the number of states $j \in S$ with positive transition probabilities $p_{ij}(u)$ is bounded by d.*
- *For each $i \in S$ and $u \in U(i)$, the expected stage costs satisfy $0 \leq g(i,u) \leq G$.*

Then, for every discount factor $0 < \alpha < 1$, for all $i_0 \in S$, and for all $\varepsilon > 0$, there is a set of states $\hat{S} \subset S$ with the following properties:

(i) $|\hat{S}| \in O\big((cd)^{\log\left(\frac{\varepsilon(1-\alpha)}{G}\right)/\log(\alpha)}\big)$, in particular, the number of states in $\hat{S}$ does not depend on $|S|$.
(ii) The function $\underline{J} : \hat{S} \to \mathbb{R}_+$ defined as an optimal solution to the reduced LP

$$\max\big\{J(i_0) \,|\, J(i) \leq g(i,u) + \alpha \sum_{j \in \hat{S}} p_{ij}(u) J(j) \;\forall i \in \hat{S}\, \forall u \in U(i)\big\} \qquad (\mathrm{L}_{\hat{S}})$$

is an ε-close lower bound to J^* in the given state i_0:

$$0 \leq J^*(i_0) - \underline{J}(i_0) \leq \varepsilon.$$

Our theorem does not apply to the case $c = d = 1$, which, however, leads only to trivial MDPs (at least in our context). Note that the reduced LP ($\mathrm{L}_{\hat{S}}$) is feasible since we assume that the expected stage costs $g(i,u) \geq 0$, e. g., $\underline{J} \equiv 0$ is always a solution.

To prove the theorem we introduce some technical notions.

Definition 1 **(H–neighborhood).** *For an MDP with state space S, a state $i_0 \in S$, and a number $H \in \mathbb{N}$, the H–*neighborhood *$S(i_0,H)$ of i_0 in S is the subset of states that can be reached from i_0 with positive probability within at most H transitions. That is $S(i_0,0) := \{i_0\}$, and for $H > 0$ we define:*

$$S(i_0,H) := S(i_0,H-1) \cup \{j \in S \mid \exists i \in S(i_0,H-1)\, \exists u \in U(i) : p_{ij}(u) > 0\}.$$

Using this definition we can prove Theorem 1.

Proof (Theorem 1). For $\alpha,\varepsilon > 0$, $\alpha < 1$, and $i_0 \in S$, we choose $\hat{S} = S(i_0,H)$ where $H > 0$ is specified later. Since the numbers of applicable controls and succeeding states for any control are bounded by c resp. d, we have that $|\hat{S}| \leq \frac{(cd)^{H+1}-1}{cd-1} \leq (cd)^{H+1}$ (as we have $cd > 1$).

Similarly to Remark 1, there is some optimal solution $\underline{J}$ of the restricted LP ($\mathrm{L}_{\hat{S}}$) that is also optimal with respect to the objective function $\sum_{i \in \hat{S}} J(i)$. Thus, for every state $i \in \hat{S}$, there is at least one control $u \in U(i)$ such that the corresponding inequality in ($\mathrm{L}_{\hat{S}}$)

is satisfied with equality by $\underline{J}$ (here we need that every state has at least one control). We call one of these controls the *critical control* $\underline{u}(i)$ of i. Furthermore, let J^* be the optimal value function (i. e., J^* is the optimal solution to (2)).

We define $\underline{J}(i) := 0$ for all $i \in S \setminus \hat{S}$. The value $\underline{J}(i_0)$ is a lower bound on $J^*(i_0)$ since ($\mathrm{L}_{\hat{S}}$) can be seen as derived from (3) with the additional restrictions $J(i) = 0$ for all $i \in S \setminus \hat{S}$.

We then have the relations

$$J^*(i) \leq g(i,\underline{u}(i)) + \alpha \sum_{j\in S} p_{ij}(\underline{u}(i))J^*(j) \qquad \forall\, i \in \hat{S},$$
$$\underline{J}(i) = g(i,\underline{u}(i)) + \alpha \sum_{j\in \hat{S}} p_{ij}(\underline{u}(i))\underline{J}(j) \qquad \forall\, i \in \hat{S},$$

which gives

$$J^*(i) - \underline{J}(i) \leq \alpha \sum_{j\in S} p_{ij}(\underline{u}(i))\big(J^*(j) - \underline{J}(j)\big) \qquad \forall\, i \in \hat{S}. \tag{4}$$

Note that in (4) we sum over the whole state space, although the equality for $\underline{J}$ provides only a sum over $\hat{S}$. However, this does not affect the validity due to $\underline{J}(j) = 0$ for $j \in S\setminus\hat{S}$.

We show by reverse induction on $h = H,\ldots,0$ that, for all $i \in S(i_0,h)$,

$$J^*(i) - \underline{J}(i) \leq \alpha^{H+1-h}\frac{G}{1-\alpha}. \tag{5}$$

Note that all i to which (5) refers are contained in $\hat{S}$ because of $h \leq H$.

For $h = H$ and for all $i \in S(i_0,h)$, due to $J^*(j) \leq G/(1-\alpha)$ for all $j \in S$, we derive (5) from (4):

$$\begin{aligned} J^*(i) - \underline{J}(i) &\leq \alpha \sum_{j\in S} p_{ij}(\underline{u}(i))\Big(\frac{G}{1-\alpha} - 0\Big) \\ &= \alpha\frac{G}{1-\alpha} \end{aligned}$$

Here, the equation follows from the fact that $\sum_{j\in S} p_{ij}\underline{u}(i) = 1$ holds for every $i \in S$.

Now assume that (5) holds for all $j \in S(i_0,h)$ with $H \geq h > 0$. For any $i \in S(i_0,h-1)$ we again apply (4):

$$J^*(i) - \underline{J}(i) \leq \alpha \sum_{j\in S} p_{ij}(\underline{u}(i))\big(J^*(j) - \underline{J}(j)\big) = \alpha \sum_{j\in S(i_0,h)} p_{ij}(\underline{u}(i))\big(J^*(j) - \underline{J}(j)\big),$$

where the equation is due to the fact that all $j \in S$ with $p_{ij}(\underline{u}(i)) > 0$ are contained in $S(i_0,h)$. We can apply the induction hypothesis for $j \in S(i_0,h)$:

$$J^*(i) - \underline{J}(i) \leq \alpha \sum_{j\in S(i_0,h)} p_{ij}(\underline{u}(i))\alpha^{H+1-h}\frac{G}{1-\alpha} = \alpha^{H+1-(h-1)}\frac{G}{1-\alpha},$$

which completes the inductive proof of (5).

For $i = i_0$ and $h = 0$, inequality (5) yields $J^*(i_0) - \underline{J}(i_0) \leq \alpha^{H+1}\frac{G}{1-\alpha}$. By choosing $H = \lceil \log\left(\frac{\varepsilon(1-\alpha)}{G}\right) / \log\alpha \rceil - 1$ the gap of the lower bound $\underline{J}(i_0)$ is bounded by ε. $\square$

Remark 2. The size of the restricted state space is optimal in some sense, as can be seen from the example of a tree like MDP (rooted at i_0), in which every state (that can be reached within H steps from i_0) has exactly c different controls, which with uniform transition probabilities, lead to exactly d "new states" that can be reached only via this control. In this case, one can show that $\hat{S} = S(i_0, H)$ as above is the smallest restricted state space to obtain the desired approximation. Of course, incorporating additional parameters of the MDP might give better results in special cases.

Though this is not important in our context, we mention that Theorem 1 still holds if the state space S is not finite.

More important is the observation that, in the same way as we computed the lower bound on the optimal cost function in a given state i_0, we can compute an upper bound as well. Instead of just dropping the cost function outside $\hat{S}$, i. e., setting it to zero, we can set the corresponding variables to the general upper bound $C = G/(1-\alpha)$. By choosing the same restricted state space $\hat{S} = S(i_0, H)$ as above, the new restricted program reads:

$$\begin{aligned} &\max\ J(i_0) \\ &J(i) \leq g(i,u) + \alpha\Big(\sum_{j\in\hat{S}} p_{ij}(u)\, J(j) + \sum_{j\in S\setminus\hat{S}} p_{ij}(u)\, C\Big) \quad \forall i \in \hat{S}\, \forall u \in U(i) \end{aligned} \tag{$\mathrm{U}_{\hat{S}}$}$$

Let $\overline{J}$ be an optimal solution to ($\mathrm{U}_{\hat{S}}$). Obviously $\overline{J}(i_0)$ is an upper bound of $J^*(i_0)$. In the same way as in the proof of Theorem 1 we focus on critical controls u^*, this time with respect to the optimal solution J^* of (2). The basic relations then read (with $\overline{J}(j) = C$ for all $j \in S \setminus \hat{S}$):

$$\begin{aligned} J^*(i) &= g(i,u^*(i)) + \alpha \sum_{j\in S} p_{ij}(u^*(i))J^*(j) && \forall\, i \in \hat{S}, \\ \overline{J}(i) &\leq g(i,u^*(i)) + \alpha \sum_{j\in S} p_{ij}(u^*(i))\overline{J}(j) && \forall\, i \in \hat{S}, \end{aligned}$$

which leads to

$$\overline{J}(i) - J^*(i) \leq \alpha \sum_{j\in S} p_{ij}(u^*(i))\big(\overline{J}(j) - J^*(j)\big) \qquad \forall\, i \in \hat{S}.$$

Following the arguments as above we deduce:

$$\overline{J}(i_0) - J^*(i_0) \leq \alpha^{H+1} \frac{G}{1-\alpha}.$$

The program ($\mathrm{U}_{\hat{S}}$) for the computation of the upper bound can be tightened if better upper bounds than $G/(1-\alpha)$ for the cost function of not included states can be found.

Remark 3. It should be noted that the size of the restricted state space we are using is considerably smaller than that used for random sampling in [9]. There states are sampled up to a depth of $H_s = \lceil \log\left(\frac{\varepsilon(1-\alpha)^3}{4G}\right) / \log\alpha \rceil$ which gives a considerably larger state space (note that due to our general assumption $G \geq 1$ we have $\varepsilon(1-\alpha)/G < 1$).

However, the setting given there is somewhat different as they assume very large values of d. In particular, they sample, for each considered state in depth smaller than H_s, only

$$T \approx \left(\frac{4G}{\varepsilon(1-\alpha)^3}\right)^2 \left(\frac{2}{\ln\alpha}\ln\left(\frac{\varepsilon(1-\alpha)^3}{4G}\right)\ln\left(\left(\frac{4G}{\varepsilon(1-\alpha)^3}\right)^2 \frac{c}{\ln\alpha}\ln\left(\frac{\varepsilon(1-\alpha)^3}{4G}\right)\ln\left(\frac{4G}{\varepsilon(1-\alpha)^2}\right)\right) + \ln\left(\frac{4G}{\varepsilon(1-\alpha)^2}\right)\right)$$

consecutive states if $T < d$. Note however that even for fairly small problems, e. g., $G = 1$, $c = 4$, $\alpha = 0.7$, and $\varepsilon = 0.1$, this yields T larger than 1.8 billion.

4 Applications

For online optimization problems, very often algorithms seem to behave well in practice or, at least, in simulation experiments, although the classical analysis tools (e. g., based on competitive ratio) do not indicate this. In order to gain evidence for such observations, we aim for a method that, at each event, provides an estimate of the expected (future) costs caused by the current decision (assuming the later decisions to be made according to a concrete algorithm or not), relative to the costs generated by an optimal algorithm. Translated into the language of MDPs, we therefore need a method for the *relative evaluation* of a given policy or of a single control. In an MDP-model of some online optimization problem, the states need to encode important information like, e. g., the currently open requests and the resources used at the moment (which usually lead to state spaces of intractable size). Stochastic information about future requests can then be translated into transition probabilities.

We showed that an ε-approximation of the optimal value function for one state can be computed by only considering a small subset of states whose size does not depend on the total number of states. In the following we highlight useful applications of this result that are obtained by considering appropriate restrictions of the state space of the MDP.

Obviously, the value function J_μ of a given policy μ equals the optimal value function of the restricted MDP, where the feasible controls are restricted to those chosen by the policy, i. e., $U'(i) = \{\mu(i)\}$ for all states $i \in S$. Therefore, we can approximate the value function of a policy in the same way as we did for the optimal value function. For the same guarantee, the value function approximation for a given policy may require substantially fewer states than the approximation of the optimal value function since there is only one feasible control per state.

As mentioned above, it sometimes is more desirable to evaluate only a single control applied in a given state (not entire policy), given that the remaining decisions are made w. r. t. an optimal policy. Using another restriction of our original MDP, our method can be used for this purpose, too. Given a state $i_0 \in S$ and a control $u \in U(i_0)$, we define J_u as the value function of a policy which is optimal among all policies that apply control u in state i_0. The value $J_u(i_0)$ reflects the impact of using control u in state i_0. We say that control u is *optimal* in state i_0 if $J_u(i_0) = J^*(i_0)$. Again, J_u is the optimal value function of an restricted MDP, where the set of feasible controls in state i_0 is restricted to $U'(i_0) = \{u\}$. Thus, we can use Theorem 1 to approximate $J_u(i_0)$.

The observations above yield the following corollary of Theorem 1:

Remark 4. Assume we are given an MDP satisfying the conditions of Theorem 1, a state $i_0 \in S$, a policy μ, and a control $u \in U(i_0)$. Then, by restricting the MDP, the values $J_\mu(i_0)$ and $J_u(i_0)$ can be approximated in the same way as described for $J^*(i_0)$ in Theorem 1.

With respect to the goals discussed at the beginning of this section, we aim at estimating and reporting for every state $i_0 \in S$, which is reached while running a simulation or real-world system, *online* the quantity

$$\frac{J_x(i_0) - J^*(i_0)}{J^*(i_0)},$$

where $x \in \{\mu, u\}$. This gives the relative cost increase in state $i_0 \in S$ when using policy μ or control $u \in U(i_0)$ instead of an optimal policy. Thus, we need both a lower bound on $J^*(i_0)$ and an upper bound on $J_x(i_0)$. They can be computed as described above.

The following observation can be useful in order to determine an optimal control in a given state.

Remark 5. Suppose, for a state $i_0 \in S$ and a control $u \in U(i_0)$, we have obtained an upper bound $\overline{J}_u(i_0)$ on $J_u(i_0)$ and lower bounds $\underline{J}_{u'}(i_0)$ on $J_{u'}(i_0)$ for all controls $u' \in U(i_0) \setminus \{u\}$, respectively, such that $\overline{J}_u(i_0) \leq \underline{J}_{u'}(i_0)$ for all u'. Then u is an optimal control at i_0.

5 Algorithmic Approach

In order to compute approximations on the optimal value function at a particular state of a given MDP, it is usually inappropriate to apply the construction of Section 3 directly. In this section, we present a suitably modified algorithmic approach. Following the lines described in the last section, the algorithm can also be used in order to evaluate a concrete policy or a specific control at a single state.

The idea is to start with a small set $\hat{S}$ containing i_0 that successively is increased by adding new states. In doing so, we obtain a sequence of sets $\hat{S}_1 \subset \hat{S}_2 \subset \cdots \subset \hat{S}_n \subset S$ together with a sequence of improving lower and upper bounds obtained as the optimal values of the corresponding linear programs. Note that each added state $i \in S \setminus \hat{S}$ yields one new variable and $|U(i)|$ new constraints in both linear programs ($\mathrm{L}_{\hat{S}}$) and ($\mathrm{U}_{\hat{S}}$).

Our method proceeds by performing column generation on the linear program ($\mathrm{L}_{\hat{S}}$). The corresponding dual program reads:

$$\begin{aligned} \min \quad & \sum_{i\in\hat{S}}\sum_{u\in U(i)} g(i,u)\pi(i,u) && (\mathrm{D}_{\hat{S}}) \\ \text{subject to} \quad & \sum_{u\in U(i_0)} \pi(i_0,u) = \alpha \sum_{i\in\hat{S}}\sum_{u\in U(i)} p_{ii_0}(u)\pi(i,u) + 1 && \\ & \sum_{u\in U(j)} \pi(j,u) = \alpha \sum_{i\in\hat{S}}\sum_{u\in U(i)} p_{ij}(u)\pi(i,u) && \forall j \in \hat{S}\setminus\{i_0\} \\ & \pi(i,u) \geq 0 && \forall i \in \hat{S}\ \forall u \in U(i) \end{aligned}$$

Given an optimal solution π^{opt} to the dual program ($\mathrm{D}_{\hat{S}}$) for some subset $\hat{S} \subset S$, the reduced profit of a state $j \in S \setminus \hat{S}$ equals $\alpha \sum_{i\in\hat{S}} \sum_{u\in U(i)} p_{ij}(u)\pi^{\mathrm{opt}}(i,u)$. The goal of the pricing problem is to find a state with maximum reduced profit.

Using our column generation algorithm, we aim to investigate how many states are required in practice to obtain a given approximation guarantee. Therefore, the algorithm is mainly not designed to be very run-time efficient, but to show how many states it requires to obtain the desired approximation. Our basic algorithm thus starts with the set $\hat{S}_1 = \{i_0\}$ and only adds one state of maximum reduced profit in each iteration.

6 Computational Results

As an example for applying our algorithm, we choose the MDP model of a small *target date assignment problem*. This kind of problems arise, e. g., in the context of dispatching service technicians [8]. In the simplified model that we considered for our experiments, one is faced with a sequence of items (of two possible sizes) arriving over time. Each item has to be assigned to one of the following four days (immediately and irrevocably). The items assigned to a particular day are packed into a minimal number of bins of size one. The goal is to distribute the items in such a way that the total number of bins used is as small as possible. We consider an instance where at most six items are released per day.

The parameters of the resulting MDP are as follows. In each state there are four feasible controls and positive transition probabilities for at most four successor states. All transition probabilities are in the range $[0.1, 0.5]$. Possible stage costs are 1 (a new bin is required) and 0 (no new bin). The MDP has more than 16 million states in total. In all computations in this section, we use a discount factor of $\alpha = 0.7$. For this MDP no explicit optimal policy is known. For more details on the MDP and the computational evaluation of associated policies using our algorithm, see [7].

First we analyze the performance of our column generation algorithm compared to the method that directly applies the construction of Theorem 1 for approximating the optimal value function at a particular state i_0. Table 1 shows for given values of $H \in \mathbb{N}$, the resulting approximation guarantee ε when using the H–neighborhood of i_0 as used in Theorem 1. Moreover, the table shows the size of the H–neighborhood and the required number of states $|\hat{S}|$ to obtain the given guarantee ε when using the column generation algorithm as well as the ratio of these two values. As expected, the column generation requires substantially fewer states as in the theoretical worst-case for non-trivial approximation guarantees. It should be mentioned that the actual bounds obtained by the state set constructed from Theorem 1 are in fact much better than guaranteed by this theorem, e. g., for $H = 6$ we already achieve an approximation guarantee of $\varepsilon = 0.1$.

H	0	1	2	3	4	5	6	7	8	9	10
ε	2.34	1.64	1.15	0.81	0.57	0.40	0.28	0.20	0.14	0.10	0.07
$\lvert S(i_0,H)\rvert$	1	17	147	855	3497	11421	31893	69613	118321	199765	328357
$\lvert\hat{S}\rvert$ by CG	1	5	25	47	118	274	558	904	1429	2064	3012
%	100	29.41	17.01	5.50	3.37	2.40	1.75	1.30	1.21	1.03	0.92

Table 1. Required number of states: H–neighborhood vs. column generation algorithm (CG). The value $J^*(i_0)$ is about 1.53 which yields an impression for the relative approximation quality.

Table 2 shows for each depth w. r. t. the considered state how many states exist and how many of them were generated by the column generation algorithm. The required approximation guarantee is 5 %. We see that for practical MDPs the required state set can be far away from being an H–neighborhood for some $H \in \mathbb{N}$.

layer	0	1	2	3	4	5	6	7	8	9	10
total	1	16	130	708	2642	7924	20472	37720	48708	81444	128592
used	1	14	46	157	440	741	768	866	729	188	7
%	100	87.5	35.38	22.18	16.65	9.35	3.75	2.30	1.49	0.23	0.005

Table 2. Distribution of states per layer.

Next we consider the approximation process of our column generation algorithm in detail. Given a state $i_0 \in S$, a policy μ, and a control $u \in U(i_0)$, we refer in the following to the values $J^*(i_0)$, $J_\mu(i_0)$, and $J_u(i_0)$ simply as the optimal cost, the cost of policy μ, and the cost of control u at state i_0, respectively. Figure 1 shows the progress of lower and upper bounds for the approximation of the optimal cost, the cost of a given policy, and the cost of the control chosen by the policy at one particular state. This state was chosen to obtain a distinctive behavior of the approximations. The required approximation guarantee was chosen to be 5 % (thinner lines indicate the progress after reaching this gap). As mentioned before, estimating the value function of a given policy requires much fewer states than estimating the optimal value function. In this example we see that the considered policy is not optimal. Moreover, the chosen control of the policy also does not seem to be optimal, even though Figure 1 does not provide a proof for that.

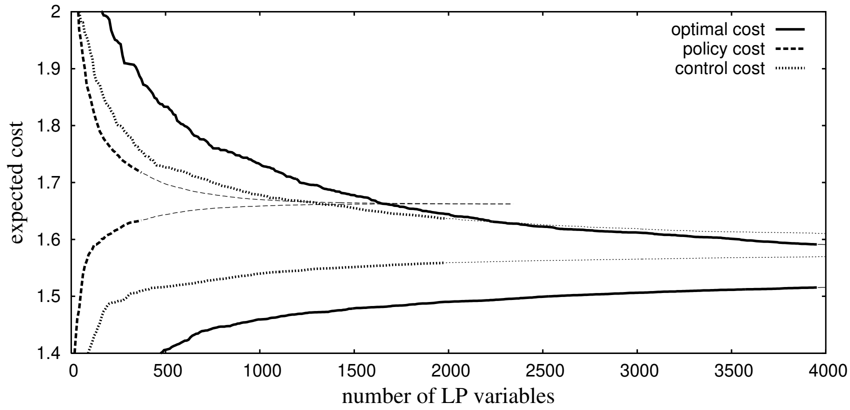

Fig. 1. Approximation progress for the optimal cost, the cost of a given policy, and the cost of the control chosen by the policy.

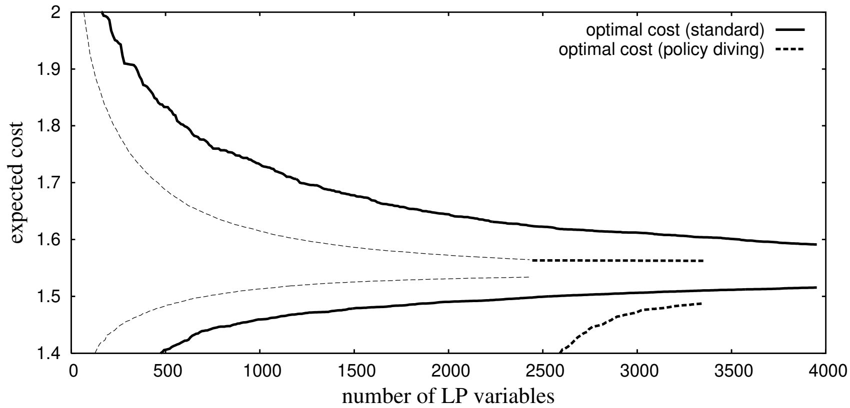

Fig. 2. Approximation of the optimal cost using fixed policy diving: up to approx. 2 500 variables we follow the fixed policy.

Note that in the beginning the upper bound on the cost of the policy considered in Figure 1 is much smaller than the upper bound on the optimal cost. This observation leads to the following algorithmic idea. One starts by approximating the cost of a known policy to some degree. The corresponding state space is then used to initialize the computation for approximating the optimal cost. This method might reduce the total number of required states, in particular if the utilized policy is good. Figure 2 depicts the progress in cost approximation by means of this algorithm that we call *fixed policy diving* for the same state as before. The thin line indicates the initial approximation of the cost function of the used policy.

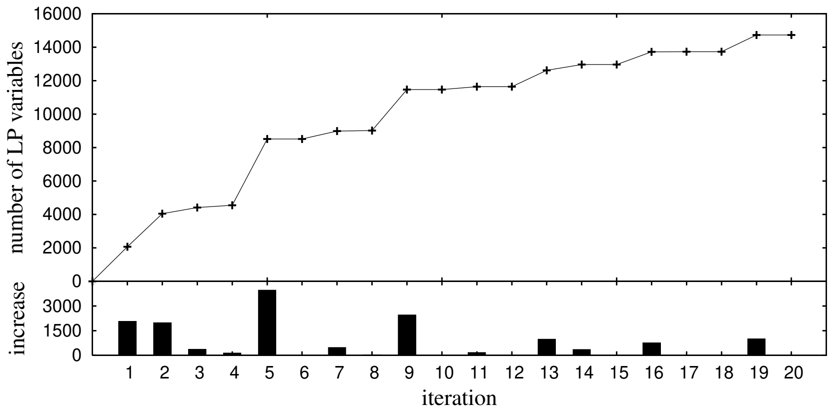

Fig. 3. Required states for approximating a sample path of states.

Finally, we investigate to what extend our algorithm can make use of a warm start when several states of a sampled path in the state space are to be considered. After approximating the cost of the first state in the sample we use the generated set $\hat{S}$ to start the approximation for the second state, and so on. Figure 3 shows how many additional states in each iteration are needed to approximate the optimal cost function at the given state up to an accuracy of 10 %. In each step the new state is sampled for the case that a currently best control is applied. Apparently, our algorithm is suitable for using such a kind of a warm start. More than the half of all used states are generated in the first five steps of the sample consisting of 20 states in total. During this process the linear program becomes very large. Therefore, one has to think about removing some states at some point in order to apply this method continuously.

References

1. Andrew G. Barto, Steven J. Bradtke, and Satinder P. Singh, *Learning to act using real-time dynamic programming*, Artificial Intelligence **72** (1995), 81–138.
2. Dimitri P. Bertsekas, *Dynamic programming and optimal control*, 2 ed., vol. 1 and 2, Athena Scientific, Belmont, 2001.

3. Allan Borodin and Ran El-Yaniv, *Online computation and competitive analysis*, Cambridge University Press, 1998.
4. Daniela P. de Farias and Benjamin Van Roy, *The linear programming approach to approximate dynamic programming*, Operations Research **51** (2003), no. 6, 850–865.
5. ______, *On constraint sampling in the linear programming approach to approximate dynamic programming*, Mathematics of Operations Research **29** (2004), no. 3, 462–478.
6. Thomas L. Dean, Leslie Pack Kaelbling, Jak Kirman, and Ann E. Nicholson, *Planning with deadlines in stochastic domains.*, AAAI, 1993, pp. 574–579.
7. Stefan Heinz, *Policies for online target date assignment problems: Competitive analysis versus expected performance*, Master's thesis, Technische Universität Berlin, 2005.
8. Stefan Heinz, Sven O. Krumke, Nicole Megow, Jörg Rambau, Andreas Tuchscherer, and Tjark Vredeveld, *The online target date assignment problem*, Proceedings of the 3rd Workshop on Approximation and Online Algorithms (Thomas Erlebach and Giuseppe Persiano, eds.), vol. 3879, Springer, 2005, pp. 230–243.
9. Michael J. Kearns, Yishay Mansour, and Andrew J. Ng, *A sparse sampling algorithm for near-optimal planning in large Markov decision processes*, International Joint Conferences on Artificial Intelligence, 1999, pp. 1324–1331.
10. Michael L. Littman, Thomas L. Dean, and Leslie Pack Kaelbling, *On the complexity of solving Markov decision problems*, Proceedings of the Eleventh Annual Conference on Uncertainty in Artificial Intelligence (UAI–95) (Montreal, Québec, Canada), 1995, pp. 394–402.
11. Martin L. Putermann, *Markov decision processes: Discrete stochastic dynamic programming*, 2 ed., John Wiley and Sons, Inc., Hoboken, New Jersey, 2005.