Lazy Approximation for Solving Continuous Finite-Horizon MDPs

Lihong Li and **Michael L. Littman**
RL^3 Laboratory
Dept. of Computer Science
Rutgers University
Piscataway, NJ 08854
{lihong,mlittman}@cs.rutgers.edu

Abstract

Solving Markov decision processes (MDPs) with continuous state spaces is a challenge due to, among other problems, the well-known curse of dimensionality. Nevertheless, numerous real-world applications such as transportation planning and telescope observation scheduling exhibit a critical dependence on continuous states. Current approaches to continuous-state MDPs include discretizing their transition models. In this paper, we propose and study an alternative, discretization-free approach we call *lazy approximation*. Empirical study shows that lazy approximation performs much better than discretization, and we successfully applied this new technique to a more realistic planetary rover planning problem.

Introduction and Previous Work

Probabilistic planning focuses on decision making under environment uncertainty, in contrast to conventional AI planning (Fikes & Nilsson 1971). One of the principal models for probabilistic planning is Markov decision processes (MDPs) (Puterman 1994). Dynamic programming (DP) (Bellman 1957) is the most common approach to solving MDPs, but it suffers from the well-known *curse of dimensionality*, which observes that state spaces increase exponentially with the number of dimensions used to describe them. In addition to high dimensionality, many real-world applications are continuous, raising additional challenges for solution algorithms. Existing methods, including those studied extensively in the reinforcement-learning (Sutton & Barto 1998) literature, are typically for discrete MDPs. Function approximation (Boyan, Moore, & Sutton 1995) provides a possible solution when the MDP is continuous, but it is usually difficult to analyze and can fail to converge in many cases (Baird 1995; Bertsekas & Tsitsiklis 1996).

More recently, several techniques have been put forward to tackle MDPs with continuous state spaces. Boyan & Littman (2001) describe a class of MDPs called *time-dependent MDPs* (TiMDPs), in which transitions take place along a single, irreversible continuous dimension. They provide an algorithm for computing an exact finite-horizon value function by DP when the transition probabilities are discrete and rewards are piecewise linear (PWL). This idea was extended to higher dimensional state spaces by Feng *et al.* (2004), who examined MDPs with two special types of reward models: piecewise constant (PWC) and piecewise linear convex (PWLC). Munos & Moore (2002) have discussed various ways for variable resolution in continuous state spaces.

Although these approaches were successfully applied to realistic problems such as transportation planning, telescope observation scheduling, and rover planning, they make a common assumption that the transition model of the MDP is *discrete*, meaning positive probabilities are only assigned to a finite set of outcomes. However, probability density functions (pdfs) of transitions of many problems encountered in practice are continuous in nature. For example, in rover planning the duration and energy consumption of an action is best modelled as a normal distribution. For the earlier algorithms to be applied to such problems, they must discretize the continuous pdfs with sufficiently high resolution to preserve accuracy. Such discretization can result in exponential blow-ups, severely limiting their applicability to large-scale problems. Furthermore, additional errors will be introduced by discretization.

In this paper, we go one step further by developing an alternative approach to handling continuous state-space MDPs with continuous transition pdfs. The philosophy of our approach is *lazy approximation* (LA, for short): In contrast to the aforementioned methods that approximate/discretize an MDP at the first step and then solve the approximate model exactly, the new method postpones the approximation stage and is able to control the tradeoff between accuracy and compactness of the computed value functions. For convenience, the discretization method will be referred to as DM.

The rest of the paper is organized as follows. The next section introduces the basic idea of LA in the simplest case, and then discusses how to extend it to more general situations. Then, we provide several insights into the advantages of LA, particularly its ability to control approximation error and compactness directly, and describe a series of empirical validations. Finally, we present the application of the technique to a planetary rover-planning problem. Space limitation does not allow us to present all technical details. Interested readers are referred to a longer paper by Li & Littman (2005).

Copyright © 2005, American Association for Artificial Intelligence (www.aaai.org). All rights reserved.

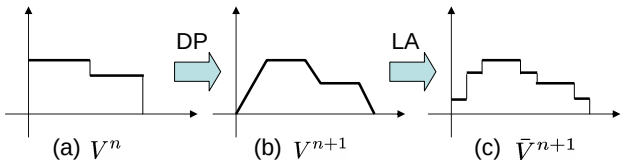

Figure 1: Illustration of lazy approximation.

The Lazy-Approximation Method

LA will be developed in three steps. Starting with the notation and background used in later discussions, we move on to the simplest case with one dimensional state space and PWC transition models. Finally, we extend the idea to more complex situations by removing these assumptions.

Preliminaries

A continuous, finite-horizon MDP is described as a four tuple: $M = \langle X, A, R, T \rangle$, where $X \subset \Re^k$ is a k dimensional state space, A is a finite set of actions, $R(x, a)$ is the expected immediate reward by taking action $a \in A$ in state $x \in X$, and $T(x'|xa) = \Pr\{x_{t+1} = x'|x_t = x, a_t = a\}$ is the Markovian transition function. For convenience, we adopt the convention that $X = [0, 1)^k$.

Given an MDP M, our objective is to compute its optimal value function at finite horizons: $V^n(x)$ with horizon $n = 0, 1, 2, \cdots, T$. The Bellman equation provides a foundation for dynamic programming:

$$V^{n+1}(x) = \max_{a \in A} \left\{ R(x, a) + \int_X T(x'|xa) V^n(x') dx' \right\}. \quad (1)$$

We define $V^0(x) \equiv 0$, and assume $R(x, a)$ is PWC. The transition function T can be either *abstract* or *relative* (Boyan & Littman 2001). As will be shown shortly, DP with a relative model T will increase the order of polynomial functions V^n as n increases, while an abstract T will retain the order. For this reason, we will focus on the more challenging relative models, i.e., $T(x'|xa) = \Pr\{x' - x|a\}$.

The Basic Idea

Consider the simplest case with $k = 1$ and PWC $T(x'|xa)$. Since $T(x'|xa)$ is for a relative model, the integral in Eqn (1) is in fact a *convolution* of $V^n(x)$ and $\Pr\{x' - x|a\}$. If both $V^n(x)$ and $\Pr\{x' - x|a\}$ are PWC, then after a DP step at horizon $n + 1$ (i.e., Eqn (1)), the value function $V^{n+1}(x)$ is in general PWL, as shown in Fig 1 (b).

Unfortunately, directly convolving V^{n+1} with $T(x'|xa)$ at horizon $n + 2$ will in turn produce a piecewise *quadratic* value function V^{n+2}. The order of $V^n(x)$ increases with n, which renders such a computation intractable in practice. To make it possible to compute the value function efficiently at higher horizons, we could do approximation before V^{n+2} is computed. Specifically, if we approximate V^{n+1} by a PWC function $\bar{V}^{n+1}$ (Fig 1 (c)), it is then used in Eqn (1) to obtain $\hat{V}^{n+2}$, which is a reasonable approximation of V^{n+2} when $\bar{V}^{n+1}$ is close enough to V^{n+1}. This approximation procedure can be iterated and continues to higher horizons. Fig 2 provides a comparative illustration of LA and DM.

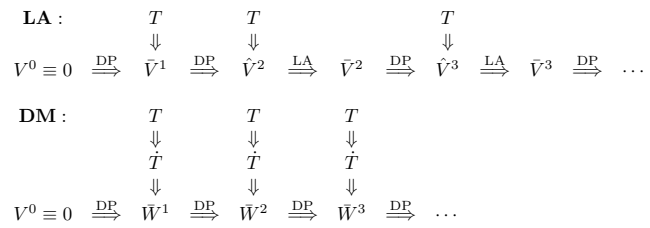

Figure 2: Complete procedures of lazy approximation (LA) and discretization method (DM) for solving finite-horizon MDPs. $\hat{V}^n$ are PWL; $\bar{V}^n$ and $\bar{W}^n$ are PWC; $\dot{T}$ is discrete.

LA needs to turn a PWL function into a PWC one. Clearly, there are a number of reasonable schemes for making this transformation. A well-known relation between $||V - \tilde{V}||_\infty$ and the quality of policy induced from $\tilde{V}$ (Singh & Yee 1994) suggests we minimize the $\mathcal{L}_\infty$ error:

$$\epsilon_n \;=\; ||\hat{V}^n - \bar{V}^n||_\infty \;=\; \sup_x |\hat{V}^n(x) - \bar{V}^n(x)|. \quad (2)$$

We will be able to bound the approximation error of $\bar{V}^n$ later in terms of ϵ_n.

Dealing with Non-PWC Transitions

In practice, however, the transition functions T are not always PWC. In such cases, we could approximate T by a PWC $\bar{T}$. Note that such an additional approximation will introduce error in the resulting value-function approximations. Our empirical studies later show that, even in this case, the LA approach still outperforms DM.

Dealing with Higher Dimensional Spaces

A function is PWC in a k-dimensional space if there is a *rectangular partition* $\mathcal{P}$ of the state space such that (i) each piece $P \in \mathcal{P}$ is a hyper-rectangle (i.e., the Cartesian product of intervals at each dimension), and (ii) the function value within each $P \in \mathcal{P}$ is constant. For simplicity, we will refer to the term "hyper-rectangle" by "rectangle".

The computation of Eqn (1) becomes more complicated when $k > 1$. We observed that the convolution of two PWC functions produces a function defined on a rectangular partition $\mathcal{P}$ and within each $P \in \mathcal{P}$ the function is of the form:

$$\hat{V}^{n+1}(\vec{x}) \;=\; \prod_{i=1}^{k} (a_i x_i + b_i), \quad \vec{x} \in P. \quad (3)$$

Although this form seems disappointing at first glance, it has a nice property that its extreme values are always obtained at the vertices of rectangle P. Therefore, we have an exact way to compute $\bar{V}^n$ as an approximation to $\hat{V}^n$ so that the $\mathcal{L}_\infty$ error, ϵ_n, is minimized: It suffices to find the minimum and maximum values and take the midpoint, which can be done in $\Theta(k)$ time.

A natural choice of data structures for rectangular partitioning of a continuous space is kd-trees (Friedman, Bentley, & Finkel 1977). But, there could be better choices in some situations by incorporating background knowledge.

Dealing with Discrete State Components

In real-life applications, it is common for the state space to consist of both continuous and discrete components. That is,

the state space is $X \times S$, where $X \subset \Re^k$ is continuous as before and S is discrete. For example, a rover has continuous state components (e.g., remaining time and energy) as well as discrete components such as whether particular subtasks have been achieved or the status of observation equipment.

As shown below, such a situation does not pose any essential difficulty to LA. Therefore, we will stick to the notation $V^n(x)$ to represent the value function in other parts of the paper, and only in this subsection will we use $V^n(x,s)$to distinguish these two types of state components. Specifically, the transition model $T(x',s'|xsa)$ can be factored into: $T(x',s'|xsa) = T(x'|xsas') \cdot T(s'|xsa)$. Consequently, the integral of Eqn (1) is rewritten as:

$$\begin{aligned} & \int_{S,X} T(x',s'|xs)V^n(x,s)dx'ds' \\ = & \int_S \left(T(s'|xsa) \int_X T(x'|xsas')V^n(x,s)dx' \right) ds' \\ = & \sum_{s' \in S} \left(T(s'|xsa) \int_X T(x'|xsas')V^n(s,s)dx' \right). \end{aligned}$$

This derivation reduces the problem to the one considered before, and each dynamic-programming step will still produce a function with pieces in the form of Eqn (3).

Theoretical Analysis

A complete analysis for LA is indeed difficult and beyond the scope of this paper. Instead, we will provide some insights on why it tends to be better than DM, including its error- and compactness-control mechanism.

Error Control

We first observe that both the $\max$ and addition operators of a set of PWC functions produces another PWC function. This allows us to focus on the integral in Eqn (1) and simplify analysis by dropping the inessential $\max$ and $R(x,a)$.

Define ϵ_n as before and ε_n to be the largest error in approximating the true V^n by $\bar{V}^n$:

$$\varepsilon_n = ||\bar{V}^n - V^n||_\infty = \sup_x |\bar{V}^n - V^n|. \quad (4)$$

We can bound ε_{n+1} in terms of ϵ_{n+1} and ε_n:

$$\begin{aligned} \varepsilon_{n+1} &= ||\bar{V}^{n+1} - V^{n+1}||_\infty \\ &\leq ||\bar{V}^{n+1} - \hat{V}^{n+1}||_\infty + ||\hat{V}^{n+1} - V^{n+1}||_\infty \\ &= \epsilon_{n+1} + \sup_x \left| \int_X T(x'|xa) \left(\bar{V}^n(x') - V^n(x') \right) dx' \right| \\ &\leq \epsilon_{n+1} + \varepsilon_n \sup_x \left| \int_X T(x'|xa)dx' \right| \\ &= \epsilon_{n+1} + \varepsilon_n. \end{aligned} \quad (5)$$

Therefore, if $\epsilon_1, \epsilon_2, \cdots, \epsilon_n$ are kept small, we will end up with an accurate value function approximation $\bar{V}^n$ at horizon n. There are examples showing that the bound in Eqn (5) is tight.

Similarly, we can bound the approximation error of DM. Let $\bar{W}^n$ be the function computed by the DM at horizon n, and the approximation error be

$$\xi_n = ||\bar{W}^n - V^n||_\infty = \sup_x |\bar{W}^n - V^n|.$$

Denote the discretized transition of T by $D = \{d_i\}$—a discrete pdf. In other words, D partitions the state space into $\bigcup_i \Delta_i$, each Δ_i corresponding to a $d_i \in D$. Now, define ρ_n, which captures the *smoothness* of V^n:

$$\rho_n = \sum_i \left(d_i \left(\sup_{D_i} V^n - \inf_{D_i} V^n \right) \right). \quad (6)$$

Hence, the smoother V^n is, the smaller ρ_n is. Using these definitions, we have the following bound on ξ_{n+1} in terms of ξ_n and ρ_n:

$$\xi_{n+1} \leq \rho_n + \xi_n. \quad (7)$$

This bound is also tight. It is desirable to have a smooth V^n or a high resolution so that ρ_n is then small and ξ_{n+1} will be close to ξ_n, meaning the approximation error is small. In practice, however, it is usually unknown a priori how smooth V^n is, and thus difficult to decide the appropriate resolution r for discretization beforehand. Besides, the smoothness of V^n may not be uniform—only a certain state subspace may need fine discretization.

In contrast, LA controls ε_{n+1} *explicitly* by keeping ϵ_n small. We emphasize here that ϵ_n is provided by the user and is controllable. We believe this direct approach is easier for people and provides more flexibility in making tradeoff between accuracy and running time. More insights will be provided after discussing the compactness-control mechanism in the next subsection.

Compactness Control

In our experiment as well as analysis, we find that the memory (space) and running time required to compute $\hat{V}^{n+1}$ (especially the integral) largely depends on the sizes of $\bar{V}^n$ and T, that is, how many rectangular pieces they have. In order to prevent their sizes from growing without control, merging neighboring regions with similar values is usually necessary. During merging, a small positive δ is provided and two neighboring regions are merged into one if their value difference is less than δ. Merging is also necessary for DM. Otherwise, if the resolution of discretization at each dimension is r, then each DP step can, in the worst case, multiply the function size by a factor of r^k.

Unlike DM, LA tries to keep the function compact by using the true transition, which is usually much more compact than a discretized representation. Consequently, the complexity of the resulting value-function approximations largely depends on how complex the target value function actually is. In this way, we could avoid unnecessarily high resolution that is difficult to eliminate in DM. If, unfortunately, the target function is too complicated to represent compactly, LA will also suffer. But, this is not because of the algorithm itself, but of the nature of the problem.

A Brief Summary

From the results above, we conclude that LA is more flexible than DM: On the one hand, it directly makes ϵ_n small to control the growth of ε_n; on the other hand, it is able to make a tradeoff between ε_n and the compactness of $\bar{V}^n$.

Empirical Study

We conducted two sets of experiments, one on synthetic problems, and the other on a larger and more realistic problem of planetary rover planning.

Experiments on Synthetic Problems

In this section, we report results on randomly generated problems. Two types of MDPs are considered: (I) There is a positive, PWC reward function at a terminating horizon 10 steps away, other rewards are 0, and the transition functions are PWC; (II) Same as (I) except that the transition function is a normal distribution.

We first study how values of ϵ_n in LA and resolution r in DM affect the running time and size of value-function approximations. Figs 3 and 4 plot the running time and size as functions of $\mathcal{L}_\infty$-error and root mean squared error (RMSE) of a typical run. The data in Fig 3 were obtained by varying r and ϵ_n in DM and LA, respectively. In Fig 4, since the true transition was approximated by a PWC function, we also investigated this affect by either (i) fixing the size of the PWC transition representation to be 20 and varying ϵ_n, or (ii) fixing $\epsilon_n = 0.01$ and varying the size of the PWC transition representation. Therefore, there are two series of data corresponding to LA in Fig 4. Several observations follow.

1. In almost all cases, LA requires much less running time and value-function representation sizes to achieve the same error level. For example, the speedup is usually about 2 orders of magnitude (notice the logarithmic scale on the y-axis).
2. It is interesting to notice that LA requires more time to achieve very small RMSE in Fig 4 (b). This is not surprising as the approximation scheme adopted in our experiment is to minimize the $\mathcal{L}_\infty$ error, rather that RMSE. In comparison, LA is very good at keeping $\mathcal{L}_\infty$ error small (c.f., Fig 4 (a,c)).
3. Comparing the growth of the fixed-resolution and fixed-epsilon curves of LA in Fig 4 suggests it may be better to fix ϵ_n and then increase the accuracy in approximating the transition model.

We next study how errors evolve as horizon increases. In order to help visualize how well LA controls the error over horizons, we show the true value functions, as well as the approximations computed by DM and LA, at horizons 1, 5, and 10 of a randomly selected but typical run (Fig 5). We found that although both algorithms produce value functions of comparable approximation errors at early horizons, the error of DM grows faster than LA, and ends up taking *much longer* to find a *much larger* representation of a *much less accurate* result at horizon 10.

Application to Rover Planning

Planning under uncertainties with resource limitation (Bresina *et al.* 2002) such as the Mars rover planning problem is the motivation behind this paper. A number of non-trivial sources of uncertainty exist in rover operation, including the duration and power consumption of actions. Furthermore, the typical size of problems is

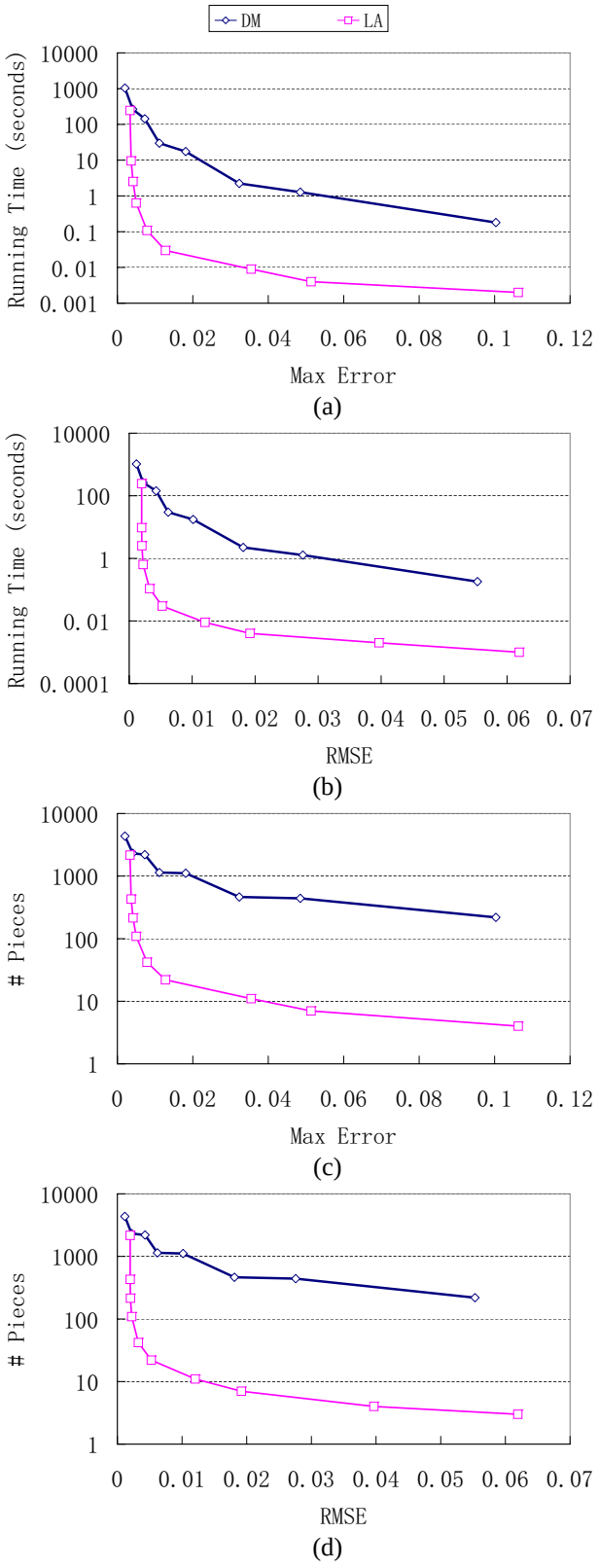

Figure 3: Results of a typical run on a problem instance of type (I). In the DM curves, we varied the resolution r from 10 to 500; in the LA curves, we varied ϵ_n from 0.1 to 0.0001.

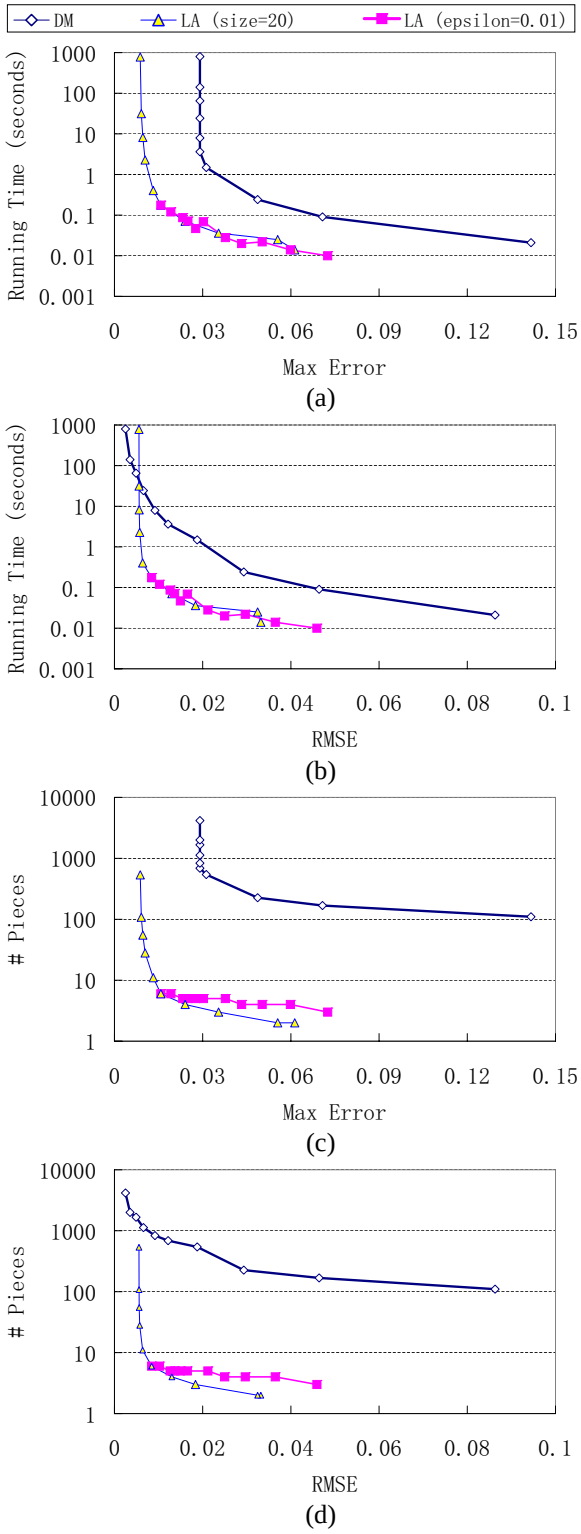

Figure 4: Results of a typical run on a problem instance of type (II). In the DM curves, we varied the resolution r from 10 to 500; in the LA curves, we either varied the size of the PWC transition from 3 to 20, fixing $\epsilon_n = 0.01$, or varied ϵ_n from 0.1 to 0.0001, fixing the size of the PWC transition to be 20.

prohibitively large and, therefore, exact, optimal solutions are not possible in most cases. Recent work modelled the problem as a finite-horizon MDP (Bresina *et al.* 2002; Feng *et al.* 2004), and given the MDP model, the objective is to approximate the optimal value function for each horizon length to generate near-optimal plans.

The rover planning problem is as follows. Given a set of locations, the associated target object at each location, and the paths between locations, the rover is to travel around and take pictures of targets before running out of time or energy. The utility of taking a picture is dependent on the target as well as when the picture is taken. Before taking a picture, the rover has to perform a sequence of actions such as identifying its location, tracking the object, navigating, and initializing its instruments, etc., and some of the actions are also dependent on other actions and the rover's state.

Even for a minimal-size problem involving only two locations and targets, the state space is very large—it has a discrete component of dimension 12 (4096 discrete states) and a continuous component of dimension 2, and there are 14 actions (some are stochastic with uncertain durations and power consumptions according to Gaussian distributions). It is impossible to discretize the MDP model and obtain accurate value functions using classical methods like value iteration in a reasonable amount of time, even if human knowledge is employed to prune unreachable states.

We applied LA to a version of this problem specified by NASA researchers. PWC functions were used to approximate normal distributions appearing in action durations and energy consumptions. We found that LA produced reasonable and consistent results even if the PWC transition approximations are not complex. In particular, Fig 6 shows the computed value function at horizon 7 for the initial rover configuration (i.e., its initial discrete state), using a PWC transition representation of size 64.

The graph, with plateaus and humps, are typical for value functions if there are finitely many goal states with positive utility and resource constraints (Feng *et al.* 2004). The flat, zero-valued regions correspond to situations where either time or energy is not sufficient to perform a task at any target (and hence the expected utility is 0). When the rover has more resources, it has positive utility expectations, and the more resources it has, the higher the expectation is. At the corner around the point $(1, 1)$ is a hump because the amount of resources allows the rover to carry out a second task, increasing its utility.

Conclusions

Future Work There are several interesting directions for future work. First, we are investigating more time- and space-efficient ways for doing the intermediate stages of lazy approximation, including improving the approximation and merging mechanisms. Second, we are seeking other data structures that best fit the purpose of efficiently representing functions for computing Eqn (1). Note that kd-trees may not be the best choice. A promising approach is using the spatial relation among pieces of the rectangular participation. Third, it may be helpful to look into and compare

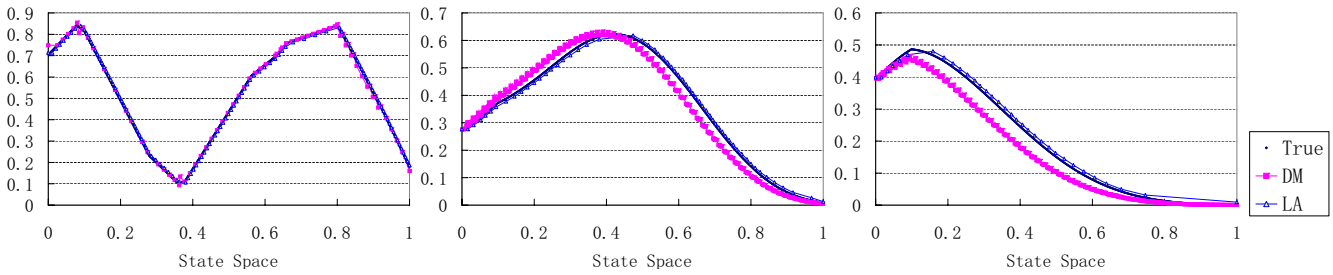

Figure 5: Value functions at horizons $1, 5, 10$ (from left to right). DM (with resolution of 20) took $0.1542s$ and produced a value function at horizon 10 of size 401. LA (with $\epsilon = 0.01$) took only $0.008s$ and ended up with a function of size 33 at horizon 10.

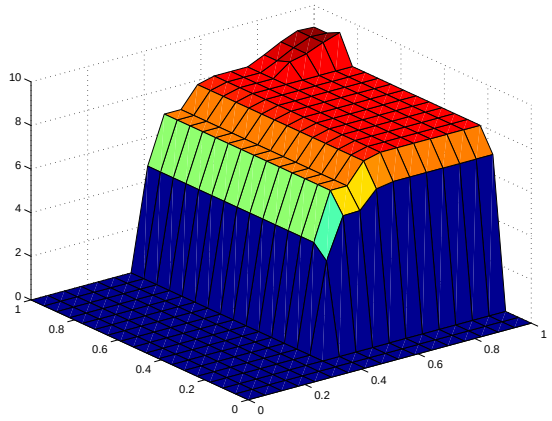

Figure 6: Computed value function for rover problem at horizon 7 for the initial configuration, with PWC transition of size 64.

rectangular partitions with others such as the Kuhn triangulations (Munos & Moore 2002), which also provides possible thoughts for better merging mechanisms.

Conclusions In this paper, we proposed and studied an approach to solving continuous, finite-horizon MDPs. In contrast to traditional discretization methods that approximate the MDP model *beforehand*, the new approach retains a continuous model and *defers* approximation until necessary. Some insights were given to explain the advantage of lazy approximation, and it was shown that, on a set of synthetic problems, lazy approximation performs consistently much better than the discretization method. It was also successfully applied to a large, real-life planning problem for planetary rovers. We look forward to its application to other non-trivial probabilistic-planning problems.

Acknowledgements

We thank Nicolas Meuleau, Zhengzhu Feng, Emmanuel Benazera, Victor Lee, and the RL^3 members for help in conducting experiments. The Rutgers Center for Advanced Information Processing (CAIP) provided their high-end machines for our experiments. This research is supported by NASA (solicitation number: NRA2-38169).

References

Baird, L. 1995. Residual algorithms: Reinforcement learning with function approximation. In *Proceedings of the Twelfth International Conference on Machine Learning (ICML-95)*, 30–37.

Bellman, R. 1957. *Dynamic Programming*. Princeton University Press.

Bertsekas, D. P., and Tsitsiklis, J. N. 1996. *Neuro-Dynamic Programming*. Athena Scientific.

Boyan, J. A., and Littman, M. L. 2001. Exact solutions to time-dependent MDPs. In *Advances in Neural Information Processing Systems 13 (NIPS-00)*, 1026–1032.

Boyan, J. A.; Moore, A. W.; and Sutton, R. S., eds. 1995. *Proceedings of the ICML-95 Workshop on Value Function Approximation*.

Bresina, J. L.; Dearden, R.; Meuleau, N.; Ramkrishnan, S.; Smith, D. E.; and Washington, R. 2002. Planning under continuous time and resource uncertainty: A challenge for AI. In *Proceedings of the Eighteenth Conference on Uncertainty in Artificial Intelligence (UAI-02)*, 77–84.

Feng, Z.; Dearden, R.; Meuleau, N.; and Washington, R. 2004. Dynamic programming for structured continuous Markov decision problems. In *Proceedings of the Twentieth Conference on Uncertainty in Artificial Intelligence (UAI-04)*, 154–161.

Fikes, R., and Nilsson, N. J. 1971. STRIPS: A new approach to the application of theorem proving to problem solving. *Artificial Intelligence* 2(3–4):189–208.

Friedman, J. H.; Bentley, J. L.; and Finkel, R. A. 1977. An algorithm for finding best matches in logarithmic expected time. *ACM Transactions on Mathematical Software* 3(3):209–226.

Li, L., and Littman, M. L. 2005. Lazy approximation: A new approach for solving continuous finite-horizon MDPs. Technical Report 577, Department of Computer Science, Rutgers University.

Munos, R., and Moore, A. W. 2002. Variable resolution discretization in optimal control. *Machine Learning* 49(2–3):291–323.

Puterman, M. L. 1994. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. New York: Wiley-Interscience.

Singh, S. P., and Yee, R. C. 1994. An upper bound on the loss from approximate optimal-value functions. *Machine Learning* 16(3):227–233.

Sutton, R. S., and Barto, A. G. 1998. *Reinforcement Learning: An Introduction*. Cambridge, MA: MIT Press.