Journal of Artificial Intelligence Research 27 (2006) 153–201 Submitted 05/06; published 10/06

Solving Factored MDPs with Hybrid State and Action Variables

Branislav Kveton BKVETON@CS.PITT.EDU
Intelligent Systems Program
5406 Sennott Square
University of Pittsburgh
Pittsburgh, PA 15260

Milos Hauskrecht MILOS@CS.PITT.EDU
Department of Computer Science
5329 Sennott Square
University of Pittsburgh
Pittsburgh, PA 15260

Carlos Guestrin GUESTRIN@CS.CMU.EDU
Machine Learning Department and
Computer Science Department
5313 Wean Hall
Carnegie Mellon University
Pittsburgh, PA 15213

Abstract

Efficient representations and solutions for large decision problems with continuous and discrete variables are among the most important challenges faced by the designers of automated decision support systems. In this paper, we describe a novel hybrid factored Markov decision process (MDP) model that allows for a compact representation of these problems, and a new hybrid approximate linear programming (HALP) framework that permits their efficient solutions. The central idea of HALP is to approximate the optimal value function by a linear combination of basis functions and optimize its weights by linear programming. We analyze both theoretical and computational aspects of this approach, and demonstrate its scale-up potential on several hybrid optimization problems.

1. Introduction

A dynamic decision problem with components of uncertainty can be very often formulated as a Markov decision process (MDP). An MDP represents a controlled stochastic process whose dynamics is described by state transitions. Objectives of the control are modeled by rewards (or costs), which are assigned to state-action configurations. In the simplest form, the states and actions of an MDP are discrete and unstructured. These models can be solved efficiently by standard dynamic programming methods (Bellman, 1957; Puterman, 1994; Bertsekas & Tsitsiklis, 1996).

Unfortunately, textbook models rarely meet the practice and its needs. First, real-world decision problems are naturally described in a factored form and may involve a combination of discrete and continuous variables. Second, there are no guarantees that compact forms of the optimal value function or policy for these problems exist. Therefore, hybrid optimization problems are usually discretized and solved approximately by the methods for discrete-state

MDPs. The contribution of this work is a principled, sound, and efficient approach to solving large-scale factored MDPs that avoids this discretization step.

Our framework is based on approximate linear programming (ALP) (Schweitzer & Seidmann, 1985), which has been already applied to solve decision problems with discrete state and action variables efficiently (Schuurmans & Patrascu, 2002; de Farias & Van Roy, 2003; Guestrin et al., 2003). These applications include context-specific planning (Guestrin et al., 2002), multiagent planning (Guestrin et al., 2002), relational MDPs (Guestrin et al., 2003), and first-order MDPs (Sanner & Boutilier, 2005). In this work, we show how to adapt ALP to solving large-scale factored MDPs in hybrid state and action spaces.

The presented approach combines factored MDP representations (Sections 3 and 4) and optimization techniques for solving large-scale structured linear programs (Section 6). This leads to various benefits. First, the quality and complexity of value function approximations is controlled by using basis functions (Section 3.2). Therefore, we can prevent an exponential blowup in the complexity of computations when other techniques cannot. Second, we always guarantee that HALP returns a solution. Its quality naturally depends on the choice of basis functions. As analyzed in Section 5.1, if these are selected appropriately, we achieve a close approximation to the optimal value function V^*. Third, a well-chosen class of basis functions yields closed-form solutions to the backprojections of our value functions (Section 5.2). This step is important for solving hybrid optimization problems more efficiently. Finally, solving hybrid factored MDPs reduces to building and satisfying relaxed formulations of the original problem (Section 6). The formulations can be solved efficiently by the cutting plane method, which has been studied extensively in applied mathematics and operations research.

For better readability of the paper, our proofs are deferred to Appendix A. The following notation is adopted throughout the work. Sets and their members are represented by capital and small italic letters as $\mathcal{S}$ and s, respectively. Sets of variables, their subsets, and members of these sets are denoted by capital letters as $\mathbf{X}$, $\mathbf{X}_i$, and X_i. In general, corresponding small letters represent value assignments to these objects. The subscripted indices D and C denote the discrete and continuous variables in a variable set and its value assignment. The function $\mathrm{Dom}(\cdot)$ computes the domain of a variable or the domain of a function. The function $\mathsf{Par}(\cdot)$ returns the parent set of a variable in a graphical model (Howard & Matheson, 1984; Dean & Kanazawa, 1989).

2. Markov Decision Processes

Markov decision processes (Bellman, 1957) provide an elegant mathematical framework for modeling and solving sequential decision problems in the presence of uncertainty. Formally, a *finite-state Markov decision process (MDP)* is given by a 4-tuple $\mathcal{M} = (\mathcal{S}, \mathcal{A}, P, R)$, where $\mathcal{S} = \{s_1, \dots, s_n\}$ is a set of states, $\mathcal{A} = \{a_1, \dots, a_m\}$ is a set of actions, $P : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, 1]$ is a stochastic transition function of state dynamics conditioned on the preceding state and action, and $R : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is a reward function assigning immediate payoffs to state-action configurations. Without loss of generality, the reward function is assumed to be nonnegative and bounded from above by a constant $R_{\max}$ (Puterman, 1994). Moreover, we assume that the transition and reward models are stationary and known a priori.

Once a decision problem is formulated as an MDP, the goal is to find a policy $\pi : \mathcal{S} \to \mathcal{A}$ that maximizes some objective function. In this paper, the quality of a policy π is measured by the *infinite horizon discounted reward*:

$$\mathrm{E}_\pi\left[\sum_{t=0}^{\infty}\gamma^t R(s^{(t)},\pi(s^{(t)}))\,\middle|\, s^{(0)}\sim\varphi\right], \tag{1}$$

where $\gamma \in [0,1)$ is a *discount factor*, $s^{(t)}$ is the state at the time step t, and the expectation is taken with respect to all state-action trajectories that start in the states $s^{(0)}$ and follow the policy π thereafter. The states $s^{(0)}$ are chosen according to a distribution φ. This optimality criterion assures that there exists an *optimal policy* π^* which is stationary and deterministic (Puterman, 1994). The policy is greedy with respect to the *optimal value function* V^*, which is a fixed point of the Bellman equation (Bellman, 1957):

$$V^*(s) = \max_a\left[R(s,a) + \gamma\sum_{s'} P(s' \mid s,a)V^*(s')\right]. \tag{2}$$

The Bellman equation plays a fundamental role in all dynamic programming (DP) methods for solving MDPs (Puterman, 1994; Bertsekas & Tsitsiklis, 1996), including value iteration, policy iteration, and linear programming. The focus of this paper is on linear programming methods and their refinements. Briefly, it is well known that the optimal value function V^* is a solution to the *linear programming (LP)* formulation (Manne, 1960):

$$\begin{aligned} \text{minimize} \quad & \sum_s \psi(s)V(s) \\ \text{subject to:} \quad & V(s) \geq R(s,a) + \gamma\sum_{s'} P(s' \mid s,a)V(s') \quad \forall\ s\in\mathcal{S}, a\in\mathcal{A}; \end{aligned} \tag{3}$$

where $V(s)$ represents the variables in the LP, one for each state s, and $\psi(s) > 0$ is a strictly positive weighting on the state space $\mathcal{S}$. The number of constraints equals to the cardinality of the cross product of the state and action spaces $|\mathcal{S}\times\mathcal{A}|$.

Linear programming and its efficient solutions have been studied extensively in applied mathematics and operations research (Bertsimas & Tsitsiklis, 1997). The simplex algorithm is a common way of solving LPs. Its worst-case time complexity is exponential in the number of variables. The ellipsoid method (Khachiyan, 1979) offers polynomial time guarantees but it is impractical for solving LPs of even moderate size.

The LP formulation (3) can be solved compactly by the *cutting plane method* (Bertsimas & Tsitsiklis, 1997) if its objective function and constraint space are structured. Briefly, this method searches for violated constraints in relaxed formulations of the original LP. In every step, we start with a relaxed solution $V^{(t)}$, find a violated constraint given $V^{(t)}$, add it to the LP, and resolve for a new vector $V^{(t+1)}$. The method is iterated until no violated constraint is found, so that $V^{(t)}$ is an optimal solution to the LP. The approach has a potential to solve large structured linear programs if we can identify violated constraints efficiently (Bertsimas & Tsitsiklis, 1997). The violated constraint and the method that found it are often referred to as a *separating hyperplane* and a *separation oracle*, respectively.

Delayed column generation is based on a similar idea as the cutting plane method, which is applied to the column space of variables instead of the row space of constraints. Bender's and Dantzig-Wolfe decompositions reflect the structure in the constraint space and are often used for solving large structured linear programs.

3. Discrete-State Factored MDPs

Many real-world decision problems are naturally described in a factored form. Discrete-state factored MDPs (Boutilier et al., 1995) allow for a compact representation of this structure.

3.1 Factored Transition and Reward Models

A *discrete-state factored MDP* (Boutilier et al., 1995) is a 4-tuple $\mathcal{M} = (\mathbf{X}, \mathcal{A}, P, R)$, where $\mathbf{X} = \{X_1, \ldots, X_n\}$ is a state space described by a set of state variables, $\mathcal{A} = \{a_1, \ldots, a_m\}$ is a set of actions[1], $P(\mathbf{X}' \mid \mathbf{X}, \mathcal{A})$ is a stochastic transition model of state dynamics conditioned on the preceding state and action, and R is a reward function assigning immediate payoffs to state-action configurations. The state of the system is completely observed and represented by a vector of value assignments $\mathbf{x} = (x_1, \ldots, x_n)$. We assume that the values of every state variable X_i are restricted to a finite domain $\mathrm{Dom}(X_i)$.

Transition model: The transition model is given by the conditional probability distribution $P(\mathbf{X}' \mid \mathbf{X}, \mathcal{A})$, where $\mathbf{X}$ and $\mathbf{X}'$ denote the state variables at two successive time steps. Since the complete tabular representation of $P(\mathbf{X}' \mid \mathbf{X}, \mathcal{A})$ is infeasible, we assume that the transition model factors along $\mathbf{X}'$ as:

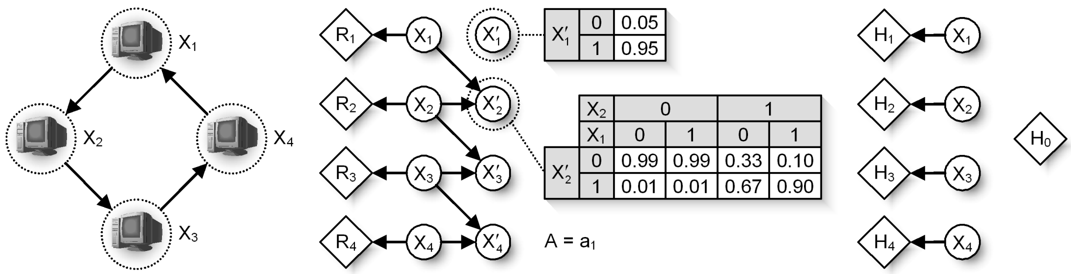
$$P(\mathbf{X}' \mid \mathbf{X}, a) = \prod_{i=1}^{n} P(X_i' \mid \mathsf{Par}(X_i'), a) \tag{4}$$

and can be described compactly by a *dynamic Bayesian network (DBN)* (Dean & Kanazawa, 1989). This DBN representation captures independencies among the state variables $\mathbf{X}$ and $\mathbf{X}'$ given an action a. One-step dynamics of every state variable is modeled by its conditional probability distribution $P(X_i' \mid \mathsf{Par}(X_i'), a)$, where $\mathsf{Par}(X_i') \subseteq \mathbf{X}$ denotes the parent set of X_i'. Typically, the parent set is a subset of state variables which simplifies the parameterization of the model. In principle, the parent set can be extended to the state variables $\mathbf{X}'$. Such an extension poses only few new challenges when solving the new problems efficiently (Guestrin, 2003). Therefore, we omit the discussion on the modeling of intra-layer dependencies in this paper.

Reward model: The reward model factors similarly to the transition model. In particular, the reward function $R(\mathbf{x}, a) = \sum_j R_j(\mathbf{x}_j, a)$ is an additive function of local reward functions defined on the subsets $\mathbf{X}_j$ and $\mathcal{A}$. In graphical models, the local functions can be described compactly by reward nodes R_j, which are conditioned on their parent sets $\mathsf{Par}(R_j) = \mathbf{X}_j \cup \mathcal{A}$. To allow this representation, we formally extend our DBN to an influence diagram (Howard & Matheson, 1984).

Example 1 (Guestrin et al., 2001) *To illustrate the concept of a factored MDP, we consider a network administration problem, in which the computers are unreliable and fail. The failures of these computers propagate through network connections to the whole network. For instance, if the server X_1 (Figure 1a) is down, the chance that the neighboring computer X_2*

1. For simplicity of exposition, we discuss a simpler model, which assumes a single action variable $\mathcal{A}$ instead of the factored action space $\mathbf{A} = \{A_1, \ldots, A_m\}$. Our conclusions in Sections 3.1 and 3.3 extend to MDPs with factored action spaces (Guestrin et al., 2002).



(a) **(b)** **(c)**

Figure 1: **a.** Four computers in a ring topology. Direction of propagating failures is denoted by arrows. **b.** A graphical representation of factored transition and reward models after taking an action a_1 in the 4-ring topology. The future state of the server X_1' is independent of the rest of the network because the server is rebooted. Reward nodes R_1 and R_j ($j \geq 2$) denote the components $2x_1$ and x_j ($j \geq 2$) of the reward model. **c.** A graphical representation of the linear value function approximation $V^{\mathbf{w}}(\mathbf{x}) = w_0 + \sum_{i=1}^{4} w_i x_i$ in the 4-ring topology. Reward nodes H_0 and H_i ($i \geq 1$) denote the value function components w_0 and $w_i x_i$ ($i \geq 1$).

crashes increases. The administrator can prevent the propagation of the failures by rebooting computers that have already crashed.

This network administration problem can be formulated as a factored MDP. The state of the network is completely observable and represented by n binary variables $\mathbf{X} = \{X_1, \dots, X_n\}$, where the variable X_i denotes the state of the i-th computer: 0 (being down) or 1 (running). At each time step, the administrator selects an action from the set $\mathcal{A} = \{a_1, \dots, a_{n+1}\}$. The action a_i ($i \leq n$) corresponds to rebooting the i-th computer. The last action a_{n+1} is dummy. The transition function reflects the propagation of failures in the network and can be encoded locally by conditioning on the parent set of every computer. A natural metric for evaluating the performance of an administrator is the total number of running computers. This metric factors along the computer states x_i and can be represented compactly by an additive reward function:

$$R(\mathbf{x}, a) = 2x_1 + \sum_{j=2}^{n} x_j.$$

The weighting of states establishes our preferences for maintaining the server X_1 and workstations $X_2, \dots, X_n$. An example of transition and reward models after taking an action a_1 in the 4-ring topology (Figure 1a) is given in Figure 1b.

3.2 Solving Discrete-State Factored MDPs

Markov decision processes can be solved by exact DP methods in polynomial time in the size of the state space $\mathbf{X}$ (Puterman, 1994). Unfortunately, factored state spaces are exponential in the number of state variables. Therefore, the DP methods are unsuitable for solving large factored MDPs. Since a factored representation of an MDP (Section 3.1) may not guarantee a structure in the optimal value function or policy (Koller & Parr, 1999), we resort to value function approximations to alleviate this concern.

Value function approximations have been successfully applied to a variety of real-world domains, including backgammon (Tesauro, 1992, 1994, 1995), elevator dispatching (Crites & Barto, 1996), and job-shop scheduling (Zhang & Dietterich, 1995, 1996). These partial successes suggest that the approximate dynamic programming is a powerful tool for solving large optimization problems.

In this work, we focus on *linear value function approximation* (Bellman et al., 1963; Van Roy, 1998):

$$V^{\mathbf{w}}(\mathbf{x}) = \sum_i w_i f_i(\mathbf{x}). \tag{5}$$

The approximation restricts the form of the value function $V^{\mathbf{w}}$ to the linear combination of $|\mathbf{w}|$ basis functions $f_i(\mathbf{x})$, where $\mathbf{w}$ is a vector of optimized weights. Every basis function can be defined over the complete state space $\mathbf{X}$, but usually is limited to a small subset of state variables $\mathbf{X}_i$ (Bellman et al., 1963; Koller & Parr, 1999). The role of basis functions is similar to features in machine learning. They are often provided by domain experts, although there is a growing amount of work on learning basis functions automatically (Patrascu et al., 2002; Mahadevan, 2005; Kveton & Hauskrecht, 2006a; Mahadevan & Maggioni, 2006; Mahadevan et al., 2006).

Example 2 *To demonstrate the concept of the linear value function model, we consider the network administration problem (Example 1) and assume a low chance of a single computer failing. Then the value function in Figure 1c is sufficient to derive a close-to-optimal policy on the 4-ring topology (Figure 1a) because the indicator functions $f_i(\mathbf{x}) = x_i$ capture changes in the states of individual computers. For instance, if the computer X_i fails, the linear policy:*

$$u(\mathbf{x}) = \arg\max_a \left[R(\mathbf{x}, a) + \gamma \sum_{\mathbf{x}'} P(\mathbf{x}' \mid \mathbf{x}, a) V^{\mathbf{w}}(\mathbf{x}') \right]$$

immediately leads to rebooting it. If the failure has already propagated to the computer X_{i+1}, the policy recovers it in the next step. This procedure is repeated until the spread of the initial failure is stopped.

3.3 Approximate Linear Programming

Various methods for fitting of the linear value function approximation have been proposed and analyzed (Bertsekas & Tsitsiklis, 1996). We focus on *approximate linear programming (ALP)* (Schweitzer & Seidmann, 1985), which recasts this problem as a linear program:

$$\begin{aligned} \text{minimize}_{\mathbf{w}} \quad & \sum_{\mathbf{x}} \psi(\mathbf{x}) \sum_i w_i f_i(\mathbf{x}) \qquad (6) \\ \text{subject to:} \quad & \sum_i w_i f_i(\mathbf{x}) \geq R(\mathbf{x}, a) + \gamma \sum_{\mathbf{x}'} P(\mathbf{x}' \mid \mathbf{x}, a) \sum_i w_i f_i(\mathbf{x}') \quad \forall\, \mathbf{x} \in \mathbf{X}, a \in \mathcal{A}; \end{aligned}$$

where $\mathbf{w}$ represents the variables in the LP, $\psi(\mathbf{x}) \geq 0$ are *state relevance weights* weighting the quality of the approximation, and $\gamma \sum_{\mathbf{x}'} P(\mathbf{x}' \mid \mathbf{x}, a) \sum_i w_i f_i(\mathbf{x}')$ is a discounted *backprojection* of the value function $V^{\mathbf{w}}$ (Equation 5). The ALP formulation can be easily derived from the standard LP formulation (3) by substituting $V^{\mathbf{w}}(\mathbf{x})$ for $V(\mathbf{x})$. The formulation is feasible if the set of basis functions contains a constant function $f_0(\mathbf{x}) \equiv 1$. We assume that such a basis function is always present. Note that the state relevance weights are no longer enforced to be strictly positive (Section 1). Comparing to the standard LP formulation (3), which is solved by the optimal value function V^* for arbitrary weights $\psi(s) > 0$, a solution $\widetilde{\mathbf{w}}$ to the ALP formulation depends on the weights $\psi(\mathbf{x})$. Intuitively, the higher the weights, the higher the quality of the approximation $V^{\widetilde{\mathbf{w}}}$ in a corresponding state.

Since our basis functions are usually restricted to subsets of state variables (Section 3.2), summation terms in the ALP formulation can be computed efficiently (Guestrin et al., 2001; Schuurmans & Patrascu, 2002). For example, the order of summation in the backprojection term can be rearranged as $\gamma \sum_i w_i \sum_{\mathbf{x}'_i} P(\mathbf{x}'_i \mid \mathbf{x}, a) f_i(\mathbf{x}'_i)$, which allows its aggregation in the space of $\mathbf{X}_i$ instead of $\mathbf{X}$. Similarly, a factored form of $\psi(\mathbf{x})$ yields an efficiently computable objective function (Guestrin, 2003).

The number of constraints in the ALP formulation is exponential in the number of state variables. Fortunately, the constraints are structured. This results from combining factored transition and reward models (Section 3.1) with the linear approximation (Equation 5). As a consequence, the constraints can be satisfied without enumerating them exhaustively.

Example 3 *The notion of a* factored constraint space *is important for compact satisfaction of exponentially many constraints. To illustrate this concept, let us consider the linear value function (Example 2) on the 4-ring network administration problem (Example 1). Intuitively, by combining the graphical representations of $P(\mathbf{x}' \mid \mathbf{x}, a_1)$, $R(\mathbf{x}, a_1)$ (Figure 1b), and $V^{\mathbf{w}}(\mathbf{x})$ (Figure 1c), we obtain a factored model of constraint violations:*

$$
\begin{aligned}
\tau^{\mathbf{w}}(\mathbf{x}, a_1) &= V^{\mathbf{w}}(\mathbf{x}) - \gamma \sum_{\mathbf{x}'} P(\mathbf{x}' \mid \mathbf{x}, a_1) V^{\mathbf{w}}(\mathbf{x}') - R(\mathbf{x}, a_1) \\
&= \sum_i w_i f_i(\mathbf{x}) - \gamma \sum_i w_i \sum_{\mathbf{x}'_i} P(\mathbf{x}'_i \mid \mathbf{x}, a_1) f_i(\mathbf{x}'_i) - R(\mathbf{x}, a_1) \\
&= w_0 + \sum_{i=1}^{4} w_i x_i - \gamma w_0 - \gamma w_1 P(x'_1 = 1 \mid a_1) - \\
&\quad\; \gamma \sum_{i=2}^{4} w_i P(x'_i = 1 \mid x_i, x_{i-1}, a_1) - 2x_1 - \sum_{j=2}^{4} x_j.
\end{aligned}
$$

for an arbitrary solution $\mathbf{w}$ (Figure 2a). Note that this cost function:

$$
\tau^{\mathbf{w}}(\mathbf{x}, a_1) = \phi^{\mathbf{w}} + \sum_{i=1}^{4} \phi^{\mathbf{w}}(x_i) + \sum_{i=2}^{4} \phi^{\mathbf{w}}(x_i, x_{i-1})
$$

is a linear combination of a constant $\phi^{\mathbf{w}}$ in $\mathbf{x}$, and univariate and bivariate functions $\phi^{\mathbf{w}}(x_i)$ and $\phi^{\mathbf{w}}(x_i, x_{i-1})$. It can be represented compactly by a cost network *(Guestrin et al., 2001), which is an undirected graph over a set of variables $\mathbf{X}$. Two nodes in the graph are connected*

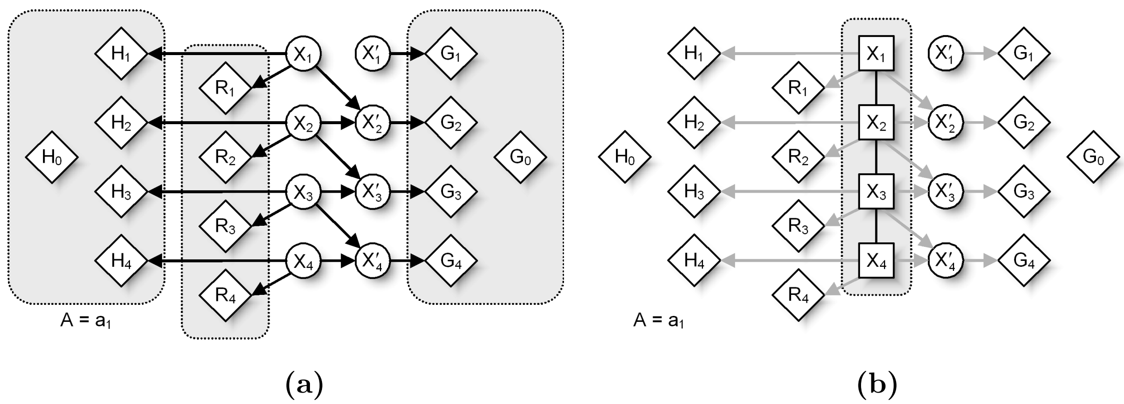


Figure 2: **a.** A graphical representation of combining factored transition and reward models (Figure 1b) with the linear approximation (Figure 1c). Reward nodes G_0 and G_i ($i \geq 1$) represent the discounted backprojection terms $-\gamma w_0$ and $-\gamma w_i x_i'$ ($i \geq 1$). Gray regions are the cost components of the constraint space. **b.** A cost network corresponding to our factored constraint space (Figure 2a). The network captures pairwise dependencies $X_1 - X_2$, $X_2 - X_3$, and $X_3 - X_4$. The treewidth of the cost network is 1.

if any of the cost terms depends on both variables. Therefore, the cost network corresponding to the function $\tau^{\mathbf{w}}(\mathbf{x}, a_1)$ *must contain edges* $X_1 - X_2$*,* $X_2 - X_3$*, and* $X_3 - X_4$ *(Figure 2b).*

Savings achieved by the compact representation of constraints are related to the efficiency of computing $\arg\min_{\mathbf{x}} \tau^{\mathbf{w}}(\mathbf{x}, a_1)$ *(Guestrin, 2003). This computation can be done by variable elimination and its complexity increases exponentially in the width of the* tree decomposition *of the cost network. The smallest width of all tree decompositions is referred to as* treewidth.

Inspired by the factorization, Guestrin et al. (2001) proposed a variable-elimination method (Dechter, 1996) that rewrites the constraint space in ALP compactly. Schuurmans and Patrascu (2002) solved the same problem by the cutting plane method. The method iteratively searches for the most violated constraint:

$$\arg\min_{\mathbf{x},a} \left[\sum_i w_i^{(t)} \left[f_i(\mathbf{x}_i) - \gamma \sum_{\mathbf{x}_i'} P(\mathbf{x}_i' \mid \mathbf{x}, a) f_i(\mathbf{x}_i') \right] - R(\mathbf{x}, a) \right] \tag{7}$$

with respect to the solution $\mathbf{w}^{(t)}$ of a relaxed ALP. The constraint is added to the LP, which is resolved for a new solution $\mathbf{w}^{(t+1)}$. This procedure is iterated until no violated constraint is found, so that $\mathbf{w}^{(t)}$ is an optimal solution to the ALP.

The quality of the ALP formulation has been studied by de Farias and Van Roy (2003). Based on their work, we conclude that ALP yields a close approximation $V^{\widetilde{\mathbf{w}}}$ to the optimal value function V^* if the weighted max-norm error $\|V^* - V^{\mathbf{w}}\|_{\infty,1/L}$ can be minimized. We return to this theoretical result in Section 5.1.

Theorem 1 (de Farias & Van Roy, 2003) *Let $\widetilde{\mathbf{w}}$ be a solution to the ALP formulation (6). Then the expected error of the value function $V^{\widetilde{\mathbf{w}}}$ can be bounded as:*

$$\left\| V^* - V^{\widetilde{\mathbf{w}}} \right\|_{1,\psi} \leq \frac{2\psi^\intercal L}{1-\kappa} \min_{\mathbf{w}} \left\| V^* - V^{\mathbf{w}} \right\|_{\infty,1/L},$$

where $\|\cdot\|_{1,\psi}$ is an $\mathcal{L}_1$-norm weighted by the state relevance weights ψ, $L(\mathbf{x}) = \sum_i w_i^L f_i(\mathbf{x})$ is a Lyapunov function *such that the inequality $\kappa L(\mathbf{x}) \geq \gamma \sup_{\mathbf{a}} \mathrm{E}_{P(\mathbf{x}' \mid \mathbf{x}, \mathbf{a})} [L(\mathbf{x}')]$ holds, $\kappa \in [0, 1)$ denotes its* contraction factor*, and $\|\cdot\|_{\infty,1/L}$ is a max-norm reweighted by the reciprocal $1/L$.*

Note that the $\mathcal{L}_1$-norm distance $\left\| V^* - V^{\widetilde{\mathbf{w}}} \right\|_{1,\psi}$ equals to the expectation $\mathrm{E}_\psi \left| V^* - V^{\widetilde{\mathbf{w}}} \right|$ over the state space $\mathbf{X}$ with respect to the state relevance weights ψ. Similarly to Theorem 1, we utilize the $\mathcal{L}_1$ and $\mathcal{L}_\infty$ norms in the rest of the work to measure the expected and worst-case errors of value functions. These norms are defined as follows.

Definition 1 *The $\mathcal{L}_1$* (Manhattan) *and $\mathcal{L}_\infty$* (infinity) norms *are typically defined as $\|f\|_1 = \sum_{\mathbf{x}} |f(\mathbf{x})|$ and $\|f\|_\infty = \max_{\mathbf{x}} |f(\mathbf{x})|$. If the state space $\mathbf{X}$ is represented by both discrete and continuous variables $\mathbf{X}_D$ and $\mathbf{X}_C$, the definition of the norms changes accordingly:*

$$\|f\|_1 = \sum_{\mathbf{x}_D} \int_{\mathbf{x}_C} |f(\mathbf{x})| \, \mathrm{d}\mathbf{x}_C \quad \text{and} \quad \|f\|_\infty = \sup_{\mathbf{x}} |f(\mathbf{x})|. \tag{8}$$

The following definitions:

$$\|f\|_{1,\psi} = \sum_{\mathbf{x}_D} \int_{\mathbf{x}_C} \psi(\mathbf{x}) \, |f(\mathbf{x})| \, \mathrm{d}\mathbf{x}_C \quad \text{and} \quad \|f\|_{\infty,\psi} = \sup_{\mathbf{x}} \psi(\mathbf{x}) \, |f(\mathbf{x})| \tag{9}$$

correspond to the $\mathcal{L}_1$ and $\mathcal{L}_\infty$ norms reweighted by a function $\psi(\mathbf{x})$.

4. Hybrid Factored MDPs

Discrete-state factored MDPs (Section 3) permit a compact representation of decision problems with discrete states. However, real-world domains often involve continuous quantities, such as temperature and pressure. A sufficient discretization of these quantities may require hundreds of points in a single dimension, which renders the representation of our transition model (Equation 4) infeasible. In addition, rough and uninformative discretization impacts the quality of policies. Therefore, we want to avoid discretization or defer it until necessary. As a step in this direction, we discuss a formalism for representing hybrid decision problems in the domains of discrete and continuous variables.

4.1 Factored Transition and Reward Models

A *hybrid factored MDP (HMDP)* is a 4-tuple $\mathcal{M} = (\mathbf{X}, \mathbf{A}, P, R)$, where $\mathbf{X} = \{X_1, \dots, X_n\}$ is a state space described by state variables, $\mathbf{A} = \{A_1, \dots, A_m\}$ is an action space described by action variables, $P(\mathbf{X}' \mid \mathbf{X}, \mathbf{A})$ is a stochastic transition model of state dynamics conditioned on the preceding state and action, and R is a reward function assigning immediate payoffs to state-action configurations.[2]

2. *General state and action space MDP* is an alternative term for a hybrid MDP. The term *hybrid* does not refer to the dynamics of the model, which is discrete-time.

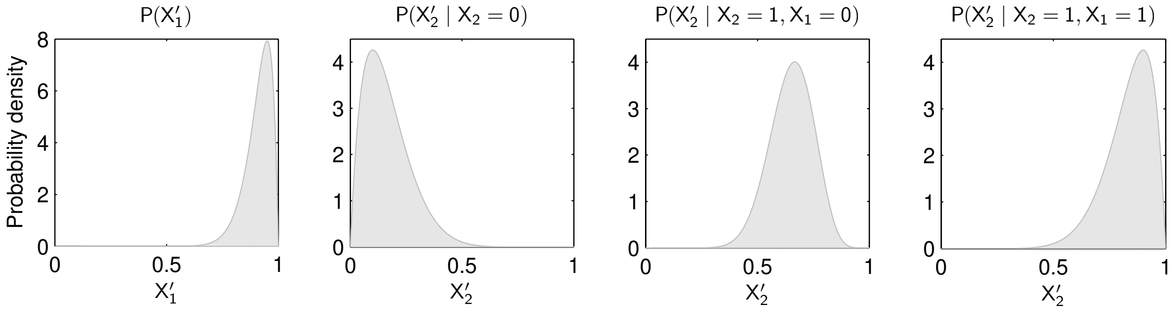


Figure 3: Transition functions for continuous variables X_1' and X_2' after taking an action a_1 in the 4-ring topology (Example 4). The densities are shown for extreme values of their parent variables X_1 and X_2.

State variables: State variables are either discrete or continuous. Every discrete variable X_i takes on values from a finite domain $\mathrm{Dom}(X_i)$. Following Hauskrecht and Kveton (2004), we assume that every continuous variable is bounded to the $[0, 1]$ subspace. In general, this assumption is very mild and permits modeling of any closed interval on $\mathbb{R}$. The state of the system is completely observed and described by a vector of value assignments $\mathbf{x} = (\mathbf{x}_D, \mathbf{x}_C)$ which partitions along its discrete and continuous components $\mathbf{x}_D$ and $\mathbf{x}_C$.

Action variables: The action space is distributed and represented by action variables $\mathbf{A}$. The composite action is defined by a vector of individual action choices $\mathbf{a} = (\mathbf{a}_D, \mathbf{a}_C)$ which partitions along its discrete and continuous components $\mathbf{a}_D$ and $\mathbf{a}_C$.

Transition model: The transition model is given by the conditional probability distribution $P(\mathbf{X}' \mid \mathbf{X}, \mathbf{A})$, where $\mathbf{X}$ and $\mathbf{X}'$ denote the state variables at two successive time steps. We assume that this distribution factors along $\mathbf{X}'$ as $P(\mathbf{X}' \mid \mathbf{X}, \mathbf{A}) = \prod_{i=1}^{n} P(X_i' \mid \mathsf{Par}(X_i'))$ and can be described compactly by a DBN (Dean & Kanazawa, 1989). Typically, the parent set $\mathsf{Par}(X_i') \subseteq \mathbf{X} \cup \mathbf{A}$ is a small subset of state and action variables which allows for a local parameterization of the transition model.

Parameterization of our transition model: One-step dynamics of every state variable is described by its conditional probability distribution $P(X_i' \mid \mathsf{Par}(X_i'))$. If X_i' is a continuous variable, its transition function is represented by a mixture of beta distributions (Hauskrecht & Kveton, 2004):

$$\begin{aligned} P(X_i' = x \mid \mathsf{Par}(X_i')) &= \sum_j \pi_{ij} P_{\mathrm{beta}}(x \mid \alpha_j, \beta_j) \\ P_{\mathrm{beta}}(x \mid \alpha, \beta) &= \frac{\Gamma(\alpha + \beta)}{\Gamma(\alpha)\Gamma(\beta)} x^{\alpha - 1}(1 - x)^{\beta - 1}, \end{aligned} \tag{10}$$

where π_{ij} is the weight assigned to the j-th component of the mixture, and $\alpha_j = \phi_{ij}^{\alpha}(\mathsf{Par}(X_i'))$ and $\beta_j = \phi_{ij}^{\beta}(\mathsf{Par}(X_i'))$ are arbitrary positive functions of the parent set. The mixture of beta distributions provides a very general class of transition functions and yet allows closed-form solutions[3] to the expectation terms in HALP (Section 5). If every $\beta_j = 1$, Equation 10 turns into a polynomial in X_i'. Due to the Weierstrass approximation theorem (Jeffreys & Jeffreys, 1988), such a polynomial is sufficient to approximate any continuous transition density over X_i' with any precision. If X_i' is a discrete variable, its transition model is parameterized by $|\mathrm{Dom}(X_i')|$ nonnegative discriminant functions $\theta_j = \phi_{ij}^{\theta}(\mathsf{Par}(X_i'))$ (Guestrin et al., 2004):

$$P(X_i' = j \mid \mathsf{Par}(X_i')) = \frac{\theta_j}{\sum_{j=1}^{|\mathrm{Dom}(X_i')|} \theta_j}. \tag{11}$$

Note that the parameters α_j, β_j, and θ_j (Equations 10 and 11) are functions instantiated by value assignments to the variables $\mathsf{Par}(X_i') \subseteq \mathbf{X} \cup \mathbf{A}$. We keep separate parameters for every state variable X_i' although our indexing does not reflect this explicitly. The only restriction on the functions is that they return valid parameters for all state-action pairs $(\mathbf{x}, \mathbf{a})$. Hence, we assume that $\alpha_j(\mathbf{x}, \mathbf{a}) \geq 0$, $\beta_j(\mathbf{x}, \mathbf{a}) \geq 0$, $\theta_j(\mathbf{x}, \mathbf{a}) \geq 0$, and $\sum_{j=1}^{|\mathrm{Dom}(X_i')|} \theta_j(\mathbf{x}, \mathbf{a}) > 0$.

Reward model: The reward model factors similarly to the transition model. In particular, the reward function $R(\mathbf{x}, \mathbf{a}) = \sum_j R_j(\mathbf{x}_j, \mathbf{a}_j)$ is an additive function of local reward functions defined on the subsets $\mathbf{X}_j$ and $\mathbf{A}_j$. In graphical models, the local functions can be described compactly by reward nodes R_j, which are conditioned on their parent sets $\mathsf{Par}(R_j) = \mathbf{X}_j \cup \mathbf{A}_j$. To allow this representation, we formally extend our DBN to an influence diagram (Howard & Matheson, 1984). Note that the form of the reward functions $R_j(\mathbf{x}_j, \mathbf{a}_j)$ is not restricted.

Optimal value function and policy: The *optimal policy* π^* can be defined greedily with respect to the *optimal value function* V^*, which is a fixed point of the Bellman equation:

$$\begin{aligned} V^*(\mathbf{x}) &= \sup_{\mathbf{a}} \left[R(\mathbf{x}, \mathbf{a}) + \gamma \mathrm{E}_{P(\mathbf{x}' \mid \mathbf{x}, \mathbf{a})} \left[V^*(\mathbf{x}') \right] \right] \\ &= \sup_{\mathbf{a}} \left[R(\mathbf{x}, \mathbf{a}) + \gamma \sum_{\mathbf{x}_D'} \int_{\mathbf{x}_C'} P(\mathbf{x}' \mid \mathbf{x}, \mathbf{a}) V^*(\mathbf{x}') \, \mathrm{d}\mathbf{x}_C' \right]. \end{aligned} \tag{12}$$

Accordingly, the *hybrid Bellman operator* $\mathcal{T}^*$ is given by:

$$\mathcal{T}^* V(\mathbf{x}) = \sup_{\mathbf{a}} \left[R(\mathbf{x}, \mathbf{a}) + \gamma \mathrm{E}_{P(\mathbf{x}' \mid \mathbf{x}, \mathbf{a})} \left[V(\mathbf{x}') \right] \right]. \tag{13}$$

In the rest of the paper, we denote expectation terms over discrete and continuous variables in a unified form:

$$\mathrm{E}_{P(\mathbf{x})}[f(\mathbf{x})] = \sum_{\mathbf{x}_D} \int_{\mathbf{x}_C} P(\mathbf{x}) f(\mathbf{x}) \, \mathrm{d}\mathbf{x}_C. \tag{14}$$

Example 4 (Hauskrecht & Kveton, 2004) *Continuous-state network administration is a variation on Example 1, where the computer states are represented by continuous variables on the interval between 0 (being down) and 1 (running). At each time step, the administrator*

3. The term *closed-form* refers to a generally accepted set of closed-form operations and functions extended by the gamma and incomplete beta functions.

selects a single action from the set $\mathcal{A} = \{a_1, \dots, a_{n+1}\}$*. The action* a_i $(i \leq n)$ *corresponds to rebooting the i-th computer. The last action* a_{n+1} *is dummy. The transition model captures the propagation of failures in the network and is encoded locally by beta distributions:*

$$P(X_i' = x \mid \mathsf{Par}(X_i')) = P_{\text{beta}}(x \mid \alpha, \beta) \quad \left| \quad \begin{array}{ll} \alpha = 20 & a = i \\ \beta = 2 & \\ \alpha = 2 + 13x_i - 5x_i \mathrm{E}[\mathsf{Par}(X_i')] & a \neq i \\ \beta = 10 - 2x_i - 6x_i \mathrm{E}[\mathsf{Par}(X_i')] & \end{array} \right.$$

where the variables x_i *and* $\mathrm{E}[\mathsf{Par}(X_i')]$ *denote the state of the i-th computer and the expected state of its parents. Note that this transition function is similar to Example 1. For instance, in the 4-ring topology, the modes of transition densities for continuous variables* X_1' *and* X_2' *after taking an action* a_1 *(Figure 3):*

$$\begin{array}{rr} \widehat{P}(X_1' \mid a = a_1) = 0.95 & \widehat{P}(X_2' \mid X_2 = 1, X_1 = 0, a = a_1) \approx 0.67 \\ \widehat{P}(X_2' \mid X_2 = 0, a = a_1) = 0.10 & \widehat{P}(X_2' \mid X_2 = 1, X_1 = 1, a = a_1) = 0.90 \end{array}$$

equal to the expected values of their discrete counterparts (Figure 1b). The reward function is additive:

$$R(\mathbf{x}, a) = 2x_1^2 + \sum_{j=2}^{n} x_j^2$$

and establishes our preferences for maintaining the server X_1 *and workstations* $X_2, \dots, X_n$*.*

4.2 Solving Hybrid Factored MDPs

Value iteration, policy iteration, and linear programming are the most fundamental dynamic programming methods for solving MDPs (Puterman, 1994; Bertsekas & Tsitsiklis, 1996). Unfortunately, none of these techniques is suitable for solving hybrid factored MDPs. First, their complexity is exponential in the number of state variables if the variables are discrete. Second, the methods assume a finite support for the optimal value function or policy, which may not exist if continuous variables are present. Therefore, any feasible approach to solving arbitrary HMDPs is likely to be approximate. In the rest of the section, we review two major classes of methods for approximating value functions in hybrid domains.

Grid-based approximation: Grid-based methods (Chow & Tsitsiklis, 1991; Rust, 1997) transform the initial state space $\mathbf{X}$ into a set of grid points $G = \left\{\mathbf{x}^{(1)}, \dots, \mathbf{x}^{(N)}\right\}$. The points are used to estimate the optimal value function V_G^* on the grid, which in turn approximates V^*. The Bellman operator on the grid is defined as (Rust, 1997):

$$\mathcal{T}_G^* V(\mathbf{x}^{(i)}) = \max_{\mathbf{a}} \left[R(\mathbf{x}^{(i)}, \mathbf{a}) + \gamma \sum_{j=1}^{N} P_G(\mathbf{x}^{(j)} \mid \mathbf{x}^{(i)}, \mathbf{a}) V(\mathbf{x}^{(j)}) \right], \tag{15}$$

where $P_G(\mathbf{x}^{(j)} \mid \mathbf{x}^{(i)}, \mathbf{a}) = \Psi_{\mathbf{a}}^{-1}(\mathbf{x}^{(i)}) P(\mathbf{x}^{(j)} \mid \mathbf{x}^{(i)}, \mathbf{a})$ is a transition function, which is normalized by the term $\Psi_{\mathbf{a}}(\mathbf{x}^{(i)}) = \sum_{j=1}^{N} P(\mathbf{x}^{(j)} \mid \mathbf{x}^{(i)}, a)$. The operator $\mathcal{T}_G^*$ allows the computation of the value function V_G^* by standard techniques for solving discrete-state MDPs.

Inputs:

a hybrid factored MDP $\mathcal{M} = (\mathbf{X}, \mathbf{A}, P, R)$
basis functions $f_0(\mathbf{x}), f_1(\mathbf{x}), f_2(\mathbf{x}), \dots$
initial basis function weights $\mathbf{w}^{(0)}$
a set of states $G = \left\{\mathbf{x}^{(1)}, \dots, \mathbf{x}^{(N)}\right\}$

Algorithm:

$t = 0$
while a stopping criterion is not met
  for every state $\mathbf{x}^{(j)}$
    for every basis function $f_i(\mathbf{x})$
      $\mathbf{X}_{ji} = f_i(\mathbf{x}^{(j)})$
    $\mathbf{y}_j = \max_{\mathbf{a}} \left[R(\mathbf{x}^{(j)}, \mathbf{a}) + \gamma \mathrm{E}_{P(\mathbf{x}' \mid \mathbf{x}^{(j)}, \mathbf{a})}\left[V^{\mathbf{w}^{(t)}}(\mathbf{x}')\right]\right]$
  $\mathbf{w}^{(t+1)} = (\mathbf{X}^{\intercal}\mathbf{X})^{-1}\mathbf{X}^{\intercal}\mathbf{y}$
  $t = t + 1$

Outputs:

basis function weights $\mathbf{w}^{(t)}$

Figure 4: Pseudo-code implementation of the least-squares value iteration ($\mathcal{L}_2$ VI) with the linear value function approximation (Equation 5). The stopping criterion is often based on the number of steps or the $\mathcal{L}_2$-norm error $\left\|V^{\mathbf{w}^{(t)}} - \mathcal{T}^* V^{\mathbf{w}^{(t)}}\right\|_2$ measured on the set G. Our discussion in Sections 5.2 and 6 provides a recipe for an efficient implementation of the backup operation $\mathcal{T}^* V^{\mathbf{w}^{(t)}}(\mathbf{x}^{(j)})$.

Rust (1997) analyzed the convergence of these methods for random and pseudo-random samples. Clearly, a uniform discretization of increasing precision guarantees the convergence of V_G^* to V^* but causes an exponential blowup in the state space (Chow & Tsitsiklis, 1991). To overcome this concern, Munos and Moore (2002) proposed an adaptive algorithm for non-uniform discretization based on the Kuhn triangulation. Ferns et al. (2005) analyzed metrics for aggregating states in continuous-state MDPs based on the notion of bisimulation. Trick and Zin (1993) used linear programming to solve low-dimensional problems with continuous variables. These continuous variables were discretized manually.

Parametric value function approximation: An alternative approach to solving factored MDPs with continuous-state components is the approximation of the optimal value function V^* by some parameterized model V^λ (Bertsekas & Tsitsiklis, 1996; Van Roy, 1998; Gordon, 1999). The parameters λ are typically optimized iteratively by applying the backup operator $\mathcal{T}^*$ to a finite set of states. The least-squares error $\left\|V^\lambda - \mathcal{T}^* V^\lambda\right\|_2$ is a commonly minimized error metric (Figure 4). Online updating by gradient methods (Bertsekas & Tsitsiklis, 1996; Sutton & Barto, 1998) is another way of optimizing value functions. The limitation of these techniques is that their solutions are often unstable and may diverge (Bertsekas, 1995). On the other hand, they generate high-quality approximations.

Parametric approximations often assume fixed value function models. However, in some cases, it is possible to derive flexible forms of V^λ that combine well with the backup operator $\mathcal{T}^*$. For instance, Sondik (1971) showed that convex piecewise linear functions are sufficient to represent value functions and their DP backups in partially-observable MDPs (POMDPs) (Astrom, 1965; Hauskrecht, 2000). Based on this idea, Feng et al. (2004) proposed a method for solving MDPs with continuous variables. To obtain full DP backups, the value function approximation is restricted to *rectangular piecewise linear and convex (RPWLC)* functions. Further restrictions are placed on the transition and reward models of MDPs. The advantage of the approach is its adaptivity. The major disadvantages are restrictions on solved MDPs and the complexity of RPWLC value functions, which may grow exponentially in the number of backups. As a result, without further modifications, this approach is less likely to succeed in solving high-dimensional and distributed decision problems.

5. Hybrid Approximate Linear Programming

To overcome the limitations of existing methods for solving HMDPs (Section 4.2), we extend the discrete-state ALP (Section 3.3) to hybrid state and action spaces. We refer to this novel framework as *hybrid approximate linear programming (HALP)*.

Similarly to the discrete-state ALP, HALP optimizes the linear value function approximation (Equation 5). Therefore, it transforms an initially intractable problem of computing V^* in the hybrid state space $\mathbf{X}$ into a lower dimensional space of $\mathbf{w}$. The HALP formulation is given by a linear program[4]:

$$\begin{aligned} \text{minimize}_{\mathbf{w}} \quad & \sum_i w_i \alpha_i \\ \text{subject to:} \quad & \sum_i w_i F_i(\mathbf{x}, \mathbf{a}) - R(\mathbf{x}, \mathbf{a}) \geq 0 \quad \forall\, \mathbf{x} \in \mathbf{X}, \mathbf{a} \in \mathbf{A}; \end{aligned} \tag{16}$$

where $\mathbf{w}$ represents the variables in the LP, α_i denotes *basis function relevance weight*:

$$\begin{aligned} \alpha_i &= \mathrm{E}_{\psi(\mathbf{x})}[f_i(\mathbf{x})] \\ &= \sum_{\mathbf{x}_D} \int_{\mathbf{x}_C} \psi(\mathbf{x}) f_i(\mathbf{x}) \, \mathrm{d}\mathbf{x}_C, \end{aligned} \tag{17}$$

$\psi(\mathbf{x}) \geq 0$ is a *state relevance density function* that weights the quality of the approximation, and $F_i(\mathbf{x}, \mathbf{a}) = f_i(\mathbf{x}) - \gamma g_i(\mathbf{x}, \mathbf{a})$ denotes the difference between the basis function $f_i(\mathbf{x})$ and its discounted *backprojection*:

$$\begin{aligned} g_i(\mathbf{x}, \mathbf{a}) &= \mathrm{E}_{P(\mathbf{x}' \mid \mathbf{x}, \mathbf{a})}\left[f_i(\mathbf{x}')\right] \\ &= \sum_{\mathbf{x}'_D} \int_{\mathbf{x}'_C} P(\mathbf{x}' \mid \mathbf{x}, \mathbf{a}) f_i(\mathbf{x}') \, \mathrm{d}\mathbf{x}'_C. \end{aligned} \tag{18}$$

4. More precisely, the HALP formulation (16) is a *linear semi-infinite optimization* problem with an infinite number of constraints. The number of basis functions is finite. For brevity, we refer to this optimization problem as linear programming.

Vectors $\mathbf{x}_D$ ($\mathbf{x}'_D$) and $\mathbf{x}_C$ ($\mathbf{x}'_C$) are the discrete and continuous components of value assignments $\mathbf{x}$ ($\mathbf{x}'$) to all state variables $\mathbf{X}$ ($\mathbf{X}'$). The linear program can be rewritten compactly:

$$\begin{aligned} \text{minimize}_{\mathbf{w}} \quad & \mathrm{E}_\psi[V^{\mathbf{w}}] \\ \text{subject to:} \quad & V^{\mathbf{w}} - \mathcal{T}^* V^{\mathbf{w}} \geq 0 \end{aligned} \tag{19}$$

by using the Bellman operator $\mathcal{T}^*$.

The HALP formulation reduces to the discrete-state ALP (Section 3.3) if the state and action variables are discrete, and to the continuous-state ALP (Hauskrecht & Kveton, 2004) if the state variables are continuous. The formulation is feasible if the set of basis functions contains a constant function $f_0(\mathbf{x}) \equiv 1$. We assume that such a basis function is present.

In the rest of the paper, we address several concerns related to the HALP formulation. First, we analyze the quality of this approximation and relate it to the minimization of the max-norm error $\|V^* - V^{\mathbf{w}}\|_\infty$, which is a commonly-used metric (Section 5.1). Second, we present rich classes of basis functions that lead to closed-form solutions to the expectation terms in the objective function and constraints (Equations 17 and 18). These terms involve sums and integrals over the complete state space $\mathbf{X}$ (Section 5.2), and therefore are hard to evaluate. Finally, we discuss approximations to the constraint space in HALP and introduce a framework for solving HALP formulations in a unified way (Section 6). Note that complete satisfaction of this constraint space may not be possible since every state-action pair $(\mathbf{x}, \mathbf{a})$ induces a constraint.

5.1 Error Bounds

The quality of the ALP approximation (Section 3.3) has been studied by de Farias and Van Roy (2003). We follow up on their work and extend it to structured state and action spaces with continuous variables. Before we proceed, we demonstrate that a solution to the HALP formulation (16) constitutes an upper bound on the optimal value function V^*.

Proposition 1 *Let $\widetilde{\mathbf{w}}$ be a solution to the HALP formulation (16). Then $V^{\widetilde{\mathbf{w}}} \geq V^*$.*

This result allows us to restate the objective $\mathrm{E}_\psi[V^{\mathbf{w}}]$ in HALP.

Proposition 2 *Vector $\widetilde{\mathbf{w}}$ is a solution to the HALP formulation (16):*

$$\begin{aligned} \text{minimize}_{\mathbf{w}} \quad & \mathrm{E}_\psi[V^{\mathbf{w}}] \\ \text{subject to:} \quad & V^{\mathbf{w}} - \mathcal{T}^* V^{\mathbf{w}} \geq 0 \end{aligned}$$

if and only if it solves:

$$\begin{aligned} \text{minimize}_{\mathbf{w}} \quad & \|V^* - V^{\mathbf{w}}\|_{1,\psi} \\ \text{subject to:} \quad & V^{\mathbf{w}} - \mathcal{T}^* V^{\mathbf{w}} \geq 0; \end{aligned}$$

where $\|\cdot\|_{1,\psi}$ is an $\mathcal{L}_1$-norm weighted by the state relevance density function ψ and $\mathcal{T}^*$ is the hybrid Bellman operator.

Based on Proposition 2, we conclude that HALP optimizes the linear value function approximation with respect to the reweighted $\mathcal{L}_1$-norm error $\|V^* - V^{\mathbf{w}}\|_{1,\psi}$. The following theorem draws a parallel between minimizing this objective and max-norm error $\|V^* - V^{\mathbf{w}}\|_{\infty}$. More precisely, the theorem says that HALP yields a close approximation $V^{\widetilde{\mathbf{w}}}$ to the optimal value function V^* if V^* is close to the span of basis functions $f_i(\mathbf{x})$.

Theorem 2 *Let $\widetilde{\mathbf{w}}$ be an optimal solution to the HALP formulation (16). Then the expected error of the value function $V^{\widetilde{\mathbf{w}}}$ can be bounded as:*

$$\left\|V^* - V^{\widetilde{\mathbf{w}}}\right\|_{1,\psi} \leq \frac{2}{1-\gamma} \min_{\mathbf{w}} \|V^* - V^{\mathbf{w}}\|_{\infty},$$

where $\|\cdot\|_{1,\psi}$ is an $\mathcal{L}_1$-norm weighted by the state relevance density function ψ and $\|\cdot\|_{\infty}$ is a max-norm.

Unfortunately, Theorem 2 rarely yields a tight bound on $\left\|V^* - V^{\widetilde{\mathbf{w}}}\right\|_{1,\psi}$. First, it is hard to guarantee a uniformly low max-norm error $\|V^* - V^{\mathbf{w}}\|_{\infty}$ if the dimensionality of a problem grows but the basis functions $f_i(\mathbf{x})$ are local. Second, the bound ignores the state relevance density function $\psi(\mathbf{x})$ although this one impacts the quality of HALP solutions. To address these concerns, we introduce non-uniform weighting of the max-norm error in Theorem 3.

Theorem 3 *Let $\widetilde{\mathbf{w}}$ be an optimal solution to the HALP formulation (16). Then the expected error of the value function $V^{\widetilde{\mathbf{w}}}$ can be bounded as:*

$$\left\|V^* - V^{\widetilde{\mathbf{w}}}\right\|_{1,\psi} \leq \frac{2\mathrm{E}_{\psi}[L]}{1-\kappa} \min_{\mathbf{w}} \|V^* - V^{\mathbf{w}}\|_{\infty,1/L},$$

where $\|\cdot\|_{1,\psi}$ is an $\mathcal{L}_1$-norm weighted by the state relevance density ψ, $L(\mathbf{x}) = \sum_i w_i^L f_i(\mathbf{x})$ is a Lyapunov function *such that the inequality $\kappa L(\mathbf{x}) \geq \gamma \sup_{\mathbf{a}} \mathrm{E}_{P(\mathbf{x}' \mid \mathbf{x}, \mathbf{a})}[L(\mathbf{x}')]$ holds, $\kappa \in [0, 1)$ denotes its* contraction factor, *and $\|\cdot\|_{\infty,1/L}$ is a max-norm reweighted by the reciprocal $1/L$.*

Note that Theorem 2 is a special form of Theorem 3 when $L(\mathbf{x}) \equiv 1$ and $\kappa = \gamma$. Therefore, the Lyapunov function $L(\mathbf{x})$ permits at least as good bounds as Theorem 2. To make these bounds tight, the function $L(\mathbf{x})$ should return large values in the regions of the state space, which are unimportant for modeling. In turn, the reciprocal $1/L(\mathbf{x})$ is close to zero in these undesirable regions, which makes their impact on the max-norm error $\|V^* - V^{\mathbf{w}}\|_{\infty,1/L}$ less likely. Since the state relevance density function $\psi(\mathbf{x})$ reflects the importance of states, the term $\mathrm{E}_{\psi}[L]$ should remain small. These two factors contribute to tighter bounds than those by Theorem 2.

Since the Lyapunov function $L(\mathbf{x}) = \sum_i w_i^L f_i(\mathbf{x})$ lies in the span of basis functions $f_i(\mathbf{x})$, Theorem 3 provides a recipe for achieving high-quality approximations. Intuitively, a good set of basis functions always involves two types of functions. The first type guarantees small errors $|V^*(\mathbf{x}) - V^{\mathbf{w}}(\mathbf{x})|$ in the important regions of the state space, where the state relevance density $\psi(\mathbf{x})$ is high. The second type returns high values where the state relevance density $\psi(\mathbf{x})$ is low, and vice versa. The latter functions allow the satisfaction of the constraint space $V^{\mathbf{w}} \geq \mathcal{T}^* V^{\mathbf{w}}$ in the unimportant regions of the state space without impacting the optimized objective function $\|V^* - V^{\mathbf{w}}\|_{1,\psi}$. Note that a trivial value function $V^{\mathbf{w}}(\mathbf{x}) = (1-\gamma)^{-1} R_{\max}$

satisfies all constraints in any HALP but unlikely leads to good policies. For a comprehensive discussion on selecting appropriate $\psi(\mathbf{x})$ and $L(\mathbf{x})$, refer to the case studies of de Farias and Van Roy (2003).

Our discussion is concluded by clarifying the notion of the state relevance density $\psi(\mathbf{x})$. As demonstrated by Theorem 4, its choice is closely related to the quality of a greedy policy for the value function $V^{\widetilde{\mathbf{w}}}$ (de Farias & Van Roy, 2003).

Theorem 4 *Let $\widetilde{\mathbf{w}}$ be an optimal solution to the HALP formulation (16). Then the expected error of a greedy policy:*

$$u(\mathbf{x}) = \arg\sup_{\mathbf{a}} \left[R(\mathbf{x}, \mathbf{a}) + \gamma \mathrm{E}_{P(\mathbf{x}' \mid \mathbf{x}, \mathbf{a})}\left[V^{\widetilde{\mathbf{w}}}(\mathbf{x}')\right]\right]$$

can be bounded as:

$$\|V^* - V^u\|_{1,\nu} \leq \frac{1}{1-\gamma} \left\|V^* - V^{\widetilde{\mathbf{w}}}\right\|_{1,\mu_{u,\nu}},$$

where $\|\cdot\|_{1,\nu}$ and $\|\cdot\|_{1,\mu_{u,\nu}}$ are weighted $\mathcal{L}_1$-norms, V^u is a value function for the greedy policy u, and $\mu_{u,\nu}$ is the expected frequency of state visits generated by following the policy u given the initial state distribution ν.

Based on Theorem 4, we may conclude that the expected error of greedy policies for HALP approximations is bounded when $\psi = \mu_{u,\nu}$. Note that the distribution $\mu_{u,\nu}$ is unknown when optimizing $V^{\widetilde{\mathbf{w}}}$ because it is a function of the optimized quantity itself. To break this cycle, de Farias and Van Roy (2003) suggested an iterative procedure that solves several LPs and adapts $\mu_{u,\nu}$ accordingly. In addition, real-world control problems exhibit a lot of structure, which permits the guessing of $\mu_{u,\nu}$.

Finally, it is important to realize that although our bounds (Theorems 3 and 4) build a foundation for better HALP approximations, they can be rarely used in practice because the optimal value function V^* is generally unknown. After all, if it was known, there is no need to approximate it. Moreover, note that the optimization of $\|V^* - V^{\mathbf{w}}\|_{\infty,1/L}$ (Theorem 3) is a hard problem and there are no methods that would minimize this error directly (Patrascu et al., 2002). Despite these facts, both bounds provide a loose guidance for empirical choices of basis functions. In Section 7, we use this intuition and propose basis functions that should closely approximate unknown optimal value functions V^*.

5.2 Expectation Terms

Since our basis functions are often restricted to small subsets of state variables, expectation terms (Equations 17 and 18) in the HALP formulation (16) should be efficiently computable. To unify the analysis of these expectation terms, $\mathrm{E}_{\psi(\mathbf{x})}[f_i(\mathbf{x})]$ and $\mathrm{E}_{P(\mathbf{x}' \mid \mathbf{x}, \mathbf{a})}[f_i(\mathbf{x}')]$, we show that their evaluation constitutes the same computational problem $\mathrm{E}_{P(\mathbf{x})}[f_i(\mathbf{x})]$, where $P(\mathbf{x})$ denotes some factored distribution.

Before we discuss expectation terms in the constraints, note that the transition function $P(\mathbf{x}' \mid \mathbf{x}, \mathbf{a})$ is factored and its parameterization is determined by the state-action pair $(\mathbf{x}, \mathbf{a})$. We keep the pair $(\mathbf{x}, \mathbf{a})$ fixed in the rest of the section, which corresponds to choosing a single constraint $(\mathbf{x}, \mathbf{a})$. Based on this selection, we rewrite the expectation terms $\mathrm{E}_{P(\mathbf{x}' \mid \mathbf{x}, \mathbf{a})}[f_i(\mathbf{x}')]$

in a simpler notation $\mathrm{E}_{P(\mathbf{x}')}[f_i(\mathbf{x}')]$, where $P(\mathbf{x}') = P(\mathbf{x}' \mid \mathbf{x}, \mathbf{a})$ denotes a factored distribution with fixed parameters.

We also assume that the state relevance density function $\psi(\mathbf{x})$ factors along $\mathbf{X}$ as:

$$\psi(\mathbf{x}) = \prod_{i=1}^{n} \psi_i(x_i), \tag{20}$$

where $\psi_i(x_i)$ is a distribution over the random state variable X_i. Based on this assumption, we can rewrite the expectation terms $\mathrm{E}_{\psi(\mathbf{x})}[f_i(\mathbf{x})]$ in the objective function in a new notation $\mathrm{E}_{P(\mathbf{x})}[f_i(\mathbf{x})]$, where $P(\mathbf{x}) = \psi(\mathbf{x})$ denotes a factored distribution. In line with our discussion in the last two paragraphs, efficient solutions to the expectation terms in HALP are obtained by solving the generalized term $\mathrm{E}_{P(\mathbf{x})}[f_i(\mathbf{x})]$ efficiently. We address this problem in the rest of the section.

Before computing the expectation term $\mathrm{E}_{P(\mathbf{x})}[f_i(\mathbf{x})]$ over the complete state space $\mathbf{X}$, we recall that the basis function $f_i(\mathbf{x})$ is defined on a subset of state variables $\mathbf{X}_i$. Therefore, we may conclude that $\mathrm{E}_{P(\mathbf{x})}[f_i(\mathbf{x})] = \mathrm{E}_{P(\mathbf{x}_i)}[f_i(\mathbf{x}_i)]$, where $P(\mathbf{x}_i)$ denotes a factored distribution on a lower dimensional space $\mathbf{X}_i$. If no further assumptions are made, the local expectation term $\mathrm{E}_{P(\mathbf{x}_i)}[f_i(\mathbf{x}_i)]$ may be still hard to compute. Although it can be estimated by a variety of numerical methods, for instance Monte Carlo (Andrieu et al., 2003), these techniques are imprecise if the sample size is small, and quite computationally expensive if a high precision is needed. Consequently, we try to avoid such an approximation step. Instead, we introduce an appropriate form of basis functions that leads to closed-form solutions to the expectation term $\mathrm{E}_{P(\mathbf{x}_i)}[f_i(\mathbf{x}_i)]$.

In particular, let us assume that every basis function $f_i(\mathbf{x}_i)$ factors as:

$$f_i(\mathbf{x}_i) = f_{i_D}(\mathbf{x}_{i_D}) f_{i_C}(\mathbf{x}_{i_C}) \tag{21}$$

along its discrete and continuous components $f_{i_D}(\mathbf{x}_{i_D})$ and $f_{i_C}(\mathbf{x}_{i_C})$, where the continuous component further decouples as a product:

$$f_{i_C}(\mathbf{x}_{i_C}) = \prod_{X_j \in \mathbf{X}_{i_C}} f_{ij}(x_j) \tag{22}$$

of univariate basis function factors $f_{ij}(x_j)$. Note that the basis functions remain multivariate despite the two independence assumptions. We make these presumptions for computational purposes and they are relaxed later in the section.

Based on Equation 21, we conclude that the expectation term:

$$\begin{aligned} \mathrm{E}_{P(\mathbf{x}_i)}[f_i(\mathbf{x}_i)] &= \mathrm{E}_{P(\mathbf{x}_i)}[f_{i_D}(\mathbf{x}_{i_D}) f_{i_C}(\mathbf{x}_{i_C})] \\ &= \mathrm{E}_{P(\mathbf{x}_{i_D})}[f_{i_D}(\mathbf{x}_{i_D})]\, \mathrm{E}_{P(\mathbf{x}_{i_C})}[f_{i_C}(\mathbf{x}_{i_C})] \end{aligned} \tag{23}$$

decomposes along the discrete and continuous variables $\mathbf{X}_{i_D}$ and $\mathbf{X}_{i_C}$, where $\mathbf{x}_i = (\mathbf{x}_{i_D}, \mathbf{x}_{i_C})$ and $P(\mathbf{x}_i) = P(\mathbf{x}_{i_D}) P(\mathbf{x}_{i_C})$. The evaluation of the discrete part $\mathrm{E}_{P(\mathbf{x}_{i_D})}[f_{i_D}(\mathbf{x}_{i_D})]$ requires aggregation in the subspace $\mathbf{X}_{i_D}$:

$$\mathrm{E}_{P(\mathbf{x}_{i_D})}[f_{i_D}(\mathbf{x}_{i_D})] = \sum_{\mathbf{x}_{i_D}} P(\mathbf{x}_{i_D}) f_{i_D}(\mathbf{x}_{i_D}), \tag{24}$$

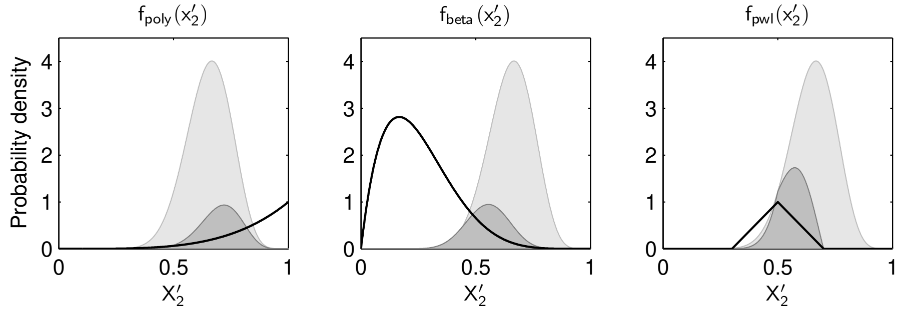


Figure 5: Expectation of three basis functions $f(x_2')$ (Example 5) with respect to the transition function $P(X_2' \mid X_2 = 1, X_1 = 0, a = a_1)$ from Figure 3. Every basis function $f(x_2')$ is depicted by a thick black line. The transition function is shown in a light gray color. Darker gray lines represent the values of the product $P(x_2' \mid \mathbf{x}, a_1) f(x_2')$. The area below corresponds to the expectation terms $\mathrm{E}_{P(x_2' \mid \mathbf{x}, a_1)}[f(x_2')]$.

which can be carried out efficiently in $O(\prod_{X_j \in \mathbf{X}_{i_D}} |\mathrm{Dom}(X_j)|)$ time (Section 3.3). Following Equation 22, the continuous term $\mathrm{E}_{P(\mathbf{x}_{i_C})}[f_{i_C}(\mathbf{x}_{i_C})]$ decouples as a product:

$$\begin{aligned} \mathrm{E}_{P(\mathbf{x}_{i_C})}[f_{i_C}(\mathbf{x}_{i_C})] &= \mathrm{E}_{P(\mathbf{x}_{i_C})}\left[\prod_{X_j \in \mathbf{X}_{i_C}} f_{ij}(x_j)\right] \\ &= \prod_{X_j \in \mathbf{X}_{i_C}} \mathrm{E}_{P(x_j)}[f_{ij}(x_j)], \end{aligned} \tag{25}$$

where $\mathrm{E}_{P(x_j)}[f_{ij}(x_j)]$ represents the expectation terms over individual random variables X_j. Consequently, an efficient solution to the local expectation term $\mathrm{E}_{P(\mathbf{x}_i)}[f_i(\mathbf{x}_i)]$ is guaranteed by efficient solutions to its univariate components $\mathrm{E}_{P(x_j)}[f_{ij}(x_j)]$.

In this paper, we consider three univariate basis function factors $f_{ij}(x_j)$: piecewise linear functions, polynomials, and beta distributions. These factors support a very general class of basis functions and yet allow closed-form solutions to the expectation terms $\mathrm{E}_{P(x_j)}[f_{ij}(x_j)]$. These solutions are provided in the following propositions and demonstrated in Example 5.

Proposition 3 (Polynomial basis functions) *Let:*

$$P(x) = P_{\mathrm{beta}}(x \mid \alpha, \beta)$$

be a beta distribution over X and:

$$f(x) = x^n (1 - x)^m$$

be a polynomial in x and $(1 - x)$. Then $\mathrm{E}_{P(x)}[f(x)]$ has a closed-form solution:

$$\mathrm{E}_{P(x)}[f(x)] = \frac{\Gamma(\alpha + \beta)}{\Gamma(\alpha)\Gamma(\beta)} \frac{\Gamma(\alpha + n)\Gamma(\beta + m)}{\Gamma(\alpha + \beta + n + m)}.$$

Corollary 1 (Beta basis functions) *Let:*

$$\begin{aligned} P(x) &= P_{\text{beta}}(x \mid \alpha, \beta) \\ f(x) &= P_{\text{beta}}(x \mid \alpha_f, \beta_f) \end{aligned}$$

be beta distributions over X. Then $\mathrm{E}_{P(x)}[f(x)]$ has a closed-form solution:

$$\mathrm{E}_{P(x)}[f(x)] = \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)} \frac{\Gamma(\alpha_f+\beta_f)}{\Gamma(\alpha_f)\Gamma(\beta_f)} \frac{\Gamma(\alpha+\alpha_f-1)\Gamma(\beta+\beta_f-1)}{\Gamma(\alpha+\alpha_f+\beta+\beta_f-2)}.$$

Proof: A direct consequence of Proposition 3. Since integration is a distributive operation, our claim straightforwardly generalizes to the mixture of beta distributions $P(x)$. □

Proposition 4 (Piecewise linear basis functions) *Let:*

$$P(x) = P_{\text{beta}}(x \mid \alpha, \beta)$$

be a beta distribution over X and:

$$f(x) = \sum_i \mathbf{1}_{[l_i, r_i]}(x)(a_i x + b_i)$$

be a piecewise linear (PWL) function in x, where $\mathbf{1}_{[l_i, r_i]}(x)$ represents the indicator function of the interval $[l_i, r_i]$. Then $\mathrm{E}_{P(x)}[f(x)]$ has a closed-form solution:

$$\mathrm{E}_{P(x)}[f(x)] = \sum_i \left[a_i \frac{\alpha}{\alpha+\beta}(F^+(r_i) - F^+(l_i)) + b_i(F(r_i) - F(l_i)) \right],$$

where $F(u) = F_{\text{beta}}(u \mid \alpha, \beta)$ and $F^+(u) = F_{\text{beta}}(u \mid \alpha+1, \beta)$ denote the cumulative density functions of beta distributions.

Example 5 *Efficient closed-form solutions to the expectation terms in HALP are illustrated on the 4-ring network administration problem (Example 4) with three hypothetical univariate basis functions:*

$$\begin{aligned} f_{\text{poly}}(x'_2) &= x'^4_2 \\ f_{\text{beta}}(x'_2) &= P_{\text{beta}}(x'_2 \mid 2, 6) \\ f_{\text{pwl}}(x'_2) &= \mathbf{1}_{[0.3,0.5]}(x'_2)(5x'_2 - 1.5) + \mathbf{1}_{[0.5,0.7]}(x'_2)(-5x'_2 + 3.5) \end{aligned}$$

Suppose that our goal is to evaluate expectation terms in a single constraint that corresponds to the network state $\mathbf{x} = (0, 1, 0, 0)$ and the administrator rebooting the server. Based on these assumptions, the expectation terms in the constraint $(\mathbf{x}, a_1)$ simplify as:

$$\mathrm{E}_{P(\mathbf{x}' \mid \mathbf{x}, a_1)}\left[f(x'_2)\right] = \mathrm{E}_{P(x'_2 \mid \mathbf{x}, a_1)}\left[f(x'_2)\right],$$

where the transition function $P(x'_2 \mid \mathbf{x}, a_1)$ is given by:

$$\begin{aligned} P(x'_2 \mid \mathbf{x}, a_1) &= P(X'_2 = x'_2 \mid X_2 = 1, X_1 = 0, a = a_1) \\ &= P_{\text{beta}}(x'_2 \mid 15, 8). \end{aligned}$$

Closed-form solutions to the simplified expectation terms $\mathrm{E}_{P(x_2' \mid \mathbf{x}, a_1)}[f(x_2')]$ *are computed as:*

$$
\begin{aligned}
\mathrm{E}_{P(x_2' \mid \mathbf{x}, a_1)}\left[f_{\mathrm{poly}}(x_2')\right] &= \int_{x_2'} P_{\mathrm{beta}}(x_2' \mid 15, 8) x_2'^4 \, \mathrm{d}x_2' \\
\text{(Proposition 3)} \quad &= \frac{\Gamma(15+8)}{\Gamma(15)\Gamma(8)} \frac{\Gamma(15+4)\Gamma(8)}{\Gamma(15+8+4)} \\
&\approx 0.20 \\
\mathrm{E}_{P(x_2' \mid \mathbf{x}, a_1)}\left[f_{\mathrm{beta}}(x_2')\right] &= \int_{x_2'} P_{\mathrm{beta}}(x_2' \mid 15, 8) P_{\mathrm{beta}}(x_2' \mid 2, 6) \, \mathrm{d}x_2' \\
\text{(Corollary 1)} \quad &= \frac{\Gamma(15+8)}{\Gamma(15)\Gamma(8)} \frac{\Gamma(2+6)}{\Gamma(2)\Gamma(6)} \frac{\Gamma(15+2-1)\Gamma(8+6-1)}{\Gamma(15+2+8+6-2)} \\
&\approx 0.22 \\
\mathrm{E}_{P(x_2' \mid \mathbf{x}, a_1)}\left[f_{\mathrm{pwl}}(x_2')\right] &= \int_{x_2'} P_{\mathrm{beta}}(x_2' \mid 15, 8) \mathbf{1}_{[0.3, 0.5]}(x_2')(5x_2' - 1.5) \, \mathrm{d}x_2' + \\
&\quad \int_{x_2'} P_{\mathrm{beta}}(x_2' \mid 15, 8) \mathbf{1}_{[0.5, 0.7]}(x_2')(-5x_2' + 3.5) \, \mathrm{d}x_2' \\
\text{(Proposition 4)} \quad &= 5 \frac{15}{15+8}(F^+(0.5) - F^+(0.3)) - 1.5(F(0.5) - F(0.3)) - \\
&\quad 5 \frac{15}{15+8}(F^+(0.7) - F^+(0.5)) + 3.5(F(0.7) - F(0.5)) \\
&\approx 0.30
\end{aligned}
$$

where $F(u) = F_{\mathrm{beta}}(u \mid 15, 8)$ *and* $F^+(u) = F_{\mathrm{beta}}(u \mid 15{+}1, 8)$ *denote the cumulative density functions of beta distributions. A graphical interpretation of these computations is presented in Figure 5. Brief inspection verifies that the term* $\mathrm{E}_{P(x_2' \mid \mathbf{x}, a_1)}[f_{\mathrm{pwl}}(x_2')]$ *is indeed the largest one.*

Up to this point, we obtained efficient closed-form solutions for factored basis functions and state relevance densities. Unfortunately, the factorization assumptions in Equations 20, 21, and 22 are rarely justified in practice. In the rest of the section, we show how to relax them. In Section 6, we apply our current results and propose several methods that approximately satisfy the constraint space in HALP.

5.2.1 Factored State Relevance Density Functions

Note that the state relevance density function $\psi(\mathbf{x})$ is very unlikely to be completely factored (Section 5.1). Therefore, the independence assumption in Equation 20 is extremely limiting. To relax this assumption, we approximate $\psi(\mathbf{x})$ by a linear combination $\psi^{\omega}(\mathbf{x}) = \sum_{\ell} \omega_{\ell} \psi_{\ell}(\mathbf{x})$ of factored state relevance densities $\psi_{\ell}(\mathbf{x}) = \prod_{i=1}^{n} \psi_{\ell i}(x_i)$. As a result, the expectation terms in the objective function decompose as:

$$
\begin{aligned}
\mathrm{E}_{\psi^{\omega}(\mathbf{x})}[f_i(\mathbf{x})] &= \mathrm{E}_{\sum_{\ell} \omega_{\ell} \psi_{\ell}(\mathbf{x})}[f_i(\mathbf{x})] \\
&= \sum_{\ell} \omega_{\ell} \mathrm{E}_{\psi_{\ell}(\mathbf{x})}[f_i(\mathbf{x})], \qquad (26)
\end{aligned}
$$

where the factored terms $\mathrm{E}_{\psi_\ell(\mathbf{x})}[f_i(\mathbf{x})]$ can be evaluated efficiently (Equation 23). Moreover, if we assume the factored densities $\psi_\ell(\mathbf{x})$ are polynomials, their linear combination $\psi^\omega(\mathbf{x})$ is a polynomial. Due to the Weierstrass approximation theorem (Jeffreys & Jeffreys, 1988), this polynomial is sufficient to approximate any state relevance density $\psi(\mathbf{x})$ with any precision. It follows that the linear combinations permit state relevance densities that reflect arbitrary dependencies among the state variables $\mathbf{X}$.

5.2.2 Factored Basis Functions

In line with the previous discussion, note that the linear value function $V^{\mathbf{w}}(\mathbf{x}) = \sum_i w_i f_i(\mathbf{x})$ with factored basis functions (Equations 21 and 22) is sufficient to approximate the optimal value function V^* within any max-norm error $\|V^* - V^{\mathbf{w}}\|_\infty$. Based on Theorem 2, we know that the same set of basis functions guarantees a bound on the $\mathcal{L}_1$-norm error $\left\|V^* - V^{\widetilde{\mathbf{w}}}\right\|_{1,\psi}$. Therefore, despite our independence assumptions (Equations 21 and 22), we have a potential to obtain an arbitrarily close HALP approximation $V^{\widetilde{\mathbf{w}}}$ to V^*.

6. Constraint Space Approximations

An optimal solution $\widetilde{\mathbf{w}}$ to the HALP formulation (16) is determined by a finite set of *active constraints* at a vertex of the feasible region. Unfortunately, identification of this active set is a hard computational problem. In particular, it requires searching through an exponential number of constraints, if the state and action variables are discrete, and an infinite number of constraints, if any of the variables are continuous. As a result, it is in general infeasible to find the optimal solution $\widetilde{\mathbf{w}}$ to the HALP formulation. Hence, we resort to approximations to the constraint space in HALP whose optimal solution $\widehat{\mathbf{w}}$ is close to $\widetilde{\mathbf{w}}$. This notion of an approximation is formalized as follows.

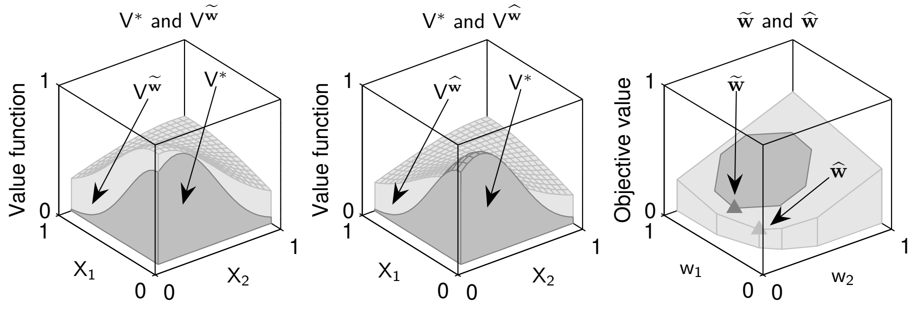
Definition 2 *The HALP formulation is* relaxed*:*

$$\begin{aligned} \text{minimize}_{\mathbf{w}} \quad & \sum_i w_i \alpha_i && (27)\\ \text{subject to:} \quad & \sum_i w_i F_i(\mathbf{x}, \mathbf{a}) - R(\mathbf{x}, \mathbf{a}) \geq 0 \quad (\mathbf{x}, \mathbf{a}) \in \mathcal{C}; \end{aligned}$$

if only a subset $\mathcal{C}$ of its constraints is satisfied.

The HALP formulation (16) can be solved approximately by solving its relaxed formulations (27). Several methods for building and solving these approximate LPs have been proposed: Monte Carlo sampling of constraints, (Hauskrecht & Kveton, 2004), ε-grid discretization of the constraint space (Guestrin et al., 2004), and an adaptive search for a violated constraint (Kveton & Hauskrecht, 2005). In the remainder of this section, we introduce these methods. From now on, we denote optimal solutions to the complete and relaxed HALP formulations by the symbols $\widetilde{\mathbf{w}}$ and $\widehat{\mathbf{w}}$, respectively.

Before we proceed, note that while $V^{\widetilde{\mathbf{w}}}$ is an upper bound on the optimal value function V^* (Figure 6a), the relaxed value function $V^{\widehat{\mathbf{w}}}$ does not have to be (Figure 6b). The reason is that the relaxed HALP formulation does not guarantee that the constraint $V^{\widehat{\mathbf{w}}} \geq \mathcal{T}^* V^{\widehat{\mathbf{w}}}$ is satisfied for all states $\mathbf{x}$. As a result, we cannot simply use Proposition 1 to prove $V^{\widehat{\mathbf{w}}} \geq V^*$.



(a) **(b)** **(c)**

Figure 6: **a.** Graphical relation between the value function V^* and its HALP approximation $V^{\widetilde{\mathbf{w}}}$. The function $V^{\widetilde{\mathbf{w}}}$ is guaranteed to be an upper bound on V^*. **b.** The relaxed HALP approximation $V^{\widehat{\mathbf{w}}}$ may not lead to an upper bound. **c.** Graphical relation between the optimal and relaxed solutions $\widetilde{\mathbf{w}}$ and $\widehat{\mathbf{w}}$. The feasible regions of the complete and relaxed HALP formulations are shown in dark and light gray colors. The value function approximations $V^{\widetilde{\mathbf{w}}}$ and $V^{\widehat{\mathbf{w}}}$ are typically nonlinear in the state space $\mathbf{X}$ but always linear in the space of parameters $\mathbf{w}$.

Furthermore, note that the inequality $\mathrm{E}_\psi\left[V^{\widehat{\mathbf{w}}}\right] \leq \mathrm{E}_\psi\left[V^{\widetilde{\mathbf{w}}}\right]$ always holds because the optimal solution $\widetilde{\mathbf{w}}$ is feasible in the relaxed HALP (Figure 6c). These observations become helpful for understanding the rest of the section.

6.1 MC-HALP

In the simplest case, the constraint space in HALP can be approximated by its Monte Carlo (MC) sample. In such a relaxation, the set of constraints $\mathcal{C}$ is selected with respect to some proposal distribution φ over state-action pairs $(\mathbf{x}, \mathbf{a})$. Since the set $\mathcal{C}$ is finite, it establishes a relaxed formulation (27), which can be solved by any LP solver. An algorithm that builds and satisfies relaxed MC-HALP formulations is outlined in Figure 7.

Constraint sampling is easily applied in continuous domains and its space complexity is proportional to the number of state and action components. Hauskrecht and Kveton (2004) used it to solve continuous-state factored MDPs and further refined it by heuristics (Kveton & Hauskrecht, 2004). In discrete-state domains, the quality of the sampled approximations was analyzed by de Farias and Van Roy (2004). Their result is summarized by Theorem 5.

Theorem 5 (de Farias & Van Roy, 2004) *Let $\widetilde{\mathbf{w}}$ be a solution to the ALP formulation (6) and $\widehat{\mathbf{w}}$ be a solution to its relaxed formulation whose constraints are sampled with respect to a proposal distribution φ over state-action pairs $(\mathbf{x}, \mathbf{a})$. Then there exist a distribution φ and sample size:*

$$N \geq O\left(\frac{A\theta}{(1-\gamma)\epsilon}\left(K \ln \frac{A\theta}{(1-\gamma)\epsilon} + \ln \frac{1}{\delta}\right)\right)$$

Inputs:

a hybrid factored MDP $\mathcal{M} = (\mathbf{X}, \mathbf{A}, P, R)$
basis functions $f_0(\mathbf{x}), f_1(\mathbf{x}), f_2(\mathbf{x}), \dots$
a proposal distribution φ

Algorithm:

```
initialize a relaxed HALP formulation with an empty set of constraints
t = 0
while a stopping criterion is not met
    sample (x, a) ~ φ
    add the constraint (x, a) to the relaxed HALP
    t = t + 1
solve the relaxed MC-HALP formulation
```

Outputs:

basis function weights $\mathbf{w}$

Figure 7: Pseudo-code implementation of the MC-HALP solver.

such that with probability at least $1 - \delta$*:*

$$\left\| V^* - V^{\widehat{\mathbf{w}}} \right\|_{1,\psi} \leq \left\| V^* - V^{\widetilde{\mathbf{w}}} \right\|_{1,\psi} + \epsilon \left\| V^* \right\|_{1,\psi},$$

where $\|\cdot\|_{1,\psi}$ *is an* $\mathcal{L}_1$*-norm weighted by the state relevance weights* ψ*,* θ *is a problem-specific constant,* A *and* K *denote the numbers of actions and basis functions, and* ϵ *and* δ *are scalars from the interval* $(0, 1)$*.*

Unfortunately, proposing a sampling distribution φ that guarantees this polynomial bound on the sample size is as hard as knowing the optimal policy π^* (de Farias & Van Roy, 2004). This conclusion is parallel to those in importance sampling. Note that uniform Monte Carlo sampling can guarantee a low probability of constraints being violated but it is not sufficient to bound the magnitude of their violation (de Farias & Van Roy, 2004).

6.2 ε-HALP

Another way of approximating the constraint space in HALP is by discretizing its continuous variables $\mathbf{X}_C$ and $\mathbf{A}_C$ on a uniform ε-grid. The new discretized constraint space preserves its original factored structure but spans discrete variables only. Therefore, it can be compactly satisfied by the methods for discrete-state ALP (Section 3.3). An algorithm that builds and satisfies relaxed ε-HALP formulations is outlined in Figure 8. Note that the new constraint space involves exponentially many constraints $O(\lceil 1/\varepsilon + 1 \rceil^{|\mathbf{X}_C| + |\mathbf{A}_C|})$ in the number of state and action variables $\mathbf{X}_C$ and $\mathbf{A}_C$.

6.2.1 Error Bounds

Recall that the ε-HALP formulation approximates the constraint space in HALP by a finite set of equally-spaced grid points. In this section, we study the quality of this approximation

Inputs:
a hybrid factored MDP $\mathcal{M} = (\mathbf{X}, \mathbf{A}, P, R)$
basis functions $f_0(\mathbf{x}), f_1(\mathbf{x}), f_2(\mathbf{x}), \ldots$
grid resolution ε

Algorithm:
discretize continuous variables $\mathbf{X}_C$ and $\mathbf{A}_C$ into $\lceil 1/\varepsilon + 1 \rceil$ equally-spaced values
identify subsets $\mathbf{X}_i$ and $\mathbf{A}_i$ ($\mathbf{X}_j$ and $\mathbf{A}_j$) corresponding to the domains of $F_i(\mathbf{x}, \mathbf{a})$ ($R_j(\mathbf{x}, \mathbf{a})$)
evaluate $F_i(\mathbf{x}_i, \mathbf{a}_i)$ ($R_j(\mathbf{x}_j, \mathbf{a}_j)$) for all configurations $\mathbf{x}_i$ and $\mathbf{a}_i$ ($\mathbf{x}_j$ and $\mathbf{a}_j$) on the ε-grid
calculate basis function relevance weights α_i
solve the relaxed ε-HALP formulation (Section 3.3)

Outputs:
basis function weights $\mathbf{w}$

Figure 8: Pseudo-code implementation of the ε-HALP solver.

and bound it in terms violating constraints in the complete HALP. More precisely, we prove that if a relaxed HALP solution $\widehat{\mathbf{w}}$ violates the constraints in the complete HALP by a small amount, the quality of the approximation $V^{\widehat{\mathbf{w}}}$ is close to $V^{\widetilde{\mathbf{w}}}$. In the next section, we extend this result and relate $V^{\widehat{\mathbf{w}}}$ to the grid resolution ε. Before we proceed, we quantify our notion of constraint violation.

Definition 3 *Let $\widehat{\mathbf{w}}$ be an optimal solution to a relaxed HALP formulation (27). The vector $\widehat{\mathbf{w}}$ is* δ-infeasible *if:*

$$V^{\widehat{\mathbf{w}}} - \mathcal{T}^* V^{\widehat{\mathbf{w}}} \geq -\delta, \tag{28}$$

where $\mathcal{T}^*$ is the hybrid Bellman operator.

Intuitively, the lower the δ-infeasibility of a relaxed HALP solution $\widehat{\mathbf{w}}$, the closer the quality of the approximation $V^{\widehat{\mathbf{w}}}$ to $V^{\widetilde{\mathbf{w}}}$. Proposition 5 states this intuition formally. In particular, it says that the relaxed HALP formulation leads to a close approximation $V^{\widehat{\mathbf{w}}}$ to the optimal value function V^* if the complete HALP does and the solution $\widehat{\mathbf{w}}$ violates its constraints by a small amount.

Proposition 5 *Let $\widetilde{\mathbf{w}}$ be an optimal solution to the HALP formulation (16) and $\widehat{\mathbf{w}}$ be an optimal δ-infeasible solution to its relaxed formulation (27). Then the expected error of the value function $V^{\widehat{\mathbf{w}}}$ can be bounded as:*

$$\left\| V^* - V^{\widehat{\mathbf{w}}} \right\|_{1,\psi} \leq \left\| V^* - V^{\widetilde{\mathbf{w}}} \right\|_{1,\psi} + \frac{2\delta}{1-\gamma},$$

where $\|\cdot\|_{1,\psi}$ is an $\mathcal{L}_1$-norm weighted by the state relevance density function ψ.

Based on Proposition 5, we can generalize our conclusions from Section 5.1 to relaxed HALP formulations. For instance, we may draw a parallel between optimizing the relaxed objective $\mathrm{E}_\psi\left[V^{\widehat{\mathbf{w}}}\right]$ and the max-norm error $\|V^* - V^{\mathbf{w}}\|_{\infty,1/L}$.

Theorem 6 *Let $\widehat{\mathbf{w}}$ be an optimal δ-infeasible solution to a relaxed HALP formulation (27). Then the expected error of the value function $V^{\widehat{\mathbf{w}}}$ can be bounded as:*

$$\left\|V^* - V^{\widehat{\mathbf{w}}}\right\|_{1,\psi} \leq \frac{2\mathrm{E}_\psi[L]}{1-\kappa} \min_{\mathbf{w}} \|V^* - V^{\mathbf{w}}\|_{\infty,1/L} + \frac{2\delta}{1-\gamma},$$

where $\|\cdot\|_{1,\psi}$ is an $\mathcal{L}_1$-norm weighted by the state relevance density ψ, $L(\mathbf{x}) = \sum_i w_i^L f_i(\mathbf{x})$ is a Lyapunov function *such that the inequality $\kappa L(\mathbf{x}) \geq \gamma \sup_{\mathbf{a}} \mathrm{E}_{P(\mathbf{x}'|\mathbf{x},\mathbf{a})}[L(\mathbf{x}')]$ holds, $\kappa \in [0, 1)$ denotes its* contraction factor*, and $\|\cdot\|_{\infty,1/L}$ is a max-norm reweighted by the reciprocal $1/L$.*

Proof: Direct combination of Theorem 3 and Proposition 5. □

6.2.2 Grid Resolution

In Section 6.2.1, we bounded the error of a relaxed HALP formulation by its δ-infeasibility (Theorem 6), a measure of constraint violation in the complete HALP. However, it is unclear how the grid resolution ε relates to δ-infeasibility. In this section, we analyze the relationship between ε and δ. Moreover, we show how to exploit the factored structure in the constraint space to achieve the δ-infeasibility of a relaxed HALP solution $\widehat{\mathbf{w}}$ efficiently.

First, let us assume that $\widehat{\mathbf{w}}$ is an optimal δ-infeasible solution to an ε-HALP formulation and $\mathbf{Z} = \mathbf{X} \cup \mathbf{A}$ is the joint set of state and action variables. To derive a bound relating both ε and δ, we assume that the magnitudes of constraint violations $\tau^{\widehat{\mathbf{w}}}(\mathbf{z}) = \sum_i \widehat{w}_i F_i(\mathbf{z}) - R(\mathbf{z})$ are Lipschitz continuous.

Definition 4 *The function $f(\mathbf{x})$ is* Lipschitz continuous *if:*

$$\left|f(\mathbf{x}) - f(\mathbf{x}')\right| \leq K \left\|\mathbf{x} - \mathbf{x}'\right\|_\infty \quad \forall\ \mathbf{x}, \mathbf{x}' \in \mathbf{X}; \tag{29}$$

where K is referred to as a Lipschitz constant*.*

Based on the ε-grid discretization of the constraint space, we know that the distance of any point $\mathbf{z}$ to its closest grid point $\mathbf{z}_G = \arg\min_{\mathbf{z}'} \|\mathbf{z} - \mathbf{z}'\|_\infty$ is bounded as:

$$\|\mathbf{z} - \mathbf{z}_G\|_\infty < \frac{\varepsilon}{2}. \tag{30}$$

From the Lipschitz continuity of $\tau^{\widehat{\mathbf{w}}}(\mathbf{z})$, we conclude:

$$\left|\tau^{\widehat{\mathbf{w}}}(\mathbf{z}_G) - \tau^{\widehat{\mathbf{w}}}(\mathbf{z})\right| \leq K \|\mathbf{z}_G - \mathbf{z}\|_\infty \leq \frac{K\varepsilon}{2}. \tag{31}$$

Since every constraint in the relaxed ε-HALP formulation is satisfied, $\tau^{\widehat{\mathbf{w}}}(\mathbf{z}_G)$ is nonnegative for all grid points $\mathbf{z}_G$. As a result, Equation 31 yields $\tau^{\widehat{\mathbf{w}}}(\mathbf{z}) > -K\varepsilon/2$ for every state-action pair $\mathbf{z} = (\mathbf{x}, \mathbf{a})$. Therefore, based on Definition 3, the solution $\widehat{\mathbf{w}}$ is δ-infeasible for $\delta \geq K\varepsilon/2$. Conversely, the δ-infeasibility of $\widehat{\mathbf{w}}$ is guaranteed by choosing $\varepsilon \leq 2\delta/K$.

Unfortunately, K may increase rapidly with the dimensionality of a function. To address this issue, we use the structure in the constraint space and demonstrate that this is not our case. First, we observe that the *global Lipschitz constant* K_{glob} is additive in *local Lipschitz constants* that correspond to the terms $\widehat{w}_i F_i(\mathbf{z})$ and $R_j(\mathbf{z})$. Moreover, $K_{\mathrm{glob}} \leq N K_{\mathrm{loc}}$, where

---

Inputs:

a hybrid factored MDP $\mathcal{M} = (\mathbf{X}, \mathbf{A}, P, R)$
basis functions $f_0(\mathbf{x}), f_1(\mathbf{x}), f_2(\mathbf{x}), \ldots$
initial basis function weights $\mathbf{w}^{(0)}$
a separation oracle $\mathcal{O}$

Algorithm:

```
initialize a relaxed HALP formulation with an empty set of constraints
t = 0
while a stopping criterion is not met
    query the oracle O for a violated constraint (x_O, a_O) with respect to w^(t)
    if the constraint (x_O, a_O) is violated
        add the constraint to the relaxed HALP
    resolve the LP for a new vector w^(t+1)
    t = t + 1
```

Outputs:

basis function weights $\mathbf{w}^{(t)}$

Figure 9: Pseudo-code implementation of a HALP solver with the cutting plane method.

N denotes the total number of the terms and K_{loc} is the maximum over the local constants. Finally, parallel to Equation 31, the δ-infeasibility of a relaxed HALP solution $\widehat{\mathbf{w}}$ is achieved by the discretization:

$$\varepsilon \leq \frac{2\delta}{NK_{\mathrm{loc}}} \leq \frac{2\delta}{K_{\mathrm{glob}}}. \tag{32}$$

Since the factors $\widehat{w}_i F_i(\mathbf{z})$ and $R_j(\mathbf{z})$ are often restricted to small subsets of state and action variables, K_{loc} should change a little when the size of a problem increases but its structure is fixed. To prove that K_{loc} is bounded, we have to bound the weights $\widehat{w}_i$. If all basis functions are of unit magnitude, the weights $\widehat{w}_i$ are intuitively bounded as $|\widehat{w}_i| \leq (1-\gamma)^{-1} R_{\max}$, where $R_{\max}$ denotes the maximum one-step reward in the HMDP.

Based on Equation 32, we conclude that the number of discretization points in a single dimension $\lceil 1/\varepsilon + 1 \rceil$ is bounded by a polynomial in N, K_{loc}, and $1/\delta$. Hence, the constraint space in the relaxed ε-HALP formulation involves $O([NK_{\mathrm{loc}}(1/\delta)]^{|\mathbf{X}|+|\mathbf{A}|})$ constraints, where $|\mathbf{X}|$ and $|\mathbf{A}|$ denote the number of state and action variables. The idea of variable elimination can be used to write the constraints compactly by $O([NK_{\mathrm{loc}}(1/\delta)]^{T+1}(|\mathbf{X}|+|\mathbf{A}|))$ constraints (Example 3), where T is the treewidth of a corresponding cost network. Therefore, satisfying this constraint space is polynomial in N, K_{loc}, $1/\delta$, $|\mathbf{X}|$, and $|\mathbf{A}|$, but still exponential in T.

6.3 Cutting Plane Method

Both MC and ε-HALP formulations (Sections 6.1 and 6.2) approximate the constraint space in HALP by a finite set of constraints $\mathcal{C}$. Therefore, they can be solved directly by any linear programming solver. However, if the number of constraints is large, formulating and solving

---

Inputs:

a hybrid factored MDP $\mathcal{M} = (\mathbf{X}, \mathbf{A}, P, R)$
basis functions $f_0(\mathbf{x}), f_1(\mathbf{x}), f_2(\mathbf{x}), \dots$
basis function weights $\mathbf{w}$
grid resolution ε

Algorithm:

discretize continuous variables $\mathbf{X}_C$ and $\mathbf{A}_C$ into $(\lceil 1/\varepsilon + 1 \rceil)$ equally-spaced values
identify subsets $\mathbf{X}_i$ and $\mathbf{A}_i$ ($\mathbf{X}_j$ and $\mathbf{A}_j$) corresponding to the domains of $F_i(\mathbf{x}, \mathbf{a})$ ($R_j(\mathbf{x}, \mathbf{a})$)
evaluate $F_i(\mathbf{x}_i, \mathbf{a}_i)$ ($R_j(\mathbf{x}_j, \mathbf{a}_j)$) for all configurations $\mathbf{x}_i$ and $\mathbf{a}_i$ ($\mathbf{x}_j$ and $\mathbf{a}_j$) on the ε-grid
build a cost network for the factored cost function:
  $\tau^{\mathbf{w}}(\mathbf{x}, \mathbf{a}) = \sum_i w_i F_i(\mathbf{x}, \mathbf{a}) - R(\mathbf{x}, \mathbf{a})$
find the most violated constraint in the cost network:
  $(\mathbf{x}_{\mathcal{O}}, \mathbf{a}_{\mathcal{O}}) = \arg\min_{\mathbf{x},\mathbf{a}} \tau^{\mathbf{w}}(\mathbf{x}, \mathbf{a})$

Outputs:

state-action pair $(\mathbf{x}_{\mathcal{O}}, \mathbf{a}_{\mathcal{O}})$

Figure 10: Pseudo-code implementation of the ε-HALP separation oracle $\mathcal{O}_\varepsilon$.

LPs with the complete set of constraints is infeasible. In this section, we show how to build relaxed HALP approximations efficiently by the cutting plane method.

The cutting plane method for solving HALP formulations is outlined in Figure 9. Briefly, this approach builds the set of LP constraints incrementally by adding a violated constraint to this set in every step. In the remainder of the paper, we refer to any method that returns a violated constraint for an arbitrary vector $\widehat{\mathbf{w}}$ as a *separation oracle*. Formally, every HALP oracle approaches the optimization problem:

$$\arg\min_{\mathbf{x},\mathbf{a}} \left[V^{\widehat{\mathbf{w}}}(\mathbf{x}) - \gamma \mathrm{E}_{P(\mathbf{x}' \mid \mathbf{x}, \mathbf{a})} \left[V^{\widehat{\mathbf{w}}}(\mathbf{x}') \right] - R(\mathbf{x}, \mathbf{a}) \right]. \tag{33}$$

Consequently, the problem of solving hybrid factored MDPs efficiently reduces to the design of efficient separation oracles. Note that the cutting plane method (Figure 9) can be applied to suboptimal solutions to Equation 33 if these correspond to violated constraints.

The presented approach can be directly used to satisfy the constraints in relaxed ε-HALP formulations (Schuurmans & Patrascu, 2002). Briefly, the solver from Figure 9 iterates until no violated constraint is found and the ε-HALP separation oracle $\mathcal{O}_\varepsilon$ (Figure 10) returns the most violated constraint in the discretized cost network given an intermediate solution $\mathbf{w}^{(t)}$. Note that although the search for the most violated constraint is polynomial in $|\mathbf{X}|$ and $|\mathbf{A}|$ (Section 6.2.2), the running time of our solver does not have to be (Guestrin, 2003). In fact, the number of generated cuts is exponential in $|\mathbf{X}|$ and $|\mathbf{A}|$ in the worst case. However, the same oracle embedded into the ellipsoid method (Khachiyan, 1979) yields a polynomial-time algorithm (Bertsimas & Tsitsiklis, 1997). Although this technique is impractical for solving large LPs, we may conclude that our approach is indeed polynomial-time if implemented in this particular way.

Finally, note that searching for the most violated constraint (Equation 33) has application beyond satisfying the constraint space in HALP. For instance, computation of a greedy policy for the value function $V^{\widehat{\mathbf{w}}}$:

$$\begin{aligned} u(\mathbf{x}) &= \arg\max_{\mathbf{a}} \left[R(\mathbf{x}, \mathbf{a}) + \gamma \mathrm{E}_{P(\mathbf{x}' \mid \mathbf{x}, \mathbf{a})} \left[V^{\widehat{\mathbf{w}}}(\mathbf{x}') \right] \right] \\ &= \arg\min_{\mathbf{a}} \left[-R(\mathbf{x}, \mathbf{a}) - \gamma \mathrm{E}_{P(\mathbf{x}' \mid \mathbf{x}, \mathbf{a})} \left[V^{\widehat{\mathbf{w}}}(\mathbf{x}') \right] \right] \end{aligned} \tag{34}$$

is almost an identical optimization problem, where the state variables $\mathbf{X}$ are fixed. Moreover, the magnitude of the most violated constraint is equal to the lowest δ for which the relaxed HALP solution $\widehat{\mathbf{w}}$ is δ-infeasible (Equation 28):

$$\begin{aligned} \underline{\delta} &= \min_{\mathbf{x}} \left[V^{\widehat{\mathbf{w}}}(\mathbf{x}) - \max_{\mathbf{a}} \left[R(\mathbf{x}, \mathbf{a}) + \gamma \mathrm{E}_{P(\mathbf{x}' \mid \mathbf{x}, \mathbf{a})} \left[V^{\widehat{\mathbf{w}}}(\mathbf{x}') \right] \right] \right] \\ &= \min_{\mathbf{x}, \mathbf{a}} \left[V^{\widehat{\mathbf{w}}}(\mathbf{x}) - R(\mathbf{x}, \mathbf{a}) - \gamma \mathrm{E}_{P(\mathbf{x}' \mid \mathbf{x}, \mathbf{a})} \left[V^{\widehat{\mathbf{w}}}(\mathbf{x}') \right] \right]. \end{aligned} \tag{35}$$

6.4 MCMC-HALP

In practice, both MC and ε-HALP formulations (Sections 6.1 and 6.2) are built on a blindly selected set of constraints $\mathcal{C}$. More specifically, the constraints in the MC-HALP formulation are chosen randomly (with respect to a prior distribution φ) while the ε-HALP formulation is based on a uniform ε-grid. This discretized constraint space preserves its original factored structure, which allows for its compact satisfaction. However, the complexity of solving the ε-HALP formulation is exponential in the treewidth of its discretized constraint space. Note that if the discretized constraint space is represented by binary variables only, the treewidth increases by a multiplicative factor of $\log_2 \lceil 1/\varepsilon + 1 \rceil$, where $\lceil 1/\varepsilon + 1 \rceil$ denotes the number of discretization points in a single dimension. Consequently, even if the treewidth of a problem is relatively small, solving its ε-HALP formulation becomes intractable for small values of ε.

To address the issues of the discussed approximations (Sections 6.1 and 6.2), we propose a novel Markov chain Monte Carlo (MCMC) method for finding the most violated constraint of a relaxed HALP. The procedure directly operates in the domains of continuous variables, takes into account the structure of factored MDPs, and its space complexity is proportional to the number of variables. This separation oracle can be easily embedded into the ellipsoid or cutting plane method for solving linear programs (Section 6.3), and therefore constitutes a key step towards solving HALP efficiently. Before we proceed, we represent the constraint space in HALP compactly and state an optimization problem for finding violated constraints in this factored representation.

6.4.1 Compact Representation of Constraints

In Section 3.3, we showed how the factored representation of the constraint space allows for its compact satisfaction. Following this idea, we define *violation magnitude* $\tau^{\mathbf{w}}(\mathbf{x}, \mathbf{a})$:

$$\begin{aligned} \tau^{\mathbf{w}}(\mathbf{x}, \mathbf{a}) &= -\left[V^{\mathbf{w}}(\mathbf{x}) - \gamma \mathrm{E}_{P(\mathbf{x}' \mid \mathbf{x}, \mathbf{a})} \left[V^{\mathbf{w}}(\mathbf{x}') \right] - R(\mathbf{x}, \mathbf{a}) \right] \\ &= -\sum_i w_i [f_i(\mathbf{x}) - \gamma g_i(\mathbf{x}, \mathbf{a})] + R(\mathbf{x}, \mathbf{a}), \end{aligned} \tag{36}$$

which measures the amount by which the solution $\mathbf{w}$ violates the constraints in the complete HALP. We represent the magnitude of violation $\tau^{\mathbf{w}}(\mathbf{x}, \mathbf{a})$ compactly by an influence diagram

(ID), where $\mathbf{X}$ and $\mathbf{A}$ are decision nodes, and $\mathbf{X}'$ are random variables. This representation is built on the transition model $P(\mathbf{X}' \mid \mathbf{X}, \mathbf{A})$, which is factored and captures independencies among the variables $\mathbf{X}$, $\mathbf{X}'$, and $\mathbf{A}$. We extend the diagram by three types of reward nodes, one for each term in Equation 36: $H_i = -w_i f_i(\mathbf{x})$ for every basis function, $G_i = \gamma w_i f_i(\mathbf{x}')$ for every backprojection, and $R_j = R_j(\mathbf{x}_j, \mathbf{a}_j)$ for every local reward function. The construction is completed by adding arcs that graphically represent the dependencies of the reward nodes on the variables. Finally, we can verify that:

$$\tau^{\mathbf{w}}(\mathbf{x}, \mathbf{a}) = \mathrm{E}_{P(\mathbf{x}'|\mathbf{x},\mathbf{a})}\left[\sum_i (H_i + G_i) + \sum_j R_j\right]. \tag{37}$$

Consequently, the decision that maximizes the expected utility in the ID corresponds to the most violated constraint. A graphical representation of the violation magnitude $\tau^{\mathbf{w}}(\mathbf{x}, \mathbf{a})$ on the 4-ring network administration problem (Example 4) is given in Figure 2a. The structure of the constraint space is identical to Example 3 if the basis functions are univariate.

We conclude that any algorithm for solving IDs can be applied to find the most violated constraint. However, most of these methods (Cooper, 1988; Jensen et al., 1994; Ortiz, 2002) are restricted to discrete variables. Fortunately, special properties of the ID representation allow its further simplification. If the basis functions are chosen conjugate to the transition model (Section 5.2), we obtain a closed-form solution to the expectation term $\mathrm{E}_{P(\mathbf{x}'|\mathbf{x},\mathbf{a})}[G_i]$ (Equation 18), and the random variables $\mathbf{X}'$ are marginalized out of the diagram. The new representation contains no random variables and is known as a cost network (Section 3.3).

Note that the problem of finding the most violated constraint in the ID representation is also identical to finding the maximum a posteriori (MAP) configuration of random variables in Bayesian networks (Dechter, 1996; Park & Darwiche, 2001, 2003; Yuan et al., 2004). The latter problem is difficult because of the alternating summation and maximization operators. Since we marginalized out the random variables $\mathbf{X}'$, we can solve the maximization problem by standard large-scale optimization techniques.

6.4.2 Separation Oracle $\mathcal{O}_{\mathrm{MCMC}}$

To find the most violated constraint in the cost network, we apply the Metropolis-Hastings (MH) algorithm (Metropolis et al., 1953; Hastings, 1970) and propose a Markov chain whose invariant distribution converges to the vicinity of $\arg\max_{\mathbf{z}} \tau^{\mathbf{w}}(\mathbf{z})$, where $\mathbf{z} = (\mathbf{x}, \mathbf{a})$ is a value assignment to the joint set of state and action variables $\mathbf{Z} = \mathbf{X} \cup \mathbf{A}$.

In short, the Metropolis-Hastings algorithm defines a Markov chain that transits between an existing state $\mathbf{z}$ and a proposed state $\mathbf{z}^*$ with the *acceptance probability*:

$$A(\mathbf{z}, \mathbf{z}^*) = \min\left\{1, \frac{p(\mathbf{z}^*)q(\mathbf{z} \mid \mathbf{z}^*)}{p(\mathbf{z})q(\mathbf{z}^* \mid \mathbf{z})}\right\}, \tag{38}$$

where $q(\mathbf{z}^* \mid \mathbf{z})$ and $p(\mathbf{z})$ are a *proposal distribution* and a *target density*, respectively. Under mild restrictions on $p(\mathbf{z})$ and $q(\mathbf{z}^* \mid \mathbf{z})$, the frequency of state visits generated by the Markov chain always converges to the target function $p(\mathbf{z})$ (Andrieu et al., 2003). In the remainder of this section, we discuss the choices of $p(\mathbf{z})$ and $q(\mathbf{z}^* \mid \mathbf{z})$ to solve our optimization problem.[5]

5. For an introduction to Markov chain Monte Carlo (MCMC) methods, refer to the work of Andrieu et al. (2003).

Target density: The violation magnitude $\tau^{\mathbf{w}}(\mathbf{z})$ is turned into a density by the transformation $p(\mathbf{z}) = \exp[\tau^{\mathbf{w}}(\mathbf{z})]$. Due to its monotonic character, $p(\mathbf{z})$ retains the same set of global maxima as $\tau^{\mathbf{w}}(\mathbf{z})$. Therefore, the search for $\arg\max_{\mathbf{z}} \tau^{\mathbf{w}}(\mathbf{z})$ can be done on the new function $p(\mathbf{z})$. To prove that $p(\mathbf{z})$ is a density, we demonstrate that $\sum_{\mathbf{z}_D} \int_{\mathbf{z}_C} p(\mathbf{z})\, \mathrm{d}\mathbf{z}_C$ is a normalizing constant, where $\mathbf{z}_D$ and $\mathbf{z}_C$ are the discrete and continuous parts of the value assignment $\mathbf{z}$. First, note that the integrand $\mathbf{z}_C$ is restricted to the space $[0,1]^{|\mathbf{Z}_C|}$. As a result, the integral $\int_{\mathbf{z}_C} p(\mathbf{z})\, \mathrm{d}\mathbf{z}_C$ is proper if $p(\mathbf{z})$ is bounded, and hence it is Riemann integrable and finite. To prove that $p(\mathbf{z}) = \exp[\tau^{\mathbf{w}}(\mathbf{z})]$ is bounded, we bound the magnitude of violation $\tau^{\mathbf{w}}(\mathbf{z})$. If all basis functions are of unit magnitude, the weights $\widehat{w}_i$ can be bounded as $|\widehat{w}_i| \leq (1-\gamma)^{-1} R_{\max}$ (Section 6.2.2), which in turn yields the bound $|\tau^{\mathbf{w}}(\mathbf{z})| \leq (|\mathbf{w}|\,(1-\gamma)^{-1}+1) R_{\max}$. Therefore, $p(\mathbf{z})$ is bounded and can be treated as a density function.

To find the mode of $p(\mathbf{z})$, we employ simulating annealing (Kirkpatrick et al., 1983) and generate a non-homogeneous Markov chain whose invariant distribution is equal to $p^{1/T_t}(\mathbf{z})$, where T_t is a cooling schedule such that $\lim_{t\to\infty} T_t = 0$. Under weak regularity assumptions on $p(\mathbf{z})$, $p^{\infty}(\mathbf{z})$ is a probability density that concentrates on the set of the global maxima of $p(\mathbf{z})$ (Andrieu et al., 2003). If our cooling schedule T_t decreases such that $T_t \geq c/\ln(t+1)$, where c is a problem-specific constant, the chain from Equation 38 converges to the vicinity of $\arg\max_{\mathbf{z}} \tau^{\mathbf{w}}(\mathbf{z})$ with the probability converging to 1 (Geman & Geman, 1984). However, this logarithmic cooling schedule is slow in practice, especially for a high initial temperature c. To overcome this problem, we select a smaller value of c (Geman & Geman, 1984) than is required by the convergence criterion. Therefore, the convergence of our chain to the global optimum $\arg\max_{\mathbf{z}} \tau^{\mathbf{w}}(\mathbf{z})$ is no longer guaranteed.

Proposal distribution: We take advantage of the factored character of $\mathbf{Z}$ and adopt the following proposal distribution (Geman & Geman, 1984):

$$q(\mathbf{z}^* \mid \mathbf{z}) = \begin{cases} p(z_i^* \mid \mathbf{z}_{-i}) & \text{if } \mathbf{z}^*_{-i} = \mathbf{z}_{-i} \\ 0 & \text{otherwise} \end{cases}, \tag{39}$$

where $\mathbf{z}_{-i}$ and $\mathbf{z}^*_{-i}$ are value assignments to all variables but Z_i in the original and proposed states. If Z_i is a discrete variable, its conditional:

$$p(z_i^* \mid \mathbf{z}_{-i}) = \frac{p(z_1, \dots, z_{i-1}, z_i^*, z_{i+1}, \dots, z_{n+m})}{\sum_{z_i} p(z_1, \dots, z_{i-1}, z_i, z_{i+1}, \dots, z_{n+m})} \tag{40}$$

can be derived in a closed form. If Z_i is a continuous variable, a closed form of its cumulative density function is unlikely to exist. To sample from the conditional, we embed another MH step within the original chain. In the experimental section, we use the Metropolis algorithm with the acceptance probability:

$$A(z_i, z_i^*) = \min\left\{1, \frac{p(z_i^* \mid \mathbf{z}_{-i})}{p(z_i \mid \mathbf{z}_{-i})}\right\}, \tag{41}$$

where z_i and z_i^* are the original and proposed values of the variable Z_i. Note that sampling from both conditionals can be performed in the space of $\tau^{\mathbf{w}}(\mathbf{z})$ and locally.

Inputs:

a hybrid factored MDP $\mathcal{M} = (\mathbf{X}, \mathbf{A}, P, R)$
basis functions $f_0(\mathbf{x}), f_1(\mathbf{x}), f_2(\mathbf{x}), \dots$
basis function weights $\mathbf{w}$

Algorithm:

initialize a state-action pair $\mathbf{z}^{(t)}$
$t = 0$
while a stopping criterion is not met
  for every variable Z_i
    sample $u \sim \mathrm{U}_{[0,1]}$
    sample $z_i^* \sim p(Z_i \mid \mathbf{z}_{-i}^{(t)})$
    if $u < \min\left\{1, \frac{p^{1/T_t-1}(z_i^* \mid \mathbf{z}_{-i}^{(t)})}{p^{1/T_t-1}(z_i^{(t)} \mid \mathbf{z}_{-i}^{(t)})}\right\}$
      $z_i^{(t+1)} = z_i^*$
    else
      $z_i^{(t+1)} = z_i^{(t)}$
  update T_{t+1} according to the cooling schedule
  $t = t + 1$
$(\mathbf{x}_\mathcal{O}, \mathbf{a}_\mathcal{O}) = \mathbf{z}^{(t)}$

Outputs:

state-action pair $(\mathbf{x}_\mathcal{O}, \mathbf{a}_\mathcal{O})$

Figure 11: Pseudo-code implementation of the MCMC-HALP oracle $\mathcal{O}_{\mathrm{MCMC}}$. The symbol $\mathrm{U}_{[0,1]}$ denotes the uniform distribution on the interval $[0, 1]$. Since the testing for violated constraints (Figure 9) is inexpensive, our implementation of the MCMC-HALP solver in Section 7 tests all constraints $\mathbf{z}^{(t)}$ generated by the Markov chain and not only the last one. Therefore, the separation oracle $\mathcal{O}_{\mathrm{MCMC}}$ returns more than one constraint per chain.

Finally, by assuming that $\mathbf{z}_{-i}^* = \mathbf{z}_{-i}$ (Equation 39), we derive a non-homogenous Markov chain with the acceptance probability:

$$\begin{aligned}
A(\mathbf{z}, \mathbf{z}^*) &= \min\left\{1, \frac{p^{1/T_t}(\mathbf{z}^*)q(\mathbf{z} \mid \mathbf{z}^*)}{p^{1/T_t}(\mathbf{z})q(\mathbf{z}^* \mid \mathbf{z})}\right\} \\
&= \min\left\{1, \frac{p^{1/T_t}(z_i^* \mid \mathbf{z}_{-i}^*)p^{1/T_t}(\mathbf{z}_{-i}^*)p(z_i \mid \mathbf{z}_{-i}^*)}{p^{1/T_t}(z_i \mid \mathbf{z}_{-i})p^{1/T_t}(\mathbf{z}_{-i})p(z_i^* \mid \mathbf{z}_{-i})}\right\} \\
&= \min\left\{1, \frac{p^{1/T_t}(z_i^* \mid \mathbf{z}_{-i})p^{1/T_t}(\mathbf{z}_{-i})p(z_i \mid \mathbf{z}_{-i})}{p^{1/T_t}(z_i \mid \mathbf{z}_{-i})p^{1/T_t}(\mathbf{z}_{-i})p(z_i^* \mid \mathbf{z}_{-i})}\right\} \\
&= \min\left\{1, \frac{p^{1/T_t-1}(z_i^* \mid \mathbf{z}_{-i})}{p^{1/T_t-1}(z_i \mid \mathbf{z}_{-i})}\right\}, \qquad (42)
\end{aligned}$$

which converges to the vicinity of the most violated constraint. Yuan et al. (2004) proposed a similar chain for finding the MAP configuration of random variables in Bayesian networks.

6.4.3 Constraint Satisfaction

If the MCMC-HALP separation oracle $\mathcal{O}_{\mathrm{MCMC}}$ (Figure 11) converges to a violated constraint (not necessarily the most violated) in polynomial time, the ellipsoid method is guaranteed to solve HALP formulations in polynomial time (Bertsimas & Tsitsiklis, 1997). Unfortunately, convergence of our chain within arbitrary precision requires an exponential number of steps (Geman & Geman, 1984). Although the bound is loose to be of practical interest, it suggests that the time complexity of proposing violated constraints dominates the time complexity of solving relaxed HALP formulations. Therefore, the oracle $\mathcal{O}_{\mathrm{MCMC}}$ should search for violated constraints efficiently. Convergence speedups that directly apply to our work include hybrid Monte Carlo (HMC) (Duane et al., 1987), Rao-Blackwellization (Casella & Robert, 1996), and slice sampling (Higdon, 1998).

7. Experiments

Experimental section is divided in three parts. First, we show that HALP can solve a simple HMDP problem at least as efficiently as alternative approaches. Second, we demonstrate the scale-up potential of our framework and compare several approaches to satisfy the constraint space in HALP (Section 6). Finally, we argue for solving our constraint satisfaction problem in the domains of continuous variables without discretizing them.

All experiments are performed on a Dell Precision 380 workstation with 3.2GHz Pentium 4 CPU and 2GB RAM. Linear programs are solved by the simplex method in the LP_SOLVE package. The expected return of policies is estimated by the Monte Carlo simulation of 100 trajectories. The results of randomized methods are additionally averaged over 10 randomly initialized runs. Whenever necessary, we present errors on the expected values. These errors correspond to the standard deviations of measured quantities. The discount factor γ is 0.95.

7.1 A Simple Example

To illustrate the ability of HALP to solve factored MDPs, we compare it to $\mathcal{L}_2$ (Figure 4) and grid-based value iteration (Section 4.2) on the 4-ring topology of the network administration problem (Example 4). Our experiments are conducted on uniform and non-uniform grids of varying sizes. Grid points are kept fixed for all compared methods, which allows for their fair comparison. Both value iteration methods are iterated for 100 steps and terminated earlier if their Bellman error drops below 10^{-6}. Both the $\mathcal{L}_2$ and HALP methods approximate the optimal value function V^* by a linear combination of basis functions, one for each computer X_i ($f_i(\mathbf{x}) = x_i$), and one for every connection $X_i \to X_j$ in the ring topology ($f_{i \to j}(\mathbf{x}) = x_i x_j$). We assume that our basis functions are sufficient to derive a one-step lookahead policy that reboots the least efficient computer. We believe that such a policy is close-to-optimal in the ring topology. The constraint space in the complete HALP formulation is approximated by its MC-HALP and ε-HALP formulations (Sections 6.1 and 6.2). The state relevance density function $\psi(\mathbf{x})$ is uniform. Our experimental results are reported in Figure 12.

Uniform ε-grid

		ε-HALP		**$\mathcal{L}_2$ VI**		**Grid-based VI**	
ε	N	Reward	Time	Reward	Time	Reward	Time
1	8	52.1 ± 2.2	< 1	52.1 ± 2.2	2		
1/2	91	52.1 ± 2.2	< 1	52.1 ± 2.2	7	47.6 ± 2.2	< 1
1/4	625	52.1 ± 2.2	< 1	52.1 ± 2.2	55	51.5 ± 2.2	20
1/8	6 561	52.1 ± 2.2	2	52.1 ± 2.2	577	52.0 ± 2.3	2 216

Heuristics

Policy	Reward
Dummy	25.0 ± 2.8
Random	42.1 ± 3.3
Server	47.6 ± 2.2
Utopian	83.0

Non-uniform grid

	MC-HALP		**$\mathcal{L}_2$ VI**		**Grid-based VI**	
N	Reward	Time	Reward	Time	Reward	Time
10	45.2 ± 5.1	< 1	45.9 ± 5.8	1	47.5 ± 2.8	< 1
50	50.2 ± 2.4	< 1	51.8 ± 2.2	4	48.7 ± 2.5	< 1
250	51.5 ± 2.4	< 1	51.9 ± 2.2	22	50.4 ± 2.3	2
1 250	51.8 ± 2.3	< 1	51.9 ± 2.2	110	51.6 ± 2.2	60

Figure 12: Comparison of three approaches to solving hybrid MDPs on the 4-ring topology of the network administration problem (Example 4). The methods are compared on uniform and non-uniform grids of varying size (N) by the expected discounted reward of policies and their computation time (in seconds).

To verify that our solutions are non-trivial, we compare them to three heuristic policies: dummy, random, and server. The dummy policy $\pi_{\text{dummy}}(\mathbf{x}) = a_5$ always takes the dummy action a_5. Therefore, it establishes a lower bound on the performance of any administrator. The random policy behaves randomly. The server policy $\pi_{\text{server}}(\mathbf{x}) = a_1$ protects the server X_1. The performance of our heuristics is shown in Figure 12. Assuming that we can reboot all computers at each time step, a utopian upper bound on the performance of any policy π can be derived as:

$$\begin{aligned} \mathrm{E}_\pi\left[\sum_{t=0}^{\infty} \gamma^t R(\mathbf{x}_t, \pi(\mathbf{x}_t))\right] &\leq \frac{1}{1-\gamma} \max_{\mathbf{x},a} \mathrm{E}_{P(\mathbf{x}' \mid \mathbf{x},a)}\left[\max_{a'} R(\mathbf{x}', a')\right] \\ &= \frac{1}{1-\gamma} \max_{\mathbf{x},a} \int_{\mathbf{x}'} 2P(x_1' \mid \mathbf{x}, a) x_1'^2 + \sum_{j=2}^{4} P(x_j' \mid \mathbf{x}, a) x_j'^2 \, \mathrm{d}\mathbf{x}' \\ &\leq \frac{5}{1-\gamma} \int_{x'} P_{\text{beta}}(x' \mid 20, 2) x'^2 \, \mathrm{d}x' \\ &\approx 83.0. \end{aligned} \tag{43}$$

We do not analyze the quality of HALP solutions with respect to the optimal value function V^* (Section 5.1) because this one is unknown.

Based on our results, we draw the following conclusions. First, grid-based value iteration is not practical for solving hybrid optimization problems of even small size. The main reason is the space complexity of the method, which is quadratic in the number of grid points N. If the state space is discretized uniformly, N is exponential in the number of state variables. Second, the quality of the HALP policies is close to the $\mathcal{L}_2$ VI policies. This result is positive since $\mathcal{L}_2$ value iteration is commonly applied in approximate dynamic programming. Third, both the $\mathcal{L}_2$ and HALP approaches yield better policies than grid-based value iteration. This result is due to the quality of our value function estimator. Its extremely good performance for $\varepsilon = 1$ can be explained from the monotonicity of the reward and basis functions. Finally, the computation time of the $\mathcal{L}_2$ VI policies is significantly longer than the computation time of the HALP policies. Since a step of $\mathcal{L}_2$ value iteration (Figure 4) is as hard as formulating a corresponding relaxed HALP, this result comes at no surprise.

7.2 Scale-up Potential

To illustrate the scale-up potential of HALP, we apply three relaxed HALP approximations (Section 6) to solve two irrigation network problems of varying complexity. These problems are challenging for state-of-the-art MDP solvers due to the factored state and action spaces.

Example 6 (Irrigation network operator) *An irrigation network is a system of irrigation channels connected by regulation devices (Figure 13). The goal of an irrigation network operator is to route water between the channels to optimize water levels in the whole system. The optimal levels are determined by the type of a planted crop. For simplicity of exposition, we assume that all irrigation channels are oriented and of the same size.*

This optimization problem can be formulated as a factored MDP. The state of the network is completely observable and represented by n *continuous variables* $\mathbf{X} = \{X_1, \dots, X_n\}$*, where the variable* X_i *denotes the water level in the* i*-th channel. At each time step, the irrigation network operator regulates* m *devices* A_i *that pump water between every pair of their inbound and outbound channels. The operation modes of these devices are described by discrete action variables* $\mathbf{A} = \{A_1, \dots, A_m\}$*. Inflow and outflow devices (no inbound or outbound channels) are not controlled and just pump water in and out of the network.*

The transition model reflects water flows in the irrigation network and is encoded locally by conditioning on the operation modes $\mathbf{A}$*:*

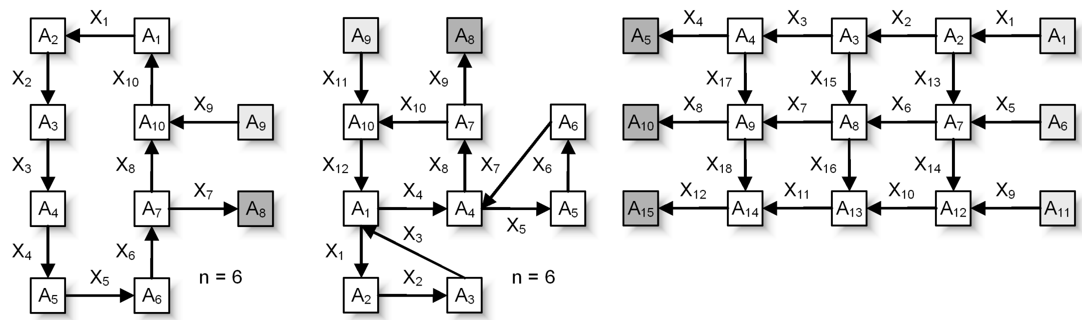
$$P(X'_{i \to j} = x \mid \mathsf{Par}(X'_{i \to j})) \;\propto\; P_{\mathrm{beta}}(x \mid \alpha, \beta) \quad \left| \quad \begin{array}{l} \alpha = 46\mu'_{i \to j} + 2 \\ \beta = 46(1 - \mu'_{i \to j}) + 2 \end{array} \right.$$

$$\begin{aligned} \mu'_{i \to j} &= \mu_{i \to j} + \sum_h \mathbf{1}_{a_{h \to i \to j}}(A_i) \min(1 - \mu_{i \to j}, \min(x_{h \to i}, \tau_i)) \\ \mu_{i \to j} &= x_{i \to j} - \sum_k \mathbf{1}_{a_{i \to j \to k}}(A_j) \min(x_{i \to j}, \tau_j) \end{aligned}$$

where $X_{i \to j}$ *represents the water level between the regulation devices* A_i *and* A_j*,* $\mathbf{1}_{a_{h \to i \to j}}(A_i)$ *and* $\mathbf{1}_{a_{i \to j \to k}}(A_j)$ *denote the indicator functions of water routing actions* $a_{h \to i \to j}$ *and* $a_{i \to j \to k}$ *at the devices* A_i *and* A_j*, and* τ_i *and* τ_j *are the highest tolerated flows through these devices. In short, this transition model conserves water mass in the network and adds some variance to the resulting state* $X'_{i \to j}$*. The introduced indexing of state and action variables is explained on the 6-ring irrigation network in Figure 14a. In the rest of the paper, we assume an inflow of 0.1 to any inflow device* A_i *(*$\tau_i = 0.1$*), an outflow of 1 from any outflow device* A_j *(*$\tau_j = 1$*), and the highest tolerated flow of* $1/3$ *at the remaining devices* A_k *(*$\tau_k = 1/3$*).*

The reward function $R(\mathbf{x}, \mathbf{a}) = \sum_j R_j(x_j)$ *is factored along individual irrigation channels and described by the univariate function:*

$$R_j(x_j) = 2x_j$$



(a) (b) (c)

Figure 13: Illustrations of three irrigation network topologies: **a.** 6-ring, **b.** 6-ring-of-rings, and **c.** 3×3 grid. Irrigation channels and their regulation devices are represented by arrows and rectangles. Inflow and outflow nodes are colored in light and dark gray. The ring and ring-of-rings networks are parameterized by the total number of regulation devices except for the last four (n).

for each outflow channel (one of its regulation devices must be outflow), and by the function:

$$R_j(x_j) = \frac{\mathcal{N}(x_j \mid 0.4, 0.025)}{25.6} + \frac{\mathcal{N}(x_j \mid 0.55, 0.05)}{32}$$

for the remaining channels (Figure 14b). Therefore, we reward both for maintaining optimal water levels and pumping water out of the irrigation network. Several examples of irrigation network topologies are shown in Figure 13.

Similarly to Equation 43, we derive a utopian upper bound on the performance of any policy π in an arbitrary irrigation network as:

$$\begin{aligned} \mathrm{E}_\pi\left[\sum_{t=0}^{\infty} \gamma^t R(\mathbf{x}_t, \pi(\mathbf{x}_t))\right] \;\leq\; & \frac{1}{1-\gamma}\Bigg[0.2 n_{\text{in}} + (n - n_{\text{out}}) \\ & \max_x \int_{x'} P_{\text{beta}}(x' \mid 46x + 2, 46(1-x) + 2) R(x')\, \mathrm{d}x'\Bigg], \end{aligned} \tag{44}$$

where n is the total number of irrigation channels, n_{in} and n_{out} denote the number of inflow and outflow channels, respectively, and $R(x') = \mathcal{N}(x' \mid 0.4, 0.025)/25.6 + \mathcal{N}(x' \mid 0.55, 0.05)/32$. We do not analyze the quality of HALP solutions with respect to the optimal value function V^* (Section 5.1) because this one is unknown.

In the rest of the section, we illustrate the performance of three HALP approximations, MC-HALP, ε-HALP, and MCMC-HALP (Section 6), on the ring and ring-of-rings topologies (Figure 13) of the irrigation network problem. The constraints in the MC-HALP formulation are sampled uniformly at random. This establishes a baseline for all HALP approximations. The ε-HALP and MCMC-HALP formulations are generated iteratively by the cutting plane

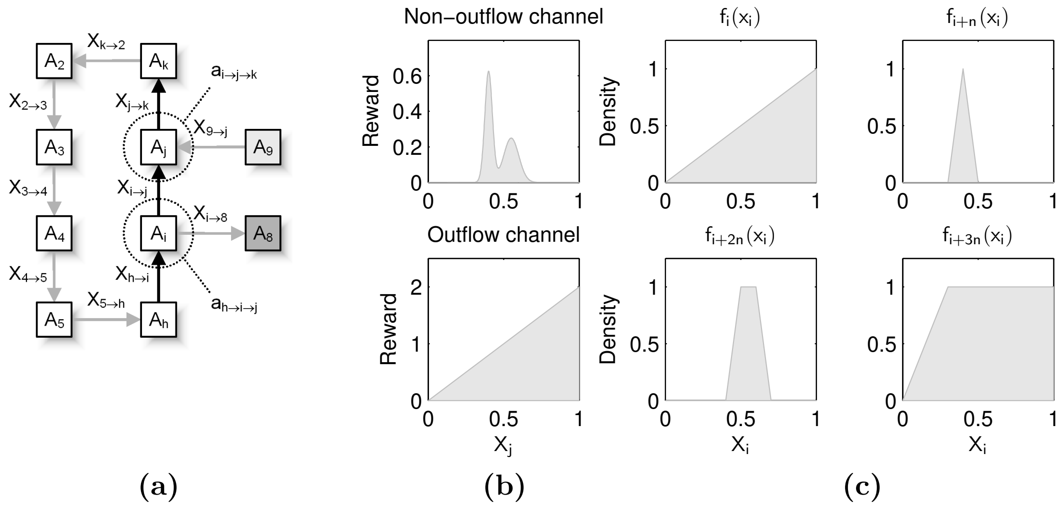


Figure 14: **a.** Indexing used in the description of the transition function in Example 6. The parameters h, i, j, and k are equal to 6, 7, 10, and 1, respectively. **b.** Univariate reward functions over water levels X_j (Example 6). **c.** Univariate basis functions over water levels X_i.

method. The MCMC oracle $\mathcal{O}_{\mathrm{MCMC}}$ is simulated for 500 steps from the initial temperature $c = 0.2$, which leads to a decreasing cooling schedule from $T_0 = 0.2$ to $T_{500} \approx 0.02$. These parameters are selected empirically to demonstrate the characteristics of the oracle $\mathcal{O}_{\mathrm{MCMC}}$ rather than to maximize its performance. The value function V^* is approximated by a linear combination of four univariate piecewise linear basis functions for each channel (Figure 14c). We assume that our basis functions are sufficient to derive a one-step lookahead policy that routes water between the channels if their water levels are too high or too low (Figure 14b). We believe that such a policy is close-to-optimal in irrigation networks. The state relevance density function $\psi(\mathbf{x})$ is uniform. Our experimental results are reported in Figures 15–17.

Based on the results, we draw the following conclusions. First, all HALP approximations scale up in the dimensionality of solved problems. As shown in Figure 16, the return of the policies grows linearly in n. Moreover, the time complexity of computing them is polynomial in n. Therefore, if a problem and its approximate solution are structured, we take advantage of this structure to avoid an exponential blowup in the computation time. At the same time, the quality of the policies is not deteriorating with increasing problem size n.

Second, the MCMC solver ($N = 250$) achieves the highest objective values on all solved problems. Higher objective values are interpreted as closer approximations to the constraint space in HALP since the solvers operate on relaxed formulations of HALP. Third, the quality of the MCMC-HALP policies ($N = 250$) surpasses the MC-HALP policies ($N = 10^6$) while both solvers consume approximately the same computation time. This result is due to the informative search for violated constraints in the MCMC-HALP solver. Fourth, the quality of the MCMC-HALP policies ($N = 250$) is close to the ε-HALP policies ($\varepsilon = 1/16$) although there is a significant difference between their objective values. Further analysis shows that the shape of the value functions is similar (Figure 17) and they differ the most in the weight

Ring topology		$n=6$			$n=12$			$n=18$		
		OV	Reward	Time	OV	Reward	Time	OV	Reward	Time
ε-HALP	1/4	24.3	34.6 ± 2.0	11	36.2	53.9 ± 2.7	44	48.0	74.3 ± 2.9	87
$\varepsilon =$	1/8	55.4	39.6 ± 2.5	41	88.1	61.5 ± 3.5	107	118.8	84.3 ± 3.8	178
	1/16	59.1	40.3 ± 2.6	281	93.2	62.6 ± 3.4	665	126.1	86.3 ± 3.8	1 119
MCMC	10	60.9	30.3 ± 4.9	38	86.3	47.6 ± 6.3	62	109.5	56.8 ± 7.4	87
$N =$	50	70.1	40.2 ± 2.6	194	110.3	62.4 ± 3.5	328	148.8	85.0 ± 3.6	483
	250	70.7	40.2 ± 2.6	940	112.0	63.0 ± 3.4	1 609	151.7	85.4 ± 3.8	2 280
MC	10^2	16.2	25.0 ± 5.1	< 1	16.9	41.9 ± 5.6	< 1	17.2	51.8 ± 8.8	< 1
$N =$	10^4	40.8	37.9 ± 2.8	10	52.8	58.8 ± 3.5	18	63.8	75.9 ± 6.6	31
	10^6	51.2	39.4 ± 2.7	855	67.1	60.3 ± 3.8	1 415	81.1	82.9 ± 3.8	1 938
Utopian			49.1			79.2			109.2	

Ring-of-rings topology		$n=6$			$n=12$			$n=18$		
		OV	Reward	Time	OV	Reward	Time	OV	Reward	Time
ε-HALP	1/4	28.4	40.4 ± 2.5	85	44.1	66.5 ± 3.2	382	59.8	93.0 ± 3.8	931
$\varepsilon =$	1/8	65.4	47.5 ± 3.0	495	107.9	76.1 ± 4.1	2 379	148.8	105.3 ± 4.2	5 877
	1/16	68.9	47.0 ± 2.9	4 417	113.1	77.3 ± 4.2	19 794	156.9	107.8 ± 4.1	53 655
MCMC	10	66.9	35.3 ± 6.1	60	94.6	54.4 ± 9.4	107	110.6	47.8 ± 13.2	157
$N =$	50	80.9	47.1 ± 2.9	309	131.9	76.6 ± 3.6	571	181.4	104.6 ± 4.4	859
	250	81.7	47.2 ± 2.9	1 522	134.1	77.3 ± 3.5	2 800	186.0	106.6 ± 3.9	4 291
MC	10^2	13.7	31.0 ± 4.9	< 1	15.4	46.1 ± 6.4	< 1	16.8	66.6 ± 9.4	1
$N =$	10^4	44.3	43.3 ± 3.2	12	59.0	68.9 ± 5.4	26	71.5	92.2 ± 6.8	49
	10^6	55.8	45.1 ± 3.1	1 026	75.1	74.3 ± 3.8	1 738	92.0	103.1 ± 4.2	2 539
Utopian			59.1			99.2			139.3	

Figure 15: Comparison of three HALP solvers on two irrigation network topologies of varying sizes (n). The solvers are compared by the objective value of a relaxed HALP (OV), the expected discounted reward of a corresponding policy, and computation time (in seconds). The ε-HALP, MCMC-HALP, and MC-HALP solvers are parameterized by the resolution of ε-grid (ε), the number of MCMC chains (N), and the number of samples (N). Note that the quality of policies improves with higher grid resolution ($1/\varepsilon$) and larger sample size (N). Upper bounds on their expected returns are shown in the last rows of the tables.

of the constant basis function $f_0(\mathbf{x}) \equiv 1$. Note that increasing w_0 does not affect the quality of a greedy policy for $V^{\mathbf{w}}$. However, this trick allows the satisfaction of the constraint space in HALP (Section 5.1).

Finally, the computation time of the ε-HALP solver is seriously affected by the topologies of the irrigation networks, which can be explained as follows. For a small ε and large n, the time complexity of formulating cost networks for the ring and ring-of-rings topologies grows by the rates of $\lceil 1/\varepsilon + 1 \rceil^2$ and $\lceil 1/\varepsilon + 1 \rceil^3$, respectively. Since the ε-HALP method consumes a significant amount of time by constructing cost networks, its quadratic (in $\lceil 1/\varepsilon + 1 \rceil$) time complexity on the ring topology worsens to cubic (in $\lceil 1/\varepsilon + 1 \rceil$) on the ring-of-rings topology. On the other hand, a similar cross-topology comparison of the MCMC-HALP solver shows that its computation times differ only by a multiplicative factor of 2. This difference is due

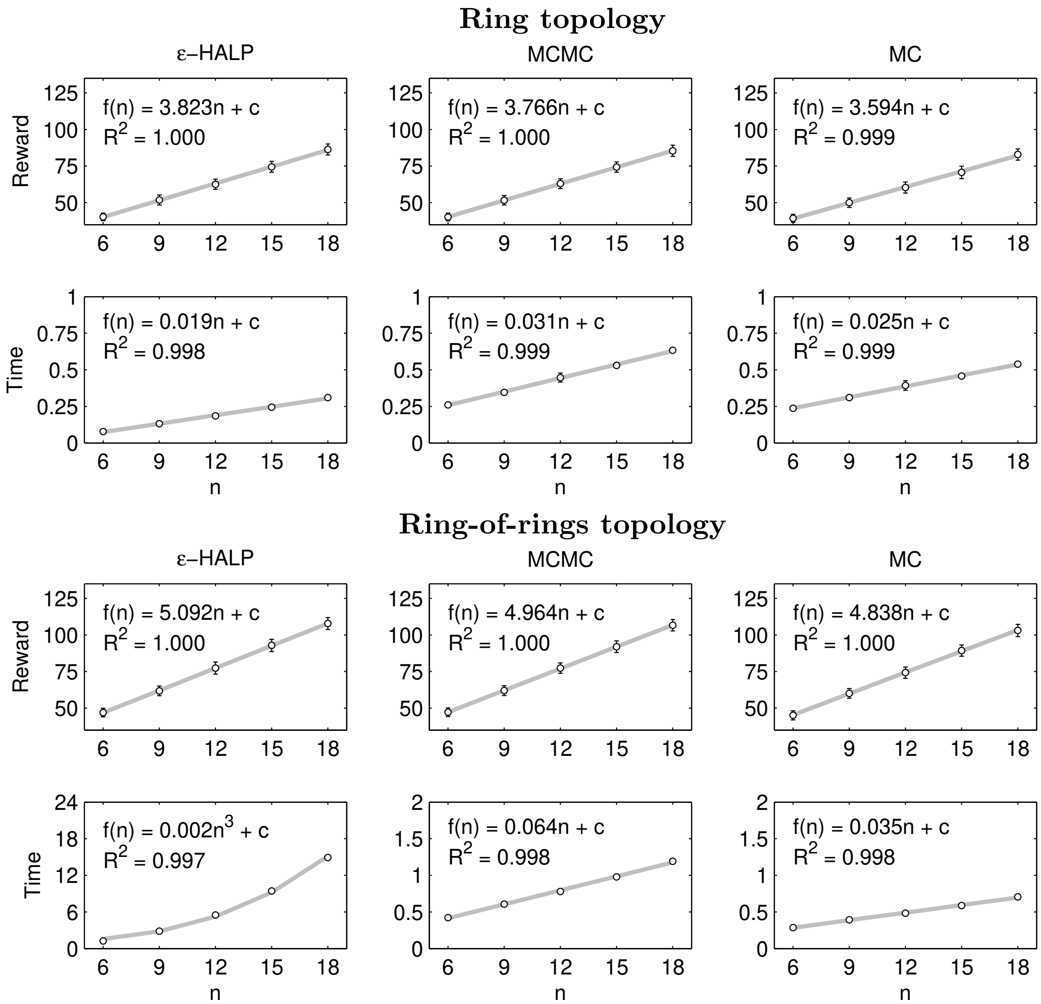


Figure 16: Scale-up potential of the ε-HALP, MCMC-HALP, and MC-HALP solvers on two irrigation network topologies of varying sizes (n). The graphs show the expected discounted reward of policies and their computation time (in hours) as functions of n. The HALP solvers are parameterized by the resolution of ε-grid ($\varepsilon = 1/16$), the number of MCMC chains ($N = 250$), and the number of samples ($N = 10^6$). Note that all trends can be approximated by a polynomial $f(n)$ (gray line) with a high degree of confidence (the coefficient of determination R^2), where c denotes a constant independent of n.

to the increased complexity of sampling $p(z_i^* \mid \mathbf{z}_{-i})$, which results from more complex local dependencies in the ring-of-rings topology and not its treewidth.

Before we proceed, note that our relaxed formulations (Figure 15) have significantly less constraints than their complete sets (Section 6.3). For instance, the MC-HALP formulation ($N = 10^6$) on the 6-ring irrigation network problem is originally established by 10^6 randomly sampled constraints. Based on our empirical results, the constraints can be satisfied greedily

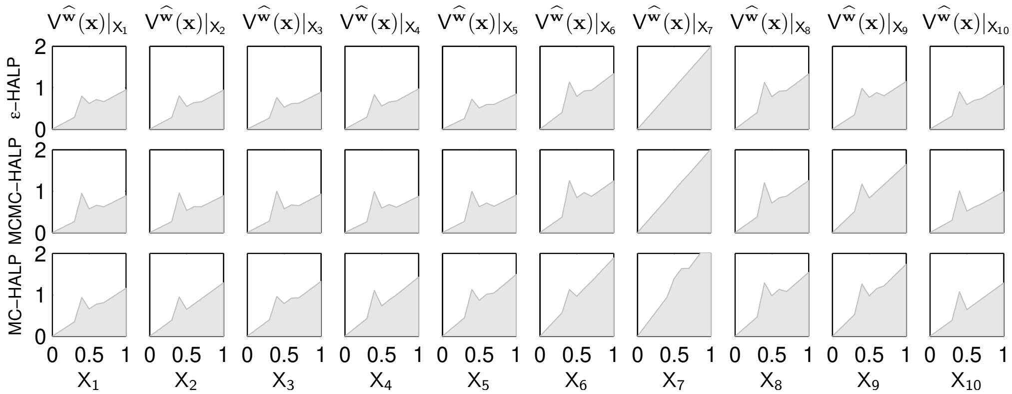


Figure 17: Univariate projections $V^{\widehat{\mathbf{w}}}(\mathbf{x})|_{X_j} = \sum_{i:X_j=X_i} \widehat{w}_i f_i(x_i)$ of approximate value functions $V^{\widehat{\mathbf{w}}}$ on the 6-ring irrigation network problem (Figure 13a). These functions are learned from 40 basis functions (Figure 14c) by the ε-HALP, MCMC-HALP, and MC-HALP solvers. The solvers are parameterized by the resolution of ε-grid ($\varepsilon = 1/16$), the number of MCMC chains ($N = 250$), and the number of samples ($N = 10^6$). Note that the univariate projections $V^{\widehat{\mathbf{w}}}(\mathbf{x})|_{X_j}$ are very similar. The proximity of their greedy policies can be explained based on this observation.

ε-HALP

ε	OV	Reward	Time
1	30.4	48.3 ± 3.0	9
1/2	42.9	58.7 ± 3.1	342
1/4	49.1	61.9 ± 3.1	9 443

MCMC

N	OV	Reward	Time
10	45.3	43.6 ± 6.5	83
50	116.2	72.2 ± 3.6	458
250	118.5	73.2 ± 3.7	2 012

MC

N	OV	Reward	Time
10^2	12.8	56.6 ± 4.5	< 1
10^4	49.9	53.4 ± 5.9	19
10^6	71.7	70.3 ± 3.9	1 400

Figure 18: Comparison of three HALP solvers on the 3×3 grid irrigation network problem (Figure 13). The solvers are compared by the objective value of a relaxed HALP (OV), the expected discounted reward of a corresponding policy, and computation time (in seconds). The ε-HALP, MCMC-HALP, and MC-HALP solvers are parameterized by the resolution of ε-grid (ε), the number of MCMC chains (N), and the number of samples (N). Note that the quality of policies improves with higher grid resolution ($1/\varepsilon$) and larger sample size (N). An upper bound on the expected returns is 87.2.

by a subset of 400 constraints on average (Kveton & Hauskrecht, 2004). Similarly, the oracle $\mathcal{O}_{\mathrm{MCMC}}$ in the MCMC-HALP formulation ($N = 250$) iterates through $250 \times 500 \times (10+10) = 2,500,000$ state-action configurations (Figure 11). However, corresponding LP formulations involve only 700 constraints on average.

7.3 The Curse of Treewidth

In the ring and ring-of-rings topologies, the treewidth of the constraint space (in continuous variables) is 2 and 3, respectively. As a result, the oracle $\mathcal{O}_\varepsilon$ can perform variable elimination for a small ε, and the ε-HALP solver returns close-to-optimal policies. Unfortunately, small treewidth is atypical in real-world domains. For instance, the treewidth of a more complex 3×3 grid irrigation network (Figure 13c) is 6. To perform variable elimination for $\varepsilon = 1/16$, the separation oracle $\mathcal{O}_\varepsilon$ requires the space of $\lceil 1/\varepsilon + 1 \rceil^7 \approx 2^{28}$, which is at the memory limit of existing PCs. To analyze the behavior of our separation oracles (Section 6) in this setting, we repeat our experiments from Section 7.2 on the 3×3 grid irrigation network.

Based on the results in Figure 18, we conclude that the time complexity of the ε-HALP solver grows by the rate of $\lceil 1/\varepsilon + 1 \rceil^7$. Therefore, approximate constraint space satisfaction (MC-HALP and MCMC-HALP) generates better results than a combinatorial optimization on an insufficiently discretized ε-grid (ε-HALP). This conclusion is parallel to those in large structured optimization problems with continuous variables. We believe that a combination of exact and approximate steps delivers the best tradeoff between the quality and complexity of our solutions (Section 6.4).

8. Conclusions

Development of scalable algorithms for solving real-world decision problems is a challenging task. In this paper, we presented a theoretically sound framework that allows for a compact representation and efficient solutions to hybrid factored MDPs. We believe that our results can be applied to a variety of optimization problems in robotics, manufacturing, or financial mathematics. This work can be extended in several interesting directions.

First, note that the concept of closed-form solutions to the expectations terms in HALP is not limited to the choices in Section 5.2. For instance, we can show that if $P(x)$ and $f(x)$ are normal densities, $\mathrm{E}_{P(x)}[f(x)]$ has a closed-form solution (Kveton & Hauskrecht, 2006b). Therefore, we can directly reason with normal transition functions instead of approximating them by a mixture of beta distributions. Similar conclusions are true for piecewise constant, piecewise linear, and gamma transition and basis functions. Note that our efficient solutions apply to any approach to solving hybrid factored MDPs that approximates the optimal value function by a linear combination of basis functions (Equation 5).

Second, the constraint space in HALP (16) $V^{\mathbf{w}} - \mathcal{T}^* V^{\mathbf{w}} \geq 0$ exhibits the same structure as the constraint space in approximate policy iteration (API) (Guestrin et al., 2001; Patrascu et al., 2002) $\left\| V^{\mathbf{w}} - \mathcal{T}^* V^{\mathbf{w}} \right\|_\infty \leq \varepsilon$, where ε is a variable subject to minimization. As a result, our work provides a recipe for solving API formulations in hybrid state and action domains. In discrete-state spaces, Patrascu et al. (2002) and Guestrin (2003) showed that API returns better policies than ALP for the same set of basis functions. Note that API is more complex than ALP because every step of API involves satisfying the constraint $\left\| V^{\mathbf{w}} - \mathcal{T}^* V^{\mathbf{w}} \right\|_\infty \leq \varepsilon$ for some fixed ε.

Third, automatic learning of basis functions seems critical for the application of HALP to real-world domains. Patrascu et al. (2002) analyzed this problem in discrete-state spaces and proposed a greedy approach to learning basis functions. Kveton and Hauskrecht (2006a) generalized these ideas and showed how to learn parametric basis functions in hybrid spaces. We believe that a combination of the greedy search with a state space analysis (Mahadevan, 2005; Mahadevan & Maggioni, 2006) can yield even better basis functions.

Finally, we proposed several bounds (Section 5.1 and 6.2.1) that may explain the quality of the complete and relaxed HALP formulations. In future, we plan to empirically evaluate their tightness on a variety of low-dimensional hybrid optimization problems (Bresina et al., 2002; Munos & Moore, 2002) with known optimal value functions.

Acknowledgment

This work was supported in part by National Science Foundation grants CMS-0416754 and ANI-0325353. The first author was supported by Andrew Mellon Predoctoral Fellowships for the academic years 2004-06. The first author also recognizes support from Intel Corporation in the summer 2005 and 2006.

Appendix A. Proofs

Proof of Proposition 1: The Bellman operator $\mathcal{T}^*$ is known to be a contraction mapping. Based on its monotonicity, for any value function V, $V \geq \mathcal{T}^* V$ implies $V \geq \mathcal{T}^* V \geq \cdots \geq V^*$. Since constraints in the HALP formulation (16) enforce $V^{\widetilde{\mathbf{w}}} \geq \mathcal{T}^* V^{\widetilde{\mathbf{w}}}$, we conclude $V^{\widetilde{\mathbf{w}}} \geq V^*$. □

Proof of Proposition 2: Based on Proposition 1, we note that the constraint $V^{\mathbf{w}} \geq \mathcal{T}^* V^{\mathbf{w}}$ guarantees that $V^{\mathbf{w}} \geq V^*$. Subsequently, our claim is proved by realizing:

$$\arg\min_{\mathbf{w}} \mathrm{E}_\psi[V^{\mathbf{w}}] = \arg\min_{\mathbf{w}} \mathrm{E}_\psi[V^{\mathbf{w}} - V^*]$$

and

$$\begin{aligned} \mathrm{E}_\psi[V^{\mathbf{w}} - V^*] &= \mathrm{E}_\psi|V^{\mathbf{w}} - V^*| \\ &= \mathrm{E}_\psi|V^* - V^{\mathbf{w}}| \\ &= \left\| V^* - V^{\mathbf{w}} \right\|_{1,\psi}. \end{aligned}$$

The proof generalizes from the discrete-state case (de Farias & Van Roy, 2003) without any alternations. □

Proof of Theorem 2: Similarly to Theorem 2 (de Farias & Van Roy, 2003), this claim is proved in three steps. First, we find a point $\overline{\mathbf{w}}$ in the feasible region of the HALP such that $V^{\overline{\mathbf{w}}}$ is within $O(\epsilon)$ distance from $V^{\mathbf{w}^*}$, where:

$$\begin{aligned} \mathbf{w}^* &= \arg\min_{\mathbf{w}} \left\| V^* - V^{\mathbf{w}} \right\|_\infty \\ \epsilon &= \left\| V^* - V^{\mathbf{w}^*} \right\|_\infty. \end{aligned}$$

Such a point $\overline{\mathbf{w}}$ is given by:

$$\overline{\mathbf{w}} = \mathbf{w}^* + \frac{(1+\gamma)\epsilon}{1-\gamma} e,$$

where $e = (1, 0, \ldots, 0)$ is an indicator of the constant basis function $f_0(\mathbf{x}) \equiv 1$. This point satisfies all requirements and its feasibility can be handily verified by solving:

$$\begin{aligned} V^{\overline{\mathbf{w}}} - \mathcal{T}^* V^{\overline{\mathbf{w}}} &= V^{\mathbf{w}^*} + \frac{(1+\gamma)\epsilon}{1-\gamma} - \left(\mathcal{T}^* V^{\mathbf{w}^*} + \frac{\gamma(1+\gamma)\epsilon}{1-\gamma} \right) \\ &= V^{\mathbf{w}^*} - \mathcal{T}^* V^{\mathbf{w}^*} + (1+\gamma)\epsilon \\ &\geq 0, \end{aligned}$$

where the last step follows from the inequality:

$$\begin{aligned}
\left\| V^{\mathbf{w}^*} - \mathcal{T}^* V^{\mathbf{w}^*} \right\|_\infty &\leq \left\| V^{\mathbf{w}^*} - V^* \right\|_\infty + \left\| V^* - \mathcal{T}^* V^{\mathbf{w}^*} \right\|_\infty \\
&= \left\| V^* - V^{\mathbf{w}^*} \right\|_\infty + \left\| \mathcal{T}^* V^* - \mathcal{T}^* V^{\mathbf{w}^*} \right\|_\infty \\
&\leq (1+\gamma)\epsilon.
\end{aligned}$$

Subsequently, we bound the max-norm error of $V^{\overline{\mathbf{w}}}$ by using the triangle inequality:

$$\begin{aligned}
\left\| V^* - V^{\overline{\mathbf{w}}} \right\|_\infty &\leq \left\| V^* - V^{\mathbf{w}^*} \right\|_\infty + \left\| V^{\mathbf{w}^*} - V^{\overline{\mathbf{w}}} \right\|_\infty \\
&= \left(1 + \frac{1+\gamma}{1-\gamma}\right)\epsilon \\
&= \frac{2\epsilon}{1-\gamma},
\end{aligned}$$

which yields a bound on the weighted $\mathcal{L}_1$-norm error of $V^{\widetilde{\mathbf{w}}}$:

$$\begin{aligned}
\left\| V^* - V^{\widetilde{\mathbf{w}}} \right\|_{1,\psi} &\leq \left\| V^* - V^{\overline{\mathbf{w}}} \right\|_{1,\psi} \\
&\leq \left\| V^* - V^{\overline{\mathbf{w}}} \right\|_\infty \\
&\leq \frac{2\epsilon}{1-\gamma}.
\end{aligned}$$

The proof generalizes from the discrete-state case (de Farias & Van Roy, 2003) without any alternations. □

Proof of Theorem 3: Similarly to Theorem 2, this claim is proved in three steps: finding a point $\overline{\mathbf{w}}$ in the feasible region of the HALP, bounding the max-norm error of $V^{\overline{\mathbf{w}}}$, which in turn yields a bound on the $\mathcal{L}_1$-norm error of $V^{\widetilde{\mathbf{w}}}$. A comprehensive proof for the discrete-state case was done by de Farias and Van Roy (2003). This proof generalizes to structured state and action spaces with continuous variables. □

Proof of Proposition 3: The proposition is proved in a sequence of steps:

$$\begin{aligned}
\mathrm{E}_{P(x)}[f(x)] &= \int_x P_{\mathrm{beta}}(x \mid \alpha, \beta) x^n (1-x)^m \, \mathrm{d}x \\
&= \int_x \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)} x^{\alpha-1}(1-x)^{\beta-1} x^n (1-x)^m \, \mathrm{d}x \\
&= \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)} \int_x x^{\alpha+n-1}(1-x)^{\beta+m-1} \, \mathrm{d}x \\
&= \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)} \frac{\Gamma(\alpha+n)\Gamma(\beta+m)}{\Gamma(\alpha+\beta+n+m)} \int_x \frac{\Gamma(\alpha+\beta+n+m)}{\Gamma(\alpha+n)\Gamma(\beta+m)} x^{\alpha+n-1}(1-x)^{\beta+m-1} \, \mathrm{d}x \\
&= \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)} \frac{\Gamma(\alpha+n)\Gamma(\beta+m)}{\Gamma(\alpha+\beta+n+m)} \underbrace{\int_x P_{\mathrm{beta}}(x \mid \alpha+n, \beta+m) \, \mathrm{d}x}_{1}.
\end{aligned}$$

Since integration is a distributive operation, our claim straightforwardly generalizes to the mixture of beta distributions $P(x)$. $\square$

Proof of Proposition 4: The proposition is proved in a sequence of steps:

$$\begin{aligned}
\mathrm{E}_{P(x)}[f(x)] &= \int_x P_{\mathrm{beta}}(x \mid \alpha, \beta) \sum_i \mathbf{1}_{[l_i, r_i]}(x)(a_i x + b_i)\,\mathrm{d}x \\
&= \sum_i \int_{l_i}^{r_i} P_{\mathrm{beta}}(x \mid \alpha, \beta)(a_i x + b_i)\,\mathrm{d}x \\
&= \sum_i \left[a_i \int_{l_i}^{r_i} P_{\mathrm{beta}}(x \mid \alpha, \beta) x\,\mathrm{d}x + b_i \int_{l_i}^{r_i} P_{\mathrm{beta}}(x \mid \alpha, \beta)\,\mathrm{d}x\right] \\
&= \sum_i \left[a_i \frac{\alpha}{\alpha + \beta} \int_{l_i}^{r_i} P_{\mathrm{beta}}(x \mid \alpha + 1, \beta)\,\mathrm{d}x + b_i \int_{l_i}^{r_i} P_{\mathrm{beta}}(x \mid \alpha, \beta)\,\mathrm{d}x\right] \\
&= \sum_i \left[a_i \frac{\alpha}{\alpha + \beta}(F^+(r_i) - F^+(l_i)) + b_i(F(r_i) - F(l_i))\right].
\end{aligned}$$

Since integration is a distributive operation, our claim straightforwardly generalizes to the mixture of beta distributions $P(x)$. $\square$

Proof of Proposition 5: This claim is proved in three steps. First, we construct a point $\overline{\mathbf{w}}$ in the feasible region of the HALP such that $V^{\overline{\mathbf{w}}}$ is within $O(\delta)$ distance from $V^{\widehat{\mathbf{w}}}$. Such a point $\overline{\mathbf{w}}$ is given by:

$$\overline{\mathbf{w}} = \widehat{\mathbf{w}} + \frac{\delta}{1 - \gamma} e,$$

where $e = (1, 0, \ldots, 0)$ is an indicator of the constant basis function $f_0(\mathbf{x}) \equiv 1$. This point satisfies all requirements and its feasibility can be handily verified by solving:

$$\begin{aligned}
V^{\overline{\mathbf{w}}} - \mathcal{T}^* V^{\overline{\mathbf{w}}} &= V^{\widehat{\mathbf{w}}} + \frac{\delta}{1 - \gamma} - \left(\mathcal{T}^* V^{\widehat{\mathbf{w}}} + \frac{\gamma\delta}{1 - \gamma}\right) \\
&= V^{\widehat{\mathbf{w}}} - \mathcal{T}^* V^{\widehat{\mathbf{w}}} + \delta \\
&\geq 0,
\end{aligned}$$

where the inequality $V^{\widehat{\mathbf{w}}} - \mathcal{T}^* V^{\widehat{\mathbf{w}}} \geq -\delta$ holds from the δ-infeasibility of $\widehat{\mathbf{w}}$. Since the optimal solution $\widetilde{\mathbf{w}}$ is feasible in the relaxed HALP, we conclude $\mathrm{E}_\psi\left[V^{\widehat{\mathbf{w}}}\right] \leq \mathrm{E}_\psi\left[V^{\widetilde{\mathbf{w}}}\right]$. Subsequently, this inequality yields a bound on the weighted $\mathcal{L}_1$-norm error of $V^{\overline{\mathbf{w}}}$:

$$\begin{aligned}
\left\|V^* - V^{\overline{\mathbf{w}}}\right\|_{1,\psi} &= \mathrm{E}_\psi\left|V^{\widehat{\mathbf{w}}} + \frac{\delta}{1 - \gamma} - V^*\right| \\
&= \mathrm{E}_\psi\left[V^{\widehat{\mathbf{w}}}\right] + \frac{\delta}{1 - \gamma} - \mathrm{E}_\psi[V^*] \\
&\leq \mathrm{E}_\psi\left[V^{\widetilde{\mathbf{w}}}\right] + \frac{\delta}{1 - \gamma} - \mathrm{E}_\psi[V^*] \\
&= \left\|V^* - V^{\widetilde{\mathbf{w}}}\right\|_{1,\psi} + \frac{\delta}{1 - \gamma}.
\end{aligned}$$

Finally, we combine this result with the triangle inequality:

$$\begin{aligned}\left\|V^* - V^{\widehat{\mathbf{w}}}\right\|_{1,\psi} &\leq \left\|V^* - V^{\overline{\mathbf{w}}}\right\|_{1,\psi} + \left\|V^{\overline{\mathbf{w}}} - V^{\widehat{\mathbf{w}}}\right\|_{1,\psi} \\ &\leq \left\|V^* - V^{\widetilde{\mathbf{w}}}\right\|_{1,\psi} + \frac{2\delta}{1-\gamma},\end{aligned}$$

which leads to a bound on the weighted $\mathcal{L}_1$-norm error of $V^{\widehat{\mathbf{w}}}$. $\square$

References

Andrieu, C., de Freitas, N., Doucet, A., & Jordan, M. (2003). An introduction to MCMC for machine learning. *Machine Learning*, *50*, 5–43.

Astrom, K. (1965). Optimal control of Markov processes with incomplete state information. *Journal of Mathematical Analysis and Applications*, *10*(1), 174–205.

Bellman, R. (1957). *Dynamic Programming.* Princeton University Press, Princeton, NJ.

Bellman, R., Kalaba, R., & Kotkin, B. (1963). Polynomial approximation – a new computational technique in dynamic programming: Allocation processes. *Mathematics of Computation*, *17*(82), 155–161.

Bertsekas, D. (1995). A counterexample for temporal differences learning. *Neural Computation*, *7*(2), 270–279.

Bertsekas, D., & Tsitsiklis, J. (1996). *Neuro-Dynamic Programming.* Athena Scientific, Belmont, MA.

Bertsimas, D., & Tsitsiklis, J. (1997). *Introduction to Linear Optimization.* Athena Scientific, Belmont, MA.

Boutilier, C., Dearden, R., & Goldszmidt, M. (1995). Exploiting structure in policy construction. In *Proceedings of the 14th International Joint Conference on Artificial Intelligence*, pp. 1104–1111.

Bresina, J., Dearden, R., Meuleau, N., Ramakrishnan, S., Smith, D., & Washington, R. (2002). Planning under continuous time and resource uncertainty: A challenge for AI. In *Proceedings of the 18th Conference on Uncertainty in Artificial Intelligence*, pp. 77–84.

Casella, G., & Robert, C. (1996). Rao-Blackwellisation of sampling schemes. *Biometrika*, *83*(1), 81–94.

Chow, C.-S., & Tsitsiklis, J. (1991). An optimal one-way multigrid algorithm for discrete-time stochastic control. *IEEE Transactions on Automatic Control*, *36*(8), 898–914.

Cooper, G. (1988). A method for using belief networks as influence diagrams. In *Proceedings of the Workshop on Uncertainty in Artificial Intelligence*, pp. 55–63.

Crites, R., & Barto, A. (1996). Improving elevator performance using reinforcement learning. In *Advances in Neural Information Processing Systems 8*, pp. 1017–1023.

de Farias, D. P., & Van Roy, B. (2003). The linear programming approach to approximate dynamic programming. *Operations Research*, *51*(6), 850–856.

de Farias, D. P., & Van Roy, B. (2004). On constraint sampling for the linear programming approach to approximate dynamic programming. *Mathematics of Operations Research*, *29*(3), 462–478.

Dean, T., & Kanazawa, K. (1989). A model for reasoning about persistence and causation. *Computational Intelligence*, *5*, 142–150.

Dechter, R. (1996). Bucket elimination: A unifying framework for probabilistic inference. In *Proceedings of the 12th Conference on Uncertainty in Artificial Intelligence*, pp. 211–219.

Duane, S., Kennedy, A. D., Pendleton, B., & Roweth, D. (1987). Hybrid Monte Carlo. *Physics Letters B*, *195*(2), 216–222.

Feng, Z., Dearden, R., Meuleau, N., & Washington, R. (2004). Dynamic programming for structured continuous Markov decision problems. In *Proceedings of the 20th Conference on Uncertainty in Artificial Intelligence*, pp. 154–161.

Ferns, N., Panangaden, P., & Precup, D. (2005). Metrics for Markov decision processes with infinite state spaces. In *Proceedings of the 21st Conference on Uncertainty in Artificial Intelligence*.

Geman, S., & Geman, D. (1984). Stochastic relaxation, Gibbs distribution, and the Bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *6*(6), 721–741.

Gordon, G. (1999). *Approximate Solutions to Markov Decision Processes*. Ph.D. thesis, Carnegie Mellon University.

Guestrin, C. (2003). *Planning Under Uncertainty in Complex Structured Environments*. Ph.D. thesis, Stanford University.

Guestrin, C., Hauskrecht, M., & Kveton, B. (2004). Solving factored MDPs with continuous and discrete variables. In *Proceedings of the 20th Conference on Uncertainty in Artificial Intelligence*, pp. 235–242.

Guestrin, C., Koller, D., Gearhart, C., & Kanodia, N. (2003). Generalizing plans to new environments in relational MDPs. In *Proceedings of the 18th International Joint Conference on Artificial Intelligence*, pp. 1003–1010.

Guestrin, C., Koller, D., & Parr, R. (2001). Max-norm projections for factored MDPs. In *Proceedings of the 17th International Joint Conference on Artificial Intelligence*, pp. 673–682.

Guestrin, C., Koller, D., & Parr, R. (2002). Multiagent planning with factored MDPs. In *Advances in Neural Information Processing Systems 14*, pp. 1523–1530.

Guestrin, C., Koller, D., Parr, R., & Venkataraman, S. (2003). Efficient solution algorithms for factored MDPs. *Journal of Artificial Intelligence Research*, *19*, 399–468.

Guestrin, C., Venkataraman, S., & Koller, D. (2002). Context specific multiagent coordination and planning with factored MDPs. In *Proceedings of the 18th National Conference on Artificial Intelligence*, pp. 253–259.

Hastings, W. K. (1970). Monte Carlo sampling methods using Markov chains and their application. *Biometrika*, *57*, 97–109.

Hauskrecht, M. (2000). Value-function approximations for partially observable Markov decision processes. *Journal of Artificial Intelligence Research*, *13*, 33–94.

Hauskrecht, M., & Kveton, B. (2004). Linear program approximations for factored continuous-state Markov decision processes. In *Advances in Neural Information Processing Systems 16*, pp. 895–902.

Higdon, D. (1998). Auxiliary variable methods for Markov chain Monte Carlo with applications. *Journal of the American Statistical Association*, *93*(442), 585–595.

Howard, R., & Matheson, J. (1984). Influence diagrams. In *Readings on the Principles and Applications of Decision Analysis*, Vol. 2, pp. 719–762. Strategic Decisions Group, Menlo Park, CA.

Jeffreys, H., & Jeffreys, B. (1988). *Methods of Mathematical Physics*. Cambridge University Press, Cambridge, United Kingdom.

Jensen, F., Jensen, F., & Dittmer, S. (1994). From influence diagrams to junction trees. In *Proceedings of the 10th Conference on Uncertainty in Artificial Intelligence*, pp. 367–373.

Khachiyan, L. (1979). A polynomial algorithm in linear programming. *Doklady Akademii Nauk SSSR*, *244*, 1093–1096.

Kirkpatrick, S., Gelatt, C. D., & Vecchi, M. P. (1983). Optimization by simulated annealing. *Science*, *220*(4598), 671–680.

Koller, D., & Parr, R. (1999). Computing factored value functions for policies in structured MDPs. In *Proceedings of the 16th International Joint Conference on Artificial Intelligence*, pp. 1332–1339.

Kveton, B., & Hauskrecht, M. (2004). Heuristic refinements of approximate linear programming for factored continuous-state Markov decision processes. In *Proceedings of the 14th International Conference on Automated Planning and Scheduling*, pp. 306–314.

Kveton, B., & Hauskrecht, M. (2005). An MCMC approach to solving hybrid factored MDPs. In *Proceedings of the 19th International Joint Conference on Artificial Intelligence*, pp. 1346–1351.

Kveton, B., & Hauskrecht, M. (2006a). Learning basis functions in hybrid domains. In *Proceedings of the 21st National Conference on Artificial Intelligence*, pp. 1161–1166.

Kveton, B., & Hauskrecht, M. (2006b). Solving factored MDPs with exponential-family transition models. In *Proceedings of the 16th International Conference on Automated Planning and Scheduling*, pp. 114–120.

Mahadevan, S. (2005). Samuel meets Amarel: Automating value function approximation using global state space analysis. In *Proceedings of the 20th National Conference on Artificial Intelligence*, pp. 1000–1005.

Mahadevan, S., & Maggioni, M. (2006). Value function approximation with diffusion wavelets and Laplacian eigenfunctions. In *Advances in Neural Information Processing Systems 18*, pp. 843–850.

Mahadevan, S., Maggioni, M., Ferguson, K., & Osentoski, S. (2006). Learning representation and control in continuous Markov decision processes. In *Proceedings of the 21st National Conference on Artificial Intelligence*.

Manne, A. (1960). Linear programming and sequential decisions. *Management Science*, *6*(3), 259–267.

Metropolis, N., Rosenbluth, A., Rosenbluth, M., Teller, A., & Teller, E. (1953). Equation of state calculations by fast computing machines. *Journal of Chemical Physics*, *21*, 1087–1092.

Munos, R., & Moore, A. (2002). Variable resolution discretization in optimal control. *Machine Learning*, *49*, 291–323.

Ortiz, L. (2002). *Selecting Approximately-Optimal Actions in Complex Structured Domains*. Ph.D. thesis, Brown University.

Park, J., & Darwiche, A. (2001). Approximating MAP using local search. In *Proceedings of the 17th Conference on Uncertainty in Artificial Intelligence*, pp. 403–410.

Park, J., & Darwiche, A. (2003). Solving MAP exactly using systematic search. In *Proceedings of the 19th Conference on Uncertainty in Artificial Intelligence*, pp. 459–468.

Patrascu, R., Poupart, P., Schuurmans, D., Boutilier, C., & Guestrin, C. (2002). Greedy linear value-approximation for factored Markov decision processes. In *Proceedings of the 18th National Conference on Artificial Intelligence*, pp. 285–291.

Puterman, M. (1994). *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley & Sons, New York, NY.

Rust, J. (1997). Using randomization to break the curse of dimensionality. *Econometrica*, *65*(3), 487–516.

Sanner, S., & Boutilier, C. (2005). Approximate linear programming for first-order MDPs. In *Proceedings of the 21st Conference on Uncertainty in Artificial Intelligence*.

Schuurmans, D., & Patrascu, R. (2002). Direct value-approximation for factored MDPs. In *Advances in Neural Information Processing Systems 14*, pp. 1579–1586.

Schweitzer, P., & Seidmann, A. (1985). Generalized polynomial approximations in Markovian decision processes. *Journal of Mathematical Analysis and Applications*, *110*, 568–582.

Sondik, E. (1971). *The Optimal Control of Partially Observable Markov Decision Processes*. Ph.D. thesis, Stanford University.

Sutton, R., & Barto, A. (1998). *Reinforcement Learning: An Introduction*. MIT Press, Cambridge, MA.

Tesauro, G. (1992). Practical issues in temporal difference learning. *Machine Learning*, *8*(3-4), 257–277.

Tesauro, G. (1994). TD-Gammon, a self-teaching backgammon program, achieves master-level play. *Neural Computation*, *6*(2), 215–219.

Tesauro, G. (1995). Temporal difference learning and TD-Gammon. *Communications of the ACM*, *38*(3), 58–68.

Trick, M., & Zin, S. (1993). A linear programming approach to solving stochastic dynamic programs. Tech. rep., Carnegie Mellon University.

Van Roy, B. (1998). *Planning Under Uncertainty in Complex Structured Environments*. Ph.D. thesis, Massachusetts Institute of Technology.

Yuan, C., Lu, T.-C., & Druzdzel, M. (2004). Annealed MAP. In *Proceedings of the 20th Conference on Uncertainty in Artificial Intelligence*, pp. 628–635.

Zhang, W., & Dietterich, T. (1995). A reinforcement learning approach to job-shop scheduling. In *Proceedings of the 14th International Joint Conference on Artificial Intelligence*, pp. 1114–1120.

Zhang, W., & Dietterich, T. (1996). High-performance job-shop scheduling with a time-delay TD(λ) network. In *Advances in Neural Information Processing Systems 8*, pp. 1024–1030.