Probabilistic Inference for Solving Discrete and Continuous State Markov Decision Processes

Marc Toussaint MTOUSSAI@INF.ED.AC.UK
Amos Storkey A.STORKEY@ED.AC.UK
School of Informatics, University of Edinburgh, 5 Forrest Hill, Edinburgh EH1 2QL, UK

Abstract

Inference in Markov Decision Processes has recently received interest as a means to infer goals of an observed action, policy recognition, and also as a tool to compute policies. A particularly interesting aspect of the approach is that any existing inference technique in DBNs now becomes available for answering behavioral questions–including those on continuous, factorial, or hierarchical state representations. Here we present an Expectation Maximization algorithm for computing optimal policies. Unlike previous approaches we can show that this actually optimizes the discounted expected future return for arbitrary reward functions and without assuming an ad hoc finite total time. The algorithm is generic in that any inference technique can be utilized in the E-step. We demonstrate this for exact inference on a discrete maze and Gaussian belief state propagation in continuous stochastic optimal control problems.

1. Introduction

The problems of planning in stochastic environments and inference in Markovian models are closely related, in particular in view of the challenges both of them face: e.g., coping with very large state spaces spanned by multiple state variables, or realizing planning (or inference) in continuous state spaces. Both fields developed techniques to address these problems. For instance, in the field of planning, they include work on Factored Markov Decision Processes (Boutilier et al., 1995; Koller & Parr, 1999; Guestrin et al., 2003; Kveton & Hauskrecht, 2005) or abstractions (Hauskrecht et al., 1998). On the other hand, in the field of probabilistic inference, techniques for approximate inference on factorial latent representations (e.g., Factorial Hidden Markov Models Ghahramani & Jordan, 1995) and a large amount of work on approximate inference in continuous state spaces does exist (ranging from particle filters to, e.g., Assumed Density Filtering; see, e.g., (Minka, 2001) for an overview).

In view of these similarities one may ask whether approaches to probabilistic inference can *exactly* be transferred to the problem of planning, in other words, whether one can translate the problem of planning exactly into a problem of inference. Clearly, the aim of this is to connect both fields more strongly but eventually also to apply, e.g., efficient methods of probabilistic inference directly in the realm of planning.

Bui et al. (2002) have used inference on Abstract Hidden Markov Models for policy recognition, i.e., for reasoning about executed behaviors, but do not address the problem of computing optimal policies from such inference. Attias (2003) recently proposed a framework which suggests a straight-forward way to translate the problem of planning to a problem of inference: A Markovian state-action model is assumed, which is conditioned on a start state $x_0 = A$ and a goal state $x_T = B$. Here, however, the total time T has to be fixed *ad hoc* and the MAP action sequence that is proposed as a solution is not optimal in the sense of maximizing an expected future reward. Thirdly, Verma and Rao (2006) used inference to compute plans (considering the maximal probable explanation (MPE) instead of the MAP action sequence) but again the total time has to be fixed and the plan is not optimal in the expected return sense. In this paper our contribution is to (1) provide a framework that translates the problem of maximizing the expected future return exactly into a problem of likelihood maximization in a latent variable mixture model, for arbitrary reward functions and without assuming a fixed time, (2) demonstrate the approach first on a discrete maze problem using exact

Appearing in *Proceedings of the 23rd International Conference on Machine Learning*, Pittsburgh, PA, 2006.

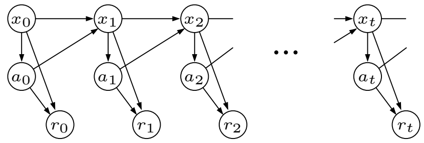

Figure 1. Dynamic Bayesian Network for a MDP. The x states denote the state variables, a the actions and r the rewards.

inference in the E-step, then on a continuous stochastic optimal control problem assuming Gaussian belief state representations and using the unscented transform to handle non-linear dynamics.

The key step to our approach is to introduce a mixture of finite-time MDPs as a model that is equivalent to the original time-unbounded MDP but has two advantages: it allows us to formulate a likelihood proportional to the expected future return, and inference in the mixture model can efficiently be realized with a single synchronous forward-backward propagation without having to fix a finite total time in advance. The next three sections introduce this framework, show equivalence to maximizing expected future rewards, and describe an EM-algorithm to compute optimal policies. Section 5 explains the intimate relation to Policy Iteration and the examples in Section 6 and 7 demonstrate the feasibility of this approach.

2. Markov Decision Processes

Figure 1 displays the Dynamic Bayesian Network for a time-unlimited Markov Decision Process (MDP), which is defined by the state transition probability $P(x_{t+1} \,|\, a_t, x_t)$, the action probability $P(a_t \,|\, x_t \,; \boldsymbol{\pi})$, and the reward probability $P(r_t \,|\, a_t, x_t)$. The random variables x and a can be discrete or continuous whereas the reward variable is, without loss of generality, assumed to be binary, $r_t \in \{0, 1\}$.[1] Throughout this paper we assume that the transition and reward probabilities are given, i.e. known a priori. Such probabilities can also be estimated from experience, but this is not addresses here.

The action probability $P(a_t \,|\, x_t \,; \boldsymbol{\pi})$ is directly parameterized by an unknown policy $\boldsymbol{\pi}$ such that $P(a_t = a \,|\, x_t = i \,; \boldsymbol{\pi}) = \pi_{ai}$, the numbers $\pi_{ai} \in [0, 1]$ are normalized w.r.t. a. The problem is to *solve the MDP*, i.e. to find a policy that maximizes the expected future return:

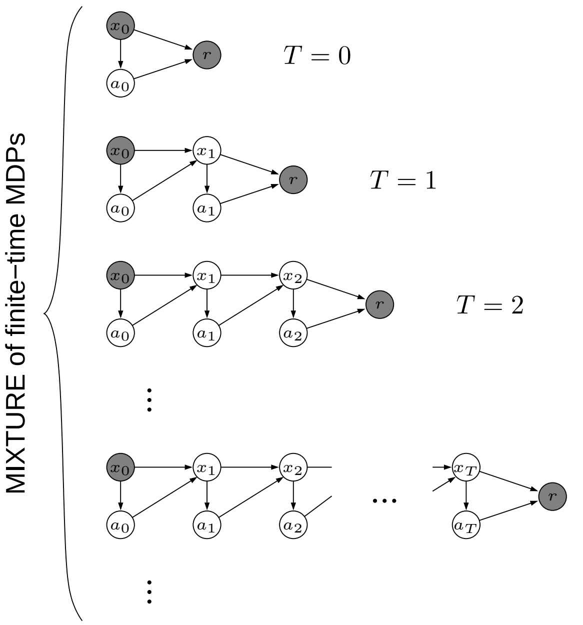

Figure 2. Mixture of finite-time MDPs.

Definition 2.1 *Solving an MDP means to find a parameter* $\boldsymbol{\pi}$ *of the graphical model in Figure 1 that maximizes the expected future return* $V^{\boldsymbol{\pi}}(i) = \mathrm{E}\,\{\sum_{t=0}^{\infty} \gamma^t\ r_t \,|\, x_0 \!=\! i \,; \boldsymbol{\pi}\}$, *where* $\gamma \in [0, 1]$ *is a discount factor.*

The classical approach to solving MDPs is anchored in Bellman's equation, which simply reflects the recursive property of the future discounted return $R_T = \sum_{t=T}^{\infty} \gamma^t\ r_t = r_T + \gamma\ R_{T+1}$ and consequently of its expectation conditioned on the current state, $V^{\boldsymbol{\pi}}(i) = \sum_{j,a} P(j \,|\, a, i)\ P(a \,|\, i \,; \boldsymbol{\pi})\ [P(r_t = 1 \,|\, a, i) + \gamma\ V^{\boldsymbol{\pi}}(j)]$. Standard algorithms for computing value functions can be viewed as iterative schemes that converge towards the Bellman equation. We discuss relations in section 5.

3. Mixture of MDPs and likelihoods

Our approach is to cast the problem of solving an MDP into a problem of optimizing the parameters of a graphical model with many hidden variables, namely all future states and actions. The only observables are the current state and rewards. To achieve exact equivalence between solving the MDP and standard likelihood maximization we need to define a likelihood that is proportional to the expected future return. This is most easily done by formally considering a *mixture of finite-time MDPs*. By a finite-time MDP we mean one which is limited in time by T and which emits a reward variable only at the very final time step, as il-

[1] Assuming binary reward variables is sufficient to associate arbitrary reward expectations $P(r_t \,|\, a_t, x_t)$ in the interval $[0, 1]$ to states and actions, which is, modulo rescaling, the general case in Reinforcement Learning scenarios.

lustrated for various T in Figure 2. The full joint for a finite-time MDP is

$$P(r, x_{0:T}, a_{0:T} \,|\, T ; \boldsymbol{\pi}) = P(r \,|\, a_T, x_T)\ P(a_0 \,|\, x_0 ; \boldsymbol{\pi})\ P(x_0) \cdot \prod_{t=1}^{T} P(a_t \,|\, x_t ; \boldsymbol{\pi})\ P(x_t \,|\, a_{t-1}, x_{t-1})\ . \qquad (1)$$

The full mixture of finite-time MDPs is then given by the joint

$$P(r, x_{0:T}, a_{0:T}, T ; \boldsymbol{\pi}) = P(r, x_{0:T}, a_{0:T} \,|\, T ; \boldsymbol{\pi})\ P(T)\ . \qquad (2)$$

Here, $P(T)$ is a prior over the total time, which we choose proportional to the discounting, $P(T) = \gamma^T\ (1-\gamma)$. Note that each finite-time MDP shares the same transition probabilities and is parameterized by the same policy $\boldsymbol{\pi}$.

Now we can consider the reward variable r as observations, condition on the start state, and define the likelihood for a single finite-time MDP,

$$L^{\pi}_T(i) = P(r{=}1 \,|\, x_0{=}i, T ; \boldsymbol{\pi}) = \mathrm{E}\,\{r \,|\, x_0{=}i, T ; \boldsymbol{\pi}\}\ , \qquad (3)$$

and for the full mixture of MDPs,

$$L^{\pi}(i) = P(r{=}1 \,|\, x_0 = i ; \boldsymbol{\pi}) = \sum_T P(T)\ \mathrm{E}\,\{r \,|\, x_0{=}i, T ; \boldsymbol{\pi}\}\ . \qquad (4)$$

This likelihood is, for the discounted time prior $P(T) = \gamma^T\ (1-\gamma)$, proportional to the expected future return,[2]

$$L^{\pi}(i) = (1-\gamma)\ V^{\pi}(i)\ . \qquad (5)$$

Hence we have established

Theorem 3.1 *Maximizing the likelihood (4) in the mixture of finite-time MDPs (Figure 2) is equivalent to solving the MDP (definition 2.1).*

Besides establishing exact equivalence between likelihood and expected future return maximization, considering the mixture of finite-time models has a second advantage: the finite-time property of every mixture component makes the E-step in the full mixture model rather simple and efficient, as detailed in the next section.

4. An EM-algorithm for computing the optimal policy

Formulating the objective function in terms of a likelihood allows us to apply Expectation-Maximization to find optimal parameters (the policy $\boldsymbol{\pi}$) of our model. All action and state variables (except for x_0) are hidden variables. The E-step will, for a given $\boldsymbol{\pi}$, compute posteriors over state-action sequences as well as T conditioned on $x_0 = A$ and $r = 1$. The M-step then adapts the model parameters $\boldsymbol{\pi}$ to optimize the expected likelihood (expectations then taken w.r.t. the posteriors calculated in the E-step). Conceptually, the E-step in this Markovian model is straight-forward. However, the special structure of the finite-time MDPs will allow for certain simplifications and save us from performing separate inference sweeps in all finite-time MDPs.

E-step: forward-backward in all MDPs synchronously. Since we assume the transition probabilities to be stationary, we may use the simpler notations $p(j|a,i) = P(x_{t+1} = j \,|\, a_t = a, x_t = i)$ and $p(j|i;\boldsymbol{\pi}) = P(x_{t+1}{=}j \,|\, x_t = i ; \boldsymbol{\pi}) = \sum_a p(j|a,i)\ \pi_{ai}$. Further, as a "seed" for backward propagation, we define

$$\widehat{\beta}(i) = P(r{=}1 \,|\, x_T{=}i ; \boldsymbol{\pi}) = \sum_a P(r{=}1 \,|\, a_T{=}a, x_T{=}i)\ \pi_{ai}\ . \qquad (6)$$

In the E-step, we consider a fixed given policy $\boldsymbol{\pi}$ and all the quantities we compute depend on $\boldsymbol{\pi}$ even if not explicitly annotated. For a single MDP of finite time T the standard forward and backward propagation computes

$$\alpha_0(i) = \delta_{i=A}\ , \quad \alpha_t(i) = P(x_t{=}i \,|\, x_0{=}A ; \boldsymbol{\pi}) = \sum_j p(i|j ; \boldsymbol{\pi})\ \alpha_{t-1}(j)\ , \qquad (7)$$

$$\tilde\beta_T(i) = \widehat{\beta}(i)\ , \quad \tilde\beta_t(i) = P(r{=}1 \,|\, x_t{=}i ; \boldsymbol{\pi}) = \sum_j p(j|i ; \boldsymbol{\pi})\ \tilde\beta_{t+1}(j)\ . \qquad (8)$$

We find that all the α-quantities do not depend on T in any way, i.e., they are valid for all MDPs of any finite time T. This is not true for the $\tilde\beta$-quantities when defined as above. However, we can use a simple trick, namely define the β's to be indexed backward in time (with the 'time-to-go' τ), an get

$$\beta_0(i) = \widehat{\beta}(i)\ , \quad \beta_\tau(i) = P(r{=}1 \,|\, x_{T-\tau}{=}i ; \boldsymbol{\pi}) = \sum_j p(j|i ; \boldsymbol{\pi})\ \beta_{\tau-1}(j)\ . \qquad (9)$$

[2] Note that the expectation term $\mathrm{E}\,\{r \,|\, x_0 = i, T ; \boldsymbol{\pi}\}$ here is exactly the same as the terms $\mathrm{E}\,\{r_t \,|\, x_0 = i ; \boldsymbol{\pi}\}$ for $t = T$ in definition 2.1: we are taking the expectation w.r.t. a full probabilistic forward-sweep through the MDP, from time 0 to time T, given the policy $\boldsymbol{\pi}$. All MDPs share the same transition probabilities.

Defined in that way, all β-quantities do indeed not depend on T. For a specific MDP of finite time T, setting $\tau = T - t$ would allow us to retrieve the traditional forward-indexed $\tilde\beta$-quantities.

This means that we can perform α- and β-propagation in parallel, incrementing t and τ synchronously, and can retrieve the α's and β's for all MDPs of any finite time T. Although we introduce a mixture of MDPs we only have to perform a single forward and backward sweep. This procedure is, from the point of view of ordinary Hidden Markov Models, quite unusual—it is possible because in our specific setup we only condition on the very first ($x_0 = A$) and very last state ($r = 1$). Apart from the implementation of discounting with the time prior, this is the main reason why we considered the mixture of finite-time MDPs (Figure 2) in the first place instead of a single time-unbounded MDP that could emit rewards at any times (Figure 1).

During α- and β-propagation, we can compute the state posteriors, which are clearly not independent of the total time T. We define

$$\begin{aligned}\gamma_{t\tau}(i) &= P(x_t = i \,|\, x_0 = A, r = 1, T = t + \tau \,; \boldsymbol{\pi}) \\ &= \frac{1}{Z(t,\tau)}\ \beta_\tau(i)\ \alpha_t(i)\ , \qquad (10)\end{aligned}$$

with the normalization

$$\begin{aligned}Z(t,\tau) &= \sum_i \alpha_t(i)\ \beta_\tau(i) \\ &= \frac{P(x_0 = A, r = 1 \,|\, T = t + \tau \,; \boldsymbol{\pi})}{P(x_0 = A)} = Z(t+\tau)\ . \qquad (11)\end{aligned}$$

Time and reward posterior. It is interesting to realize that the normalization constant Z only depends on the sum $t + \tau$, i.e., we can define $Z(t+\tau) = Z(t,\tau)$. Further, $Z(t+\tau)$ is related to the likelihood $P(x_0 = A, r = 1 \,|\, T = t + \tau \,; \boldsymbol{\pi})$, i.e., the likelihood that the start state is A and the final state leads to reward if one assumes an MDP of specific length T. Using Bayes rule, this leads us to the *posterior over* T (in abbreviated notation),

$$P(T \,|\, x_0, r \,; \boldsymbol{\pi}) = \frac{P(x_0, r \,|\, T \,; \boldsymbol{\pi})}{P(x_0, r \,; \boldsymbol{\pi})}\ P(T) = \frac{Z(T)\ P(T)}{P(r \,|\, x_0 \,; \boldsymbol{\pi})}\ , \qquad (12)$$

and the reward posterior

$$P(r \,|\, x_0 \,; \boldsymbol{\pi}) = \sum_T P(T)\ Z(T)\ . \qquad (13)$$

Action and state posteriors. Let us briefly derive the action and state posteriors that are relevant for the later discussion. For convenience, let

$$\begin{aligned}q_{t\tau}(a,i) &= P(r = 1 \,|\, a_t = a, x_t = i, T = t + \tau \,; \boldsymbol{\pi}) \\ &= \left\{ \begin{array}{ll} \sum_j p(j|i,a)\ \beta_{\tau-1}(j) & \tau > 1 \\ P(r = 1 \,|\, a_T = a, x_T = i) & \tau = 0 \end{array} \right. . \qquad (14)\end{aligned}$$

As expected, this quantity is independent of A and t because we conditioned on x_t and the history before time t becomes irrelevant in the Markov chain. We may use the simpler notation $q_\tau(a,i) = q_{t\tau}(a,i)$. Multiplying with the time prior and eliminating the total time we get the *action-conditioned likelihood*

$$\begin{aligned}&P(r = 1 \,|\, a_t = a, x_t = i \,; \boldsymbol{\pi}) \\ &\qquad = \frac{1}{C} \sum_{\tau=0}^{\infty} P(T = t + \tau)\ q_\tau(a,i)\ , \qquad (15)\end{aligned}$$

where $C = \sum_{\tau'} P(T = t + \tau')$, and which is for the discounted time prior independent of t because $P(t + \tau) = \gamma^t P(\tau)$ which is absorbed in the normalization. Further, with Bayes rule we get the *action posterior*

$$\begin{aligned}&P(a_t = a \,|\, x_t = i, r = 1 \,; \boldsymbol{\pi}) \\ &\qquad = \frac{\pi_{ai}}{C'} \sum_{\tau=0}^{\infty} P(T = t + \tau)\ q_\tau(a,i)\ , \qquad (16)\end{aligned}$$

where $C' = P(r = 1 \,|\, x_t = i \,; \boldsymbol{\pi})\ \sum_{\tau'} P(T = t + \tau')$ can be computed from normalization, and which is also independent of t. Finally, in the experimental results we will display the posterior probability of visiting a certain state i. This is derived as

$$\begin{aligned}&P(i \in x_{0:T} \,|\, x_0, r = 1 \,; \boldsymbol{\pi}) \qquad (17) \\ &\quad = \sum_T \frac{P(T)\ Z(T)}{P(r = 1 \,|\, x_0 \,; \boldsymbol{\pi})} \Big[1 - \prod_{t=0}^{T} \big[1 - \gamma_{t,T-t}(i)\big]\Big]\ .\end{aligned}$$

M-step: the policy update. The standard M-step in an EM-algorithm maximizes the expected complete log-likelihood $Q(\pi^*, \pi) = \sum_T \sum_{x_{0:T}, a_{0:T}} P(x_{0:T}, a_{0:T}, T \,|\, r = 1 \,; \boldsymbol{\pi}) \log P(r = 1, x_{0:T}, a_{0:T}, T \,; \boldsymbol{\pi}^*)$ w.r.t. the new parameters π^*, where expectations over the latent variables $(T, x_{0:T}, a_{0:T})$ were taken w.r.t. the posterior given the old parameters π. In our case, in strong analogy to the standard HMM case, this turns out to assign the new parameters to the posterior action probability given in equation (16),

$$\pi^*_{ai} = P(a_t = a \,|\, x_t = i, r = 1 \,; \boldsymbol{\pi})\ . \qquad (18)$$

However, exploiting the structure of the MDP, we can also write the likelihood as

$$\begin{aligned}P(r = 1 \,|\, x_0 = i \,; \boldsymbol{\pi}^*) &= \sum_{aj} P(r = 1 \,|\, a_t = a, x_t = j \,; \boldsymbol{\pi}^*)\ \pi^*_{aj} \\ &\qquad \cdot P(x_t = j \,|\, x_0 = i \,; \boldsymbol{\pi}^*)\ . \qquad (19)\end{aligned}$$

Maximizing this expression w.r.t. π^*_{aj} can be done separately for every j and is thus independent of $P(x_t = j \,|\, x_0 = i\,; \boldsymbol{\pi}^*)$ and thereby also independent of t because $P(r = 1 \,|\, a_t = a, x_t = j\,; \boldsymbol{\pi}^*)$ is so (see equation (15)). Using the E-step we can approximate the first term and maximize $\sum_a P(r=1 \,|\, a_t=a, x_t=j\,; \boldsymbol{\pi})\ \pi^*_{aj}$ via

$$\pi^*_{ai} = \delta_{a=a^*(i)}\ , \qquad a^*(i) = \operatorname*{argmax}_a P(r=1 \,|\, a_t=a, x_t=i\,; \boldsymbol{\pi}) \tag{20}$$

where $a^*(i)$ maximizes the action-conditioned likelihood (15). Given the relations between equation (15) and (16), the main difference to the standard M-step is that this update is much greedier and converges faster in the MDP case. We will use the update (20) rather than (18) in the experiments.

5. Relation to Policy Iteration

The β-quantities computed during backward propagation are actually the value function for a single MDP of finite time T. More precisely, comparing (9) with the reward likelihood (3) for the MDP of finite time T and the definition 2.1 of the value function, we have $\beta_\tau(i) \propto \big(V^\pi(i) \text{ of the MDP of time } T = \tau\big)$. Accordingly, the full value function is the mixture of the β's, $V^\pi(i) = \frac{1}{1-\gamma} \sum_T P(T)\ \beta_T(i)$, when a discount time prior is chosen. Further, the quantities $q_\tau(a,i)$ defined in (14) are equally related to the Q-function, in the sense that $Q^\pi(a,i) = \frac{1}{1-\gamma} \sum_T P(T)\ q_T(a,i)$ for a discount time prior. Note that this is also the action-conditioned likelihood (15) and, interestingly, the action posterior (16) is proportional to $\pi_{ai}\ Q^\pi(a,i)$. Hence, *the E-step performs a policy evaluation* which yields the classical value function but additionally the time, state and action posteriors, which have no traditional analogue. Given this relation to policy evaluation, *the M-step performs a policy update* which (for the greedy M-step (20)) is the standard policy update in Policy Iteration (Sutton & Barto, 1998), maximizing the Q-function w.r.t. the action a in state i. Thus, the EM-algorithm using exact inference and belief representation is effectively equivalent to Policy Iteration (also w.r.t. convergence) but computes the necessary quantities in a different way. However, when using approximate inference or belief representations (as in the second example) the EM-algorithm amounts to a qualitatively different algorithm. One should also mention the related approach by (Ng et al., 1999), using density propagation for *forward* simulation of a policy (the α-propagation) and estimation of the policy gradient. In practical implementation, knowing the time, state and action posteriors can be exploited by pruning unnecessary computations as discussed in the following example.

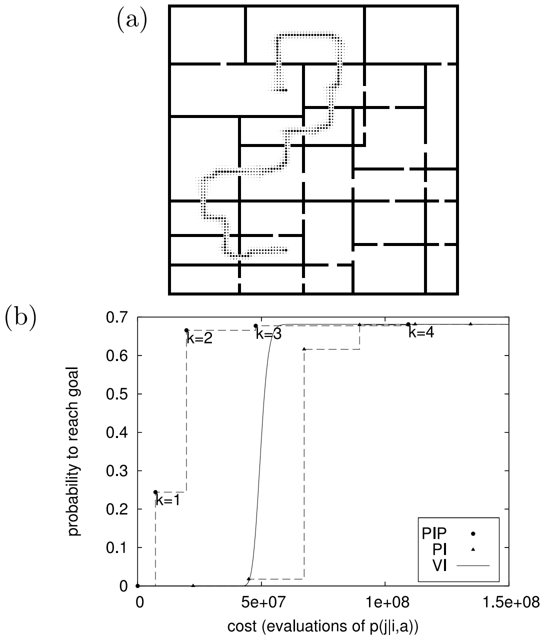

Figure 3. (a) State visiting probability calculated by PIP for some start and goal state. The radii of the dots are proportional to (17). (b) The probability of reaching the goal (for PIP) and the value calculated for the start state (PS) against the cost of the planning algorithms (measured by evaluations of $p(j|i,a)$) for both start/goal configurations.

6. Discrete maze example

We tested our probabilistic inference planning (PIP) algorithm on a discrete maze of size 100×100 and compared it to standard Value Iteration (VI) and Policy Iteration (PI). Walls of the maze are considered to be trap states (leading to unsuccessful trials) and actions (north, south, east, west, stay) are highly noisy in that with a probability of 0.2 they lead to random transitions. In the experiment we chose a uniform time prior (discount factor $\gamma = 1$) and iterated the policy update $k = 5$ times. To increase computational efficiency we exploited that the algorithm explicitly calculates posteriors which can be used to prune unnecessary message passings as explained in appendix A. For policy evaluation in PI we performed 100 iterations of standard value function updates.

Figure 3(a) displays the posterior state visiting probabilities generated by our probabilistic inference planner (PIP) for a problem where rewards are given when some goal state is reached. Computational costs are measured by the number of evaluations of the environment $p(j|i,a)$ needed during the planning procedure.

Figure 3(b) displays the probability of reaching the goal $P(r=1 \,|\, x_0 = A\,; \boldsymbol{\pi})$ against these costs for the same two start/goal state configurations. Note that for PIP (and PI) we can give this information only after a complete E- and M-step cycle (policy evaluation and update) which are the discrete dots (triangles) in the graph. The graph also displays the curve for VI, where the currently calculated value V_A of the start state (which converges to $P(B|A)$ for the optimal policy) is plotted against how often VI evaluated $p(j|i,a)$.

In contrast to VI and PI, the PIP algorithm takes considerable advantage of knowing the start state in this planning scenario: the forward propagation allows for the pruning and the early decision on cutoff times of the E-step as described in appendix A. It should thus not be a surprise and not overstated that PIP is significantly more efficient in this specific scenario. Certainly, some kind forward propagations could also be introduced for VI or PI to achieve similar efficiency. Nonetheless, our approach provides a principled way of pruning by exploiting the computation of proper posteriors.

A detailed inspection of the policies computed by all three methods showed that they are equal for states which have significantly non-zero state visiting probabilities.

7. Stochastic optimal control

Gaussian belief state propagation. In the second example we want to show that the framework naturally allows to transfer other inference techniques to the problem of solving MDPs. We address the problem of stochastic optimal control in the case of a continuous state and control space. A standard inference technique in continuous state spaces is to assume Gaussian belief states as representations for α's and β's and propagate forward-backward and using the unscented transform to handle also non-linear transition dynamics (see (Murphy, 2002) for an overview on inference techniques in DBNs). Note that using Gaussian belief states implies that the effective value function (section 5) becomes a mixture of Gaussians.

All the equations we derived remain valid when reinterpreted for the continuous case (summations become integrations, etc) and the exact propagation equations (7) and (9) are replaced by propagations of Gaussian belief states using the unscented transform. In more detail, let $\mathcal{N}(x,a,A)$ be the normal distribution over x with mean a and covariance A and let $\overline{\mathcal{N}}(x,a,A)$ be the respective *non-normalized* Gaussian function with $\overline{\mathcal{N}}(a,a,A)=1$. As a transition model we assume

$$P(x'|u,x) = \mathcal{N}(x', \phi(u,x), Q + (|u|/\mu)^2\; I) \qquad (21)$$

where $\phi(u,x)$ is an non-linear function, Q is a constant noise covariance, and we introduced a parameter μ for an additional noise term that is squared in the control signal.

With the parameterization $\alpha_t(x) = \mathcal{N}(x, a_t, A_t)$ and $\beta_t(x) = \overline{\mathcal{N}}(x, b_t, B_t)$ (note that β's always remain non-normalized Gaussian functions during propagation), forward and backward propagation read

$$\begin{aligned}(a_t, A_t) &= UT_\phi(a_{t-1}, A_{T-1})\\ (b_t, B_t) &= UT_{\phi^{-1}}(b_{t-1}, B_{T-1})\ ,\end{aligned}$$

where $UT_\phi(a,A)$ denotes the unscented transform of a mean and covariance under a non-linear function. In brief, this transform deterministically considers $2n+1$ points (say with standard deviation distance to the mean) representing the Gaussian, maps these point forward using ϕ (also taking care of the noise terms), and returns the Gaussian that approximates the mapped points (and their associated covariances). Further we have

$$\begin{aligned}&Z_{t\tau} = \mathcal{N}(a_t, b_\tau, A_t + B_\tau)\\ &\gamma_{t\tau}(x) = \mathcal{N}(x, c_{t\tau}, C_{t\tau})\ ,\quad C_{t\tau} = (A_t^{-1} + B_\tau^{-1})^{-1}\ ,\\ &\qquad c_{t\tau} = C_{t\tau}\ (A_t^{-1}\ a_t + B_\tau^{-1}\ b_\tau)\end{aligned}$$

The policy and the M-step. In general, the policy is given as an arbitrary non-linear function $\pi:\ x \mapsto u$. Clearly, we cannot store such a function in memory. However, via the M-step the policy can always be implicitly expressed in terms of the β-quantities of the previous E-step and numerically evaluated at specific states x. This is particularly feasible in our case because the unscented transform used in the belief propagation (of the next E-step) only needs to evaluate the transition function at some states; and we have the advantage of not needing to approximate the function π in any way. For the M-step we need equation (14),

$$\begin{aligned}q_\tau(x,u) &= \int_{x'} P(x'|u,x)\ \overline{\mathcal{N}}(x', b_{\tau-1}, B_{\tau-1})\\ &= |2\pi B|^{1/2} \mathcal{N}(b_{\tau-1}, \phi(x,u), B_{\tau-1} + Q + (|u|/\mu)^2\ I)\ ,\end{aligned} \qquad (22)$$

and we maximize (15) with a gradient ascent using

$$\begin{aligned}\partial_u q_\tau(x,u) = -q_\tau(x,u) \Big[\ h^T \Big(\partial_u \phi(x,u)\Big) -&\\ u\frac{1}{\mu^2}\Big(\mathrm{tr}(A^{-1}) - h^T h\Big)\Big]&\\ A := B_{\tau-1} + Q + (|u|/\mu)^2\ I\ ,\quad h := A^{-1}\ (\phi(x,u) - b)&\end{aligned} \qquad (23)$$

Examples. Consider a simple 2-dimensional problem where the start state is distributed around zero via $\alpha_0(x) = \mathcal{N}(x,\, (0,0),\, .01I)$ and the goal region is determined by $P(r = 1 \,|\, x) = \overline{\mathcal{N}}(x,\, (1,1),\, \text{diag}(.0001,.1))$. Note that this goal regions around $(1,1)$ is heavily skewed in that rewards depend more on the precision in the $x-$dimension than the $y-$dimension. The control law is simply $\phi(u,x) = x + .1\,u$ and the discount factor $\gamma = 1$. When choosing $\mu = 0$ (no control-dependent noise), the optimal control policy will try to jump directly to the goal $(1,1)$. Hence we first consider the solution when manually constraining the norm of $|u|$ to be small (effectively following the gradient of $P(r = 1 \,|\, u_t = u, x_t = x\,; \boldsymbol{\pi})$). Figure 4ab shows the learned control policy π and the forward simulation given this policy by displaying the covariance ellipses for $\alpha_{0:T}(x)$ after $k = 3$ iterations. What we find is a control policy that reduces errors in $x-$dimension more strongly than in $y-$direction, leading to the tangential approach to the goal region. This is related to studies on redundant control or the so-called uncontrolled manifold.

Next we can investigate what the effect of control-dependent noise is without a constraint on the amplitude of u. Figure 4cd displays results (after $k = 3$ iterations) for $\mu = 1$ and no additional constraints on u. The process actually resembles a golf player: the stronger the hit, the more noise. The optimal strategy is to hit fairly hard in the beginning, hopefully coming closer to the goal, such that later a number of smaller and more precise hits can be made. The reason for the small control signals around the goal region is that small steps have much more accuracy and reward expectation is already fairly large for just the $x-$coordinate being close to 1.

Finally we think of x being a phase space and consider the dynamics $\phi(x,u) = (x_1 + .1x_2, x_2 + .1u)$ where u is the 1-dimensional acceleration of the velocity x_2, and x_1 is a position. This time we set the start and goal to $(0,0)$ and $(1,0)$ respectively, both with variance .001 and choose $\mu = 10$. Figure 4ef display the result and show nicely how the learned control policy approaches the new position on the $x-$axis by first gaining and then reducing velocity.

8. Conclusion

In this paper we presented a model that translates the problem of planning into a problem of probabilistic inference. Our main contribution over previous approaches to inference in MDPs is that we do not have to fix a total time and likelihood maximization is equivalent to maximization of the expected future return. A key step for this result was to consider the mixture of finite-time MDP as an equivalent model, which leads to a simple and efficient synchronous forward-backward procedure for the E-step. We can compute posteriors over actions, states, and the total time. We showed that for exact inference and belief representation the resulting EM-algorithm (with greedy M-step) is in effect equivalent to Policy Iteration.

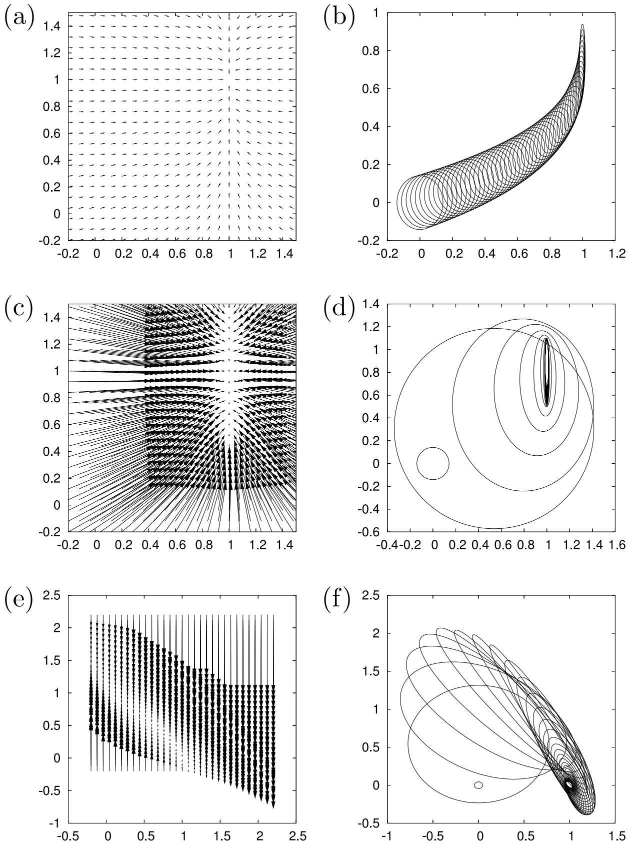

Figure 4. Actions to approach an aspheric Gaussian-shaped target. Learned policy (a,c) and forward simulation (α's) of this learned policy (b,d) for the cases of restricted action amplitude - the walker model (a,b) and unconstrained amplitude - the golfer model (c,d). And the approach to a new position under phase space dynamics (e,f).

However, the general motivation for us to consider inference for solving MDPs is that the full variety of existing inference techniques can be applied. This refers especially to more complex structured state representations for MDPs (continuous, factorial and hierarchical DBNs, options or macro-policies) in which there is growing interest recently. We believe it is of great interest to be able to apply existing inference methods directly to such MDPs. The second example demonstrates such transfer: We combined standard inference methods, usually used for Kalman smoothing in non-linear dynamics, with our framework to solve a stochastic optimal control problem.

A. Pruning computations

Consider a finite state space and assume that we fixed the maximum allowed time T by some upper limit T_M (e.g., by deciding on a cutoff time based on the time posterior computed on the fly, see below). Then there are potentially large regions of the state space on which we may prune computations, i.e., states i for which the posterior $\gamma_{t\tau}(i) = 0$ for any t and τ with $t + \tau \leq T_M$. Let us consider the α-propagation only (all statements apply conversely for the β-propagation). For iteration time t we define a set of states

$$X_\alpha(t) = \{i \,|\, \alpha_t(i) \neq 0 \ \wedge\ (t < T_M/2 \vee \beta_{T_M-t}(i) \neq 0)\} \ .$$

Further, given $\beta_\tau(i) = 0 \Rightarrow \forall \tau' \leq \tau : \beta_{\tau'}(i) = 0$, it follows

$$\begin{aligned} i \in X_\alpha(t) &\Leftarrow\ \alpha_t(i) \neq 0 \ \wedge\ \beta_{T_M-t}(i) \neq 0 \\ &\Leftarrow\ \exists_{\tau' \leq T_M - t} :\ \alpha_t(i) \neq 0 \ \wedge\ \beta_{\tau'}(i) \neq 0 \\ &\iff \exists_{\tau : t+\tau \leq T_M} :\ \gamma_{t\tau}(i) \neq 0 \end{aligned} \tag{24}$$

Thus, every state that is potentially visited at time t (for which $\exists_{\tau : t+\tau \leq T_M} : \ \gamma_{t\tau}(i) \neq 0$) is included in $X_\alpha(t)$. We will exclude all states $i \notin X_\alpha(t)$ from the propagation procedure and not deliver their messages. The constraint $t < T_M/2$ concerning the β's was inserted in the definition of $X_\alpha(t)$ only because of the feasibility of computing $X_\alpha(t)$ at iteration time t. Initializing $X_\alpha(0) = \{A\}$, we can compute $X_\alpha(t)$ recursively via

$$X_\alpha(t) = \Big[X_\alpha(t-1) \cup \mathsf{OUT}(X_\alpha(t-1)) \Big] \cap \left\{ \begin{array}{ll} S & t < T_M/2 \\ \{i \,|\, \beta_{T_M-t}(i) \neq 0\} & t \geq T_M/2 \end{array} \right. ,$$

where $\mathsf{OUT}(X_\alpha(t-1))$ is the set of states which have non-zero probability transitions from states in $X_\alpha(t-1)$.

For the discount prior, we can use a time cutoff T_M for which we expect further contributions to be insignificant. The choice of this cutoff involves a payoff between computational cost and accuracy of the E-step. Let T_0 be the minimum time for which $Z(T_0) \neq 0$. It is clear that the cutoff needs to be greater than T_0. In the experiment in section 6 we used an increasing schedule for the cutoff time, $T_M = (1 + 0.2\,k)\,T_0$, depending on the iteration k of the EM-algorithm.

Acknowledgments

Marc Toussaint was supported by the German Research Foundation (DFG), Emmy Noether fellowship TO 409/1-1.

References

Attias, H. (2003). Planning by probabilistic inference. *Proc. of the 9th Int. Workshop on Artificial Intelligence and Statistics.*

Boutilier, C., Dearden, R., & Goldszmidt, M. (1995). Exploiting structure in policy construction. *Proc. of the 14th Int. Joint Conf. on Artificial Intelligence (IJCAI 1995)* (pp. 1104–1111).

Bui, H., Venkatesh, S., & West, G. (2002). Policy recognition in the abstract hidden markov models. *Journal of Artificial Intelligence Research*, *17*, 451–499.

Ghahramani, Z., & Jordan, M. I. (1995). Factorial hidden Markov models. *Advances in Neural Information Processing Systems, NIPS* (pp. 472–478). MIT Press.

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

Hauskrecht, M., Meuleau, N., Kaelbling, L. P., Dean, T., & Boutilier, C. (1998). Hierarchical solution of Markov decision processes using macro-actions. *Proc. of Uncertainty in Artificial Intelligence (UAI 1998)* (pp. 220–229).

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

Kveton, B., & Hauskrecht, M. (2005). An MCMC approach to solving hybrid factored MDPs. *Proc. of the 19th Int. Joint Conf. on Artificial Intelligence (IJCAI 2005).*

Minka, T. (2001). A family of algorithms for approximate bayesian inference. PhD thesis, MIT.

Murphy, K. (2002). Dynamic bayesian networks: Representation, inference and learning. PhD Thesis, UC Berkeley, Computer Science Division. See particularly the chapter on DBN at http://www.cs.ubc.ca/ murphyk/Papers/dbnchapter.pdf.

Ng, A. Y., Parr, R., & Koller, D. (1999). Policy search via density estimation. *Advances in Neural Information Processing Systems 12* (pp. 1022–1028).

Sutton, R., & Barto, A. (1998). *Reinforcement learning.* MIT Press, Cambridge.

Verma, D., & Rao, R. P. N. (2006). Goal-based imitation as probabilistic inference over graphical models. *Advances in Neural Information Processing Systems 18 (NIPS 2005).*