Model Transfer for Markov Decision Tasks via Parameter Matching

Funlade T. Sunmola
Wolfson Computer Laboratory,
University Hospital Birmingham, NHS Foundation Trust,
Edgbaston, Birmingham, B15 2TH. UK.

Jeremy L. Wyatt
School of Computer Science,
University of Birmingham,
Edgbaston, Birmingham, B15 2TT. UK.

Abstract

Model transfer refers to the process of transferring information from a model that was previously identified for one task (source) to a new task (target). For decision tasks in unknown Markov environments, we profit through model transfer by using information from related tasks, e.g. transition knowledge and solution (policy) knowledge, to quickly determine an appropriate model of the new task environment. A difficulty with such transfer is typically the non-linear and indirect relationship between the available source knowledge and the target's working prior model of the unknown environment, provided through a complex multidimensional transfer function. In this paper, we take a Bayesian view and present a probability perturbation method that conditions the target's model parameters to a variety of source knowledge types. The method relies on pre-posterior distributions, which specifies the distribution of the target's parameter set given each individual knowledge types. The pre-posteriors are then combined to obtain a posterior distribution for the parameter set that matches all the available knowledge. The method is illustrated with an example.

Introduction

Planning under uncertainty is central to solving many important real-world problems. An increasingly popular computational framework for modelling such tasks is Markov Decision Processes (MDP). An MDP is a stochastic control process characterized by a set of states that can be perceived exactly, actions, and transition probability functions that specify the transition probabilities from one state to the next. Essentially, the process is Markovian in the sense that, the outcome of applying an action to a state depends only on the current action and state. The MDP framework is rich and quite expressive in capturing the essence of purposeful activity in a wide variety of tasks, including applications in artificial intelligence, engineering, financial and medical decision making. However, in practice the computational effort of solving an MDP may be prohibitive and, moreover, the model parameters of the MDP may be unknown.

Quite often, there is knowledge available on other tasks that one can transfer to a new task. For humans, knowledge transfer is an integral part of life. It represents the transmission of knowledge (conveying the knowledge of a source task to a target task) and the appropriate use of the transmitted knowledge. The goal is to promote/facilitate knowledge sharing, collaboration and networking as a vehicle for problem solving. Recent research into knowledge transfer for tasks modelled as MDPs justifies this thinking (Singh, 1992; Mahadevan, 2005; Sherstov and Stone, 2005; Liu and Stone, 2006). The goal is to find ways to transfer knowledge acquired in a task (the source) into a new task (the target) with a desired outcome of reducing the computational effort expended on solving the target MDP.

There are typically several types of source information, including relevance of model variables (feature selection), conditional independence among variables, and parameter-specific domain knowledge. This information set can be used to specify priors on the transition and the solution (value or policy function) for the new task. The prior on solution may be derived from theoretical or observed historical (target) solutions of related tasks, separate from data (observations) acquired during interactions with the target environment. The transition knowledge and source solutions impose specific constraints/preferences on parameters of the true model of the new task.

Ideally, one would want to determine the parameters of the true model such that when the model is applied with the parameters to solve the target task, the output that results are in close correspondence to a totality of the prior knowledge available. This is a problem of parameter matching. In many practical situations, especially those involving task transfers, parameter matching is an important step. Unfortunately searching for models that parameter match is nontrivial because of variety of knowledge types which makes the relationship between the available prior information and the model of the new task difficult (sometimes non-linear and indirect), typically, provided through a complex multidimensional model transfer function.

In this paper, we present a model transfer framework that allows one to combine a variety of knowledge types when transferring prior information between models of Markov decision tasks. The framework is presented within the context of Bayesian modelling using probability perturbation(Caers and Hoffman, 2006). Probability perturbation is designed as an alternative to the standard Bayesian method that decomposes posterior distribution into likelihood and prior information. It uses pre-posterior to decompose the source knowledge into 'easy' (or linear knowledge) and 'difficult' (or non-linear knowledge). The method relies on fast non-iterative sequential simulation to generate model realizations. The mechanisms of probability perturbation allow one to match the difficult knowledge by perturbing an initial realization. The probability perturbation method moves the initial realization closer to matching the difficult knowledge, while maintaining the prior model statistics and conditioning to the linear knowledge.

The remainder of the paper is structured into six sections. Section two contains an overview of current formalisms for model transfer including those that use Bayesian paradigms. A standard Bayesian framework to model learning for Markov decision tasks is presented in section three with an assumption that a working prior is available for the target task. This is followed in section four by an extension of the standard Bayesian framework to model transfer for Markov decision tasks. A variety of transfer techniques using the Bayesian framework is presented in section five and results of empirical tests on a sample problem is presented in section six. The paper ends in section seven with concluding remarks and suggestions for future work.

Formalisms for Model Transfer

Model transfer is long recognised in many fields as a way of taking advantage of previously acquired task knowledge when determining a model of a new task. Unfortunately, it is widely acknowledged, especially in machine learning, that systems do not take sufficient advantage of model transfer. This is partly due to the fact that there is no complete theory of how task knowledge can be retained and then selectively transferred when learning a new task (Thrun, 1997). Existing formalisms for model transfer differs primarily in their procedures for using prior knowledge, representation of the source and target tasks, and the specific goal of model transfer.

There are two main reasons for the incorporation of related existing knowledge. First, existing knowledge can be used to improve the global accuracy of models when only impoverished data are available. Second, existing knowledge can be used to improve the success of model transfer efforts. The transferring procedures were either Bayesian or frequentist in their use of prior information.

We assume that the task to accomplish is modelled as a standard Markov decision processes (MDP) $\langle \mathcal{S}, \mathcal{A}, \mathcal{R}, \mathcal{P} \rangle$ with finite state and action sets $\mathcal{S}, \mathcal{A}$, reward function $R : \mathcal{S} \mapsto \mathcal{R}$, and dynamics P. The dynamics P refers to a set of transition distributions p_{ij}^{a} that captures the probability of reaching state j after we execute an action a at state i such that $i, j \in S$. We assume throughout that R is known but not the dynamics P of the MDP. Once the dynamics is learnt, the planning problem in the MDP is straightforwardly finding a policy $\pi : \mathcal{S} \mapsto \mathcal{A}$ that optimise the expected discounted total reward $V = E(\sum_{t=1}^{\infty} \gamma^{t-1} r_t)$, where r_t is the reward received t steps into the future and $\gamma \in [0, 1]$ is a discount factor.

In the area of Markov decision tasks, model transfer paradigms are relatively new. Whilst a generally acceptable formalism for model transfer in this area is yet to emerge, there have been a number of important findings. Representation wise, it has been noted that traditional MDP definition is adequate only for solving problems in isolation - it is not expressive enough to capture similarities across problems and as such it is poorly suited for accomplishing knowledge transfer. A suggested way of overcoming this difficulty is a MDP formalism that use classes(Guestrin, 2003) and outcomes to remove the undesirable independence of model description on the state space(Sherstov and Stone, 2005).

Transferring procedures used for models of Markov decision tasks are predominantly frequentist, the use of Bayesian methods is limited. Bayesian approach proceeds as follows. Start with a prior distribution that encodes the learners initial belief about the possible values of each unknown MDP. Then, whenever a sampled realization of the unknown MDP is observed, the current belief is updated to reflect the observed data. In the context of an MDP with unknown transition probabilities, each of the unknown transition probability is an unknown parameter taking values in the [0, 1]-interval.

Bayesian approaches have been considered from the outset for MDPs (Bellman, 1961; Martin, 1967) and interest has re-emerged in this approach (see, for example (Dearden, 2000; Poupart *et al.*, 2006; Strens, 2000; Wang *et al.*, 2005; Wyatt, 2001)). The Bayesian-MDP framework is known to have a number of inherent difficulties – they are heavily reliant on planning for action selection, uses various myopic/non-myopic strategies as surrogate for dealing with the effects that actions have on future value estimates, and are intractable for typical real world tasks because it involves dynamic programming over a tree of information states. Researchers have generally had difficulty developing efficient and accurate Bayesian-MDP algorithms. To date, several approximate algorithms have been proposed and are based for example on confidence intervals (Sunmola and Wyatt, 2003; Wyatt, 2001) and sampling (Dearden, 2000; Strens, 2000; Wang *et al.*, 2005), but they tend to be computationally intensive at run time, preventing online learning or make drastic approximations such as myopically optimizing the policy. An alternative efficient point based value iteration algorithm called BEETLE was developed recently (Poupart *et al.*, 2006), framing the Bayesian-MDP problem as a partially observable Markov decision process (Duff, 2002) and showing through analytical derivation that the optimal value function is the upper envelope of a set of multivariate polynomials. Also relevant is the recently emerging body of work on using imitation techniques to accelerate reinforcement learning (Price, 2003). Imitation frameworks typically use observations of other agents to provide an observer agent with information about its action capabilities in unexperienced situations.

Finally, the specific goal of model transfer for Markov decision tasks has also significantly influenced existing formalisms and transfer procedures. The formalisms split between those that consider transition and reward knowledge, and those that focus primarily on the solution knowledge i.e. the value / policy functions(Mahadevan, 2005; Bernstein, 1999; Liu and Stone, 2006). Typically, the formalisms do not explicitly provide mechanisms for combining these sets of knowledge during model transfer.

The transfer problem may be formulated as a misfit minimisation problem, misfit between the targets working prior model of the task environment and the source's model of the task environment. The problem may be solved naively by trial and error. The difficulties with this approach are as follows: (a) the parameter spaces for the model are typically high dimensional; (b) each solution of the model can take a long time with several dynamic programming backups; (c) there may be many or no solutions to this high dimensional problem as the source may contain errors; d) one would need to integrate on-line data (experience tuples); and e) take cognisance of exploration control.

Learning with a Working Prior

In a standard Bayesian framework, we assume that there is a space $\mathcal{P}$ of unknown transition functions (parametric models) for the MDP and that there exists a *belief state* over this space. The belief state defines a probability density $f(P|M)$ over the MDPs. The density is parameterised by $M \in \mathcal{M}$. In the Bayesian approach, the unknown parameter P is treated as a random variable, and a working prior distribution $f(P|M)$ is chosen to represent what one knows about the parameter before observing transitions. In particular, $f(P|M)$ is the real task-specific prior describing actual beliefs which may be a non-informative prior when we have no prior knowledge about P.

At each step in the environment, we start at state s, choose an action a and then observe a new state s' and a reward r. We summarise our experience by a sequence of experience tuples $< s, a, r, s' >$. When we observe transitions, we update the prior with the new experience. Given an experience tuple $< s, a, r, s' >$ we can compute the *posterior* belief state by Bayes rule:

$$\begin{aligned} f(P|M) &= \frac{f(< s, a, r, s' > |P) f(P|M)}{f(< s, a, r, s' >)} \\ &= \frac{1}{Z} f(< s, a, r, s' > |P) f(P|M) \end{aligned} \quad (1)$$

in which Z is a normalising constant. Thus, the standard Bayesian approach starts with a working prior probability distribution over all possible MDPs (we assume that the sets of possible states, actions, and rewards are delimited in advance). As we gain experience, the approach focuses the mass of the posterior distribution on those MDPs in which the observed experience tuples are most probable. In summary, we update the prior with each data point $\langle s, a, r, t \rangle$ to obtain a posterior M which we use to approximate the expected state values. The Bayesian estimator of expected return under the optimal policy is:

$$V_i(M) = E[\tilde{V}_i|M] = \int_{\mathcal{P}} V_i(P) f(P|M) dP \quad (2)$$

where $V_i(P)$ is the value of i given the transition function P. When this integral is evaluated we transform our problem into one of solving an MDP with unknown transition probabilities, defined on the information space $\mathcal{M} \times \mathcal{S}$:

$$V_i(M) = \max\nolimits_a \ \{\sum_i \bar{p}^a_{ij}(M)(r^a_{ij} + \gamma V_j(T^a_{ij}(M)))\} \quad (3)$$

in which, for convenience, the transformation on M due to a single observed transition $i \overset{a,r}{\rightsquigarrow} j$ is denoted $(T^a_{ij}(M))$, $\bar{p}^a_{ij}(M)$ is the marginal expectation of the posterior distribution, and r^a_{ij} is the reward associated with the transition $i \overset{a,r}{\rightsquigarrow} j$. The optimal policy is to act greedily with respect to the Bayes Q-values.

The integral of equation (2) involved in the inference process may be approximated through naive global sampling (NGS) using the basic idea in Monte Carlo simulation in which a set of weighted particles (samples $\hat{P}$), drawn from the posterior distribution of the model parameters $f(P|M)$, is used to map the integration, to discrete sums. When, for simplicity, the model dimension is known and fixed, the integral may be approximated as follows:

$$V_i(M) = E[\tilde{V}_i|M] \approx \frac{1}{NS} \sum_{l=1}^{NS} V_i(\hat{P}_l) \quad (4)$$

where the particles $\hat{P}_l, l = 1, \ldots, NS$ are drawn from the constrained posterior $f(P|M)$, and assumed to be 'sufficiently' independent for the approximation to hold. NS is the sample size. Monte Carlo sampling techniques are an improvement over direct numerical approximation in that they automatically select particles in regions of high probability. NGS is naive in the sense that it involves global solutions to MDPs which can be expensive. Reusing MDPS from previous steps and doing sampling with repair will generally alleviate the deficiencies (Dearden, 2000).

Product of Dirichlet Densities

Typically, prior update and computation of $f(P|M)$ are rendered tractable by assuming a convenient, natural conjugate, working prior: M is a product of local independent densities for each transition distribution, and each density is Dirichlet. The probability density of the Dirichlet distribution for variables $\vec{p}_s^{\,a}$ with parameters $M = [\vec{m}^a_s : \ \forall s \in S \ \forall a \in A]$ is defined by:

$$f(P|M) = \frac{1}{Z(M)} \prod_{a=1}^{A} \prod_{s=1}^{S} (\vec{p}_s^{\,a})^{\vec{m}^a_s - 1} \quad (5)$$

where $\vec{m}_s^{\,a} = \{m^a_{s1}, m^a_{s2}, \ldots, m^a_{sN}\}$ for the possible N successor states of state $s \in S$ and action $a \in A$, given that $m^a_{ss'} > 0 \ \forall s' \in N$. The parameters $\vec{m}_s^{\,a}$ can be interpreted as *prior observation counts* for events governed by $\vec{p}_s^{\,a}$. The normalisation constant $Z(M)$ becomes:

$$Z(M) = \frac{\prod_{a=1}^{A} \prod_{s=1}^{S} \Gamma(\vec{m}^a_s)}{\Gamma(\sum_{s=1}^{S} \vec{m}^a_s)} \quad (6)$$

Let $\vec{m}^a_0 = \sum_{s=1}^{S} \vec{m}^a_s$. The mean and variance of the Dirichlet distribution are:

$$E[\vec{p}_s^{\,a}] = \frac{\vec{m}^a_s}{\vec{m}^a_0} \quad (7)$$

and

$$Var[\vec{p}_s^{\,a}] = \frac{\vec{m}^a_s(\vec{m}^a_0 - \vec{m}^a_s)}{(\vec{m}^a_0)^2(\vec{m}^a_0 + 1)} \quad (8)$$

When $\vec{m}_s^a \to 0$, the distribution becomes non-informative. This means that all the $\vec{p}_s{}^a$ stay the same if all are scaled with the same multiplicative constant. The variances will, however, get smaller as the parameter $\vec{m}_s^a$ grows.

Working Prior Plus

We consider situations where, in addition to a working prior, alternative source of information on the underlying MDP is available through a *task class model* (TCM). A task class model is a collection of information previously identified for a family of related tasks. The class model describes the structure and behaviour of a set of tasks which are its instances.

Specifically, we consider situations in which information in the task class model is classified into *direct* ϕ - 'easy' (or linear knowledge) and *indirect* I - 'difficult' (or non-linear knowledge). The direct knowledge is parameterised by the model parameters, typically of the form:

$$\phi_i = h(P) + \epsilon_i \quad [w_i] \tag{9}$$

in which each expression of direct knowledge ϕ_i is specified with a tolerance ϵ_i and an assigned weight w_i that determines the importance of the expression in the TCM. Figure 1 shows a sample task class model fragment for a family of chain tasks. The example TCM contains five equally-weighted direct expressions that are parameterised by specific model parameters. In addition, there is one indirect expression that provides prior information about policies for the family of tasks modelled by the TCM. In the example, we know through the TCM that the same action should be selected for internal states 2 and 3.

We choose to transfer information from the TCM to a working prior M. In principle, one could achieve the transfer by specifying a transfer function g such that

$$M = g(M, \langle s,a,r,t\rangle, \phi, I) \tag{10}$$

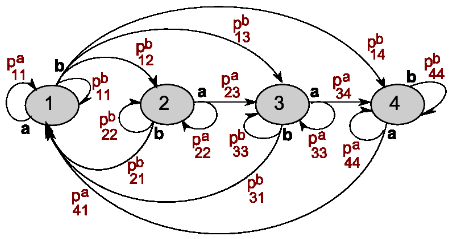
However, for non-trivial task instances, g represents a complex multi-dimensional transfer function that is difficult to compute. Instead of seeking to compute g, we use the working prior and the indirect knowledge to produce a probability distribution over all possible models of the MDP subject to constraints on the model-space specified by the direct information ϕ.

The Bayesian estimator of expected return under the optimal policy $\tilde{V}_i$ then becomes:

$$V_i(M) = E[\tilde{V}_i|M] = \int_{\mathcal{P}_{C(\phi)}} V_i(P) f(P|M,I)dP \tag{11}$$

in which the quantity of interest V is expressed as an expectation of functional that depends on trajectories defined by unknown transition function P parameterised by a) a matrix M of transition counts, and b) indirect information I, subject to constraint $\mathcal{P}_{C(\phi)}$ on model space.

Computation of the expectation equation 11 requires calculating integrals that, for all but simple tasks, are difficult to compute in closed form. There are two significant computational difficulties - computation of $f(P|M,I)$ subject to $\mathcal{P}_{C(\phi)}$ and computation of $V_i(P)$ for large samples of P.



(a) A four-state chain task

direct – ϕ

$\phi_1 = 1 - p^b_{21}p^b_{22}$	[0.2]
$\phi_2 = 1 - p^b_{31}p^b_{33}$	[0.2]
$\phi_3 = 1 - p^a_{41}p^a_{44}$	[0.2]
$\phi_4 = p^a_{34} - p^a_{22}p^a_{23}$	[0.2]
$\phi_5 = p^a_{11} - p^b_{44}$	[0.2]

indirect – I

equal(π_2,π_3)

(b) Task class model

Figure 1: A sample task class model fragment for a family of chain tasks. ϕ_i are task class model parameters. The indirect expression specifies same policy for internal states 2 and 3.

We focus here on the former. We use $\mathcal{P}_{C(\phi)}$ to constrain the working prior and compute $f(P|M,I)$ using pre-posterior densities.

Constrained Prior Density

A convenient choice of prior distribution over the parameters given C_Θ, i.e. $P_{C_\Theta} = [\vec{p}_s{}^a : \ \forall s \in S \ \forall a \in A | C_\Theta]$ is:

$$f(P|M) = \begin{cases} \frac{1}{Z_{C_\Theta}(M)} \prod_{a=1}^{A} \prod_{s=1}^{S} (\vec{p}_s{}^a)^{\vec{m}_s^a - 1} & \text{for } p \in \mathcal{C}_\Theta \\ 0 & \text{otherwise,} \end{cases} \tag{12a}$$

where $\vec{m}_s{}^a = \{m^a_{s1}, m^a_{s2}, \ldots, m^a_{sN}\}$ for the possible N successor states of state $s \in S$ and action $a \in A$, given that $m^a_{ss'} > 0 \, \forall s' \in N$.

$$\mathcal{C}_\Theta = \{\phi; 0 \le p^a_{ss'} \le 1; \sum_{s'=1}^{N} p^a_{ss'} = 1 \ \ \forall s, s' \in S, a \in A\} \tag{12b}$$

and

$$(Z_{C_\Theta}(M))^{-1} = \int_{\mathcal{C}_\Theta} \prod_{a=1}^{A} \prod_{s=1}^{S} (\vec{p}_s{}^a)^{\vec{m}_s^a - 1} d\vec{p}_s{}^a \tag{12c}$$

Pre-posterior Densities of $f(P|M,I)$

After observing a transition with experience tuple $< s, a, r, s' >$, the posterior distribution from which samples of models are drawn is, in a standard Bayesian context, de-

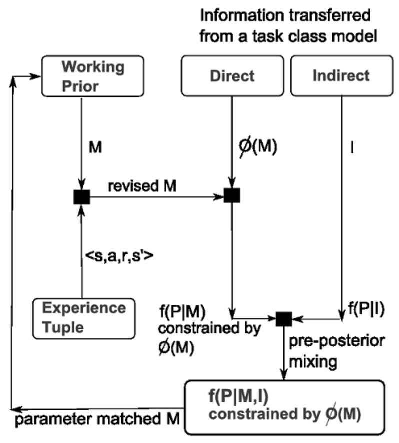


Figure 2: Computing $f(P|M, I)$ through pre-posterior mixing.

composed into a likelihood and prior distribution as follows.

$$\begin{aligned} f(P|M, I) &= \frac{f(<s, a, r, s'>, I|P)f(P|M)}{f(<s, a, r, s'>, I)} \\ &= \frac{f(<s, a, r, s'>|P)f(I|P)f(P|M)}{f(<s, a, r, s'>, I)} \end{aligned} \quad (13)$$

where the likelihood $f(<s, a, r, s'>, I|P)$ is further decomposed into $f(<s, a, r, s'>|P)$ and $f(I|P)$ under the assumptions of conditional independence. Whilst the conditional independence assumption makes the inference of the likelihood feasible, it is difficult to verify and may have considerable consequence in the computation of the posterior distribution $f(P|M, I)$ and the resulting expected values.

We take an approach that differs from the standard Bayesian approach. Instead of working with the likelihoods $f(<s, a, r, s'>|P)$ and $f(I|P)$, we use two pre-posteriors $f(P|<s, a, r, s'>)$ and $f(P|I)$ and have the pre-posteriors combined into $f(P|M, I)$ using the framework illustrated in Fig. 2.

Posterior Inference

Learning a model of a task environment using the framework of Fig. 2 and deriving a policy based on the learned model requires the following main steps.

```
start with current state $s$, action $a$, transfer window ($t_0, t_1$), $\phi$, $I$, and working prior $M$
initialise current time step $t$
repeat
    observe transition $s \overset{a,r}{\rightsquigarrow} s'$
    if $t$ falls within the transfer window $(t_0, t_1)$ then
        update $M$ using $s \overset{a,r}{\rightsquigarrow} s'$ and $I$, subject to $\phi$.
    else
        update $M$ with $s \overset{a,r}{\rightsquigarrow} s'$
    end if
    evaluate $V$ using equation (3)
    select action for the current state from $V$
    move $t$ to next time step
until termination condition
```

We assume that information transfer takes place only within a transfer window specified by a transfer start time t_0 and a transfer end time t_1. The learner is assumed to know both t_0 and t_1.

The learner starts by performing an action in its initial state, observing transition and reward. It then uses the resulting experience tuple $<s, a, r, s'>$ to update the working prior M and obtain a posterior density. This is done using information in the TCM i.e. ϕ and I if the current time falls within the transfer window otherwise the TCM is ignored and M is updated using only the observed transition. The value function V is computed using the posterior density and a revised policy is derived from V. The learner then moves to the next time step and the learning loop repeats until a termination point is reached. An example of a termination condition is when a specified end of learning time steps is reached.

Updating the working prior in a transfer window entails transferring information from the TCM to the working prior. Two techniques for accomplishing the transfer are presented below.

Transfer Techniques

There are at least two computational challenges in transferring information from a task class model to a working prior for a new task.

- *plausible model realisations:* we need to ensure that when we generate model realisations P from a posterior density, not only do the realisations reflect the data so far acquired in the form of experience on the new task but that they also satisfy the conditions specified in a TCM.
- *efficient dynamic programming back-ups during model transfer:* which is a problem when there is a need to match indirect expressions in a TCM and evaluation of a number of alternative models is required to obtain a satisfactory match.

Naive Transfer

In a naive transfer approach (see algorithm 1), we first update the working prior M with the observed transition $s \overset{a,r}{\rightsquigarrow} s'$ and then randomly sample a model realisation from the updated working prior constrained by ϕ. A variety of methods exists for sampling P from a constrained dirichlet prior in a constrained parameter MDP task environment (see for example (Sunmola and Wyatt, 2006)).

We next evaluate the sample P using dynamic programming to obtain a corresponding value function V. The sample is accepted if the estimated V satisfies the indirect expressions I in the TCM, within specified tolerances. If a sample is accepted, we increase the number of samples by 1 and add P to a list of plausible model realisations. When the size of the list is equal to required number of samples, we use the list to adjust the working prior M and estimate the posterior distribution $f(P|M, I)$ from the adjusted M.

Algorithm 1 Estimate $f(P|M,I)$ using Naive Sampling

```
Input: $M$, $s \overset{a,r}{\rightsquigarrow} s'$, $I$, $\phi$, and requiredSampleSize
Output: $f(P|M,I)$, revised $M$
 1: set noOfSamples = 0
 2: update $M$ with $s \overset{a,r}{\rightsquigarrow} s'$
 3: repeat
 4:     generate model realisation $P$ from a $\phi$-constrained prior $M$
 5:     evaluate $V$ for the model $P$ using dynamic programming.
 6:     if $V$ satisfies the indirect expressions in $I$ within specified tolerances then
 7:         increase noOfSamples by 1
 8:         add $P$ to a list of plausible model realisations.
 9:     end if
10: until (noOfSamples=requiredSampleSize)
11: adjust $M$ using the list of plausible model realisations.
12: estimate $f(P|M,I)$ corresponding to the adjusted $M$.
```

Perturbation-based Transfer

A probability perturbation method (see algorithm 2) is used to estimate the posterior distribution of $f(P|M,I)$ from pre-posterior densities $f(P| < s,a,r,s' >)$ and $f(P|I)$. It is easy to compute $f(P| < s,a,r,s' >)$ given a dirichlet prior density. However, since I is indirect, estimating $f(P|I)$ is not as straightforward. We estimate $f(P|I)$ through perturbations of the current probability model $f(P| < s,a,r,s' >)$. We start with a set of initial model realisations that is drawn from $f(P| < s,a,r,s' >)$ constrained by ϕ. We evaluate the value functions for the model realisations and use it to estimate the mismatch of the model realisations given I.

Algorithm 2 Estimate $f(P|M,I)$ using Probability Perturbation

```
Input: $M$, $\langle s,a,r,t \rangle$, $I$, $\phi$, $\delta$, and sample size $nSample$
Output: $f(P|M,I)$, revised $M$
 1: generate an initial $nSample$ model realisations from $f(P| < s,a,r,s' >)$ constrained by $\phi$
 2: obtain value estimates for the $nSample$ model realisations
 3: calculate mismatch of the value estimates given $I$.
 4: repeat
 5:     set $f(P|I) = (1-\delta) \times f(P| < s,a,r,s' >) + \delta \times f(P|M)$
 6:     combine pre-posteriors $f(P| < s,a,r,s' >)$ and $f(P|I)$ using a pre-posterior mixing algorithm to obtain $f(P|M,I)$
 7:     draw a new $nSample$ of model realizations from $f(P|M,I)$ constrained by $\phi$
 8:     obtain value estimates for the new $nSample$ of model realizations
 9:     calculate mismatch of the current value estimates given $I$.
10:     find an optimal $\delta*$ that minimises mismatch
11:     set $\delta = \delta*$
12: until ($I$ is matched to some desired level)
```

To improve on the match, we perturb the probability model $f(P| < s,a,r,s' >)$ used to generate the initial realizations rather than perturbing the initial realizations directly. This is done through a perturbation parameter γ to obtain an estimate for $f(P|I)$ as $(1-\gamma) \times f(P| < s,a,r,s' >) + \gamma \times f(P|M)$. We combine both pre-posteriors $f(P|I)$ and $f(P| < s,a,r,s' >)$ into a new model $f(P|M,I)$ using Journel's tau-model(Journel, July 2002) of the following type:

$$\begin{aligned} f(P|M,I) &= \frac{1}{1+x} \text{ with } x = b(\frac{c}{a})^{\tau}, \text{ where:} \\ b &= \frac{1-f(P| < s,a,r,s' >)}{f(P| < s,a,r,s' >)}, \; c = \frac{1-f(P|I)}{f(P|I)}, \\ a &= \frac{1-f(P|M)}{f(P|M)} \end{aligned} \tag{14}$$

τ allows modelling explicitly the full dependency between $< s,a,r,s' >$ and I. The assumption of standardised conditional independence is obtained when $\tau = 1$.

The combined probability model $f(P|M,I)$ is used to populate the next realisations. There may exist a value of δ, for which the perturbed model realisations matches I better than the initial realizations. We find the optimum model realisations by optimising the value δ.

EMPIRICAL RESULTS

To illustrate the benefits of transferring information from a task class model to a new task, we experiment on a task instance drawn from the task class model shown in Figure 1. The task instance is a two action, four states, chain task. Computing an optimal policy for the chain task when the model is unknown requires exploration control.

The actual transition probabilities for the chain task instance are as shown in table (1). Whilst the actual transition probabilities are unknown to the learner at the start of the task, the learner is aware of the parameters ϕ of the task class model which were set as follows $(0.91, 0.79, 0.8725, 0.6156, 0)$. Rewards of 1000 units accrue for transitions to state 4 under action b. Transitions to state 1 for each of the two actions attracts a reward of 10. All other transitions have zero rewards attached. The learner starts with a working prior of 1 for transitions to states 1, 2, and 3 and 20 for transitions to state 4 under each action.

We use a naive global sampling method with a sample size of 50 to obtain value estimates. Throughout, in the experiments, we adopt the usual standardised conditional independence assumption. Performance of the learning agent can be measured in several ways. To account for exploration and exploitation trade-off we measured the discounted total reward to-go at each point at each time step. More precisely, suppose the agent receives the following rewards $r_1, r_2, \ldots, r_t$ in a run of time length t. The reward to go at time t' is defined to be $\sum_{t' \geq t} r_t{'} \gamma^{(t'-t)}$. In Figure (3), we plot the discounted total reward-to-go for learning with and without model transfer in comparison to the optimal policy (assuming model of the task environment is known from start), as a function of time averaged over 10 runs. The transfer window was set to (0, 50) time steps. As expected, the

Table 1: State transition probabilities p_{ij}^a and p_{ij}^b.

p_{ij}^a	1	2	3	4
1	0.80	0.15	0.05	0.00
2	0.00	0.16	0.84	0.00
3	0.00	0.00	0.25	0.75
4	0.85	0.00	0.00	0.15

p_{ij}^b	1	2	3	4
1	0.20	0.60	0.15	0.05
2	0.90	0.10	0.00	0.00
3	0.70	0.00	0.30	0.00
4	0.20	0.00	0.00	0.80

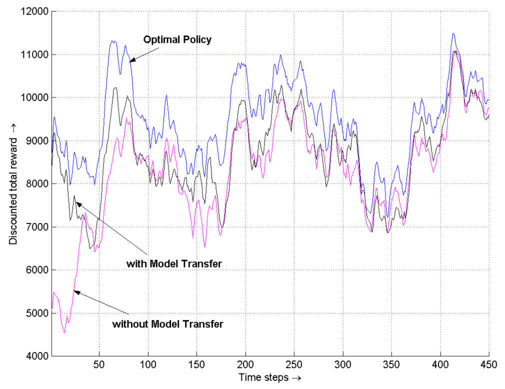


Figure 3: Plot of discounted total rewards over time for the chain task with and without model transfer

optimum policy gave the largest discounted total reward at the initial time steps, until the model of the task environment is learnt by the learning agent. The performance of the learning agent is better with model transfer using information in the TCM when compared with learning without model transfer under the same starting working prior settings. This simple experiment illustrates the potential benefits of model transfer for Markov decision tasks based on a task class model. The task class model provides a constraining influence on exploration early in learning for related task instances covered by the class model.

Conclusions and Future Work

We have studied a Bayesian framework to model transfer for Markov decision tasks in which information is transferred from a task class model to a working prior model. We presented two transfer techniques (naive and perturbation-based) that enable us to match the parameters of the working priors to the information in the task class model. The potential benefit of model transfer in the framework is illustrated with a simple example.

Work is in progress on several aspects of the framework. We are carrying out detailed empirical comparisons of the transfer techniques to not only study the effectiveness of the techniques with regards to discounted total reward to-go but also with respect to their efficiency in terms of computational time. One attraction of the perturbation-based approach is the ability to explicitly account for dependencies between the working prior and the task class model. Studying the significance of this attraction for Markov decision processes is an area of future work. Other important areas of future work includes studying other sampling-based model transfer techniques for the task class model framework, inferring the optimum transfer window when it is unknown to the learner, and understanding the sensitivity of learning through model transfer to variations in a task class model.

References

R. E. Bellman. *Adaptive control processes: A guided tour*. Princeton University Press, 1961.

D. S. Bernstein. Reusing old policies to accelerate learning on new MDPs. Technical Report 23, University of Massachusetts, Amherst, 1999.

J. Caers and T. Hoffman. The Probability Perturbation Method: A New Look at Bayesian Inverse Modeling. *Mathematical Geology*, 38(1):81–100, January 2006.

Richard Dearden. *Learning and Planning in Structured World*. PhD thesis, Department of Computer Science, University of British Columbia, Canada, 2000.

M. Duff. *Optimal learning: Computational procedures for Bayes-adaptive Markov decision processes*. Ph.D.Thesis, University of Massassachusetts Amherst, 2002.

Carlos Guestrin. *Planning Under Uncertainty in Complex Structured Environments*. Ph.D. Dissertation, Computer Science Department, Stanford University, Dept. of Computer Science, Stanford, 2003.

A. G. Journel. Combining knowledge from diverse sources: An alternative to traditional data independence hypotheses. *Mathematical Geology*, 34:573–596(24), July 2002.

Y. Liu and P. Stone. Value-function-based transfer for reinforcement learning using structure mapping. In *Proceedings of the Twenty-First National Conference on Artificial Intelligence AAAI-06*, Boston, MA, July 2006.

S. Mahadevan. Proto-value functions: Developmental reinforcement learning. *Proceedings of the Twenty Second International Conference on Machine Learning (ICML 05)*, 2005.

J. Martin. *Bayesian decision problems and Markov chains*. Wiley, New York, 1967.

P. Poupart, N. Vlassis, J. Hoey, and K. Regan. An analytic solution to discrete Bayesian reinforcement learning. In *Proc. Int. Conf. on Machine Learning*, To appear, Pittsburgh, USA, 2006. Morgan Kaufmann.

Bob Price. *Accelerating Reinforcement Learning with Imitation*. PhD thesis, University of British Columbia, 2003.

A. Sherstov and Peter Stone. Improving action selection in mdp's via knowledge transfer. In *Proceedings of the 20th National Conference on Artificial Intelligence AAAI-05*, Pittsburgh, USA, July 9-13 2005.

Satinder P. Singh. Transfer of learning by composing solutions of elemental sequential tasks. *Machine Learning*, 8:323–339, 1992.

M. Strens. A Bayesian framework for reinforcement learning. In *Proceedings of the Seventeenth International Conference on Machine Learning*, pages 943–950, 2000.

Funlade Sunmola and Jeremy Wyatt. Optimistic model selection in structure based reinforcement learning. In *Proceedings of the Sixth European Workshop on Reinforcement Learning*, 2003.

F. Sunmola and J. Wyatt. Bayesian Inference in Constrained Parameter Markov Decision Processes with Uncertain Transition Probabilities. In *Proceedings of the ECAI 2006 Workshop on Planning, Learning and Monitoring with Uncertainty and Dynamic Worlds*, 2006.

S. Thrun. Lifelong learning algorithms. In Sebastian Thrun and Lorien Y. Pratt, editors, *Learning To Learn*. Morgan Kaufmann, Boston, MA, 1997.

T. Wang, D. Lizotte, M Bowling, and D. Schuurmans. Bayesian sparse sampling for on-line reward optimization. In *Proc. Int. Conf. on Machine Learning*, 2005.

J. L. Wyatt. Exploration control in reinforcement learning using optimistic model selection. *International Conference on Machine Learning (ICML)*, 2001.