Clustering MDPs For Continual Transfer

Subramanian Ramamoorthy
Joint work with:
M. M. Hassan Mahmud, Majd Hawasly and Benjamin Rosman

School of Informatics
University of Edinburgh

April 30, 2013

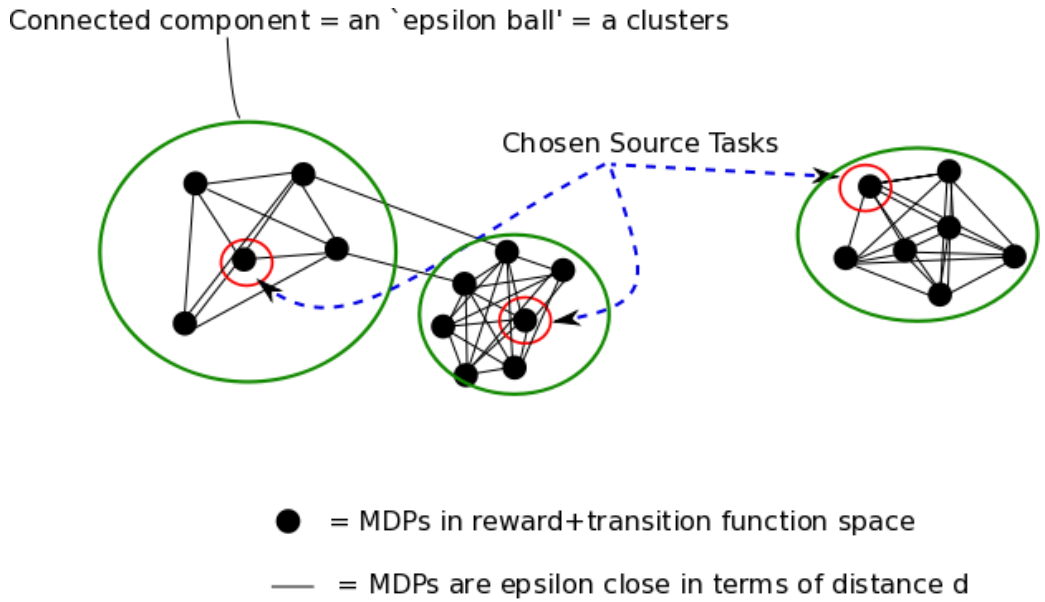
Outline

Problem of Continual Transfer

- **Continual:** The learner faces a never-ending sequence of MDPs $\mathcal{M}_1, \mathcal{M}_2, \cdots$.
- **Assume:** MDPs all defined on the same state and action space.
- **Transfer Objective:** RE-USE OPTIMAL POLICIES π_i^* of previous N MDPs – use π_i^* in $\mathcal{M}_{N+1}$ if it gives good/near optimal reward.
- **Problem:** N too high $\Rightarrow$ too much time wasted evaluating suitability of π_i^*, $\Rightarrow$ transfer useless/harmful.
- **Solution:** Two Steps...
  - **First: Reduce large number of previous tasks to a smaller, representative/landmark set of *source tasks*.**
  - **Second: Use the policies ρ_i of the sources to do policy re-use.**

Finding Source MDPs by Clustering

- **Clustering:** We cluster N previous MDPs into c clusters and use the set of CLUSTER CENTERS as the source tasks.



- **Cost of a Clustering A:** Regret of policy reuse (with respect to the TRUE OPTIMAL POLICY) when using the optimal policies of the cluster centers as input.

Our Approach

- **A Policy Reuse Algorithm:** EXP-3-TRANSFER algorithm; performance bound depends on number of policies to reuse.
- **Input Policies For EXP-3-Transfer:** Cluster *N* previous MDPs; optimal policies of the clusters centers are input.
- **Cluster Cost:** Regret of EXP-3-TRANSFER w.r.t. the TRUE OPTIMAL POLICY. Defining this requires a distance function...
- **Distance Between MDPs:** Upper bounds regret when using optimal policy of one MDP in the other.
- **Clustering Algorithm:** Solves the discrete/global optimization problem of finding the clustering with the lowest cost.

Related Work

- As far as we know, no prior work on clustering MDPs to extract a representative set.
- **Finding abstractions:** Lots of work,
  - MDP homomorphisms [1], [2], [3], [4], [5], [6]
  - Proto-value function based approach [7].
  - Scaling is an important issue.
- **Fundamental Difference:** We are interested in closeness in terms of the policies replicating optimal behavior; they are interested in closeness in terms of replicating transition/reward/value functions in the other MDP.
- **Policy Reuse:** Previous work [8] [9]. Our EXP-3 based algorithm is similar, but also automatically gives regret bounds.

Outline

Policy Reuse Problem

- **Given:** A set of policies $\rho_1, \rho_2, \cdots, \rho_c$ for a MDP.
- **Problem:** Balance reusing these policies and pure RL algorithm [8].
- **Our Solution:** Use EXP-3 bandit algorithm [10] with $c + 1$ arms:
  - c **arms:** One arm for each ρ_i.
  - $+1$ **arm:** for pure RL (such as Q-learning etc.).
- **Each step** Choose an arm according to EXP-3 strategy and run it for an episode.
- **Arm/Policy Payoff:** Total discounted reward observed in the episode.

EXP-3-Transfer

Algorithm 1 EXP-3-Transfer$(\mathcal{M}, \{\rho_1, \rho_2, \cdots, \rho_c\}, \beta, T)$

1: **Input:** MDP $\mathcal{M}$, arms 1 to c: the source policies $\rho_1, \cdots, \rho_c$ and EXP-3 parameters β and T.
2: **Initialize:** Set $w_i(1) = 1$, let $x_i(t)$ be the per-episode payoff of the arms, Q-learning policy as the $c + 1^{th}$ arm.
3: **for** $t = 1$ to T **do**
4: Set $p_i(t) = (1 - \beta)\frac{w_i(t)}{\sum_{i=1}^{c+1} w_i(t)} + \frac{\beta}{c+1}$.
5: Select arm i_t for step t to be i with probability p_i.
6: Run chosen arm for one-episode, and observe discounted payoff $x_{i_t}(t)$.
7: Set $\hat{x}_j(t) \leftarrow x_j/p_j(t)$ if $j = i_t$; otherwise $\hat{x}_j(t) \leftarrow 0$. Update $w_j(t + 1) \leftarrow w_j(t) \exp[\beta \hat{x}_j(t)/(c + 1)]$.
8: **end for**

Performance

- **Regret Bounds:** EXP-3 regret bounds apply:

$$G_{\max} - \mathbb{E}[X] \leq \sqrt{e-1}\sqrt{G(c+1)\ln(c+1)}$$
$$\leq 2.63\sqrt{(c+1)\ln(c+1)G}$$

  where X is the payoff of EXP-3-Transfer $G_{\max}$ is the payoff of the best of the $c+1$ arms and G is an upper-bound on $G_{\max}$.

- Define

$$\mathbf{g(c)} \triangleq \mathbf{2.63}\sqrt{\mathbf{(c+1)\ln(c+1)G}}$$

Outline

To do:

- **Step 1:** Define a class of distance functions d between MDPs, which upper bounds regret when using optimal policy of one MDP in another.
- **Step 2:** Derive the cost of a clustering **A** with c clusters using distance d & EXP-3-Transfer regret bound $g(c)$.
  - **What is this cost ?** This is the regret of EXP-3-Transfer when using the c source tasks as arms.
- **Step 3:** Construct a clustering algorithm to find the cluster minimizing cost.

MDP Clustering: The Basic Idea

- Divide the *N* previous MDPS into *c* clusters and then use the cluster centers as source tasks.
- Optimal policy of source tasks = input to EXP-3-Transfer.
- EXP-3-Transfer will perform well if the source tasks are GOOD REPRESENTATIVES of the *N* previous tasks IN TERMS OF POLICY REUSE.
- **in terms of policy reuse... ?** The optimal policy of the cluster center should perform well in the other MDPs that are members of that cluster.
- **So:** We need a distance function that measures this aspect.

Illustration

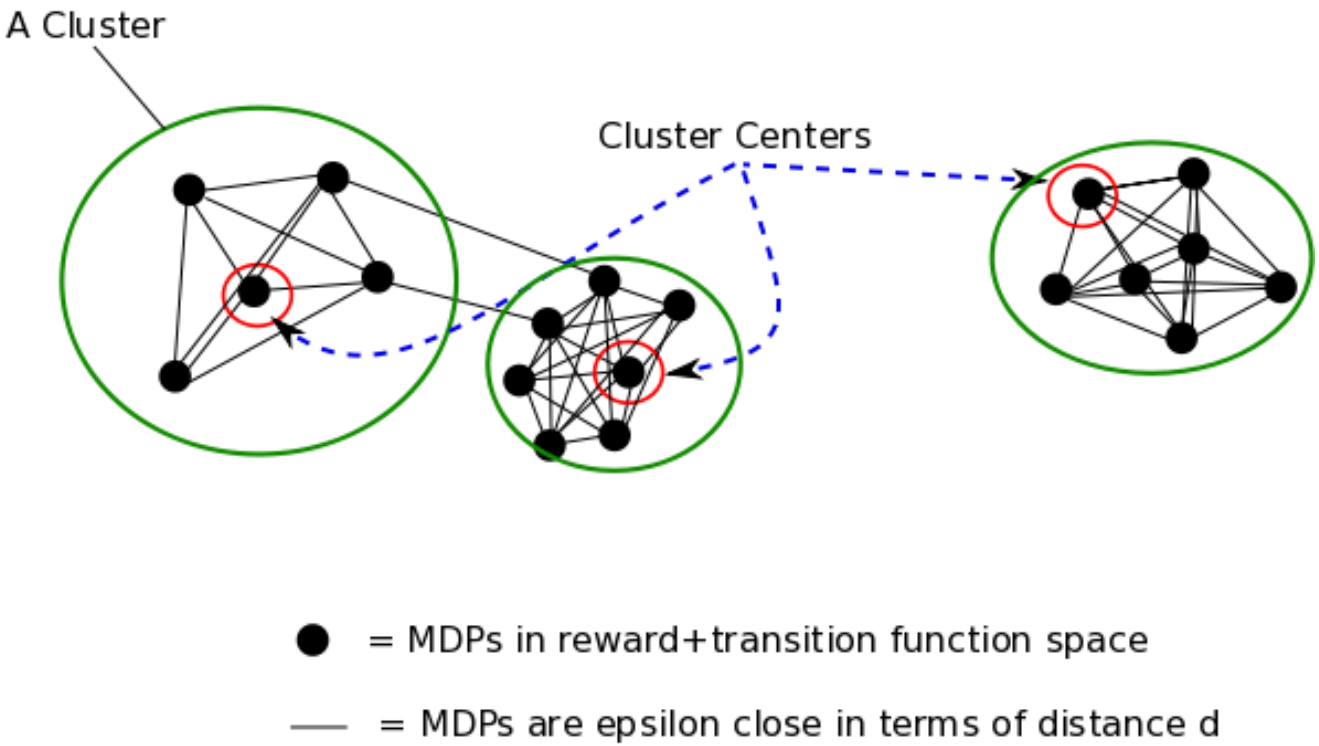


Figure: The cluster centres are such that their optimal policies perform well in the other MDPs of the cluster.

What Should a Distance Measure ?

- **Our aim:** Use optimal policy of one MDP in another another with same state-action space.
- **This Implies:** The distance function between two MDPs $\mathcal{M}_1, \mathcal{M}_2$,

$$d_V(\mathcal{M}_1, \mathcal{M}_2) \triangleq \max\{\mathbb{E}_{In_1}[V_1^*(s) - V_1^{\pi_2^*}(s)], \mathbb{E}_{In_2}[V_2^*(s) - V_2^{\pi_1^*}(s)]\}$$

- **Problem:** Not a metric, so bad for clustering and deriving cost function later on.

VPL Metric

- **New Class Of Distances:** A Value Preserving Lipschitz (VPL) metric $d(\mathcal{M}, \mathcal{M}')$
- **Metric Like Properties:** Should be coincident, symmetric and positive.
- **Instead of Subadditivity...:** Is Lipschitz-subadditive:

$$d(\mathcal{M}, \mathcal{M}') \leq K[d(\mathcal{M}, \mathcal{M}'') + d(\mathcal{M}', \mathcal{M}'')]$$

- **... and Value Preserving:** For some bounded, monotonic function k

$$d(\mathcal{M}, \mathcal{M}') < \epsilon \Rightarrow d_V(\mathcal{M}, \mathcal{M}') < k(\epsilon)$$

Example VPL Metric

- **Reward+Transition Based:** In our experiments use the following VPL metric :

$$d(\mathcal{M}, \mathcal{M}') \triangleq \max_{s,a} \max\{|R(s,a) - R'(s,a)|, \|T(\cdot|s,a) - T'(\cdot|s,a)\|_1\}$$

where $\|\cdot\|_1$ is the L_1 norm of the two probability vectors.

- **Can show:** This is VPL with Lipschitz constant 1, and $k(\epsilon) \triangleq \frac{\epsilon(1+\gamma R_{\max})}{(1-\gamma)^2}$.
- **Too simple:** Does not look at structure of policies – we have a improved distance based on ϵ-optimal policies.
- Our results are valid for any VPL metric.

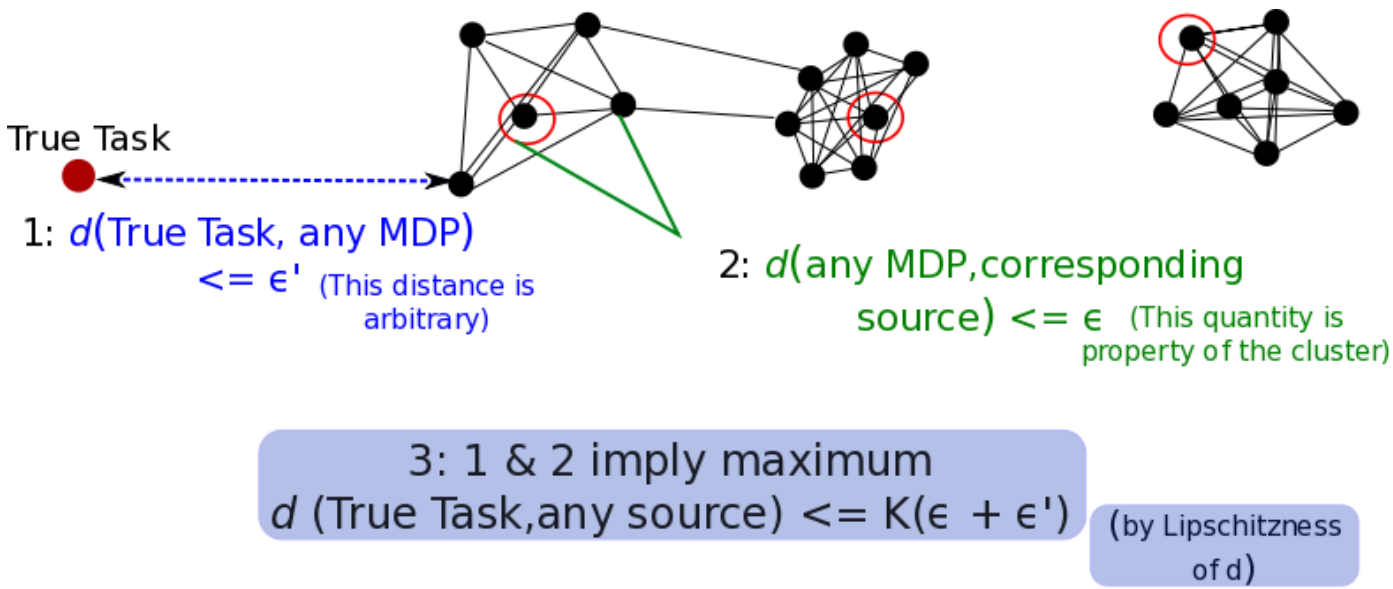
Clusters of MDPs

- **Given:** N different MDPs $\mathcal{M}_1, \mathcal{M}_2, \cdots, \mathcal{M}_N$.
- **To Do:** Split into c clusters $\mathbf{A} = \{A_1, A_2, \cdots, A_c\}$.
- **Why ?** To obtain a set c of source tasks, $\mathcal{J}_1, \mathcal{J}_2, \cdots, \mathcal{J}_c$, where

$$\mathcal{J}_i \triangleq \arg\min_{\mathcal{M} \in A_i} \max_{\mathcal{M}' \in A_i} d(\mathcal{M}, \mathcal{M}')$$

each $\mathcal{J}_i$ is a CLUSTER CENTER.

Cost Function For Cluster **A**: Illustrated



- **a:** 3 $\Rightarrow$ Regret of any source w.r.t. true task $\leq k[K(\epsilon + \epsilon')]$ (by value preservation of d).
- **b:** Regret of EXP-3-Transfer with respect to any source $\leq g(c)$.
- **a & b $\Rightarrow$:** Regret of EXP-3-Transfer w.r.t. TRUE TASK $\leq g(c) + k[K(\epsilon + \epsilon')] \triangleq cost(\mathbf{A})$.

A Cost Function For Clusters

- **Step 0:** Assume clustering $\mathbf{A} = \{A_1, A_2, \cdots, A_c\}$ with maximum diameter of any cluster $\max_{j,\mathcal{M}\in A_j} d(\mathcal{J}_j, \mathcal{M}) \leq \epsilon$.
- **Step 1:** Regret of Exp-3-Transfer given c source tasks $\leq 2.63\sqrt{(c+1)\ln(c+1)G} \triangleq g(c)$
- **Step 2:** True task $\mathcal{M}_{N+1}$ can be (in the worst case) ϵ' from any $\mathcal{M}_i$ (ϵ' symmetric and hence arbitrary/irrelevant).
- **Step 3:** $\forall j, d(\mathcal{J}_j, \mathcal{M}_{N+1}) \leq K(\epsilon + \epsilon')$ (because d is K-Lipschitz).
- **Step 4:** $\forall j, d_V(\mathcal{J}_j, \mathcal{M}_{N+1}) \leq k(K(\epsilon + \epsilon')$ (because d is value preserving.
- **Finally:** $d_V(\mathcal{J}, \mathcal{M}_{N+1})$ = regret of $\mathcal{J}$, overall regret of Exp-3-Transfer using $\mathbf{A}$ = $g(c) + k(K(\epsilon + \epsilon') \triangleq \text{cost}(\mathbf{A})$.

Clustering Algorithm

- **Objective:** Find the cluster **A** minimizing cost(**A**).
- **Problem:** It's a combinatorial optimization problem.
- **Solution:** We use a novel version of Simulated Annealing.
- **Main Issue With SA:** Setting the temperature schedule.
- **Our Approach:** No need to set the temperature schedule.

Simulated Annealing

- A standard way to solve global optimization.
- Uses Metropolis-Hastings (MH) chain as a inner step.
- For each value of $\lambda_1, \lambda_2, \cdots$, run MH with target $\pi(\mathbf{A}) \triangleq \lambda_i^{-cost(\mathbf{A})}$ for T_i steps.
- $\lambda_i < \lambda_{i+1}$, $\lim_{i\to\infty} \lambda_i = \infty$ are the **temperatures**.
- $T_1, T_2, \cdots$ is the **temperature schedule**.
- **Key Idea:** Smaller $\lambda_i \Rightarrow$ high exploration; larger $\lambda_i \Rightarrow$ stick with the best so far.
- **Main Problem:** Wrong schedule $\Rightarrow$ insufficient exploration and non-convergence.

Search-Clusterings: SA Without a Schedule

- **Key Idea:** Keep trying different exploration rates λ_i, with rate chosen stochastically.
- Set $\Lambda = \{\lambda_1, \cdots, \lambda_n\}$, $\lambda_i < \lambda_{i+1}$.
- $\lambda_m \cdots, \lambda_n$ should satisfy,

$$\sum_{i=m}^{n} \lambda_i^{-cost(\mathbf{A}^*)} \gg \sum_i \sum_a \lambda_i^{-cost(\mathbf{A})} \tag{1}$$

- Run MH on state space $\Lambda \times \{\mathbf{A}_i\}_{(i)}$; target function $\pi(\lambda, \mathbf{A}) \triangleq \lambda^{-cost(\mathbf{A})}$ (proposal distribution described in paper).
- Condition (1) ensures $\pi(\lambda, A)$ **concentrates around optimal clustering $\mathbf{A}^*$.**
- State space $\Lambda \times \{\mathbf{A}_i\}_{(i)}$ ensures **MH chain keeps exploring**.
- Ergodic Markov chain convergence results $\Rightarrow$ MH chain converges to π and hence we find $\mathbf{A}$ (at a geometric rate).

Final Algorithm

Algorithm 2 Continual-Transfer($d, J, \beta, T_1, term$)

1: **Input:** A VPL metric d, clustering period J, EXP-3-Transfer parameters β, T_1, Search-Clustering termination condition *term*.
2: **Initialize:** Initial clustering $\mathbf{A} = \emptyset$, collection of previous MDPs M.
3: **for** $h = 1$ to ∞ **do**
4: Get unknown MDP $\mathcal{M}_h$ from the environment and run EXP-3-Transfer($\mathcal{M}_h$, *sourcePol*($\mathbf{A}$), β, T_1).
5: Set M $\leftarrow$ M $\cup \{\mathcal{M}_h\}$
6: **if** $h \mod J = 0$ **then** $\mathbf{A}$ = Search-Clusterings (M, d, *term*).
7: **end for**

Outline

Experiment Setup

- Two sets of experiments.
- **First Set:** We clustered a simple set of domains by themselves to illustrate our clustering algorithm.
  - Our algorithm gives reasonable clusters
- **Second Set:** We ran the full algorithm on a set of corridor domains.
  - Clustering does not hurt when the number of previous tasks are small.
  - Helps significantly when the number of previous tasks is large.

Simple Chain Domain

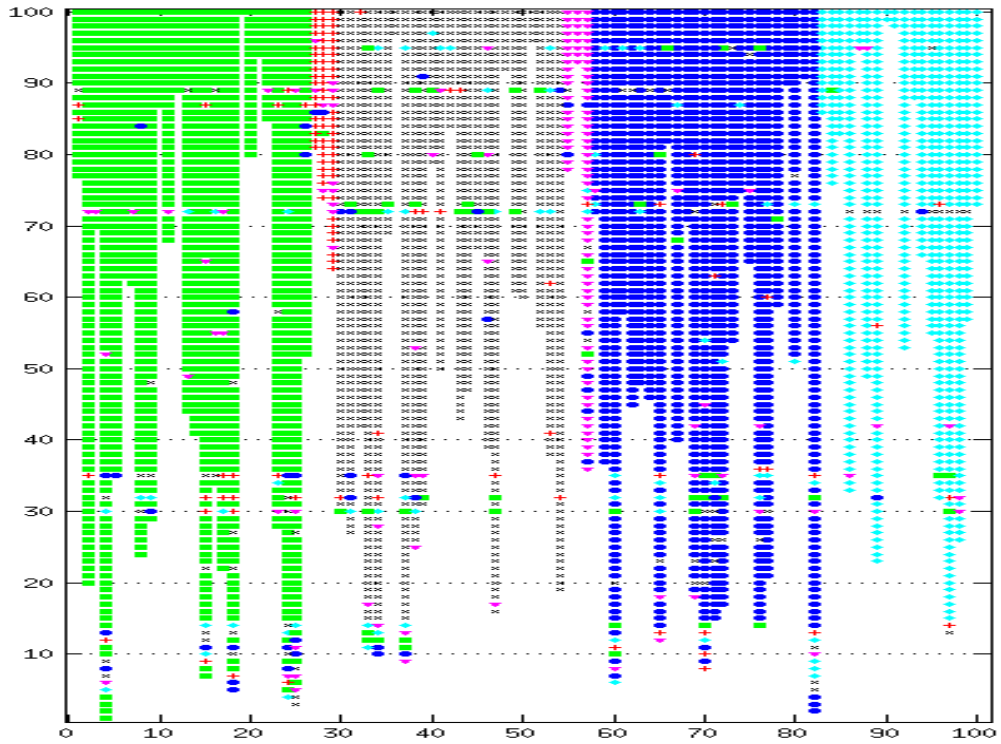


Figure: Clustering a set of simple chain domains.

Windy Corridor Domain

- A set of gridworld domains, with 10 corridors and wind.

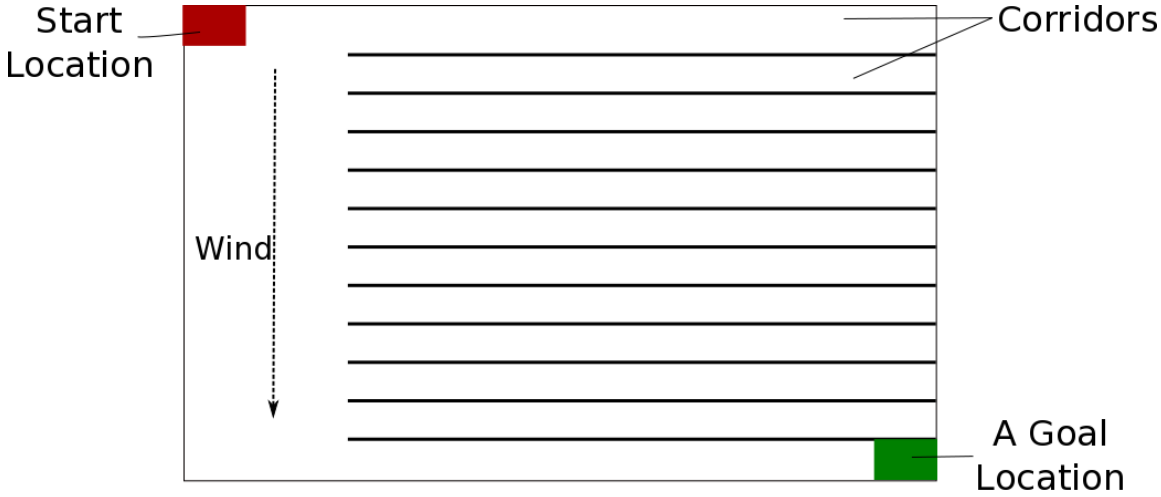


Figure: Illustration of the Domain. Tasks vary in terms of the corridor the goal state is at, the reward at goal and per step, the transition function, and the wind.

- We transferred from 10,20,30 and 100 tasks to 10 different target tasks.
- Results averaged over 10 trials.

Results: Target Task 1

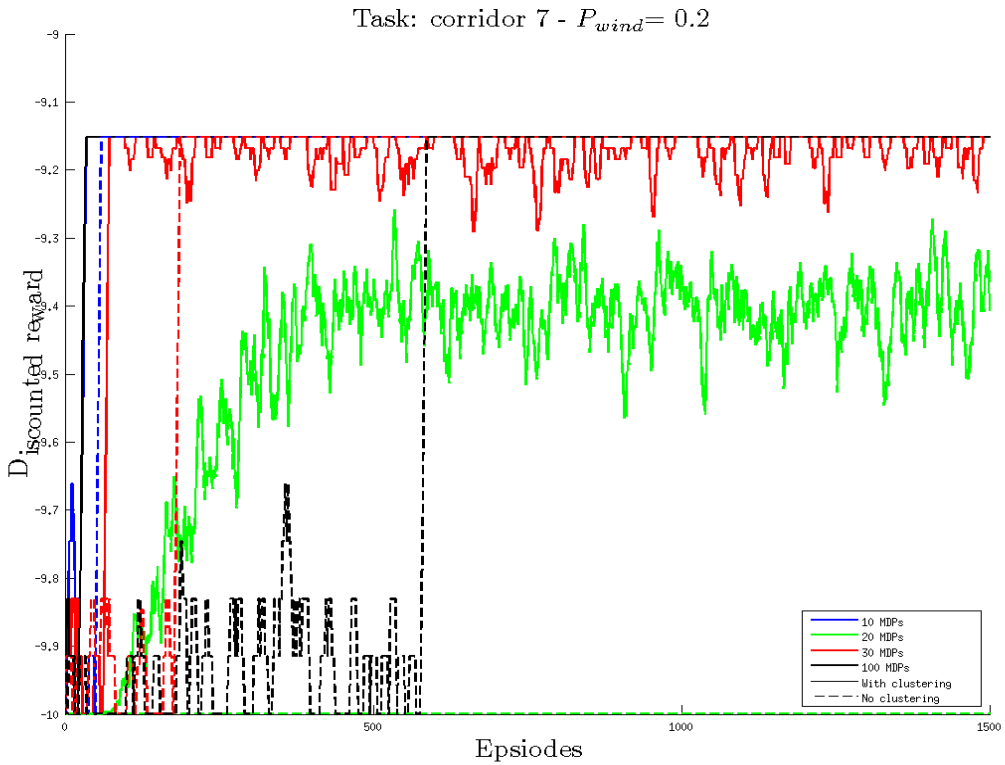


Figure: Target Task 1, first 1000 episodes. Clustering outperforms non-clustering, more so when the number of previous tasks increase.

Results: Target Task 1

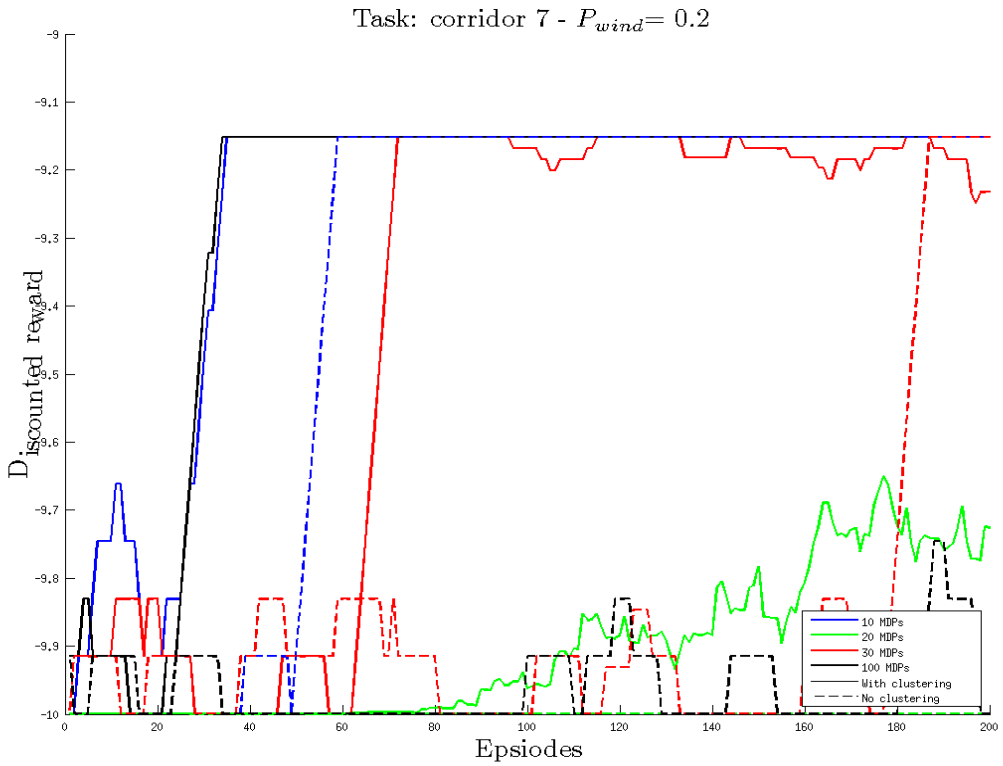


Figure: Target Task 1, first 200 episodes. Clustering outperforms non-clustering, more so when the number of previous tasks increase.

Results: Target Task 2

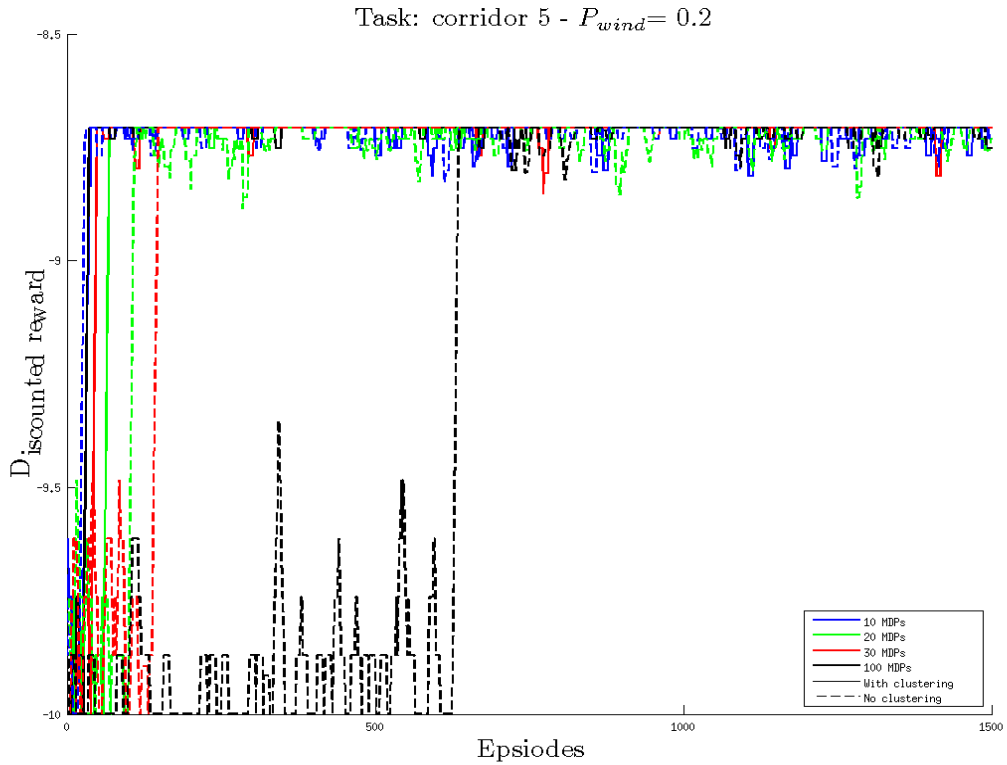


Figure: Target Task 2, first 1000 episodes. Clustering outperforms non-clustering, more so when the number of previous tasks increase.

Results: Target Task 2

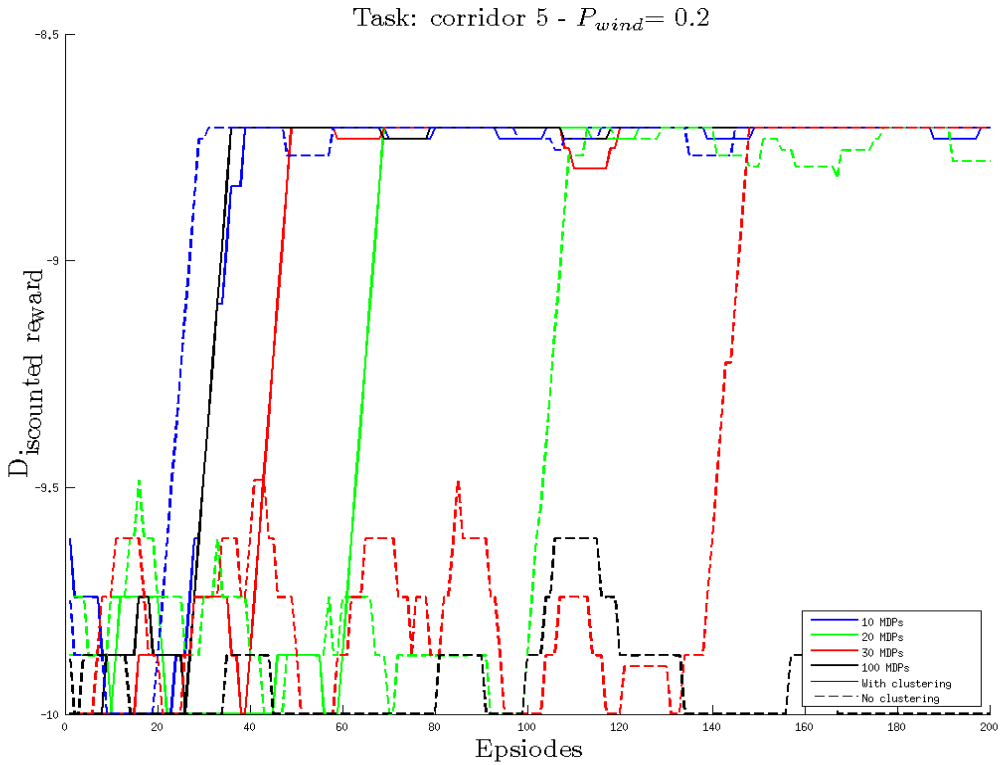


Figure: Target Task 2, first 200 episodes. Clustering outperforms non-clustering, more so when the number of previous tasks increase.

Results: Target Task 6

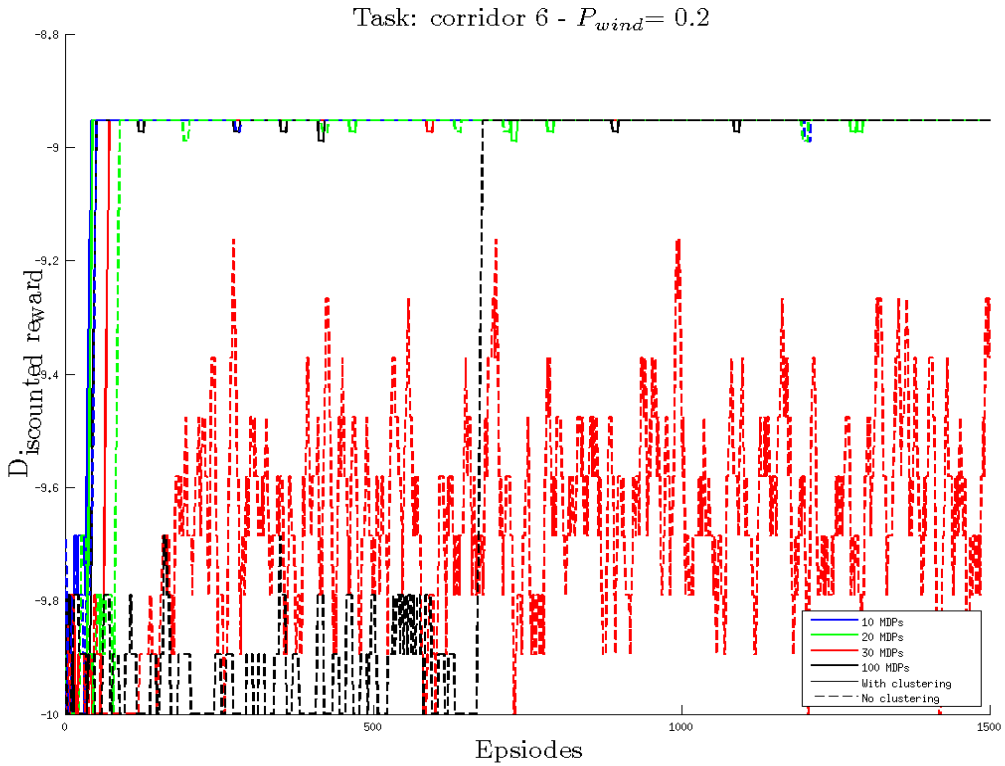


Figure: Target Task 6, first 1000 episodes. Clustering outperforms non-clustering, more so when the number of previous tasks increase.

Results: Target Task 6

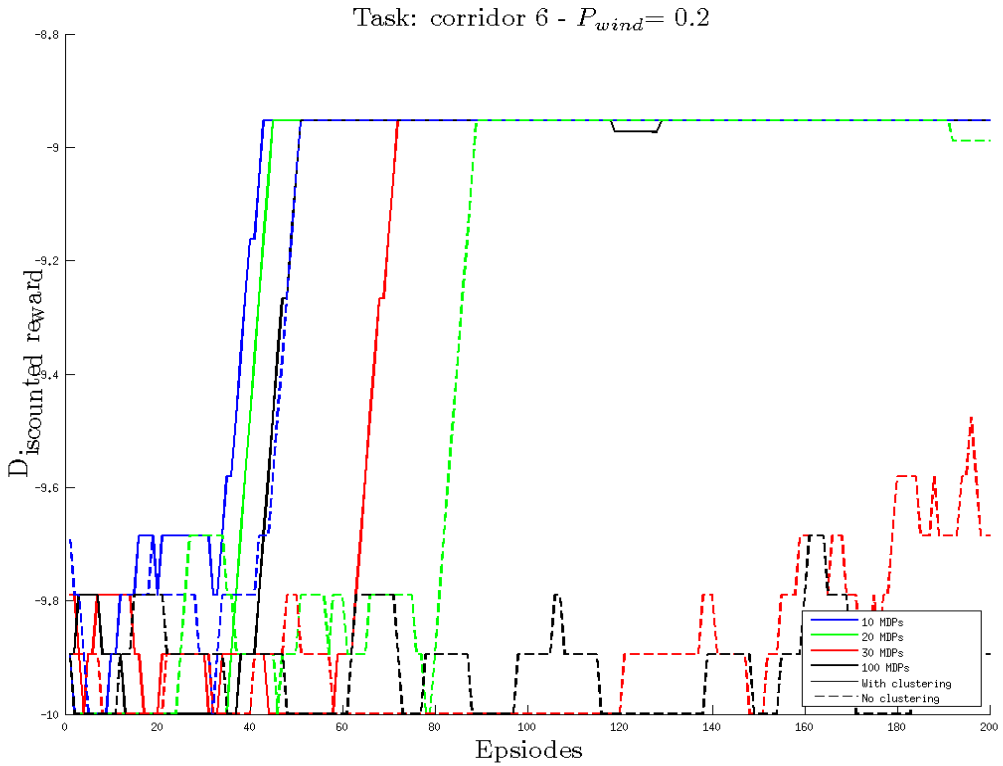


Figure: Target Task 6, first 200 episodes. Clustering outperforms non-clustering, more so when the number of previous tasks increase.

Results: Target Task 10

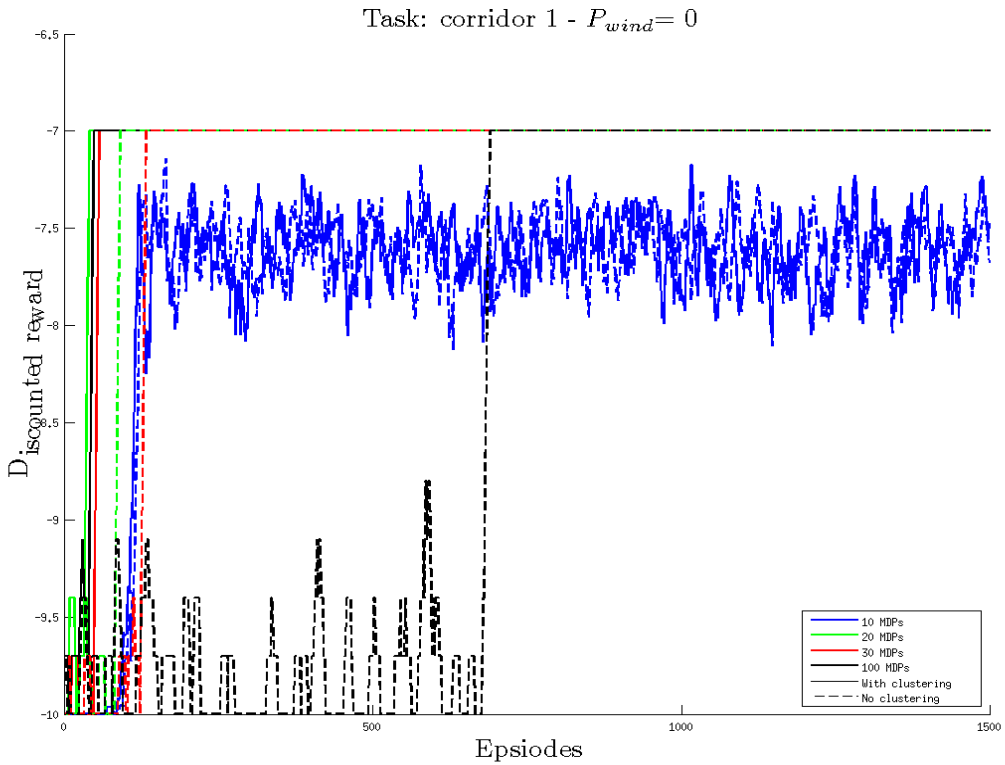


Figure: Target Task 10, first 1000 episodes. Clustering outperforms non-clustering, more so when the number of previous tasks increase.

Results: Target Task 10

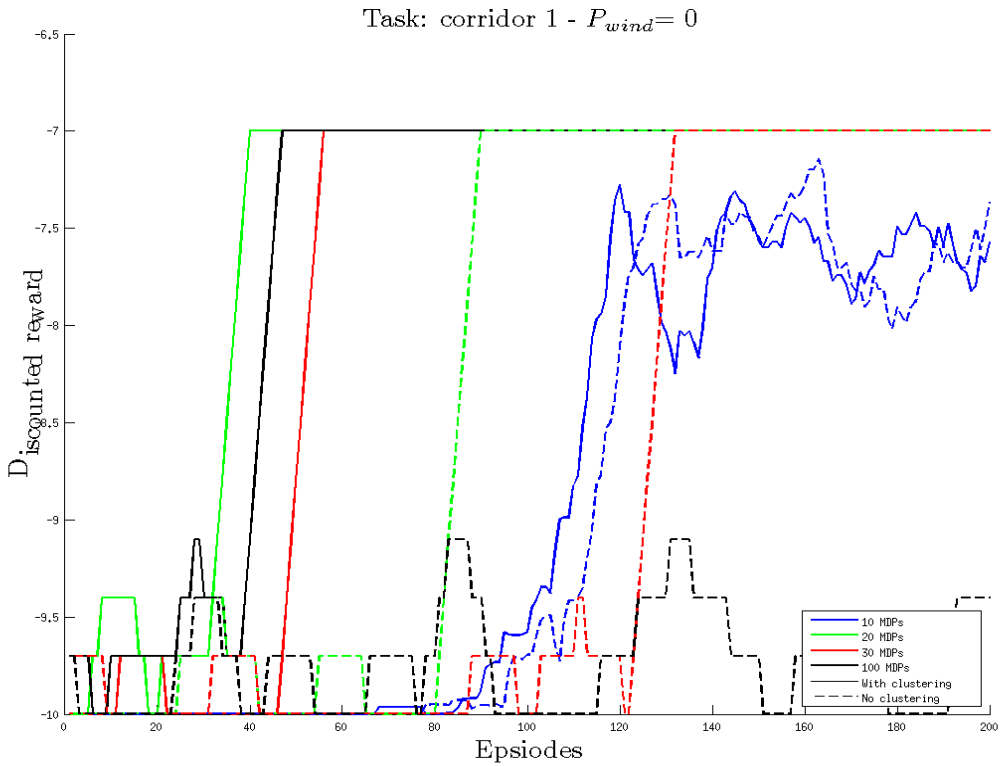


Figure: Target Task 10, first 200 episodes. Clustering outperforms non-clustering, more so when the number of previous tasks increase.

Outline

Future Work

- **VPL Metric:** Reward/transition based metric too simple - should explore ones that look at the structure of optimal policies.
- **Topology over MDPs:** Are there more effective ways of putting topology on the space of MDPs, possibly involving hierarchies ?
- **Continous Space:** Possible – none of our algorithms rely on the discreteness of state-action space – need to 'write it down'.
- **Simulated Annealing:** Schedule-less SA of independent interest.
- **Experiments:** Many more and on much more complex problems.


Balaraman Ravindran and Andrew G. Barto.
SMDP homomorphisms: An algebraic approach to astraction in semi-markov decision processes.
In *Proceedings of the International Joint Conference on Artificial Intelligence*, 2003.


Norm Ferns, Prakash Panangaden, and Doina Precup.
Metrics for finite markov decision processes.
2004.


Balaraman Ravindran.
Relativized hierarchical decomposition of markov decision processes.
Decision making: neural and behavioural approaches, 42:465–488, 2013.


George Konidaris and Andrew G. Barto.
Building portable options: skill transfer in reinforcement learning.

In *Proceedings of the 20th International Joint Conference on Artificial Intelligence*, 2007.


Jonathan Sorg and Satinder Singh.
Transfer via soft homomorphisms.
In *Proceedings of the 8^{t}h International Conference on Autonomous Agents and Multiagent Systems*, 2009.


Pablo Samuel Castro and Doina Precup.
Using bisimulation for policy transfer in mdps.
In *Proceedings of the the 24th AAAI Conference on Artificial Intelligence*, 2010.


E. Ferrante, A Lazaric, and M. Restelli.
Transfer of task representation in reinforcement learning using policy-based protovalue functions.
In *Proceedings of the 7^{t}h International Confernce on Autonomous Agent And Multiagent Systems*, 2008.


Fernando Fernandez, Javier Garcia, and Manuela Veloso.
Probabilistic policy reuse in a reinforcement learning agent.
In *Proceedings of the 5th International Conference on Autonomous Agents and Multiagent Systems*, 2006.


Fernando Fernandez, Javier Garcia, and Manuela Veloso.
Probabilistic policy reuse for inter-task transfer learning.
Robotics and Autonomous Systems, 58:866–871, 2010.


Peter Auer, Nicolo Cesa-Bianchi, Yoav Freund, and Robert E. Schapire.
The nonstochastic multiarmed bandit problem.
SIAM Journal on Computing, 32:48–77, 2002.