Transfer learning across heterogeneous robots with action sequence mapping

Balaji Lakshmanan and Ravindran Balaraman

***Abstract*— Transfer learning refers to reusing the knowledge gained while solving a task, to solve a related task more efficiently. Much of the prior work on transfer learning, assumes that identical robots were involved in both the tasks. In this work we focus on transfer learning across heterogeneous robots while solving the same task.The action capabilities of the robots are different and are unknown to each other. The actions of one robot cannot be mimicked by another even if known. Such situations arise in multi-robot systems. The objective then is to speed-up the learning of one robot, i.e., reduce its initial exploration, using very minimal knowledge from a different robot. We propose a framework in which the knowledge transfer is effected through a pseudo reward function generated from the trajectories followed by a different robot while solving the same task. The framework can effectively be used even with a single trajectory. We extend the framework to enable the robot to learn an equivalence between certain sequences of its actions and certain sequences of actions of the other robot. These are then used to learn faster on subsequent tasks. We empirically validate the framework in a rooms world domain.**

I. INTRODUCTION

One of the drawbacks associated with the reinforcement learning paradigm is the initial period of nearly random exploration while the robot looks for the goal. One approach for addressing this problem is to adopt a transfer learning framework. In a transfer learning setting, the knowledge acquired in solving one task is used to bootstrap the solving of another task. There have been several approaches proposed for transfer learning. In most of the approaches, a mapping is used to relate the new task to the task for which a policy had been learned (Torrey, Walker, Shavlik and Maclin 2005; Taylor, Whiteson, Stone 2007). In few others, a mapping between the base actions of the robots or the transitions is learnt and re-used. They also require substantial amount of accurate transition information collected in the source and target tasks (Taylor, Kuhlmann and Stone 2008). In all the earlier work on transfer learning the assumption has been that the same (or an equivalent) robot is involved in both the tasks. The problem of transfer between robots with different action capabilities has not received much attention in the literature.

In multi-robot systems, the robots have different action capabilities. The action capabilities may be unknown to each other or one robot may not be able to imitate another robot. Hence, even if the policy (state to action mapping) to solve the task is available from another robot, it cannot be used as the action capabilities are different. If the action models of the two robots are known, a complete mapping between the base actions may not always be possible or they would often need to be hand coded. Even if such a mapping is possible, it may not be precise. It is hard to obtain an exact action model of a robot as some amount of stochasticity is always involved and might require several self calibrations and result in control related problems.

Our work is primarily related to transfer learning across heterogeneous robots when the action models of the robots are different and are either unknown to each other or a mapping between their base actions do not exist. The robots do not have any prior knowledge of the environment. The objective is to speed-up learning, i.e., reduce initial exploration, with very minimal prior information (single trajectory) from a different robot. The focus is to solve the task quickly and not the optimality of the solution. In general, for any robot, it might take many exploration and exploitation steps in order to solve the task. The knowledge gained by one robot while solving a task in an environment can be reused by a second robot for learning a policy faster and solving the same task in the same environment. For instance, the first robot might have a circular action model and move in arcs while the second robot has a linear action model as seen in *Figure 2*. Given that the first robot has solved the task and learnt a path, our aim is to use the knowledge with the second robot and accelerate the learning to solve the same task. The framework we propose can be used for multi-robot system including different robotic arms or robotic arms with different degrees of freedom as shown in *Figure 1*.

We present a pseudo reward function framework for the second robot based on the knowledge obtained from the first robot. The second robot also learns to map its action sequences to certain sequence of actions of the first robot and uses this mapping while solving subsequent task. There are many real world applications of the work like quick response robots, search and rescue systems, etc., where gathering information would be expensive.

The rest of the paper is structured as follows: the next section gives the background of the problem. Then we describe our solution framework. After that, we present the experimental set-up and results that serve to support our solution framework. We then review some of the recent transfer learning work followed by the discussion and directions for future work. We will be using the terms, agent and robot interchangeably, for better understanding in appropriate sections.

Balaji Lakshmanan is student with the Department of Computer Science and Engineering, Indian Institute of Technology Madras, India balajil@cse.iitm.ac.in

Ravindran Balaraman is faculty with the Department of Computer Science and Engineering, Indian Institute of Technology Madras, India ravi@cse.iitm.ac.in

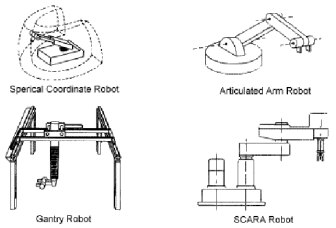


Fig. 1. Different Robotic Arm
Image courtesy: www.osha.gov

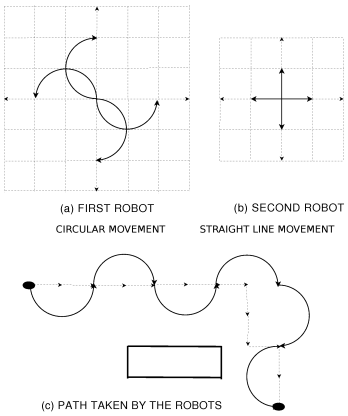


Fig. 2. Action models of the two robots and their respective paths to reach the goal. The actions of one robot cannot be imitated by the other robot even if it they are known.

II. Background

A. Reinforcement Learning

Markov decision process (MDP) is the modeling paradigm of choice for reinforcement learning problems. An MDP is quadruple {S, A, T, R}, where S is the finite set of possible states of the agent, A is the finite set of actions the agent can take, T(s, a, s') is the transition function which defines the probability of a transition from state s $\in$ S to $s' \in$ S when action a $\in$ A is taken. R(s, a) is the reward function, which defines an expected immediate reward for taking action a $\in$ A in state s $\in$ S. An MDP defines a formal model of an environment that an agent can interact with and learn about. In the reinforcement-learning framework, the agent has knowledge of its state and its action spaces, but does not have knowledge of the transition and reward functions. The task of the agent is encoded in the reward function and the objective of the agent is to obtain a policy by interacting with the environment over a period of time.

In the MDP formalism, the objective of the agent is to learn to maximize the expected value of reward received over time. It does this by learning an optimal mapping from states to actions defined as policy $\pi : S \rightarrow A$.

$Q^{\pi}(s,a) = E_{\pi} \left\{ \sum \gamma^{i} r_i | s_0 = s, a_0 = a \right\}$

where, $\gamma \in [0,1)$ is the discount factor and r_i is the reward from the environment at step i.

B. Transfer Learning

Transfer Learning has been traditionally observed as a method of utilizing the knowledge gained by solving one or more tasks, to speed up the learning in a related but different task. Speeding up the learning is important as the learner takes large number of steps to reach an unknown goal state in an environment with unknown dynamics and thus comes up with a feasible policy. If there are two MDPs with the same state and action space and related transition and reward functions, transfer learning reuses the knowledge gained from the first MDP with the second MDP, to obtain a feasible or nearly optimal policy, faster.

Our work is aimed at using the knowledge gained by one agent having one action model with another agent having a different action model, given the same environment and task. Hence, the two MDPs differ in their action space and transition function while the state space and reward function remain the same. We aim to rapidly learn and solve a task using very minimal prior knowledge from another agent with a different action model. [1]

III. Solution Framework

A. Problem Space

As in most transfer learning frameworks, the first agent is assumed to have solved the task. However, it is not necessary for the first agent to have learnt a value function or a policy. The agent is assumed to have knowledge of the sequence of states in the respective order it had covered to reach the goal. If the agent had learnt a policy, the sequence of states can be generated from the policy. The policy may or may not be optimal and the agent could have learned the policy using any algorithm. The second agent needs to solve the same task solved by the first agent, in the same environment. It has a different action model compared to first agent and has no knowledge of the state connectivity of the environment. It has only the state perception and can identify its current state. It also has no knowledge of the action model of the first agent and it doesn't understand the action capabilities of the first agent.

B. Knowledge Based Pseudo Rewards

For the first agent, there is an MDP $\{S_t, A_t, T_t, R_t\}$ and a task for which either a solution or a policy has been learnt. For the second agent there is another MDP {S, A, T, R} and a task for which a solution needs to be learnt. The state spaces S_t and S are the same. The reward functions R_t and R encode the same task. The action models A_t and A and transition function T_t and T are different. From a given start state s_{t0}, a trajectory consisting of a sequence of state action pairs $\{(s_{t0}, a_{t0}),(s_{t1}, a_{t1})...(s_{tn}, a_{tn}),s_{tg}\}$ is known by the first agent or generated using the first agent's policy and is transferred to the second agent to reach the goal state s_{tg}. s_{t0}, s_{t1}, .., s_{tn} are intermediate states visited in that order, by the first robot to reach the goal state. The action taken at each intermediate state is transferred as a label and is used for action sequence mapping.

For the second robot, we now introduce a pseudo reward function based on the trajectory of the first robot. This pseudo

[1]The reward function can be different as long as the encoded task of the agent remains the same.

reward function is inversely proportional to the distance of the intermediate states from the goal state in the sequence. This pseudo reward function is combined with the regular reward function from the environment and used during exploration. This accelerates the learning and helps to solve the task faster and reach the goal. The pseudo reward is defined as follows:

$$r'(s) = \begin{cases} iK & \text{if } s = s_{ti} \\ 0 & \text{otherwise} \end{cases}$$

where, K is a constant chosen to ensure that the pseudo reward is much smaller than the tasks reward functions. The pseudo rewards are linearly scaled depending on the order of the state in the transferred sequence. Now the modified Q function is defined as:

$Q^{\pi'}(s,a) = E_{\pi'} \{\sum \gamma^i (r'_i + r_i)\}$.

This can be used with any of the reinforcement learning methods to solve the task. We have used sarsa(λ), shown in *Algorithm 1*.

The transfer learning framework we propose does not require the first agent to learn a value function or a policy for solving the task or a transition model. The pseudo reward function proposed is characterized by the order of states in the trajectory transferred and does not depend on policy or value function. The framework is not bound to a deterministic environment and can be extended to stochastic environments as the pseudo reward framework for any given transition is dependent on the destination state and not on state action pair.

C. Action Sequence Mapping

One way of further speeding up learning in future transfer is to use the learnt knowledge to establish mapping between the actions of the two robots. With robots that are very different in their capabilities, it is difficult to establish a one to one correspondence between the actions. Hence we look to mapping fragments of policies, or action sequences of the first agent to action sequences of the second agent. It is easier to quantify what such a correspondence implies in a deterministic world. Two action sequences correspond to each other, if starting from the same state, say x, executing the action sequences by the respective robots will result in them ending up at the same state, say y. This notion has to be extended to preservation of state distributions in a stochastic environment, and is harder to establish.

During transfer, let a sequence of state, action index pair $\{(s_{t0},a_{t0}),(s_{t1},a_{t1}). ..(s_{tn},a_{tn}),s_{tg}\}$ be framed by the first robot and transferred to the second robot. The second robot, on reaching the goal, converges on a policy and let the sequence generated from the policy be $\{(s_0,b_0),(s_1,b_1)...(s_m,b_m),s_g\}$. We have the start states s_0 same as s_{t0} while goal state s_g is same as s_{tg}. Let there be two pairs of equivalent states, say s_{ti} and s_j are equivalent, s_{tk} and s_l are equivalent, satisfying the condition $0 \leq i < k \leq n$ and $0 \leq j < l \leq m$. Now the sequences of actions $\{a_{ti}, ..., a_{t(k-1)}\}$ and $\{b_j, ..., b_{(l-1)}\}$ are mapped as equivalent. $a_{t(k-1)}$ is the index of the action that takes first agent from $s_{t(k-1)}$ to s_{tk} and $b_{(l-1)}$ is the action that takes second agent from $s_{(l-1)}$ to s_l . The mapped action sequence is replaced by another action equivalent action sequence if it contains lesser number of actions. A sample of mapping computed is shown in *Table 3* and *Figure 7*. Once we have such a mapping we can re-use it while solving subsequent tasks.

In the subsequent tasks, if the same sequence of actions are observed in transferred action index sequence, the mapped action sequence of the second robot can be directly used during exploration. This can take the second robot faster to the goal, though not always guaranteed. In some instances, the mapped action sequence might take the second robot away from the goal and make it longer to solve the task. Hence, we use the mapped action sequence (1 - ε time) at the useful intermediate state during the exploration phase of the robot and for ε time, the regular algorithm *Algorithm 1* is used. However, if the new task is a combination of mapped action sequences, the convergence is very quick.

TABLE I

DIFFERENT SETS OF TASK FOR TRANSFER FROM LARGER TO FINER ACTION MODEL (ROBOT)

TASK	START STATE	GOAL STATE	DISTANCE*
TASK 1	(0,0)[ROOM 1]	(-8,12)[ROOM 2]	20
TASK 2	(2,-2)[ROOM 1]	(-12,18)[ROOM 2]	34
TASK 3	(2,6)[ROOM 1]	(-6,-6)[ROOM 3]	40
TASK 4	(20,18)[ROOM 1]	(-22,-16)[ROOM 3]	76

*DISTANCE is the optimal number of steps required to reach the Goal State from Start State

Algorithm 1 Transfer Knowledge motivated Learning

Input: s ∈ S, a ∈ A, $s_{t0}, s_{t1}...s_{tn}, s_{tg} \in S_t$, $a_{t0}, a_{t1}...a_{tn} \in A_t$, Intrinsic reward r'
Initialize Q(s,a) arbitrarily and e(s,a) =0, for all s,a
Initialize s,a
repeat
  Take action a and observe next state s' and reward r from the environment
  Choose action a' from s' using policy derived from Q (ε-greedy)
  if $s' = s_{ti}$ (i = 1 to n) $\in S_t$ **then**
    $\delta \leftarrow r'$ + r + γ Q(s',a') - Q(s,a)
  else
    $\delta \leftarrow$ r + γ Q(s',a') - Q(s,a)
  end if
  e(s,a)$\leftarrow$ e(s,a)+1
  for all (s,a) **do**
    Q(s,a)$\leftarrow$ Q(s,a)+ α δ e(s,a)
    e(s,a)$\leftarrow$ γ λ e(s,a)
  end for
  s$\leftarrow$ s'; a$\leftarrow$ a'
until s is terminal

IV. EXPERIMENTAL SET-UP AND RESULTS

To evaluate our algorithm, we created a rooms world environment, shown in *Figure 3*. The states of the domain are based on 50x50 grid world and the states are represented

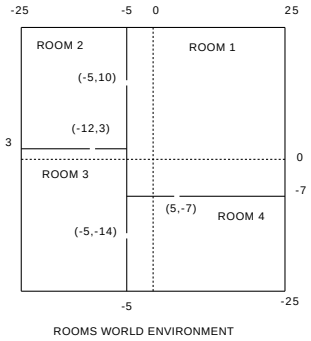
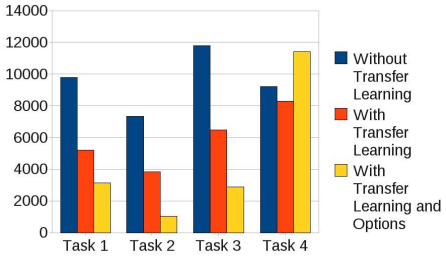


Fig. 3. Rooms world environment used in the experiment



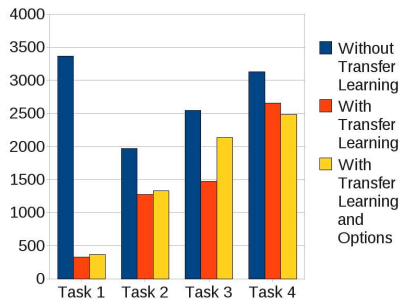
X axis: Different Tasks as given in Table I
Y axis: Average number of exploration steps to reach the Goal

Fig. 4. Comparison of the number of exploration steps before reaching the goal for transfer from larger to finer action model (Robot). (averaged across ten trials)

as (x,y) [-25 $\leq$ x, y $\leq$ 25]. There are 4 rooms with different orientations and 4 doors connecting them. The agent can move from one room to another only through these doors. The knowledge transferred is a sequence of state and corresponding action indices. The sequence of states is used for generating the pseudo reward function while the action indices are used for action sequence matching. The experiments have been run in deterministic and stochastic environments. We have also run trials to evaluate transfer learning from the robot with the larger action model *(Robot 1)* to the robot with the finer action model *(Robot 2)* and the reverse. An example of the knowledge transferred is {{(-4,6),1},{(-4,8),1},{(-4,10),4},{(-6,10),4},{(-8,10),2},{(-8,8),2},{(-8,6),2},{-8,4}}. The action model of *Robot 1* is larger and *Robot 2* is finer. A transfer learning solution framework is effective for the robot, if it speeds up the time to solve the task comparing the steps it takes otherwise. Hence, we measure the effectiveness by comparing the number of exploration steps taken by the robot to reach the goal without transfer and with transfer.

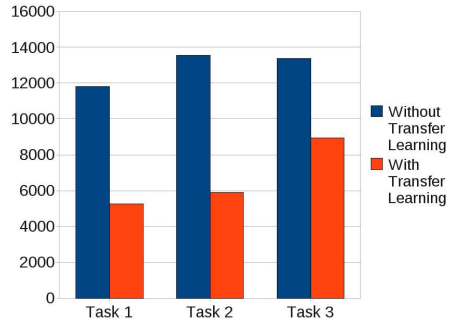
A. Transfer from Larger to Finer Action Model (Robot) in Deterministic Environment

Robot 1 having action model, shown in *Figure 2(a)* is assumed to have solved the task. We can represent index the actions as {1, 2, 3, 4}. The second robot *Robot 2*,



X axis: Different Tasks as given in Table I
Y axis: Average number of exploration steps to reach the Goal

Fig. 5. Comparison of the number of exploration steps before reaching the goal for transfer from finer to larger action model (Robot). (averaged across ten trials)



X axis: Different Tasks as given in Table I
Y axis: Average number of exploration steps to reach the Goal

Fig. 6. Comparison of the number of exploration steps to reach the goal in stochastic environment

which needs to learn a policy to reach the goal has action model, shown in *Figure 2(b)* {Forward, Backward, Right, Left} represented as {F,B,R,L} and move one step in the corresponding direction of each action. Experiments were conducted with four different sets of start and goal states in a deterministic environment. These states were distributed in different rooms and were of varying complexities, shown in *Table 1*.

The results show that there has been a speed up in learning because of transfer, shown in *Figure 4* indicated by the reduction in number of exploration steps to reach the goal. From our experiments, we find that there is an improvement of 10% to 47% , shown in *Table 2*. The speed-up may vary depending on the knowledge transfered and the resulting pseudo reward function. For all meaningful knowledge transferred, there is an appropriate speed-up happening for the second robot. The speed-up comparisons across different knowledge (sequence of states) transferred for the same task are indicated in *Table 4*.

B. Transfer from Finer to Larger Action Model (Robot) in Deterministic Environment

Here *Robot 2* is assumed to have solved the task is assumed to be having action model, shown in *Figure 2(c)*.

TABLE II

COMPARISON BETWEEN NUMBER OF EXPLORATION STEPS TO REACH THE GOAL FOR TRANSFER FROM LARGER TO FINER ACTION MODEL (ROBOT). (AVERAGED ACROSS TEN TRIALS)

TASK	A	B	C	D	E	F
TASK 1	9777.80	5200.1	46.82	3152.4	NA	NA
TASK 2	7338.3	3841.7	47.65	1039.6	34.38	85.83
TASK 3	11797.5	6467.8	45.18	2880.10	56.82	75.59
TASK 4	9222.2	8282.9	10.19	11419.4	0	-23.83

A: Number of exploration steps taken to reach the goal without transfer
B: Number of exploration steps taken to reach the goal with transfer
C: Percentage Improvement using transfer learning
D: Number of exploration steps taken to reach the goal with transfer and action sequence mapping
E: Percentage of mapped action sequences used in the final path
F: Percentage Improvement using transfer learning and action sequence mapping

TABLE III

ACTION SEQUENCE MAPPING EXAMPLE FOR TRANSFER FROM LARGER TO FINER ACTION MODEL (ROBOT)

SET	ROBOT 1 ACTION INDEX SEQUENCE	ROBOT 2 MAPPED ACTION SEQUENCE
I	11	RFFRFFLL
II	4	LL
III	2	BB
IV	1	BRFFFL
V	4411	FLFRFFLLBLFL
VI	3	FRRB
VII	222244	LLLBBLBRBBLBBLBR
VIII	444	RBRBLBLLLFFLBLFLLF
IX	44	LBLLLF
X	422	BLBBBL
XI	22	BRBBLB

The second robot which needs to learn a policy to reach the goal has action model represented as $\{1,2,3,4\}$ and move two steps in the corresponding direction of each action. The speed up in learning because of transfer is shown in *Figure 5*. There is an improvement in performance using the mapped action sequences.

C. Transfer from Larger to Finer Action Model (Robot) in Stochastic Environment

In the rooms world environment, shown in *Figure 3*, a wind is modeled to blow from left to right with a probability *0.01*. It moves the agent to the right by one step. *Robot 1* is assumed to have solved the task and transfers the sequence of states it has traversed to reach the goal to *(Robot 2)*. From *Figure 6*, it is evident that there has been a speed-up because of the transfer even if the environment is stochastic.

V. DISCUSSION

Time to converge on a policy for a given task is a measure of evaluating a transfer learning approach. A transfer learning solution framework is effective, if it speeds up the learning of the agent comparing the time it takes otherwise. In almost all the experiments we conducted using our transfer learning framework, there has been a considerable improvement in time to learn and converge on a policy. We have analyzed the solution framework using different tasks and using different transfer for same task in both deterministic and stochastic environments. One of our observation is that the performance is not completely dependent on the optimality of the transferred knowledge. From *Table 4*, we find that the transfer based

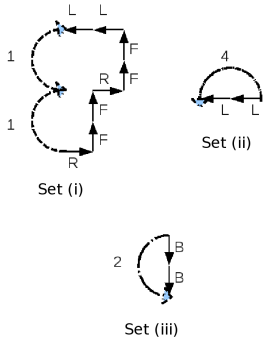


Fig. 7. Action sequence Mapping Example for transfer from larger to finer action model (Robot)

TABLE IV

COMPARISON ACROSS DIFFERENT KNOWLEDGE TRANSFER FOR SAME TASK FOR TRANSFER FROM LARGER TO FINER ACTION MODEL (ROBOT). (AVERAGED ACROSS TEN TRIALS)

TASK	DISTANCE*	A	B	C	D
NO TRANSFER	NA	7338.80	NA	104.4	NA
OPTIMAL PATH	24	3841.7	47.65	80.2	23.18
SET 1	32	7256.6	1.12	89.2	14.56
SET 2	48	5081.2	30.76	54	48.28
SET 3	56	3772.2	48.6	68	34.87

*DISTANCE is the optimal number of steps required to reach the Goal State from Start State
A: Number of exploration steps taken to reach the goal with transfer
B: Percentage Improvement on exploration steps taken to reach the goal using transfer learning
C: Number of greedy steps on converged policy taken to reach the goal with transfer
D: Percentage Improvement on greedy steps on converged policy using transfer learning

on Set 3 provides faster learning, though it is based on a non-optimal path with larger path length. As our approach provides pseudo rewards based on the distance of the state from the goal in the transfer, the less relevant states are bypassed by more important states. In case of loops in the transferred sequence, the second agent bypasses the loop to more useful states.

Learning action sequence mapping accelerates the convergence of the second robot with a high degree, if the mapped action sequences are useful with respect to the task. In few instances if the new task is a collection of mapped action sequences, the convergence, understandably, is extremely fast. If none of the mapped action sequences are relevant to the current task, the proposed transfer learning algorithm *Algorithm 1* is used. But in tasks, where the mapped action sequences are relevant but not useful, it affects the learning and results in slowing down the learning. The result in *Table 2* for TASK 4 is an example of this behavior. An interesting observation is that, in the case of loops in the sequence of the first robot, the second robot maps ”no action” as an equivalent action sequence. This helps to identify the loops and bypass them straight away. However, computing the usefulness of a mapped action sequence, for the current task need to be fine tuned for better performance. While it is easier to establish such an equivalence in a deterministic world, it is harder to establish in stochastic domain.

TABLE V

COMPARISON BETWEEN NUMBER OF EXPLORATION STEPS TO REACH THE GOAL FOR TRANSFER FROM FINER TO LARGER ACTION MODEL (ROBOT). (AVERAGED ACROSS TEN TRIALS)

TASK	A	B	C	D	E
TASK 1	3365.89	331.89	90.14	371.	NA
TASK 2	1967.3	1277.4	35.07	1333.2	32.23
TASK 3	2544.8	1468.6	42.29	2133.1	16.17
TASK 4	3126.7	2652.0	15.18	2480.2	20.67

A: Number of exploration steps taken to reach the goal without transfer
B: Number of exploration steps taken to reach the goal with transfer
C: Percentage Improvement using transfer learning
D: Number of exploration steps taken to reach the goal with transfer and action sequence mapping
E: Percentage Improvement using transfer learning and action sequence mapping

VI. RELATED WORK AND COMPARISON

There have been several methods proposed for transfer learning. Often, a mapping is used to relate the new task to the task for which a policy had been learnt. There have been several work in learning such a mapping. The mapping is used to find the similarities between the state variables in source and target task. Most of the approaches exploit the already learnt policy of one task and use it during the exploration of new task. Madden and Howley (Madden and Howley 2004) use symbolic learner and a propositional representation of the task to build a generalized policy and is used to aid exploration. Fernandez and Veloso (2006)re-use the learned policy as an option during exploration and hence use either exploration action or exploit learned policy. Liu and Stone (2006) use specialized version of the structure mapping theory, to find similarities in the tasks based on similar patterns in their state transitions. Kuhlmann and Stone (2007) use graph based domain mapping for value function transfer learning. Talvitie and Singh (2007) use an experts algorithm to select the best policy amongst a number of given policies. By creating different target task policies using the source task policy and different mappings, they can hence find the best mapping. Transfer learning, recently have also been achieved using transfer actions, where actions are transferred from source task to target task and used to guide during the exploration (Tom Croonenborghs, Kurt Driessens and Maurice Bruynooghe 2008). In all the transfer learning related work,the transfer happens across two reinforcement learning tasks and the assumption has been that the same robot is involved in both the tasks and their action models are the same. They also assume the availability of significant knowledge. There haven't been approaches where the robots are different and are trying to solve the tasks with very minimal information. Our approach on the other hand is for the condition where very limited information is available and also makes no assumptions on the action model of the other robot or its transition probabilities. Since our approach and available knowledge are different it is hard to absolutely compare the results of our method with the existing approaches.

VII. CONCLUSION AND FUTURE WORK

We introduced transfer learning across heterogeneous robots. The action capabilities of the robots are different and unknown, while the task and environment are the same. The knowledge gained by one robot in solving a task is modeled as a sequence of states and action indices. It is passed to the other robot. The sequence of states are used to develop a pseudo reward function which are used along with the regular reward function from the environment. It has been shown to speed up the learning and solve the task faster with minimal knowledge. The action indices are used to come up with a mapping between the sequence of base actions of the two robots. It is shown that if the mapped action sequences are useful for subsequent task, it accelerates the learning and helps to converge on the policy quicker.

The framework we have proposed use minimal information, a single trajectory. It can also be extended to be used with more information, say multiple trajectories, partial action equivalence or partial map of the environment etc. Our current work is based on the assumption that that environment and task are same. We can work towards extending our model for related but different tasks, while the robots are different. Our experiments have been performed on a discrete state space environment. Working in a continuous state space environment and exploiting the state space approximations can be done. Exploiting the symmetries and sub goal discovery and combining it with transfer is promising. Methods to identify useful action sequences for a given task is also a direction for future work.

True hetrogenity of robots is when the robots have different sensory input but however the underlying state space cab be mapped and the proposed framework can be used. We can extend the model to the condition when the observation of the state space is partial for the robot. Besides mobile robots, the framework can be extended to the robotic arm space.

REFERENCES

[1] Taylor, M. E., Whiteson, S., and Stone, P. Transfer learning for policy search methods. *In ICML workshop on Structural Knowledge Transfer for Machine Learning*, 2006.

[2] Liu, Y., and Stone, P. Value-function-based transfer for reinforcement learning using structure mapping. *In Proceedings of the Twenty-First National Conference on Artificial Intelligence*, 2006.

[3] Fernandez, F., and Veloso, M. Probabilistic policy reuse in a reinforcement learning agent. *In AAMAS06: Proceedings of the fifth international joint conference on Autonomous agents and multiagent systems*, 2006.

[4] Taylor, M. E., Whiteson, S., and Stone, P. Transfer via inter-task mappings in policy search reinforcement learning. *In The Sixth International Joint Conference on Autonomous Agents and Multiagent Systems*. 2007.

[5] Talvitie, E., and Singh, S. An experts algorithm for transfer learning. *In Proceedings of the 20th International Joint Conference on Artificial Intelligence*, 2007.

[6] Tom Croonenborghs, Kurt Driessens and Maurice Bruynooghe. Learning a Transfer Function for Reinforcement Learning Problems. *In Proceedings of the Annual Belgian-Dutch Conference on Machine Learning*, 2008.

[7] Ravindran.B and Barto,A.G. SMDP Homomorphisms: An Algebraic Approach to Abstraction in Semi Markov Decision Processes. *In the Proceedings of the Fifth International Conference on Knowledge Based Computer Systems*, 2004.

[8] Sandeep Goel and Manfred Huber. Subgoal Discovery for Hierarchical Reinforcement Learning Using Learned Policies. *In Proceedings of the 16th International FLAIRS Conference*, 2003

[9] Matthew E. Taylor, Gregory Kuhlmann, and Peter Stone. Autonomous Transfer for Reinforcement Learning. *In Proceedings of the Seventh International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, 2008.