Transferring Experience in Reinforcement Learning through Task Decomposition

(Extended Abstract)

Ioannis Partalas
Department of Informatics
Aristotle University of
Thessaloniki, 54124, Greece
partalas@csd.auth.gr

Grigorios Tsoumakas
Department of Informatics
Aristotle University of
Thessaloniki, 54124, Greece
greg@csd.auth.gr

Konstantinos Tzevanidis
Department of Informatics
Aristotle University of
Thessaloniki, 54124, Greece
ktzevani@csd.auth.gr

Ioannis Vlahavas
Department of Informatics
Aristotle University of
Thessaloniki, 54124, Greece
vlahavas@csd.auth.gr

ABSTRACT

Transfer learning refers to the process of conveying experience from a simple task to another more complex (and related) task in order to reduce the amount of time that is required to learn the latter task. Typically, in a transfer learning procedure the agent learns a behavior in a *source task*, and it uses the gained knowledge in order to speed up the learning process in a *target task*. Reinforcement Learning algorithms are time expensive when they learn from scratch, especially in complex domains, and transfer learning comprises a suitable solution to speed up the training process. In this work we propose a method that decomposes the target task in several instances of the source task and uses them to extract an advised action for the target task. We evaluate the efficacy of the proposed approach in the robotic soccer Keepaway domain. The results demonstrate that the proposed method helps to reduce the training time of the target task.

Categories and Subject Descriptors

I.2.6 [**Artificial Intelligence**]: Learning

General Terms

Algorithms

Keywords

transfer learning, reinforcement learning

1. INTRODUCTION

Transfer learning refers to the process of conveying experience from a simple task to another more complex (and related) task in order to reduce the amount of time that is required to learn the latter task. Typically, in a transfer learning procedure the agent learns a behavior in a *source task*, and then it uses the gained knowledge in order to speed up the learning process in a *target task*.

Recently, transfer learning among Reinforcement Learning (RL) agents has received a lot of attention [1]. This is due to the fact that RL algorithms are time expensive when they learn from scratch, especially in complex domains, and transfer learning comprises a suitable solution to speed up the training process.

In this work we propose a novel method for transfer learning among RL agents, which is based on the decomposition of the target task in several different instantiations of the source task. The identified source instances are used in order to provide an advice to the target task.

We evaluate the proposed approach in the Keepaway domain [2]. The empirical evidence show that the proposed approach reduces the complete training time in the target task. Additionally, experiments we conducted in large tasks show that the proposed approach scales well with respect to the size of the problem.

2. THE PROPOSED APPROACH

One of the key novel contributions, and at the same time a basic assumption, of the proposed approach is the observation that in certain domains the target task (states and actions) can be mapped to a number of different instantiations (states and actions) of the source task. We hypothesize that the different mappings will increase the effectiveness of transfer learning algorithms, compared to a single mapping.

In this work we propose an approach that receives an action advice based on the source instances. The basic steps of the proposed approach can be summarized as follows:

- Decompose the target task to valid instances of the source task.
- Extract an advice from the instances
- According to a strategy follow or not the advice

Cite as: Transferring Experience in Reinforcement Learning through Task Decomposition (Short Paper), I. Partalas, G. Tsoumakas, K. Tzevanidis and I. Vlahavas, *Proc. of 8th Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS 2009)*, Decker, Sichman, Sierra and Castelfranchi (eds.), May, 10–15, 2009, Budapest, Hungary, pp. XXX-XXX.

2.1 Mapping the target task

Formally, given that the target task can be mapped to N instances of the source task, the specification of N pairs of mapping functions is required. Each pair (f_S^i, f_A^i), $i = 1 \ldots N$ consists of a function $f_S^i : S_{target} \rightarrow S_{source}$ mapping the set of target states, S_{target}, to the set of source states, S_{source}, and a function $f_A^i : A_{target} \rightarrow A_{source}$ that *partially* maps the set of target actions A_{target} to the set of source actions A_{source}. By partially, we mean that certain actions in the target domain, might not have a corresponding action in the source domain in some of the instantiations.

2.2 Extracting Advice

Having defined the functions that establish a mapping from a target task to several instantiations of the source task, the next step is to use the knowledge acquired in the source task to improve the learning procedure in the target task. In order to accomplish this, the proposed method uses the experience gained from the source task to extract an advice for the target task.

We assume that the RL agent has been trained in the source task and that it has access to a function $Q'(s', a')$ returning an estimation of the Q value for a state s' and action a' of the source task. The agent is currently being trained in the target task, learning a function $Q(s, a)$ that approximates the Q function, and senses the state s. The action that will be executed by the agent is ruled by the $\epsilon - Advice$ procedure which is depicted in Algorithm 1. This is a variation of the well known $\epsilon - greedy$ rule which is used to balance the exploration and the exploitation in RL algorithms.

Algorithm 1 The ϵ-Advice strategy.

```
1:  procedure ε-ADVICE($\epsilon, s, f_S, f_A, Q, Q'$)
2:      $p \leftarrow RandomReal(0, 1)$
3:      if $p \leq \epsilon$ then
4:          return random action
5:      else
6:          $Q'_{max} \leftarrow 0$
7:          $a'_{max} \leftarrow \varnothing$
8:          $i_{max} \leftarrow 0$
9:          for $i \leftarrow 1 \ldots N$ do
10:             $s' \leftarrow f_S^i(s)$
11:             for all $a' \in A_{source}^{s'}$ do
12:                 if $Q'_{max} < Q'(s', a')$ then
13:                     $Q'_{max} \leftarrow Q'(s', a')$
14:                     $i_{max} \leftarrow i$
15:                     $a'_{max} \leftarrow a'$
16:                 end if
17:             end for
18:         end for
19:         $a_{adv} \leftarrow g_A^{i_{max}}(a'_{max})$
20:         $a_{cur} \leftarrow \arg\max_a Q(s, a)$
21:         if $Q'_{max} - Q(s, a_{cur}) > Q(s, a_{adv})$ then
22:             return $a_{adv}$
23:         else
24:             return $a_{cur}$
25:         end if
26:     end if
27: end procedure
```

According to the $\epsilon - Advice$ strategy the learning agent can select to explore the state space, to exploit the current knowledge or to exploit the experience from the source task. In lines 9 to 19 the algorithm extracts the advised action from the source task.

More specifically, for each instance i of the source task that is recognized in the target task, the corresponding mapping function f_S^i is used to transform the target state s to its source representation s'. Then for each available action in the state of the source task, the corresponding Q-values are computed. The computation of the Q-values depends on the function approximation method that is used.

As the algorithm iterates over the instances, it stores the maximum Q-value, Q'_{max} and the corresponding action a_{max}, along with the index, i_{max}, of the instance that they correspond to.

After the advised action is extracted, it is transformed to its target representation using a mapping function g_A, which is an inverse function of f_A and maps a source action to its equivalent target action. After that, the algorithm checks (lines 21-25) whether to act according to the current learned policy or to follow the action that is suggested by the source task.

The agent must prefer the advised action if the difference between the Q_{max} value and the maximum value of the current learned Q-function, $Q(s, a_{cur})$ is greater than the value of the current learned Q-function for the advised action $Q(s, a_{adv})$.

In the initial stages of learning, the agent will be biased to prefer the recommended actions depending on the impact of the source task Q-values. The impact of the source task depends strongly on how much time we spent in training it. More specifically, if the source task was trained for a small number of episodes then the Q-values will be also small and afterwards the agent in the target task will use the advised actions for a small period. As learning proceeds, the values of the target Q-function will increase and the initial bias will be overridden.

Training the source task in more episodes will cause greater values of Q' and a more valuable Q-function. So its more desirable to use the advised (and more reliable) actions for a longer period in the initial stages of learning in the target task. This is achieved by the $\epsilon - Advice$ strategy as the target values of Q will need more time to exceed the source Q' values.

The complexity of the proposed algorithm is linear with respect to the instances that are identified in the target task.

3. EXPERIMENTS

In order to evaluate the efficacy of the proposed approach we run a series of experiments in the Keepaway domain [2]. Due to space limitations, it is not possible to present the experimental evaluation. Therefore, we refer the reader to the following URL, where the complete experimental part can be found: http://mlkd.csd.auth.gr/tlExper.pdf.

4. REFERENCES

[1] V. Soni and S. Singh. Using homomorphisms to transfer options across continuous reinforcement learning domains. In *AAAI Conference on Artficial Intelligence*, pages 494–499, 2006.

[2] P. Stone and R. Sutton. Keepaway soccer: A machine learning test bed. pages 207–237. 2002.