# Approach for knowledge transfer for building autonomous agent behaviour

## Ansatz fur den Wissenstransfer fur den Aufbau eines autonomen Agentenverhaltens

Vanya Markova* and Ventseslav Shopov†

* Institute of Robotics with Bulgarian Academy of Sciences,
139 Rouski blvd, Plovdiv, 4000, Bulgaria, e-mail: markovavanya@yahoo.com
†139 Rouski blvd, Plovdiv, 4000, Bulgaria, e-mail:vkshopov@yahoo.com

***Abstract*** — In this paper building of autonomous agent behaviour is discussed. We study an application of three different algorithms: greedy algorithm, reinforcement learning and reinforcement learning with knowledge transfer. We present two case studies that demonstrates the effectiveness of knowledge transfer approach. Finally we present that transfer learning is an essential mean of agent control learning task.

***Zusammenfassung*** — In diesem Artikel wird der Aufbau des Verhaltens eines autonomen Agenten betrachtet. Wir erforschen die Anwendung von drei verschiedenen Algorithmen: Greedy-Algorithmus, bestärkendes Lernen und bestärkendes Lernen mit Wissenstransfer. Wir stellen zwei Fallstudien vor, die die Effektivität des Wissensaustauschansatzes demonstrieren. Schließlich stellen wir vor, dass Transferlernen ein wesentliches Mittel der Lernaufgabe der Agentenkontrolle ist.

## I. Introduction

In this paper we present a new approach for transfer learning trough reinforcement learning.

An essential quality of a cognitive being is its ability to learn, that is, to gain new knowledge or skills as well as to improve existing knowledge or skills based on experience. Cognitive beings are able to cope with situations they have been previously confronted with as well as they are able to adapt to new situations sensibly. Thus, when designing an artificial agent that shall exhibit cognitive qualities—which we refer to in short as a cognitive agent—one central task is to develop computational means that enable the agent to learn. In this paper, we subscribe to one of the most influential paradigms of machine learning, reinforcement learning (RL) [1]. Reinforcement learning is very valuable when the characteristics of the underlying system are not known and/or difficult to describe or when the environment of an acting agent is only partially known or completely unknown.

Agents situated in the real world can perceive a great variety of information. Two situations are unlikely to lead to the same perception even if the situations are very similar. Learning requires the agent to acknowledge important details in a perception, to recognise commonalities across situations, and, most importantly, to disregard irrelevant information. In other words, the ability to learn involves the ability to abstract. Abstraction enables us to conceptualise the surrounding world, to build categories, and to derive reactions from these categories that can adapt to different situations. Complex and overly detailed circumstances can be reduced to much simpler concepts and not until then it becomes feasible to deliberate about conclusions to draw and actions to take. Abstract concepts explicate commonalities in different scenes and, thus, can cover new situations[2] [3] [4].

Transfer learning(TL) or inductive transfer is a research problem in machine learning that focuses on storing knowledge gained while solving one problem and applying it to a different but related problem.[5].

The main goal of this paper is to apply transfer learning trough reinforcement learning in area of building autonomous agent behaviour.

In second part of this paper we will describe the underlying theory of methods in our study: the RL/QL and MDP , Autonomous Agent behaviour and TL. In addition we will describe the implementation of our new approach for using TL in building of Autonomous Agent behaviour. In third part we describe the experiments and Gathers evidence to support our hypothesis.

## II. Methods and materials

### A. Autonomous Agent behaviour

The information about past and current states of the agent and environment allow the agents to estimate its own progress. Moreover this information allow the agent to make the corrections in existing pans if any needed and even to ma make new plans if it is necessary.

However if the agent makes corrections in the existing plans too often then this could lead to poor overall performance. Moreover the frequently dropping and building plans could make the things even worse. Hence it is desirable to reduce (or completely avoid) situations in which the agent should changes its mind. There are two main approaches to do that: the first is to make the changing of the plans less recourse consuming task and the e second is to make such plans that are able to deal with volatile environment behaviour. This material is concerning the second approach.

As a key issue in building more efficient plans in rapidly changing environment we can point the ability of the agent to makes its plans in accordance not only with past and current states of the environment but also bearing in mind the future. To do that the agent needs to predict or forecast the future st states of the environment. So if we describe the stat es of the agent and environment as a time series then the task of making efficient plans will be significantly aided if the agent could forecasts the future with desirable accuracy.

An n-tipple (vector) is a result of one cycle of a work of the agent. It consists of the parameters of the behaviour of the agent: $b(b1, b2, \ldots, bn)$. The data from environment are collected and transformed into time series in the knowledge base of the agent.

## B. Markov Decision Process

We formulate the transfer learning problem in sequential decision making domains using the following framework of Markov Decision Process (MDP) We use the following definition of MDP as a 5-tuple

$$< S, A, P, R, \gamma > \tag{1}$$

where the set of states, set of actions, transition function and reward function are described. And

$$P : S \times A \rightarrow \Pi(S) \tag{2}$$

is a transition function that maps the probability of moving to a new state given an action and the current state,

$$R : S \times A \rightarrow R \tag{3}$$

is a reward function. that gives the immediate reward of taking an action in a state.

And

$$\gamma \in [0, 1) \tag{4}$$

is the discount factor.

So the MDP of the agent is described in (1), where $S$ is the set of states, $A$ is the set of actions, $P$ is transition function and $R$ is a reward function. The transition function $P$ maps the the probability of moving to a new state given an action and the current states is shown in (2). The reward functions $R$ that gives the immediate reward of taking an action is described in (3). An the discount factor $\gamma$ is bounded as is shown in (4).

## C. Reinforcement learning

Reinforcement learning [1] is a popular and effective method to solve an MDP.

In our work we reference the reinforcement learning algorithm Q-learning as is described in [6]. At each moment of time, the agent is in a given state $s \in S$, and the agent's view is represented by a feature vector. Upon this information the agent makes the decision which action $a$ from a set of all possible actions $A$ to take in order to reach its goal. The outcome of Q-learning is a Q-function

$$Q(s, a) \tag{5}$$

that attaches to any state-action tuple $(s, a)$ the expected reward over time. We discuss here the overall reward when starting in $s$ and executing action $a$. From that Q-function, one can derive the policy $\pi$ by always choosing the action with the highest Q-value:

$$\pi(s) = \underset{a \in A}{\operatorname{argmax}}(Q(s, a)) \tag{6}$$

Under these conditions, Q-learning should converge to an optimal Q-function

$$Q* = \pi(s) = \underset{a \in A}{\operatorname{argmax}}(Q(s; a)) \tag{7}$$

that returns the highest reward for any state-action tuple $(s, a)$. Hence in this way we establish an optimal policy $\pi*$.

## D. Transfer learning

Machine learning and data mining techniques have been used in numerous real-world applications. An assumption of traditional machine learning methodologies is the training data and testing data are taken from the same domain, such that the input feature space and data distribution characteristics are the same. However, in some real-world machine learning scenarios, this assumption does not hold. There are cases where training data is expensive or difficult to collect. Therefore, there is a need to create high-performance learners trained with more easily obtained data from different domains. This methodology is referred to as transfer learning.[7]

There is a hierarchical Bayesian framework for transfer in sequential decision making tasks of transferring two basic kinds of knowledge [8] [9]

In our paper we uses meta-data (e.g., attribute-value pairs) associated with each task to learn the expected benefit of transfer given a source-target task pair. An example of such a metadata is given in [10].

## E. Implementation

In our implementation of QL, we claim that if the values of the P matrix at the beginning of the training are zero, then we will reach an optimal policy for a final number of epochs (steps). In order to speed up the training, it is good that the coefficients of the P matrix are somewhat closer to the desired policy. This can be achieved through a TL in a simpler environment (or just a part of the environment). The classic reinforcement learning consists of finding an optimal policy for the whole area with high details.

Our approach is based on [10]:

- loading the whole map and scraping all details but geometric obstacles
- find a reinforcement learning solution for this plain map
- transfer the Q matrix as predefined knowledge
- load full map and using predefined knowledge enhance the learning process

Notation and transfer learning: Let $G$ be the set of all possible tasks. Let $G_{source} \subset G$ be a set of source tasks for which the agent has already learned a policy and let $G_{target} \subset G$ be another set of target tasks to be learned by the agent. For each task

$$g_i \in G, let D_i in R^n \tag{8}$$

is a descriptor of features for the given task.We assume that $g_i$ and $D_i$ that are known to the agent.

We define a target task $g_j \in G_{target}$, as the goal of the agent. Thereafter to achieve $g_j$ we have to select a task $g_i \in G_{source}$ such that $g_i$ serves as an effective source for learning $g_j$ . For task pair, $g_i$ and $g_j$ , let

$$f_u(g_i, g_j) \in R^n \tag{9}$$

denote the function of usefulness. This is the function of transferring the policy learned in $g_i$ to the task $g_j$ , where $f_u(g_i, g_j) > 0$ indicates positive benefit of transfer, while $f_u(g_i, g_j) < 0$ indicates negative benefit of transfer.

We assume that for each pair of tasks $(g_i, g_j)$ such that $g_i, g_j \in G_{source}$, the agent could reliable estimate $f_u(g_i, g_j)$. So the agent can use these estimates to predict the expected transfer benefit between tasks in $G_{source}$ and tasks in $G_{target}$.

## III. Experiments and results

We Gathers evidence to support the hypothesis that building of Autonomous Agent behaviour trough transfer learning will significantly speed up the process of constructing Autonomous Agent behaviour. We claim that building of Autonomous Agent behaviour trough transfer learning is more efficient that applying the RL direct approach.

We perform the following experiment: For a given map we should find an optimal autonomous agent behaviour. The map is described by its size nxn and complexity rate Rc. And we chose a start point and final target in this map. The goal is to find the fastest route from start point to final target.

We have three methods Greedy Approach, Reinforcement learning without knowledge transfer and Reinforcement learning whit knowledge transfer. And two cases: Plain map and rough terrain map. Plain map have all terrain with same squares that cost 1 clique and obstacles have inf cliques. In the rough terrain map the squares could be 1,2,4 or inf cliques each.

We study following algorithms:

- case I - a Greedy Approach(GA)
- case II - Reinforcement Learning without knowledge transfer(RL)
- case III -Reinforcement Learning with Transfer Learning(RLTL)

We do the following task: for a given map we need to find optimal behaviour of an autonomous agent. The agent's task is to travel on a closed route for a minimum of time. The environment is represented as a two-dimensional obstacle map. The map is described by its size nxn and the

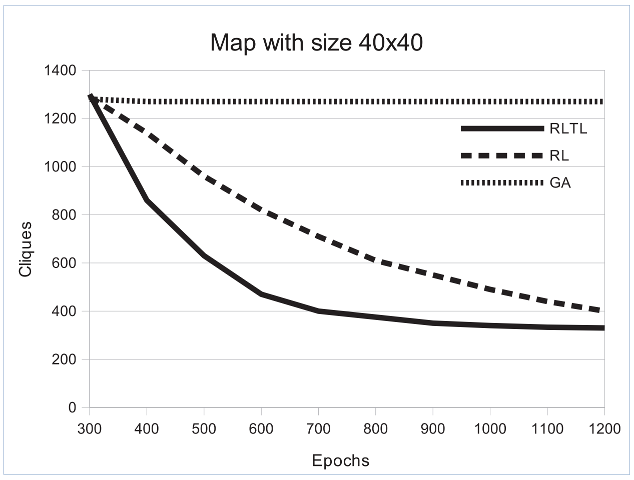

Fig. 1. Comparison of three approaches: Greedy Algorithm(GA), Reinforcement Learning without Transfer Learning(RL) and Reinforcement learning with transfer learning(RLTL) om map with size 40x40.

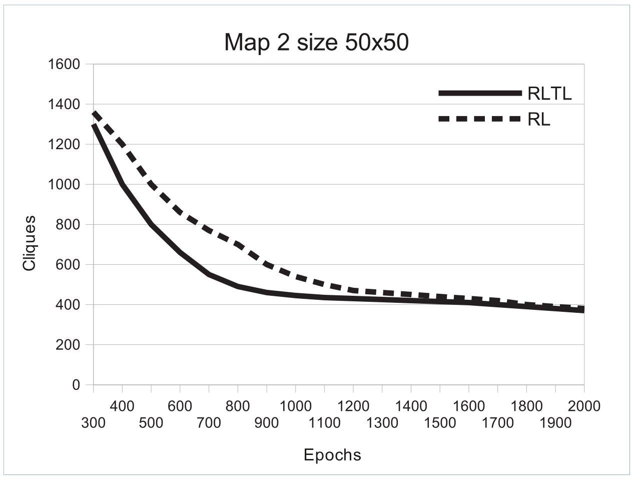

Fig. 2. Compare two algorithms: Reinforcement Learning without Transfer Learning(RL) and Reinforcement learning with transfer learning(RLTL) on map 2 with size 50x50.

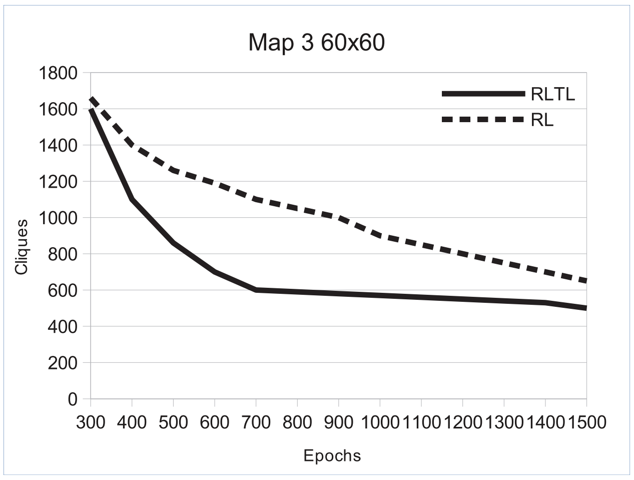

Fig. 3. Compare two algorithms: Reinforcement Learning without Transfer Learning(RL) and Reinforcement learning with transfer learning(RLTL) on map 3 with size 60x60.

rate of complexity Rc. And we chose the starting and ending points of the route on this map. The goal is to find the fastest way from the starting point to the end goal.

We discuss two cases: A simple regular 2D map and a rough 2D map of the pitch. The regular map has all the terrains with the same squares that cost 1 click and the obstacles have inf clicks. In the rough map of the terrain the squares can take 1,2,4 or inf cliques respectively. The fastest its the route with minimum cliques. Agents do not know in advance how many cliques each point of the map takes. So the environment is partially observable. We have three algorithms: Greedy Approach , Reinforcement learning without knowledge transfer(RL) and Reinforcement learning with transfer learning(RLTL) The goal is to achieve optimal policy: the route which takes minimum cliques from the starting point to the end goal. And we perform up to 2000 epochs to finding an optimal policy. As an assessment measure, we use the number of clicks to reach the goal of the final policy. We use three 2D maps: map 1 - 40x40, map 2 - 50x50 and map 3 - 60x60. All maps have random placed obstacles. Moreover all maps have random placed rough points. The distribution of the rough points however for all maps is the same.

In the first experiment, we compare the three approaches on the first map : GA, RL, RLTL. GA does not improve with time because it does not take into account the unobserved properties of the terrain. The second and third experiments show that the RLTL starts much faster and converges to the optimal policy earlier.

Three different algorithms are presented: Greedy Approach, Reinforcement Learning and Reinforcement Learning with Transfer Learning. In two case studies we demonstrate that knowledge transfer could make more effective the learning part of building the behaviour of autonomous agent. From 1 one can see that Greedy Approach does not improve over time. Figures 2 and 3 shows that Reinforcement learning with transfer learning converge much faster to optimal policy than Reinforcement Learning without Transfer Learning. The RLTL and RL are compared on three 2D maps: map 1 - 40x40, map 2 - 50x50 and map 3 - 60x60.

## IV. Conclusion

The impact of different approaches for building of autonomous agent behaviour is discussed in this paper. Three different algorithms are presented. greedy algorithm, reinforcement learning and reinforcement learning with knowledge transfer. In two case studies we demonstrate that transfer learning could make more effective the learning part of building the behaviour of autonomous agent. The agent is able to adapt to new scenarios trough transferring knowledge from simplified environments to more complex maps with rough terrain. Finally we present that transfer learning is an essential mean of agent control learning task.

## References

[1] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press Cambridge, 1998, vol. 1, no. 1.

[2] V. Markova, “Adaptive behaviour approach for autonomous mobile sensor agent,” in *Proc. Int’l Conference InfoTech-2012*, 2012, pp. 1314–1023.

[3] ——, “Autonomous agent design based on jade framework,” in *Proceedings of the International Conference on Information Technologies (InfoTech-2013), no*, 2013, pp. 19–20.

[4] V. Markova and V. Shopov, “An approach to build fuzzy cognitive map for time series,” in *Proc. of the Int’l Conference InfoTech*, 2014, pp. 207–211.

[5] D. Ventura and S. Warnick, “A theoretical foundation for inductive transfer,” *Brigham Young University, College of Physical and Mathematical Sciences*, 2007.

[6] C. J. C. H. Watkins, “Learning from delayed rewards,” Ph.D. dissertation, King’s College, Cambridge, 1989.

[7] K. Weiss, T. M. Khoshgoftaar, and D. Wang, “A survey of transfer learning,” *Journal of Big Data*, vol. 3, no. 1, p. 9, 2016.

[8] A. Wilson, A. Fern, and P. Tadepalli, “Transfer learning in sequential decision problems: A hierarchical bayesian approach,” in *Proceedings of ICML Workshop on Unsupervised and Transfer Learning*, 2012, pp. 217–227.

[9] A. Wilson, A. Fern, S. Ray, and P. Tadepalli, “Multi-task reinforcement learning: a hierarchical bayesian approach,” in *Proceedings of the 24th international conference on Machine learning*. ACM, 2007, pp. 1015–1022.

[10] J. Sinapov, S. Narvekar, M. Leonetti, and P. Stone, “Learning inter-task transferability in the absence of target task samples,” in *Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems*. International Foundation for Autonomous Agents and Multiagent Systems, 2015, pp. 725–733.