J Intell Robot Syst (2018) 91:301–312
https://doi.org/10.1007/s10846-017-0731-2

Heuristically Accelerated Reinforcement Learning by Means of Case-Based Reasoning and Transfer Learning

Reinaldo A. C. Bianchi[1] · Paulo E. Santos[1] · Isaac J. da Silva[1] · Luiz A. Celiberto Jr[2] · Ramon Lopez de Mantaras[3]

Received: 19 December 2016 / Accepted: 17 October 2017 / Published online: 30 October 2017

Abstract Reinforcement Learning (RL) is a well-known technique for learning the solutions of control problems from the interactions of an agent in its domain. However, RL is known to be inefficient in problems of the real-world where the state space and the set of actions grow up fast. Recently, heuristics, case-based reasoning (CBR) and transfer learning have been used as tools to accelerate the RL process. This paper investigates a class of algorithms called Transfer Learning Heuristically Accelerated Reinforcement Learning (TLHARL) that uses CBR as heuristics within a transfer learning setting to accelerate RL. The main contributions of this work are the proposal of a new TLHARL algorithm based on the traditional RL algorithm $Q(\lambda)$ and the application of TLHARL on two distinct real-robot domains: a robot soccer with small-scale robots and the humanoid-robot stability learning. Experimental results show that our proposed method led to a significant improvement of the learning rate in both domains.

Keywords Reinforcement learning · Transfer learning · Case-based reasoning · Robotics

Mathematics Subject Classification (2010) 07.05.Mh

1 Introduction

Reinforcement Learning (RL) [43, 51] is a machine learning method that exploits the interactions of an agent in its environment when searching for a solution of a given problem, in order to maximise the agent's cumulative reward. However, to achieve convergence, RL algorithms typically need a large number of iterations, thus implying on a slow learning rate in continuous state, high-dimensional, tasks. One of the causes for the lack of performance of most RL algorithms is their assumption that no knowledge of the problem is available beforehand.

The acceleration of the RL process has been attempted by means of transferring learnt knowledge from one source task to another (related) task, in what is currently known as transfer learning [46], by the use of heuristics [9], by means of case-based reasoning [19, 54], or by a combination of CBR, heuristics and transfer learning [8, 14, 15], called Transfer Learning Heuristically Accelerated Reinforcement Learning algorithms (TLHARL).

This paper investigates TLHARL algorithms, extending our previous work [14] in three ways: first, it introduces a new algorithm, the TLHAQ(λ); second, this paper provides a more complete description of the tests conducted on the robot-soccer domain described in [14]; and, third,

✉ Reinaldo A. C. Bianchi
rbianchi@fei.edu.br

Paulo E. Santos
psantos@fei.edu.br

Isaac J. da Silva
isaacjesus@fei.edu.br

Luiz A. Celiberto Jr
luiz.celiberto@ufabc.edu.br

Ramon Lopez de Mantaras
mantaras@iiia.csic.es

[1] Centro Universitario FEI, São Bernardo do Campo, SP, Brazil

[2] Federal University of ABC, Santo Andre, SP, Brazil

[3] Artificial Intelligence Research Institute (IIIA-CSIC), Bellaterra, Catalunya, Spain

it introduces novel results on applying TLHARL to the challenging domain of humanoid-robot stability learning.

Transfer learning (TL) [46, 47, 52] is a technique "that aims at reusing the knowledge accumulated in a solution of a previous (source) task to speed up the solution of a distinct (but related) target task" [45]. TL can be understood as a technique to speed up the acquisition of skills in a task as a result of previous training. For example, the knowledge acquired whilst learning to play a musical instrument, say a violin, could be used to speed up the process of learning to play the cello. I.e., the years spent in learning to hold the violin bow, and producing a proper sound with it, could be used to condense into weeks the learning of a cello bow.

A case-based reasoning method (summarised in Section 2.2) aims at using a base of pre-existing solutions (cases) of problems in order to solve new problems. The algorithms investigated in this work, presented in Section 4, use Reinforcement Learning to learn a task on a source domain, storing the knowledge thus obtained in a case base; and then use this case base as heuristics to speed up the learning process in the target domain. This paper considers the application of TLHARL in two domains with real robots: *robot soccer* (Section 5.1) and *humanoid-robot stability learning* (Section 5.2). The experimental results show that TLHARL led to a significant improvement in the learning rate in both domains, in contrast to the base RL algorithms.

2 Background

Reinforcement Learning (RL), Case-Based Reasoning and Transfer Learning, constitute the basis of the Transfer Learning Heuristically Accelerated Reinforcement Learning algorithms investigated in this paper, where reinforcement learning is used in the source domain to create a case base that could be transfered to a target domain. Below we introduce in more details the foundations of RL and Case-base reasoning.

Algorithm 1 Q-learning [43].

Initialise $\hat{Q}_t(s, a)$ arbitrarily.
Repeat (for each episode):
  Initialise s.
  Repeat (for each step):
    Select an action α using a pre-defined policy (such as an ϵ-greedy one).
    Execute the action α, observe $r(s, a)$, s'.
    Update the values of $Q(s, a)$ according to equation 2.
    $s \leftarrow s'$.
  Until s is terminal.
Until some stopping criterior is reached.

2.1 Reinforcement Learning

A Reinforcement Learning (RL) [43, 51] problem is traditionally defined as a finite Markov Decision Process (MDP) with the 4-tuple $\langle \mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R} \rangle$, where:

- $\mathcal{S}$: is a finite set of states.
- $\mathcal{A}$: is a finite set of actions.
- $\mathcal{T} : \mathcal{S} \times \mathcal{A} \rightarrow \Pi(\mathcal{S})$: is a state transition function.
- $\mathcal{R} : \mathcal{S} \times \mathcal{A} \rightarrow \mathbb{R}$: is a reward function.

The optimal behaviour of an agent that learns using RL is called the optimal policy $\pi^* : \mathcal{S} \rightarrow \mathcal{A}$, which returns the action that is the best one to be executed by the agent. A common way of obtaining π^* is by learning the action-value function $Q : \mathcal{S} \times \mathcal{A} \rightarrow \mathbb{R}$. Sutton and Barto [43] defines this function as "the value of taking action a in state s under a policy π, i.e., the expected return starting from s, taking the action a, and thereafter following policy π" [43, Section 3.7]. This can be formulated as:

$$\begin{aligned} Q^{\pi}(s, a) &= E_{\pi}\{R_t | s_t = s, a_t = a\} \\ &= E_{\pi}\left\{\sum_{k=0}^{\infty} \gamma^k r_t + k + 1 | s_t = s, a_t = a\right\} \end{aligned} \quad (1)$$

where s is the current state, a is the action performed in s, r is the reward received and γ is the discount factor ($0 \leq \gamma < 1$).

Q–learning algorithm [51] (Algorithm 1) learns an optimal policy π^* using the action-values by iteratively approximating the Q function, using the following update rule:

$$\hat{Q}(s, a) \leftarrow \hat{Q}(s, a) + \alpha \cdot \left[r + \gamma \max_{a'} \hat{Q}(s', a') - \hat{Q}(s, a) \right], \quad (2)$$

where s, a, r, and γ are the same as in Eq. 1, α is the learning rate and s' is the resulting next state.

Q-Learning is considered one of the most important, and the most widely used, RL algorithm due to its power to solve problems that can be modelled as an MDP and have bounded rewards. One of the criticisms of Q-Learning is that, in order to converge to the optimal policy, it needs to explore the states and actions in the domain many times, making it ill suited for solving large-scale problems.

According to Sutton and Barto [43], one of the basic mechanisms to speed up RL are the eligibility traces, which can be combined with almost any algorithm leading to general and efficient learning methods. Eligibility traces [37, 41] speed up the learning process by recording each state visited during an episode, whilst associating to each of these states some of the reward received at the end of the episode. In this way, in each iteration, all visited state-action pairs

are updated, facilitating temporal generalisation. Successful applications of eligibility traces date back to 1990's (e.g. Araujo and Grupen [2]).

The $Q(\lambda)$ algorithm is proposed by Watkins [51] to implement eligibility traces in Q-Learning. In this algorithm, at each time step, the eligibility trace is updated according to Eq. 3.

$$e(u,v) = \begin{cases} e(u,v)+1 & \text{if } u = s_t \text{ and } v = a_t \\ \lambda \cdot e(u,v) & \text{otherwise} \end{cases} \quad (3)$$

By using Eq. 3, the eligibility of one pair state-action is increased by 1 every time it is visited, and is decreased by a factor of λ ($0 \leq \lambda \leq 1$) at any other time step. Finally, $Q(\lambda)$ updates all state-action pair rule at each time step using Eq. 4.

$$\hat{Q}(s,a) \leftarrow \hat{Q}(s,a) + \alpha \cdot e(s,a) \cdot \left[r + \gamma \max_{a'} \hat{Q}(s',a') - \hat{Q}(s,a) \right], \quad (4)$$

where the parameters are as described in Eq. 2.

The convergence of RL algorithms can also be accelerated by means of an heuristic function in a way similar to that used in informed search algorithms. The Heuristically Accelerated Q–Learning (HAQL) [10] is an extension of the Q–Learning algorithm that includes an heuristic function in the Q-learning action choice rule (Eq. 5) in order to speed up the learning process:

$$\pi(s) = \begin{cases} \arg\max_a \left[\hat{Q}(s,a) + \xi H(s,a) \right] & \text{if } q \leq p, \\ a_{random} & \text{Otherwise,} \end{cases} \quad (5)$$

where ξ is a parameter that weights the effects of the heuristic function, q is a random value and p defines the tradeoff between exploration and exploitation: during the action selection, if q is larger than p, a randomly chosen action a_{random} is used to explore the problem domain.

To make sure that the action $\pi^H(s)$ is executed after being selected, the value of $H(s, \pi^H(s))$ must be larger than the variation of the $H(s,a)$ values, for a given $s \in S$ and for all $a \in A$. The values of the heuristic function can be computed using Eq. 6 below:

$$H(s,a) = \begin{cases} \max_i \hat{Q}(s,i) - \hat{Q}(s,a) + \eta \text{ if } a = \pi^H(s), \\ 0 \text{ otherwise.} \end{cases} \quad (6)$$

where η is a small real number. By computing H in this way, Eq. 5 selects the action suggested by the heuristic in all greedy steps. HAQL is presented in Algorithm 2.

Q-learning, $Q(\lambda)$ and HAQL are used in this paper as base algorithms whose performances are compared with the TLHARL algorithms on the two robotic domains considered.

Algorithm 2 Heuristically Accelerated Q–Learning (HAQL) [10].

Initialise $\hat{Q}_t(s,a)$ and $H_t(s,a)$ arbitrarily.
Repeat (for each episode):
  Initialise s.
  Repeat (for each step):
    Update the values of $H_t(s,a)$ as desired
    Select an action a using equation 5.
    Execute the action α, observe $r(s,a)$, s'.
    Update the values of $Q(s,a)$ according to equation 2.
    $s \leftarrow s'$.
  Until s is terminal.
Until some stopping criterior is reached.

2.2 Case-Based Reasoning

Case-based reasoning (CBR) [28] is a subfield of AI that aims at developing algorithms capable of using a case-base that contains pre-existing solutions of a problem for solving new problems that are similar to the ones described in the known cases.

According to Lopez de Mántaras et al [28], CBR begins with a problem description, which is matched with cases stored in a case base using some pre-defined similarity measure; if similar cases are found, they are retrieved, adapted and reused in the problem at hand.

In general, in CBR a case is composed of two parts: the problem description (P) and the solution description (A). The problem description P is a representation of the situation in which the case can be used, and is similar to the definition of state in the set of states $\mathcal{S}$ in RL. The solution description A stores the sequence of actions that an agent has to perform in order to solve the problem P. This concept is related to the optimal policy π^* in RL, that maps states into the most desirable actions a to be performed.

The case retrieval process computes the similarity between the current problem and the cases in the base in order to find the most similar case to be retrieved. Similarity is defined by means of a function on the distance between agents and objects in the problem and in the case in consideration.

There is no guarantee that the retrieved case is an exact solution to the problem at hand. When the most similar case is not an exact solution, it is necessary to adapt the retrieved case to solve the problem at hand before reusing it. This is done by modifying the case, usually by adding steps to the solution, or by rotating or translating it.

3 Related Work

Transfer Learning has been studied since the beginning of the 20th century, with the work of Thorndike and

Woodworth [49] considering the influence of one mental function to enhance the efficiency of other functions. In the field of Artificial Intelligence, transfer learning inherits ideas from several subfields of machine learning, such as inductive transfer [30], metalearning [26], multi-task learning [12, 13, 50], imitation learning [3] and human advice [21]. The first example of an algorithm that uses the transference of knowledge from one task to another to accelerate a reinforcement learning procedure was proposed in [16]. Since then, there has been an increasing interest in applying transfer learning to RL [1, 6, 18, 24, 47]. The work reported by Patricia and Caputo [34] extends the idea of TL to use prior knowledge from any source domain independently of the origin of the distribution mismatch between source and target domains, resulting in a class of algorithms applicable in any binary or multi-class transfer learning with single or multiple source domain adaptation settings. The work presented in this paper differs from these early approaches by using a case-base, and case-based reasoning techniques, to transfer knowledge across domains.

There has been an increasing interest in applications of transfer learning combined with RL. For instance, Zhang et al. [55] applies this combination on the optimisation of large-scale power systems where a matrix is used to store state-action pairs that are, then, transfered to distinct optimisation tasks. Zhang et al. [56] applies the transference of knowledge, allied with Q-learning, in the development of an optimal controller for the automatic generation control of highly interconnected power grids. Tan et al. [44] propose a combination of TL with transitive inference that allows the transference of knowledge across unrelated domains by selecting intermediate domains to act as a *bridge* connecting source and target. Gupta et al. [22] examine how reinforcement learning algorithms can transfer knowledge between different agents, by training invariant feature spaces that can then be used to transfer skills from one agent to another. This work is interesting, as it transfers knowledge between morphologically different robots, in simulation. Recent surveys on TL can be found in [32, 52].

The present paper falls within the context of applications of transfer learning to RL and, to the best of our knowledge, this is the first work that investigates the application of these ideas to accelerate the learning in real robot domains.

We believe that the work proposed here could be further extended to be used within a Deep Reinforcement Learning setting. The use of Deep Reinforcement Learning together with Transfer Learning is very promising, as it allows an agent to learn its behaviour with respect to a variety of, possibly simultaneous, tasks, and then to generalise this behaviour to new domains [33]. One interesting application of this idea was described by Glatt et al. [20] on the Atari game playing domain. Similarly, Du et al. [17] applies transfer learning to the Deep Q-network in two different tasks: Atari game playing and cart-pole balancing. The results reported show that TL improved the time efficiency of the Deep Q-network.

The first TLHARL algorithm proposed was the Transfer Learning Heuristically Accelerated Q–learning (TLHAQL) [15], which implements the TLHARL concepts using the Q–learning algorithm. One extension of TLHAQL that automatically maps the source-domain actions to the target-domain actions was evaluated in simulated tasks in [8].

The main difference between the work presented here and our previous work presented in [8] is that in the present work the focus is on the investigation of algorithms that can be used within the transfer-learning framework and testing then in real-world, real-robot, domains; whereas, the focus of our previous work was on evaluating the use of a neural network for the task of automatic mapping the source-domain actions to the target-domain actions.

A complete survey (up to 2009) of transfer learning for reinforcement learning was described in [46]. More recent surveys related to TL can be found in [25, 27, 32]. Finally, Bengio [7] presents a survey of deep learning methods and how they can be used in the transfer-learning scenario.

Algorithm 3 Transfer Learning Heuristically Accelerated Reinforcement Learning (TLHARL).

1. Use any RL algorith to learn the optimal policy π_S^* for the source domain.
2. Create a case-base from the π_S^* learned.
3. Use the case-base as heuristics to speed-up the learning in the target domain, using any HARL algorithm.

The next section introduces the Transfer Learning Heuristically Accelerated Reinforcement Learning algorithms, that are the main subject of investigation in this work.

4 Transfer Learning Heuristically Accelerated Reinforcement Learning

The Transfer Learning Heuristically Accelerated Reinforcement Learning (TLHARL) class of algorithms achieves the transference of knowledge across domains by combining Reinforcement Learning, Heuristics and Case-based reasoning.

TLHARL works by, first, applying an RL algorithm to learn a single task (considered as the source task). When learning stabilises, i.e., $\hat{Q}(s', a') - \hat{Q}(s, a)$ is close to zero, a case base with a pre-defined number of cases is built.

After that, this case base is transferred to be used on another learning task (the target task). During this phase, cases are retrieved and adapted to the current situation following CBR ideas. A case is retrieved if a similarity measurement is above a certain threshold. When this happens, the action suggested by the case is selected and used as heuristic in a HARL algorithm. The fundamental structure of a TLHARL algorithm is presented in Algorithm 3.

One of the main advantages of using TLHARL is that, if a case is not found that is similar to the situation at hand, the algorithm is reduced to the underlying RL algorithm. Thus, if the case base contains a case that can be used in a given situation, the RL process is accelerated using this case as heuristics. On the other hand, if a similar case is not found, or even if a misleading case is used as heuristics, the algorithm still converges to the optimal solution by means of a basic RL procedure.

The present paper proposes a novel TLHARL algorithm: TLHAQ(λ), based on the traditional RL algorithm $Q(\lambda)$ [51] (described in Section 2.1). The implementation of TLHAQ(λ) is similar to that of TLHAQL extended with eligibility traces, where at each time step the eligibility trace is updated according to Eq. 3 and all state-action pair rules are updated using Eq. 4. The TLHAQ(λ) is presented in Algorithm 4.

Next section describes the tests conducted to investigate the performances of TLHAQL and TLHAQ(λ) in two robotic domains.

Algorithm 4 Transfer Learning Heuristically Accelerated $Q(\lambda)$.

Initialise $\hat{Q}(s, a)$ and $H_t(s, a)$ arbitrarily.
Repeat (for each episode)
  Initialise s.
  Repeat (for each step):
    Compute similarity
    If there is a case that can be reused:
      Compute $H_t(s, a)$ using equation 6 with the actions suggested by the case selected.
      Select an action a using equation 5.
    If not:
      Select an action a using an ϵ-greedy policy.
    Execute the action a, observe $r(s, a)$, s'
    Update the values of $e(u, v)$ for all $u \in S$ and $v \in A$ according to equation 3.
    Update the values of all $Q(s, a)$ according to equation 4.
    $s \leftarrow s'$
  Until s terminal.
Until some stopping criteria is reached.

5 Experiments and Results

This section describes experiments of applying the TLHARL algorithms in two distinct real-robot domains. The first experiment (Section 5.1) investigates the acceleration gained by using TLHAQL to learn and transfer the strategy learnt from a simulated multi-robot soccer game to an equivalent match with a group of real robots. This was the first experimental evaluation of TLHAQL in a real-robotic domain, described in Celiberto et al. [14].

The second experiment (Section 5.2) aims at making a more complete analysis of the two TLHARL algorithms, comparing TLHAQL and TLHAQ(λ) in the task of transferring the control policy learnt from a two-link pendulum operating in a vertical plane (an Acrobot) to speed up the stability learning of a two-legged humanoid robot.

5.1 Experiment 1: Robot Soccer Using a Team of Small-Scale Robots

The first experiment investigates the transference of cases acquired in the RoboCup Soccer 2D simulator [31] (the source domain, illustrated in Fig. 1) to speed up the learning of a defence strategy of a group of Kilobot robots [36] (shown in Fig. 2), which is the target domain for transfer learning.

The RoboCup Soccer 2D [31] (Fig. 1) is a simulator that allows soccer matches between pairs of independent agent teams. The simulator is comprised of a server (that controls the game and computes the movements of all players and the ball) and clients, which are autonomous agents that control each individual player. The communication between the server and the clients is accomplished via TCP/IP sockets.

Fig. 1 The RoboCup Soccer 2D simulator, source domain.

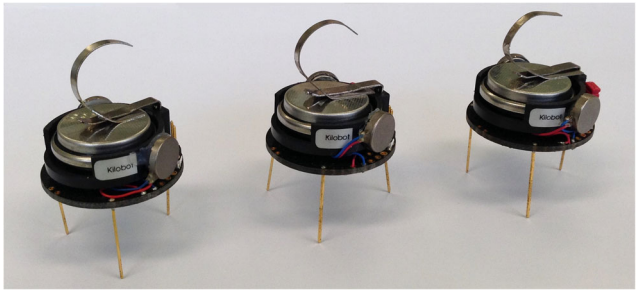

Fig. 2 The Kilobot Robots that compose the target domain

The simulation is executed in cycles of approximately 100 milliseconds: in each cycle the server executes the actions requested by the clients and updates the world state.

In the RoboCup simulator, vision is limited to a 90°–wide cone; the maximum distance up to which an object can be seen is also limited. Only objects within the view angle, and within visible distance of the viewing agent, can be picked out by the sensors. These limitations are mutually related: the wider the cone of view (limited to 90°), the shortest is the visible distance. These limitations create a number of hidden states (for example, when the ball is behind the robot), which are not relevant for the learning process in the target domain. To overcome these limitations, some of the default settings of the simulation were adapted in this work, allowing players to have a 360° view of the soccer field, without imposing a maximum distance for object perception.

The target domain is composed of three Kilobot robots (Fig. 2), that are low-cost small-scale robots suitable for multiagent experiments [36]. Kilobots are mobile robots of around 3cm of diameter that move by vibrating stilt-like legs using two motors. Each robot is controlled by an ATmega 328 microprocessor, and communicates with the others by pulsing an infrared LED located underneath it, whose reflections on the floor are picked out by the other agents up to a distance of 10cm around the robot [48]. The robots are powered by a rechargeable battery that enables experiments lasting about 3 hours.

In the source domain, the case base was learnt by implementing $Q-Learning$ on one of the simulated teams, whilst the opponent was running the straightforward strategy of *kicking the ball forward* only.[1] The learning procedure considered the following actions for each agent in the team: *hold the ball*, *pass the ball* and *kick to goal*. Other actions were allowed in the match (such as *search the ball*, *intercept the ball* or *dribble*), but not included in the learning process, since they are not relevant for the learning task in the target domain.

[1]We used for the latter the implementation available in [11], used by the UvA Trilearn 2003 Team.

In this domain, 10 cases were randomly retrieved after the end of the learning phase. Every retrieved case consists of the problem description P, defined as the distances between player and ball, that are compatible with the acceptance rate of the Kilobot robot ($0 \sim 10$ cm), and the action A taken in this situation. These 10 cases constituted the case base. No more than 10 cases could be considered here due to the Kilobot's limited memory.

The goal of the learning robot in the target domain is to stay between two other robots: one assuming the role of the ball and the other assuming the role of an opponent player. Only the learning robot has the ability to move in the environment: the other two robots are used only as beacons, to send information about distance between them and the learning robot. The set of possible actions of the learning agent is: *stay in the same position*, *move forward*, *turn right* (45°) and *turn left* (45°). The learning environment is composed of a 0,3 × 0,2 metre reflexive surface, where the fixed robots are separated by 10 cm.

In this experiment, the learning robot uses a simple tabular $Q-Learning$ algorithm (due to its limited computing capabilities). At every step, the robot observes its state and then searches the case base for cases that are similar to the current state, using the Euclidean distance as similarity function. If a similar case is found, an heuristic is defined in the following manner:

$$H(s,a) = \begin{cases} 10 \text{ if } a = \pi^H(s), \\ -1 \text{ otherwise.} \end{cases} \quad (7)$$

where $\pi^H(s)$ is the action suggested by the case, that leads the learning agent to the goal position.

The HAQL algorithm used a simple heuristics: *turn right* if the robot is further than 6 cm from opponent, and *turn left* if it is further than 6 cm from the ball.

The results, presented in Fig. 3, show the performances of TLHAQL, Heuristically Accelerated Q–learning (HAQL) and the $Q-Learning$ (QL) algorithms. In the X-axis of this figure are the episodes, which consist of 3.000 steps taken by the learning agent. In the Y-axis is shown the number of times that the learning robot reached the goal, i.e, the number of times that it placed itself between the opponent and the ball. From these results it can be seen that the proposed TLHAQL algorithms present a superior performance when compared to both $Q-Learning$ and $HAQL$.

In the experimental sciences it is important to use statistical tests to validate the hypothesis that two sets of data are significantly different from each other. These same tests can be used to verify if a newly proposed algorithm is better than existing ones.

One statistical hypothesis test that can be used to verify if a new finding is statistically significant is Student's t-Test [38]. The t-statistics was proposed in 1908 by William

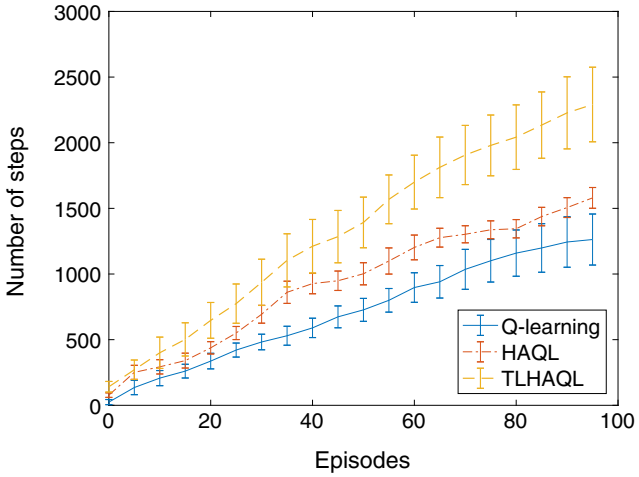

Fig. 3 The learning curves for the Q–learning, HAQL and TLHAQL algorithms

Gosset, who used the test as a method to monitor que quality of stout in the Guiness brewing company [40].

To use this test to verify if the performance of two algorithms are statistically different, the value of T is computed using the following Equation:

$$T = \frac{\bar{X}_1 - \bar{X}_2}{\sqrt{\frac{s_1^2}{N_1} + \frac{s_2^2}{N_2}}} \tag{8}$$

where $\bar{X}_1$, s_1 and N_1 are the sample mean, the variance and the sample size of the first algorithm and $\bar{X}_2$, s_2 and N_2 are the sample mean, the variance and the sample size of the second algorithm. This equation is known as Welch's unequal variances t-test [53], and is used when the two samples have distinct variances, as is the case with respect to results of computer algorithms.

From this equation, it can be seen that the grater the difference between the results (the difference between the two mean values), the higher the value of T. As a consequence, the higher the value of T, the more significantly different are the results under comparison.

From the Student's t-distribution table [38], one can determine the value of T that defines a confidence level treshold: if the computed value of T is above the confidence level treshold, then the performance of the algorithms are statistically different, with the probability that this result is wrong being less than a pre-defined level. This level, called statistical significance, is usually set to 5%, 2.5% or 1%.

In this experiment, the t-test is used to verify if the TLHAQL is statistically better than $Q-Learning$ (Fig. 4a) and the Heuristically Accelerated Q–learning (Fig. 4b). For each pair of algorithms, the value of T is computed at each episode, using the same data presented in Fig. 3. The value of T for a confidence level with a statistical significance of 1% are also presented in these Figures, plotted as dotted lines. It is possible to see in both Figures that TLHAQL is significantly better than $Q-Learning$ from the initial episodes, and that it is better than HAQL from the 50^{th} episode onwards, whereas the probability that this result is wrong falls within less than 1%.

5.2 Experiment 2: RoboCup KidSize Humanoid-Robot Stabilisation

In the second experiment the Acrobot [39] is used as source domain for testing the rate of acceleration provided by TLHAQ(λ) to learn the stabilisation of a humanoid robot used in the RoboCup Humanoid KidSize League.

The Acrobot is a traditional problem that has been used as testbed for new robotics, control and learning algorithms by several researchers [4, 5, 29, 42, 57]. It consists of a vertical, planar, two-link pendulum, where only the second link has an actuator that can be controlled, i.e, the first link is passive (Fig. 5). The task, in this problem, is to learn how to

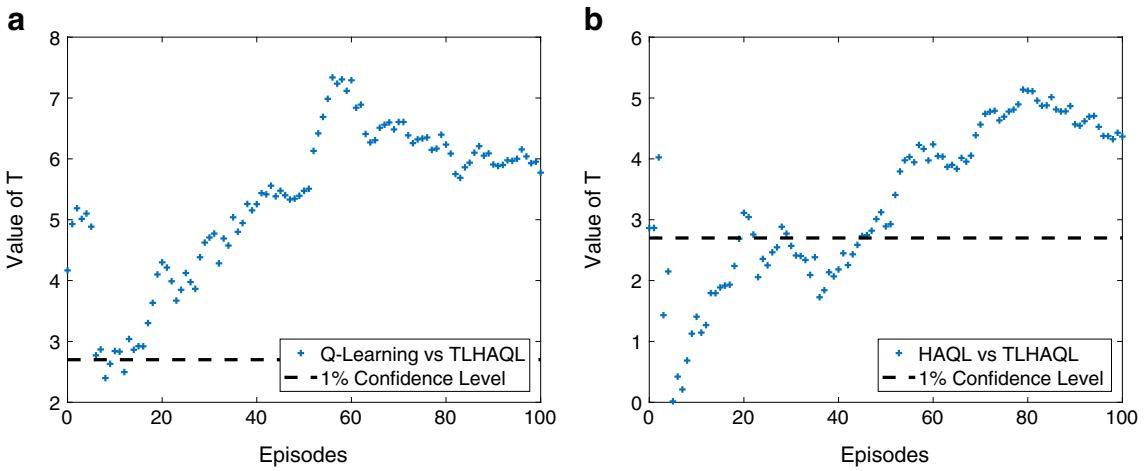

Fig. 4 Results from Student's t-Test between $Q-Learning$ and TLHAQL (**a**) and HAQL and TLHAQL (**b**)

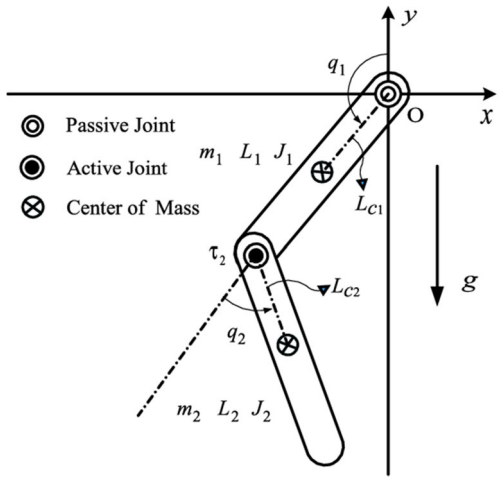

Fig. 5 The model of an Acrobot (Figure from Zhang et al. [57]), source domain

maintain the pendulum in an upward, unstable equilibrium position, starting from the rest position. The state of this system is defined by four continuous variables, $\theta_1, \theta_2, \dot{\theta_1}$ and $\dot{\theta_2}$, the position and velocity of each joint, respectively. The actions that can be used in the Acrobot are: to apply a positive or a negative torque in the second link actuator, or to do nothing.

In order to test the transfer to the target domain, a humanoid robot developed to compete in the RoboCup Humanoid League was used. The robot, shown in Fig. 6, is a modification of the open source DARwIn-OP robot [23], which includes the addition of a NUC Intel i5 computer as main processor, new electronics and new mechanical parts made by additive manufacturing. The robot, which is described in more detail in Perico et al. [35], has 6 DOF in each leg, 3 in each arm and 2 on the head, each one actuated by a Dynamixel RX-28 motor. The robot's weight is 3.0 Kg, its height is 490 millimetres and it can walk 10 cm/s.

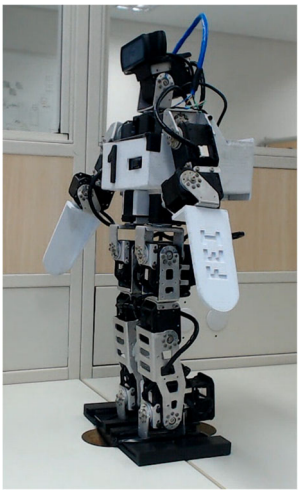

Fig. 6 The humanoid robot used on the target domain

The set of states used in the target domain is the position of three joints, the Hip, the Knee and the Foot, that can be moved from $-20°$ to $+20°$. All other joints in the robot remain in a fixed position. The set of actions that can be used by the robot are: $+1°$ *Hip Pitch*, $+1°$ *Knee Pitch*, $+1°$ *Foot Pitch*, $-1°$ *Hip Pitch*, $-1°$ *Knee Pitch*, $-1°$ *Foot Pitch* or *no action*. The initial state is a random position in the $-20°$ to $+20°$ range, and the final position is with the robot standing upright.

At the beginning of this experiment, the $Q(\lambda)$ Algorithm is used on the source domain to build the case base containing 500 cases. The case base is built after learning stabilises, which happens around the 9,000th episode, by sampling the action-state set, and selecting actions that are the best ones for the state (the worst action for one state is never selected to be included in the case base). An episode ends at goal state, or when a limit of 20,000 steps is reached.

After the case base is built, the TLHAQ(λ) algorithm is used in the target domain. The similarity between a case in the case base and the state of the robot in the target domain is given by the difference between the joint angles in the two domains.

One decision that had to be made in this experiment is the selection of the robot's joint where the knowledge transfer would be more effective. To transfer the learning from Acrobot to the humanoid robot, three different joints could be used: the *feet joint*, the *knee joint* or the *hip joint*. In this experiment, TLHAQ(λ) was used independently in two of the robots joints: *Hip* and *Ankle*. In other words, a learning session was conducted using the Acrobot's case base applied to the humanoid's *Hip* and another applying the case base to the humanoid's *Ankle*.

The results obtained are presented in Fig. 7, which shows the number of times *per* episode taken by the agent to reach the goal. Both versions of TLHAQ(λ) (applied to the *Hip* and to the *Ankle*) were compared with Q(λ). The results present the average of thirty training sessions for each of the algorithms, each of these sessions consists of 400 episodes of 120 seconds each.

The following parameters were used throughout these experiments: the learning rate α used was 0.10, the discount factor γ was 0.9, the exploration versus exploitation rate was set to 0.15, $\lambda = 0.3$ and the Q table was initialised with zeros. TLHAQ(λ) used $\eta = 1$. Both algorithms received a reward of -1 on all steps that did not lead to the goal, the goal state was rewarded with +1000 and hitting the final course of the servomotor was rewarded with -100.

From the results presented we can conclude that TLHAQ(λ) outperforms Q(λ) in the initial learning phase, and both algorithms converge to a similar performance

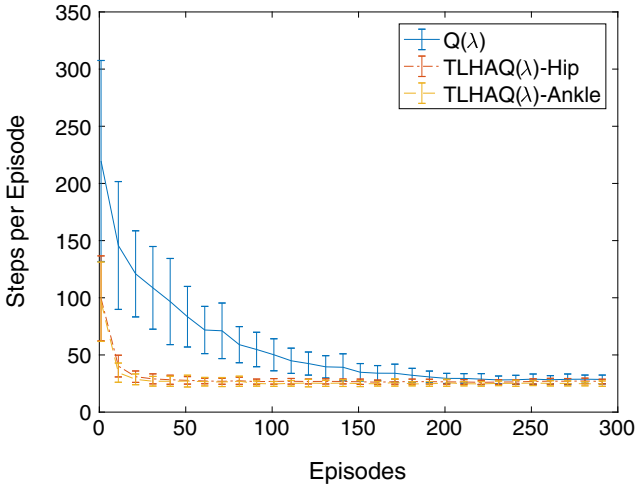

Fig. 7 The learning curves for Q(λ) and TLHAQ(λ) for the Humanoid Robot Stabilisation Problem

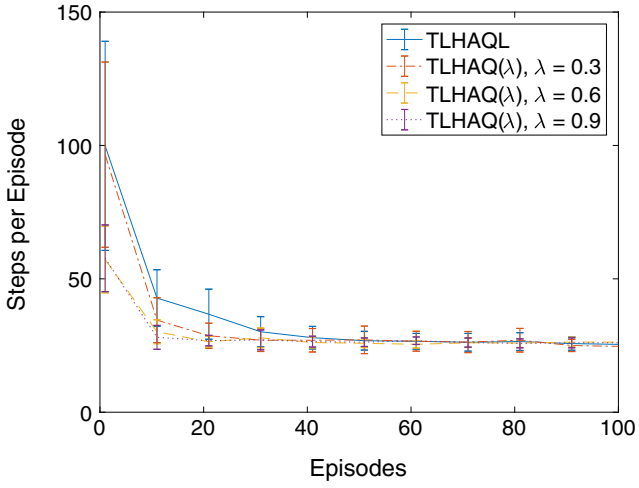

Fig. 9 The learning curves for TLHAQL and TLHAQ(λ), with $\lambda = 0.3$, 0.6 and 0.9, for the Humanoid Robot Stabilisation Problem

result in later episodes, as expected. Student's t-Test was used again to verify if the performance of the algorithms are statistically different. The results of the Student's t-Test, shown in Fig. 8, show that the performance obtained for TLHAQ(λ) is statistically superior than the performances of Q(λ), up to the 200^{th} episode, with a confidence level of 1%. After that, the results are statistically indistinguishable.

The experimental results described above show a significant improvement of the method proposed in this paper compared with standard Q(λ) algorithm. To verify that TLHAQ(λ) improves the learning rate when compared with the TLHAQL without eligibility traces (described in our previous work [14]), thirty training sessions were executed, each of these sessions consists of 100 episodes of 120 seconds using the TLHAQL and the TLHAQ(λ) algorithms. In the case of the TLHAQ(λ), three different values of λ were considered: 0.3, 0.6 and 0.9. This algorithm used the Acrobot's case base applied to the humanoid's *Hip* only. The parameters and rewards used in this experiment are the same as before.

The results obtained are presented in Fig. 9, which shows the number of times *per* episode taken by the agent to reach the goal. This figure clearly shows the positive effect of using eligibility traces in the TLHAQ(λ) algorithm, as the greater is the value of the λ, the faster the algorithm converges to the optimal solution.

The results of the Student's t-Test are shown in Fig. 10, where we can see that the performance obtained for TLHAQ(λ) with $\lambda = 0.6$ and 0.9 are statistically better than the performance of TQHAQL up to the 30^{th} episode, with the probability that this result is wrong being less than 1% (a 99% confidence level).

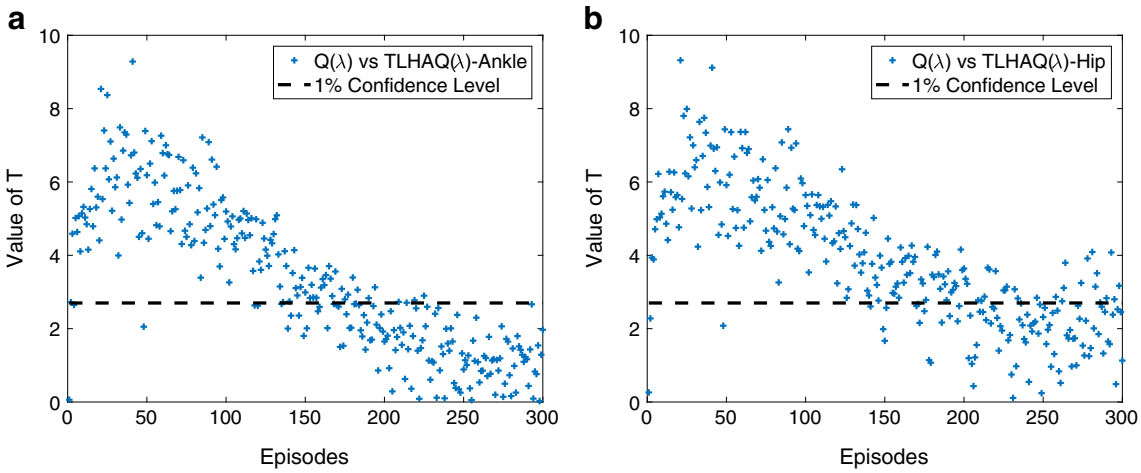

Fig. 8 Student's t-test between Q(λ) and TLHAQ(λ) used in the Ankle **a** and the Hip **b** joint

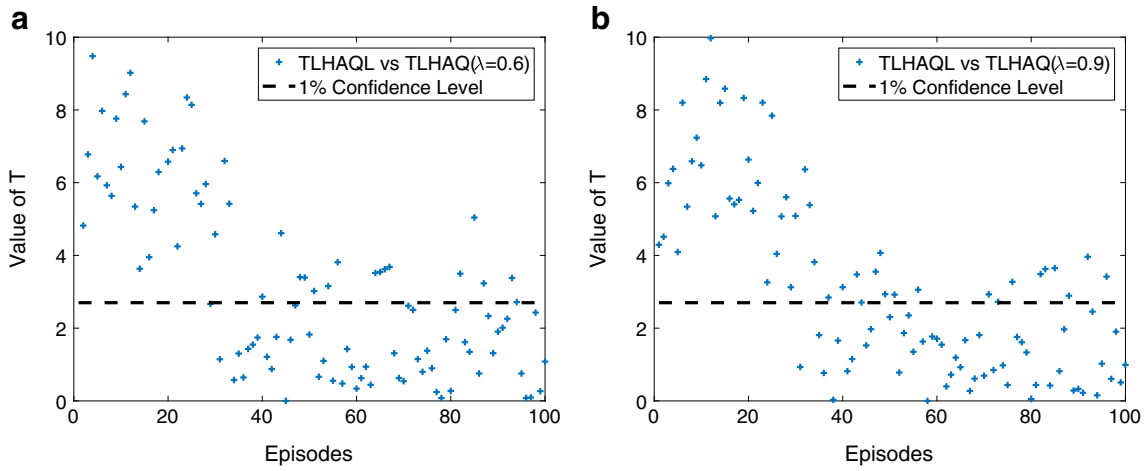

Fig. 10 Student's t-test between TLHAQL and TLHAQ(λ) used in the Hip joint, with $\lambda = 0.6$ (**a**) and $\lambda = 0.9$ (**b**)

6 Concluding Remarks

This paper investigated a class of algorithms, called Transfer Learning Heuristically Accelerated Reinforcement Learning Algorithms (TLHARL), that combines case-based reasoning and transfer learning to speed-up the convergence of reinforcement learning algorithms. The main contribution of this work is twofold: first, the introduction of a novel TLHARL algorithm based on the well-known reinforcement learning algorithm Q(λ) (the TLHAQ(λ) algorithm) and, second, the evaluation of TLHARL on two real-robot domains.

The domains considered in this work, upon which TLHARL was evaluated, consisted of learning the strategy in multi-robot soccer game and the stabilisation learning of a humanoide robot. In the first domain, a 2D robot soccer simulation was used as the source from where a set of heuristics were learnt as a case base to be transfered to a domain with three small-scale physical robots. The second domain consisted of a two-link pendulum (an Acrobot) from where a basic notion of balance was learnt that was transfered to accelerate the stabilisation learning of two joints of a humanoid robot (one joint at a time).

The results obtained show that the use of a case-base to transfer learning across domains resulted in algorithms that outperform other techniques for accelerating reinforcement learning, such as the use of eligibility traces or heuristics. The fast acceleration performed by TLHAQL and TLHAQ(λ), in contrast to that obtained by HAQL and HAQ(λ), can be explained by the fact that cases retrieved from a case base (built from a related domain) provide a better estimation of the distance between current and goal states than the fixed heuristic present in HAQL, reducing thus the search space.

Future work includes the use of TLHARL on more complex tasks, with larger state spaces, various actions and multiple tasks, characteristics that will increase the difficulty and complexity of the problem to be solved. Problems such as the multiobjective, multiparametric optimisation of robot walking and the solution of higher-level tasks in the Humanoid Soccer RoboCup domain, drone flight control and autonomous driving are within our future research interests.

Acknowledgements Reinaldo Bianchi acknowledges support from FAPESP (2016/21047-3), Paulo E. Santos acknowledges support from FAPESP-IBM (2016/18792-9) and CNPq (307093/2014-0), Isaac J. da Silva acknowledges support from CAPES, and Ramon Lopez de Mantaras acknowledges support from Generalitat de Catalunya Research Grant 2014 SGR 118 and CSIC Project 201550E022.

References

1. Aha, D.W., Molineaux, M., Sukthankar, G.: Case-based reasoning in transfer learning. In: Proceedings of the 8th International Conference on Case-Based Reasoning: Case-Based Reasoning Research and Development, ICCBR '09, pp. 29–44. Springer-Verlag, Berlin (2009)
2. Araujo, E.G., Grupen, R.A.: Learning control composition in a complex environment. In: Proceedings of the Fourth International Conference on Simulation of Adaptive Behavior, pp. 333–342. MIT Press/Bradford Books (1996)
3. Argall, B.D., Chernova, S., Veloso, M., Browning, B.: A survey of robot learning from demonstration. Robot. Auton. Syst. **57**(5), 469–483 (2009)
4. Astrom, K.J., Furuta, K.: Swinging up a pendulum by energy control. Automatica **36**(2), 287–295 (2000)
5. Atkeson, C.G., Schaal, S.: Robot learning from demonstration. In: International Conference on Machine Learning, pp. 12–20 (1997)
6. Banerjee, B., Stone, P.: General game learning using knowledge transfer. In: The 20th International Joint Conference on Artificial Intelligence, pp. 672–677 (2007)
7. Bengio, Y.: Deep learning of representations for unsupervised and transfer learning. In: Guyon, I., Dror, G., Lemaire, V., Taylor, G., Silver, D. (eds.) Proceedings of ICML Workshop on Unsupervised and Transfer Learning, Proceedings of Machine Learning

Research, vol. 27, pp. 17–36. PMLR, Bellevue, Washington, USA (2012). http://proceedings.mlr.press/v27/bengio12a.html
8. Bianchi, R., Celiberto, L.A., Matsuura, J., Santos, P., de Mántaras, R.L.: Transferring knowledge as heuristics in reinforcement learning: a case base approach. Artif. Intell. **226**, 102–121 (2015)
9. Bianchi, R.A.C., Ribeiro, C.H.C., Costa, A.H.R.: Heuristically Accelerated Q-Learning: a new approach to speed up reinforcement learning. Lect. Notes Artif. Intell. **3171**, 245–254 (2004)
10. Bianchi, R.A.C., Ribeiro, C.H.C., Costa, A.H.R.: Accelerating autonomous learning by using heuristic selection of actions. J. Heuristics **14**(2), 135–168 (2008)
11. de Boer, R., Kok, J.: The Incremental Development of a Synthetic Multi-Agent System: The UvA Trilearn 2001 Robotic Soccer Simulation Team. Master's Thesis. University of Amsterdam, Amsterdam (2002)
12. Caruana, R.: Learning many related tasks at the same time with backpropagation. In: Advances in Neural Information Processing Systems 7, pp. 657–664. Morgan Kaufmann (1995)
13. Caruana, R.: Multitask learning. Mach. Learn. **28**(1), 41–75 (1997)
14. Celiberto, L.A. Jr., Bianchi, R.A.C., Santos, P.E.: Transfer learning heuristically accelerated algorithm: a case study with real robots. In: 2016 Latin American Robotics Symposium and Intelligent Robotics Meeting, pp. 311–315 (2016)
15. Celiberto, L.A. Jr., Matsuura, J.P., de Mantaras, R.L., Bianchi, R.A.C.: Using transfer learning to speed-up reinforcement learning: A cased-based approach. In: 2010 Latin American Robotics Symposium and Intelligent Robotics Meeting, pp. 55–60 (2010)
16. Drummond, C.: Accelerating reinforcement learning by composing solutions of automatically identified subtasks. J. Artif. Intell. Res. **16**, 59–104 (2002)
17. Du, Y., de la Cruz, G.V., Irwin, J., Taylor, M.E.: Initial progress in transfer for deep reinforcement learning algorithms. In: International Joint Conference on Artificial Intelligence (2016)
18. Fernández, F., Veloso, M.: Probabilistic policy reuse in a reinforcement learning agent. In: Proceedings of the Fifth International Joint Conference on Autonomous Agents and Multiagent Systems, AAMAS '06, pp. 720–727. ACM, New York, NY, USA (2006)
19. Ferreira, L.A., Costa Ribeiro, C.H., da Costa Bianchi, R.A.: Heuristically accelerated reinforcement learning modularization for multi-agent multi-objective problems. Appl. Intell. **41**(2), 551–562 (2014)
20. Glatt, R., da Silva, F.L., Costa, A.H.R.: Towards knowledge transfer in deep reinforcement learning. In: Proceedings of the Brazilian Conference on Intelligent System (BRACIS), pp. 91–96 (2016)
21. Griffith, S., Subramanian, K., Scholz, J., Isbell, C.L., Thomaz, A.L.: Policy shaping: Integrating human feedback with reinforcement learning. In: Burges, C.J.C., Bottou, L., Ghahramani, Z., Weinberger, K.Q. (eds.) NIPS, pp. 2625–2633 (2013)
22. Gupta, A., Devin, C., Liu, Y., Abbeel, P., Levine, S.: Learning invariant feature spaces to transfer skills with reinforcement learning. In: Proceedings of the Fifth International Conference on Learning Representations. OpenReview, Toulon, France (2017)
23. Ha, I., Tamura, Y., Asama, H., Han, J., Hong, D.W.: Development of open humanoid platform darwin-op. In: SICE Annual Conference 2011, pp. 2178–2181 (2011)
24. von Hessling, A., Goel, A.K.: Abstracting reusable cases from reinforcement learning. In: Brüninghaus, S. (ed.) 6th International Conference on Case-Based Reasoning, ICCBR 2005, Chicago, IL, USA, August 23-26, 2005, Workshop Proceedings, pp. 227–236 (2005)
25. Lazaric, A.: Transfer in Reinforcement Learning: A Framework and a Survey, pp. 143–173. Springer Berlin Heidelberg, Berlin (2012)
26. Lemke, C., Budka, M., Gabrys, B.: Metalearning: a survey of trends and technologies . Artif. Intell. Rev. **44**, 1–14 (2013)
27. Lu, J., Behbood, V., Hao, P., Zuo, H., Xue, S., Zhang, G.: Transfer learning using computational intelligence. Know.-Based Syst. **80**(C), 14–23 (2015). https://doi.org/10.1016/j.knosys.2015.01.010
28. de Mántaras, R.L., McSherry, D., Bridge, D., Leake, D., Smyth, B., Craw, S., Faltings, B., Maher, M.L., Cox, M.T., Forbus, K., Keane, M., Aamodt, A., Watson, I.: Retrieval, reuse, revision and retention in case-based reasoning. Knowl. Eng. Rev **20**(3), 215–240 (2005)
29. Nichols, B.D.: Continuous action-space reinforcement learning methods applied to the minimum-time swing-up of the acrobot. In: 2015 IEEE International Conference on Systems, Man, and Cybernetics (SMC), pp. 2084–2089 (2015)
30. Niculescu-Mizil, A., Caruana: Inductive transfer for Bayesian network structure learning. In: Unsupervised and Transfer Learning - Workshop held at ICML 2011, Bellevue, Washington, USA, July 2, 2011, pp. 167–180 (2012)
31. Noda, I.: Soccer server: a simulator of robocup. In: Proceedings of AI Symposium of the Japanese Society for Artificial Intelligence, pp. 29–34 (1995)
32. Pan, S.J., Yang, Q.: A survey on transfer learning. IEEE Trans. Knowl. Data Eng. **22**(10), 1345–1359 (2010). https://doi.org/10.1109/TKDE.2009.191
33. Parisotto, E., Ba, L.J., Salakhutdinov, R.: Actor-mimic: Deep multitask and transfer reinforcement learning. arXiv:1511.06342 (2015)
34. Patricia, N., Caputo, B.: Learning to learn, from transfer learning to domain adaptation: A unifying perspective. In: Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition, CVPR '14, pp. 1442–1449. IEEE Computer Society, Washington, DC, USA (2014)
35. Perico, D.H., Silva, I.J., Vilão Junior, C.O., Homem, T.P.D., Destro, R.C., Tonidandel, F., Bianchi, R.A.C.: Newton: A high level control humanoid robot for the robocup soccer kidsize league. In: Osório, F.S., Wolf, D.F., Castelo Branco, K., Grassi, V. Jr.., Becker, M., Romero, R.A.F. (eds.) Robotics: Joint Conference on Robotics, LARS 2014, SBR 2014, Robocontrol 2014, São Carlos, Brazil, October 18-23, 2014. Revised Selected Papers, pp. 53–73. Springer Berlin Heidelberg, Berlin (2015)
36. Rubenstein, M., Ahler, C., Nagpal, R.: Kilobot: A low cost scalable robot system for collective behaviors. In: 2012 IEEE International Conference on Robotics and Automation, pp. 3293–3298 (2012)
37. Singh, S.P., Sutton, R.S.: Reinforcement learning with replacing eligibility traces. Mach. Learn. **22**(1), 123–158 (1996)
38. Spiegel, M.R.: Statistics. McGraw-Hill, New York (1998)
39. Spong, M.W.: The swing up control problem for the Acrobot. IEEE Control Syst. **15**(1), 49–55 (1995)
40. Student: The probable error of a mean. Biometrika **6**(1), 1–25 (1908)
41. Sutton, R.S.: Learning to predict by the methods of temporal differences. Machine Learning **3**(1), 9–44 (1988)
42. Suttom, R.S.: Generalization in reinforcement learning: Successful examples using sparse coarse coding. Adv. Neural Inf. Proces. Syst. **8**, 1038–1044 (1996)
43. Sutton, R.S., Barto, A.G.: Reinforcement Learning: An Introduction. MIT Press, Cambridge (1998)
44. Tan, B., Song, Y., Zhong, E., Yang, Q.: Transitive transfer learning. In: Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '15, pp. 1155–1164. ACM, New York, NY, USA (2015)

45. Taylor, M.E.: Autonomous Inter-task Transfer in Reinforcement Learning Domains. Ph.D. Thesis, Department of Computer Sciences, The University of Texas at Austin (2008)
46. Taylor, M.E., Stone, P.: Transfer learning for reinforcement learning domains: A survey. J. Mach. Learn. Res. **10**(1), 1633–1685 (2009)
47. Taylor, M.E., Stone, P., Jong, N.K.: Transferring instances for model-based reinforcement learning. In: Machine Learning and Knowledge Discovery in Databases, Lecture Notes in Artificial Intelligence, vol. 5212, pp. 488–505 (2008)
48. Tharin, J.: Kilobot User Manual. K-Team (2010)
49. Thorndike, E.L., Woodworth, R.S.: The influence of improvement in one mental function upon the efficiency of other functions. Psychol. Rev. **8**, 247–261 (1901)
50. Thrun, S., Mitchell, T.M.: Learning one more thing. In: IJCAI'95: Proceedings of the 14th International Joint Conference on Artificial Intelligence, pp. 1217–1223. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA (1995)
51. Watkins, C.J.C.H.: Learning from Delayed rewards. Ph.D. Thesis. University of Cambridge, Cambridge (1989)
52. Weiss, K., Khoshgoftaar, T.M., Wang, D.: A survey of transfer learning. J. Big Data **3**(1), 9 (2016)
53. Welch, B.L.: The generalization of "Student's" problem when several different population variances are involved. Biometrika **34**(1), 28–35 (1947)
54. Wender, S., Watson, I.: Combining case-based reasoning and reinforcement learning for tactical unit selection in real-time strategy game AI, pp. 413–429. Springer International Publishing, Berlin (2016)
55. Zhang, X., Yu, T., Yang, B., Cheng, L.: Accelerating bio-inspired optimizer with transfer reinforcement learning for reactive power optimization. Knowledge-Based Systems pp. – (2016)
56. Zhang, X.S., Li, Q., YU, T., Yang, B.: Consensus transfer q-learning for decentralized generation command dispatch based on virtual generation tribe. IEEE Trans. Smart Grid **PP**(99), 1–1 (2016). https://doi.org/10.1109/TSG.2016.2607801
57. Zhang, A., She, J., Lai, X., Wu, M.: Motion planning and tracking control for an acrobot based on a rewinding approach. Automatica **49**(1), 278–284 (2012)

Reinaldo A. C. Bianchi received the M.Sc. and Ph.D. degrees in electrical engineering from Escola Politécnica, Universidade de São Paulo, São Paulo, Brazil, in 1998 and 2004, respectively. From 2007 to 2009, he was a Visiting Researcher at the Artificial Intelligence Research Institute (IIIA-CSIC), Bellaterra, Barcelona, Spain. He is currently a Full Professor at the Department of Electrical Engineering, Centro Universitario FEI, São Bernardo do Campo, Brazil. His current research interests include Robotics, Artificial Intelligence, computer vision, machine learning and multiagent systems.

Paulo E. Santos Prof. Santos holds a degree in Physics from the Institute of Physics, University of São Paulo (1995), a Master in Electrical Engineering from the Polytechnic School, University of São Paulo (1997), a Master of Logic from the Institute for Logic, Language and Computation, University of Amsterdam (1998) and a Ph.D. in Artificial Intelligence from the Imperial College, London (2003). During 2003 and 2004 Prof. Santos was a Research Assistant in the computing department at University of Leeds (UK), and in 2005 he was appointed as a full-time lecturer in Artificial Intelligence at the Department of Electrical Engineering, Centro Universitário da FEI, São Paulo, Brazil, where he has been conducting high-level research, supervision and teaching graduate and undergraduate students in Artificial Intelligence and related areas since then.

Isaac J. da Silva is a PhD student of Electrical Engineering at Centro Universitario FEI. He holds a bachelor degree and a Master in Electrical Engineering from Centro Universitario FEI in 2013 and 2015, with research on the field of Artificial Intelligence. Currently, his research interests include Deep Learning applied to autonomous robots, Robot Soccer, Mobile Robots, Computer Vision and Humanoid Robots Control.

Luiz A. Celiberto Jr, Graduate Engineering at FEI (2004), Master in Electrical Engineering (focus Artificial Intelligence) at FEI (2007), Ph.D. in Electronic & Computer Engineering at Technological Institute of Aeronautics (ITA) (2012) and Postdoctoral at FEI (2013) in Robotics. Since 2014 is professor at Federal University of ABC in Center of Engineering, Modeling and Applied Social Sciences area and coordinator of the Robotic Intelligent System Group. The current focus of study includes: Transfer Learning, Reinforcement Learning, Multi-Agents System, Robotics and Autonomous Vehicles.

Ramon Lopez de Mantaras Research Professor of the CSIC and Director of the Artificial Intelligence Research Institute (IIIA). MSc in Computer Science from the University of California Berkeley, PhD in Physics from the University of Toulouse, and PhD in Computer Science from the Technical University of Barcelona. Author of nearly 300 papers. Editorial board member of several international journals and former Associate Editor of the Artificial Intelligence Journal. Recipient, among other awards, of the "City of Barcelona" Research Prize in 1981, the "2011 American Association of Artificial Intelligence (AAAI) Robert S. Engelmore Memorial Award", the "2012 Spanish National Computer Science Award" from the Spanish Computer Society, the Distinguished Service Award of the European Association of Artificial Intelligence (EurAI) in 2016, and the IJCAI Donald E. Walker Distinguished Service Award in 2017. President of the Board of Trustees of IJCAI from 2007 to 2009 and EurAI Fellow. He serves on a variety of panels and advisory committees for public and private institutions based in the USA and Europe. Presently working on case-based reasoning, machine learning, and AI applications to music. For additional information: http://www.iiia.csic.es/~mantaras.