Chapter 4
Case Based Reasoning, Reinforcement Learning and Heuristics: a Successful Combination

Reinaldo A. C. Bianchi

Abstract This paper presents some results of the collaboration between me and Professor Lopez de Mantaras, who received me as a visiting researcher at the IIIA from 2007 to 2009. The aim of the collaboration was to investigate a new approach that allows the use of cases in a case base as heuristics to speed up Single and Multiagent Reinforcement Learning algorithms, combining Case-Based Reasoning and Reinforcement Learning techniques. This approach, called Case-Based Heuristically Accelerated Reinforcement Learning, builds upon an emerging technique, Heuristic Accelerated Reinforcement Learning, in which RL methods are accelerated by making use of heuristic information. Algorithms that incorporates CBR techniques into the Heuristically Accelerated Q-Learning and Minimax–Q were proposed and a set of empirical evaluations were conducted in a simulator for the robot soccer domain. Experimental results showed that the algorithms proposed learn faster than methods using RL, CBR or Heuristics alone.

Reinaldo A. C. Bianchi
Centro Universitário da FEI São Bernardo do Campo, São Paulo, Brazil. e-mail: rbianchi@fei.edu.br

4.1 Introduction

Heuristic Accelerated Reinforcement Learning (HARL) [5] is an emerging technique in which Reinforcement Learning (RL) methods are sped up by making use of a conveniently chosen heuristic function, which is used for selecting appropriate actions to perform in order to guide exploration during the learning process.

HARL techniques are very attractive: as RL, they are based on firm theoretical foundations. As the heuristic function is used only in the choice of the action to be taken, many of the conclusions obtained for RL remain valid for HARL algorithms, such as the guarantee of convergence to equilibrium in the limit – given that some predefined conditions are satisfied – and the definition of an upper bound for the error [5]. Although several methods have been successfully applied for defining the heuristic function, a very interesting option has only recently been explored: the reuse of previously learned policies, using a Case-Based Reasoning approach.

This paper is the result of a collaboration that started in 2007, when Professor Lopez de Mantaras received the author as a visiting researcher at the IIIA. The goal of the collaboration was to investigate possible combinations of Case-Based Reasoning (CBR) and Single and Multiagent Heuristically Accelerated Reinforcement Learning (HARL and HAMRL) [5, 4] techniques, aiming the speeding up of algorithms by using previous domain knowledge, stored as a case base. We also proposed two new algorithms, the Case-Based Heuristically Accelerated Q–Learning (CB-HAQL), which incorporates CBR techniques into an existing HARL algorithm, the Heuristically Accelerated Q–Learning, (HAQL), and the Case-Based Heuristically Accelerated Minimax–Q (CB-HAMMQ), which incorporates CBR into the Heuristically Accelerated Minimax–Q (HAMMQ).

The paper is organized as follows: section 4.2 briefly reviews the Single and Multiagent Heuristically Accelerated Reinforcement Learning problem, while section 4.3 describes Case-Based Reasoning. Section 4.4 shows how to incorporate CBR techniques into RL algorithms, in a modified formulation of the HAQL and HAMMQ algorithms. Section 4.5 describes the domain used in the experiments, presents the experiments performed, and shows the results obtained. Finally, Section 4.6 provides our conclusions.

4.2 Single and Multiagent Heuristically Accelerated Reinforcement Learning

The single agent RL problem can be formulated as a discrete time, finite state, finite action Markov Decision Process (MDP), while systems where multiple agents compete among themselves to accomplish their tasks can be modeled as a discrete time, finite state, finite action Markov Game (MG) – also known as Stochastic Game (SG). MDPs and MGs are well described in several good surveys, such as [11, 13].

Formally, a Heuristically Accelerated Reinforcement Learning (HARL) algorithm [5] is a way to solve a MDP problem with explicit use of a heuristic function $\mathscr{H} : \mathscr{S} \times \mathscr{A} \rightarrow \Re$ for influencing the choice of actions by the learning agent. In the multiagent case (HAMRL), a heuristic function $\mathscr{H} : \mathscr{S} \times \mathscr{A} \times \mathscr{O} \rightarrow \Re$ is used. $H(s,a)$ or $H(s,a,o)$ define a heuristic that indicates the importance of performing the action a when visiting state s. The heuristic function is strongly associated with the policy: every heuristic indicates that an action must be taken regardless of others.

The first HARL algorithm proposed was the Heuristically Accelerated Q–Learning (HAQL) [5], as an extension of the Q–Learning algorithm. The only difference between them is that in the HAQL makes use of an heuristic function in the action choice rule, a modified $\varepsilon - greedy$ mechanism where a heuristic formalized as a function $H(s,a)$ is considered:

$$\pi(s) = \begin{cases} \arg\max_a \left[\hat{Q}(s,a) + \xi H(s,a)^\beta\right] & \text{if } q \leq p, \\ a_{random} & \text{otherwise,} \end{cases} \tag{4.1}$$

where ξ and β are design parameters that control the influence of the heuristic function, q is a random value uniformly distributed over $[0,1]$ and p ($0 \leq p \leq 1$) is a parameter that defines the exploration/exploitation tradeoff, and a_{random} is an action randomly chosen among those available in state s.

In a similar way, the first HAMRL algorithm proposed was the Heuristically Accelerated Minimax Q (HAMMQ) [4], as an extension of the Minimax–Q algorithm. Again, the only difference between them is that in the HAMMQ the heuristic function is used in the action choice rule:

$$\pi(s) = \begin{cases} \arg\max_a \min_o \left[\hat{Q}(s,a,o) + \xi H_t(s,a,o)\right] \text{ if } q \leq p, \\ a_{random} \text{ otherwise,} \end{cases} \tag{4.2}$$

.

As a general rule, the value of $H(s,a)$ or $H_t(s,a,o)$ used in HAQL should be higher than the variation among the $\hat{Q}(s,a)$ values for the same $s \in \mathscr{S}$, in such a way that it can influence the choice of actions, and it should be as low as possible in order to minimize the error. For example, it can be defined for the single agent case as:

$$H(s,a) = \begin{cases} \max_i \hat{Q}(s,i) - \hat{Q}(s,a) + \eta \text{ if } a = \pi^H(s), \\ 0 \text{ otherwise.} \end{cases} \tag{4.3}$$

where η is a small real value (usually 1) and $\pi^H(s)$ is the action suggested by the heuristic policy. Convergence of this algorithm was presented by Bianchi, Ribeiro and Costa [4], together with the definition of an upper bound for the error.

Despite the fact that RL is a method that has been traditionally applied in the Robotic Soccer domain, only recently have HARL methods been used in this domain. Bianchi, Ribeiro and Costa [4] investigated the use of a HAMRL algorithm in a simplified simulator for the robot soccer domain and Celiberto *et al.* [7] studied the use of the HARL algorithms to speed up learning in the RoboCup 2D Simulation domain. The heuristic used in both of these papers were very simple ones: in the first paper the heuristic was 'if the agent is with the ball, go to the opponent's goal', and in the second paper it was simply 'go to the ball'.

4.3 Case-Based Reasoning

Case-based reasoning (CBR) [8] uses knowledge of previous situations (cases) to solve new problems, by finding a similar past case and reusing it in the new problem situation. According to López de Màntaras *et al.* [8], solving a problem by CBR involves "obtaining a problem description, measuring the similarity of the current problem to previous problems stored in a case base with their known solutions, retrieving one or more similar cases, and attempting to reuse the solution of the retrieved case(s), possibly after adapting it to account for differences in

problem descriptions". In the CBR approach, a case usually describes a problem and its solution, i.e., the state of the world in a given instant and the sequence of actions to perform to solve that problem.

In this work, a case is composed of three parts [14]: the problem description (P), the solution description (A) and the case scope (K), and it is formally described as a 3-tuple:

$$case = (P, A, K). \tag{4.4}$$

The problem description P corresponds to the situation in which the case can be used. For example, for a Robotic Soccer problem, the description of a case can include the robot position, the ball's position and the positions of the other robots in the game. For a game with n robots (teammates and opponents), P can be:

$$P = \{x_B, y_B, x_{R_1}, y_{R_1}, \ldots, x_{R_n}, y_{R_n}\}. \tag{4.5}$$

The solution description is composed by the sequence of actions that each robot must perform to solve the problem, and can be defined as:

$$A = \{R_1 : [a_{1_1}, a_{1_2}, ..., a_{1_{p_1}}], \ldots, R_m : [a_{m_1}, a_{m_2}, ..., a_{m_{p_m}}]\},$$

where m is the number of robots in the team, a_{i_j} is an individual or joint action that robot R_i must perform and p_i corresponds the number of actions the robot R_i performs.

The case scope defines the applicability boundaries of the cases, to be used in the retrieval step. In the case of a robot soccer problem, K can be represented as circles or ellipsoids centered on the ball's and opponents' positions indicated in the problem description. It can be defined as:

$$K = \{\tau_B, \tau_{R_1}, \ldots, \tau_{R_n}\}, \tag{4.6}$$

where τ_B is the radius of the region around the ball and $\tau_{R_1} \ldots \tau_{R_n}$ the radius of the regions around the n robots in the game (teammates and opponents). The case retrieval process consists in obtaining from the base the most similar case, the retrieved case. Therefore, it is necessary to compute the similarity between the current problem and the cases in the base. The similarity function indicates how similar a problem and a case are. In most cases, the function is defined by the distance between the ball and the robots in the problem and in the case.

$$Sim(p,c) = dist(B^c, B^p) + \sum_{i=1}^{n} dist({R_i}^c, {R_i}^p), \tag{4.7}$$

where B^c is the position of the ball in the case and B^p its position in the problem, ${R_i}^c$ the position of the Robot i in the case and ${R_i}^p$ its position in the problem, and $dist(a,b)$ is the gaussian distance between object a and b. This distance is computed as follows:

$$dist(a,b) = e^{-\left((a_x-b_x)^2+(a_y-b_y)^2\right)/2\tau^2}, \tag{4.8}$$

where τ is the radius of the scope around the object. In this work, τ is the same for the ball and robots positions. The Gaussian distance is used because the larger the distance between two points, the lower the similarity between them. Finally, τ is used as a threshold that defines a maximum distance allowed for two points to have some degree of similarity: if $dist(a,b) > \tau$, $Sim(a,b) = 0$.

Before a case can be reused, it might be necessary to adapt it to the present situation. Adaptation of a case means that the retrieved solution is modified, by translation, rotation or the addition of steps to the sequence of actions in the solution before it can be used. In this work, we assume that rotation and translation costs are small when compared to the cost of the additional steps, because the first two are trivial computations, while the performance of additional steps by the robots are actions that must be executed (in the simulator or in the real world), taking more time. Therefore, we define the cost as the number of steps added to the adapted solution. In this work, the case that will be reused is the one that maximizes the similarity while minimizing the adaptation cost.

In recent years, CBR has been used by several researchers in the Robotic Soccer domain. By far, the Robocup 2D Simulation League is the domain where most work has been done. To mention a few, Lin, Liu and Chen [12] presented a hybrid architecture for soccer players where the deliberative layer corresponds to a CBR system, Ahmadi *et al.* [1] presented a two-layered CBR system for prediction for the coach and Berger and Lämmel [3] proposed the use of a CBR system to decide whether a pass should be performed. A more extensive review of the use of CBR in Robotic Soccer can be found in the work by Ros *et al.* [14].

Table 4.1: The CB-HAQL algorithm.

Initialize $\hat{Q}_t(s,a)$ and $H_t(s,a)$ arbitrarily.
Repeat (for each episode):
  Initialize s.
  Repeat (for each step):
    Compute similarity and cost.
    If there is a case that can be reused:
      Retrieve and Adapt if necessary.
      Compute $H_t(s,a)$ using Equation 4.3 with the
        actions suggested by the case selected.
    Select an action a using Equation 4.1.
    Execute the action a, observe $r(s,a)$, s'.
    Update the values of $Q(s,a)$ in the traditional way.
    $s \leftarrow s'$.
  Until s is terminal.
Until some stopping criterion is reached.

4.4 Combining Case-Based Reasoning and Multiagent Reinforcement Learning

In order to provide HARL algorithms with the capability of reusing previous knowledge from a domain, we propose two new algorithms, the Case-Based HAQL, that extends the HAQL algorithm, being capable of retrieving a case stored in a base, adapting it to the current situation, and building a heuristic function that corresponds to the case, and the Case-Based HAMMQ, that extends the HAMMQ in the same way.

As the problem description P corresponds to one defined state of the set of states $\mathscr{S}$ in an MDP, an algorithm that uses the RL loop can be implemented. Inside this loop, before action selection, we added steps to compute the similarity of the cases in the base with the current state and the cost of adaptation of these cases. A case is retrieved if the similarity is above a certain threshold, and the adaptation cost is low. After a case is retrieved, a heuristic is computed using Equation 4.3 and the actions suggested by the case selected. The complete CB-HAQL algorithm is presented in Table 4.1. The CB-HAMMQ algorithm is essentially the same one, using $\hat{Q}_t(s,a,o)$ and $H_t(s,a,o)$ instead.

Although this is the first work that combines CBR with RL using an explicit heuristic function, this is not the first work on combining the both fields. Drummond [9] was probably the first to use CBR to speed

up RL, proposing to accelerate RL by transferring parts of previously learned solutions to a new problem. Sharma *et al.* [15] made use of CBR as a function approximator for RL, and RL as a revision algorithm for CBR in a hybrid architecture system; Juell and Paulson [10] exploited the use of RL to learn similarity metrics in response to feedback from the environment and Auslander *et al.* [2] used CBR to adapt quickly an RL agent to changing conditions of the environment by the use of previously stored policies.

Our approach differs from all previous works combining CBR and MRL because of the heuristic use of the retrieved case. Bianchi, Ribeiro and Costa [4] proved that if the heuristic used is an admissible one, there will be a speed up in convergence time, if not, the use of the heuristic will not impede the RL method to converge to the optimal policy. As we use the case base as a heuristic, if the case base corresponds to an admissible heuristic there will be a speed up in the convergence time. But if the case base does not contain any useful case – or even if it contains cases that implement wrong solutions to the problem, the agent will learn the optimal solution anyway, by using the RL component of the algorithm [4].

4.5 Experiments in the Robotic Soccer Domain

Soccer competitions, such as RoboCup, have been proven to be an important challenge domain for research, and one where RL techniques have been widely used. The application domain of this paper is a simulator for the robot soccer domain that extends the one proposed by Littman [13], called "Expanded Littman's Soccer". Nevertheless, the technique proposed in this work is domain independent.

In this domain two teams, A and B, of three players each compete in a 10 by 15 grid presented in figure 4.1. Each team is composed by the goalie (g), the defender (d) and the attacker (a). Each cell can be occupied by only one player. The actions that are allowed are: keep the agent still, move – north, south, east and west – or pass the ball to another agent. The action "pass the ball" from agent a_i to a_j is successful if there is no opponent in between them. If there is an opponent, it will catch the ball and the action will fail. Actions are taken in turns: all actions from one team's agents are executed at the same instant, and

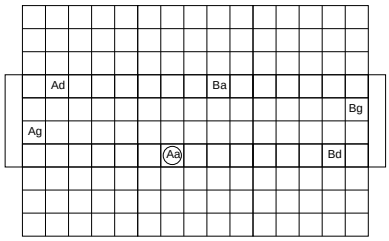

Fig. 4.1: The “Expanded Littman’s Soccer” environment.

then the opponents’ actions are executed. The ball is always with one of the players. When a player executes an action that would finish in a cell occupied by the opponent, it loses the ball and stays in the same cell. If an action taken by one agent leads it out the board, the agent stands still. When a player with the ball gets into the opponent’s goal, the trial ends and its team scores one point. The starting positions of all players are random, and the ball is given to one of the agents in a random fashion at the beginning of a trial.

To solve this problem, six algorithms were used: two traditional RL algorithms, Q–Learning and Minimax–Q; two Heuristically Accelerated algorithms, HAQL and HAMMQ; and the CB-HAQL and CB-HAMMQ algorithms, proposed in section 4.4.

The heuristic used in the HAQL and the HAMMQ algorithms was defined using a simple rule: if holding the ball, go to the opponents’ goal, not taking into account the teammates’ and opponents’ positions, leaving tasks such as learning to pass the ball or to divert the opponent to the learning process.

The heuristic value used in the CB-HA algorithms is computed during the games, as described in section 4.4. The case base used contains a set of basic cases that can be used without adaptation costs. The case base used in this experiment is composed of 5 basic cases, which cover the most significant situations that are observed during a game in the expanded Littman’s Soccer environment. These cases can be described as:

- If the agent is with the ball and there is no opponent blocking it, then move to the goal.
- If the agent is with the ball and there is an opponent blocking it, then move up.

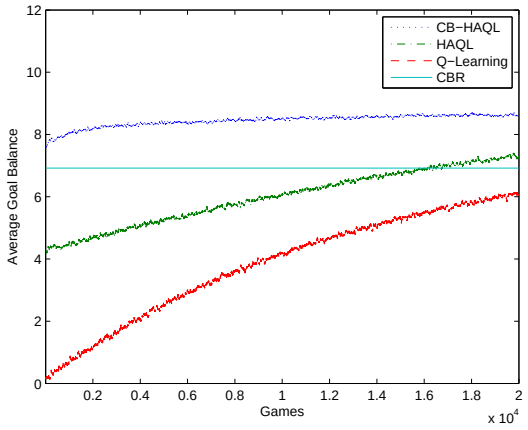

Fig. 4.2: Goals balance for the CBR, Q–learning, the HAQL and the CB-HAQL algorithms against a random opponent for the Expanded Littman's Robotic Soccer.

- If the agent is with the ball and there is an opponent blocking it, then move down.
- If the agent is with the ball and a teammate is closer to the goal, then pass the ball to the other agent.
- If the ball is with an opponent and the agent is close to the opponent, then stay in front of the opponent.

Is important to notice that this case base does not correspond to the optimal solution of the problem.

The reward the agents receive are the same for all algorithms: the agent that is holding the ball receives $+100$ every time it reaches the goal.

Thirty training sessions were run for the six algorithms, with each session consisting of 20,000 games of 10 trials.

Figure 4.2 shows the learning curves for the Single agent (Q-Learning, HAQL and CB-HAQL) algorithms when the learning team plays against an opponent moving randomly, and presents the average goal balance, which is the difference between goals scored and goals received by the learning team in each match. It is possible to verify that at the beginning of the learning phase Q-Learning has worse performance than HAQL,

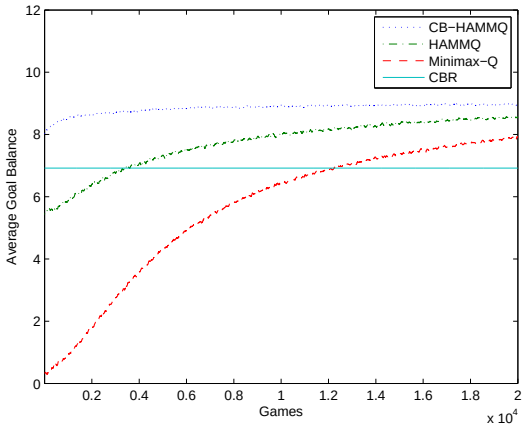

Fig. 4.3: Goals balance for the CBR, Minimax–Q, the HAMMQ and the CB-HAMMQ algorithms against a random opponent for the Expanded Littman's Robotic Soccer.

and that this has a worse performance than CB-HAQL. As the matches proceed, the performance of the three algorithms become similar, as expected.

Figure 4.3 shows the learning curves for the Multiagent (Minimax-Q, HAMMQ and CB-HAMMQ) algorithms when the learning team plays against an opponent moving randomly. It is possible to verify that, similarly to the single agent case, HAMMQ performs better than Minimax–Q, and that CB-HAMMQ is the best one. As it can be seen in this figure, the Minimax–Q is still learning after 20,000 games: as it is slower than the other two algorithms, it will only reach the optimal solution after 100,000 games.

In both figures the performance of a team of agents using only the case base (CBR) can also be observed: a line with values close to 7. As the case base does not contain the optimal solution to the problem, the agents have a performance that is worse than the one presented by the other teams at the end of the learning process. It is also worth noticing that for similar algorithms, the multiagent implementation have better results than the single agent one (Minimax–Q is better than Q–Learning, HAMMQ is better than HAQL and CB-HAMMQ is better than CB-HAQL). Finally, Table 4.2 shows the average number of goals scored at

the end of 20,000 games while playing against a random opponent, for all algorithms, and Table 4.3 shows and the average number of games won. It can be seen that the agents that are using CBR did not lose a single game.

Table 4.2: Goals made against Random opponent.

Algorithm	Goals made × Goals conceded
Q–Learning	$(133768 \pm 306) \times (57473 \pm 276)$
HAQL	$(158469 \pm 265) \times (38971 \pm 257)$
CB-HAQL	$(184279 \pm 448) \times (15417 \pm 436)$
Minimax–Q	$(140207 \pm 174) \times (38498 \pm 164)$
HAMMQ	$(166208 \pm 150) \times (22065 \pm 153)$
CB-HAMMQ	$(188168 \pm 155) \times (11292 \pm 140)$

Table 4.3: Game results against Random opponent.

Algorithm	Games won × Games lost
Q–Learning	$(16550 \pm 60) \times (1955 \pm 47)$
HAQL	$(19254 \pm 29) \times (227 \pm 15)$
CB-HAQL	$(19987 \pm 3) \times (0 \pm 0)$
Minimax–Q	$(18297 \pm 33) \times (1037 \pm 28)$
HAMMQ	$(19469 \pm 9) \times (27 \pm 4)$
CB-HAMMQ	$(19997 \pm 1) \times (0 \pm 0)$

The parameters used in the experiments were the same for all the algorithms. The learning rate is $\alpha = 0,9$, the exploration/ exploitation rate was defined as being equal to 0.2 and the discount factor $\gamma = 0.9$ (these parameters are similar to those used by Littman [13]). The value of η was set to 1. Values in the Q table were randomly initialized, with $0 \leq Q(s_t, a_t, o_t) \leq 1$.

4.6 Conclusion

This work presented a new approach to combine Case Based Reasoning, Reinforcement Learning and the use of Heuristics in Reinforcement Learning, which was the result of the work with Professor Lopez de Mantaras that started in 2007. We have proposed and evaluated two new algorithms, called Case-Based Heuristically Accelerated Q–Learning and Minimax–Q (CB-HAQL and CB-HAMMQ), which allow the use of a case base to define heuristics to speed up Single and Multiagent Reinforcement Learning algorithms.

The experimental results obtained using a new domain proposed for the Robotic Soccer games showed that CB-HAMMQ attained better results than HAMMQ and Minimax–Q alone. For example, after playing 1000 learning trials against a random opponent (Figure 4.3), the Minimax-Q, still could not produce policies that scored many goals on the opponent, while the HAMMQ was able to score some goals but less than the CBR alone and the CB-HAMMQ. Another interesting finding is that the number of goals scored by the CB-HAMMQ after 1000 trials was even higher than the number of goals scored by the CBR approach alone, indicating that the combination of the Reinforcement Learning and the case base out-performs the use of the case base on its own.

The algorithms presented here are only the initial results of the collaboration and friendship between me and Professor Lopez de Mantaras. As our work together continued, some articles about using CB-HARL in Transfer Learning problems, a natural extension of this work, have already been published [6].

Finally, we are convinced that heuristic functions will allow RL algorithms to solve problems where the convergence time is critical, as in many real time applications. Future works includes incorporating CBR in other well known RL algorithms, like SARSA, Minimax-SARSA, Minimax–Q(λ) and expanding this framework to deal with General Sum Markov Games [13] using algorithms such as Nash-Q and Friend-or-Foe Q-Learning.

Acknowledgments

Reinaldo Bianchi acknowledge the support of the CNPq (Grant No. 201591/2007-3). This work has been partially funded by the AGAUR 2009-SGR-1434 grant of the Generalitat de Catalunya and the NEXT-CBR MICINN TIN2009-13692 project.

References

1. M. Ahmadi, A. K. Lamjiri, M. M. Nevisi, J. Habibi, and K. Badie. Using a two-layered case-based reasoning for prediction in soccer coach. In H. R. Arabnia and E. B. Kozerenko, editors, *In Proc. of the Intern. Conf. of Machine Learning; Models, Technologies and Applications*, pages 181–185. CSREA Press, 2003.
2. B. Auslander, S. Lee-Urban, C. Hogg, and H. Muñoz-Avila. Recognizing the enemy: Combining reinforcement learning with strategy selection using case-based reasoning. In K.-D. Althoff, R. Bergmann, M. Minor, and A. Hanft, editors, *ECCBR*, volume 5239 of *Lecture Notes in Computer Science*, pages 59–73. Springer, 2008.
3. R. Berger and G. Lämmel. Exploiting past experience – case-based decision support for soccer agents. In *KI 2007: Advances in Artificial Intelligence - 30th Annual German Conference on AI*, volume 4667, pages 440–443. Springer, 2007.
4. R. A. C. Bianchi, C. H. C. Ribeiro, and A. H. R. Costa. Heuristic selection of actions in multiagent reinforcement learning. In M. M. Veloso, editor, *IJCAI*, pages 690–695, 2007.
5. R. A. C. Bianchi, C. H. C. Ribeiro, and A. H. R. Costa. Accelerating autonomous learning by using heuristic selection of actions. *Journal of Heuristics*, 14(2):135–168, 2008.
6. L. A. Celiberto, J. P. Matsuura, R. L. de Mantaras, and R. A. C. Bianchi. Using cases as heuristics in reinforcement learning: A transfer learning application. In *IJCAI'11*, pages 1211–1217. IJCAI/AAAI, 2011.
7. L. A. Celiberto, C. H. C. Ribeiro, A. H. R. Costa, and R. A. C. Bianchi. Heuristic reinforcement learning applied to robocup simulation agents. In U. Visser, F. Ribeiro, T. Ohashi, and F. Dellaert, editors, *RoboCup*, volume 5001 of *Lecture Notes in Computer Science*, pages 220–227. Springer, 2007.
8. R. L. de Mántaras, D. McSherry, D. Bridge, D. Leake, B. Smyth, S. Craw, B. Faltings, M. L. Maher, M. T. Cox, K. Forbus, M. Keane, A. Aamodt, and I. Watson. Retrieval, reuse, revision and retention in case-based reasoning. *Knowl. Eng. Rev.*, 20(3):215–240, 2005.
9. C. Drummond. Accelerating reinforcement learning by composing solutions of automatically identified subtasks. *Journal of Artificial Intelligence Research*, 16:59–104, 2002.
10. P. Juell and P. Paulson. Using reinforcement learning for similarity assessment in case-based systems. *IEEE Intelligent Systems*, 18(4):60–67, 2003.

11. L. P. Kaelbling, M. L. Littman, and A. W. Moore. Reinforcement learning: A survey. *Journal of Artificial Intelligence Research*, 4:237–285, 1996.
12. Y. Lin, A. Liu, and K. Chen. A hybrid architecture of case-based reasoning and fuzzy behavioral control applied to robot soccer. In *Workshop on Artificial Intelligence, International Computer Symposium (ICS2002)*, Hualien, Taiwan, 2002. National Dong Hwa University, National Dong Hwa University.
13. M. L. Littman. Markov games as a framework for multi-agent reinforcement learning. In *Proceedings of the 11th International Conference on Machine Learning (ICML'94)*, pages 157–163, 1994.
14. R. Ros, J. L. Arcos, R. L. de Mántaras, and M. Veloso. A case-based approach for coordinated action selection in robot soccer. *Artificial Intelligence*, 173(9-10):1014–1039, 2009.
15. M. Sharma, M. Holmes, J. C. Santamaría, A. Irani, C. L. I. Jr., and A. Ram. Transfer learning in real-time strategy games using hybrid cbr/rl. In M. M. Veloso, editor, *IJCAI*, pages 1041–1046, 2007.