On Measuring Social Intelligence: Experiments on Competition and Cooperation

Javier Insa-Cabrera, José-Luis Benacloch-Ayuso, and José Hernández-Orallo

DSIC, Universitat Politècnica de València, Spain
{jinsa,jorallo}@dsic.upv.es

Abstract. Evaluating agent intelligence is a fundamental issue for the understanding, construction and improvement of autonomous agents. New intelligence tests have been recently developed based on an assessment of task complexity using algorithmic information theory. Some early experimental results have shown that these intelligence tests may be able to distinguish between agents of the same kind, but they do not place very different agents, e.g., humans and machines, on a correct scale. It has been suggested that a possible explanation is that these tests do not measure social intelligence. One formal approach to incorporate social environments in an intelligence test is the recent notion of Darwin-Wallace distribution. Inspired by this distribution we present several new test settings considering competition and cooperation, where we evaluate the "social intelligence" of several reinforcement learning algorithms. The results show that evaluating social intelligence raises many issues that need to be addressed in order to devise tests of social intelligence.

1 Introduction

Social intelligence has been defined in many ways in psychology and cognition, but it can be just worded, with the terminology of agents, as the ability to perform well in the context of other agents. One problem of this definition is that we have to be more precise about what the 'other agents' are. If we evaluate humans and the other agents are worms or sea sponges, then our intuitive notion of social intelligence does not work well, because working well in the context of other agents with low intelligence is not necessarily related to social intelligence as we know it. In psychometrics and human cognition, social intelligence clearly sets these other agents as other humans. But what about artificial agents? If we use a society of dull agents, the useful abilities might be very different to those which are required if we introduce an agent into, e.g., a society of humans.

The difference between social intelligence and general intelligence is that in the latter an agent could perform well if it were able to solve non-social tasks, such as escaping from a maze, solving a puzzle or predicting the next number in a series. On the contrary, social intelligence implies that tasks involve competing and collaborating with other agents.

Dating back from the late nineties, we can find several works [1,2,9,4] addressing the problem of measuring agent intelligence in a principled and general way. Using notions taken from (algorithmic) information theory, MML and two-part

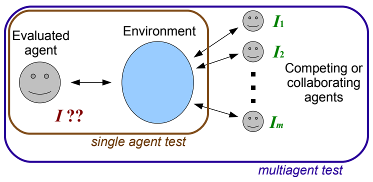

Fig. 1. A multiagent intelligence test compared to a single agent intelligence test. In a multiagent (social) intelligence test, other agents also interact (and become integral part) of the environment. In order to assess the intelligence of the evaluated agent, we need to know the intelligence of the other agents.

compression, Kolmogorov complexity and Solomonoff priors (see [10] for proper definitions of all these notions), some of these works present definitions and tests to evaluate agent intelligence. One important feature in some of these tests is that the difficulty of a problem, task or environment can be derived from its Kolmogorov complexity. This allows for the application of the setting to many different fields in artificial intelligence, including inductive or deductive tasks.

A *universal* test as introduced in [4] is a test which aims at evaluating any kind of subject including, e.g., humans and reinforcement learning agents. Some preliminary results of this evaluation [7] show that the setting is able to compare and evaluate different kinds of agents, but it fails at placing them on the same scale, since humans usually get similar scores to those of other relatively simple agents. One possible explanation for these results is that it is virtually impossible to find other agents in the test, so social intelligence is not measured. The question, therefore, is what and how agents should be introduced in the test. This is related to the Turing test and the question of evaluating intelligence with games (also suggested in [4]), where the difficulty is not only given by the complexity of the game, but from the opponent's intelligence. This leads to a circular problem: we need to know the opponent's intelligence first in order to know the difficulty of the problem. Fig. 1 shows this situation.

One recent proposal to overcome this problem is turning this circularity into a recursion. The Darwin-Wallace distribution [5] establishes a distribution of agents based on an evolutionary process. The first 'generation' just uses a Solomonoff prior over agents, with very simple agents predominating. These agents are set to interact in a random environment. The second generation is constructed by selecting the agents according to their performance. The result of this evolutionary process is a distribution of mind forms, i.e., a distribution of agents. The higher the generation i is, the more socially intelligent their agents should be or, in other words, the more demanding the 'society' will be, in the sense that competing and collaborating with other socially intelligent agents requires social intelligence. Note that this does not mean that minds have to evolve as in a true evolutionary process, or as in evolutionary game theory [14].

The previous proposal has many implementation issues. Nonetheless, it gives some clues about how social intelligence can be measured and how the other agents can be chosen. In fact [5] suggests that intelligence tests could be used to make this choice of agents from off-the-shelf algorithms in AI.

In this paper, we perform some experiments on a general intelligence test setting in order to examine the way in which simple competitive and cooperative scenarios may have a big impact on the performance of some simple agents. This is crucial to determine how social environments affect the results obtained by different agents in order to get more information about how to approximate the Darwin-Wallace distribution. We will use very simple reinforcement learning (RL) algorithms: SARSA [11], Q-learning [13] and QV-learning [15]. The goal of the paper is not showing how these three algorithms behave nor comparing them. We just use them as off-the-shelf agents which can learn from an environment to see how performance is affected by the introduction of more agents in an environment. Rather, the true goal of the paper is to analyse the behaviour of intelligence tests when environments are populated with agents, and how this affects the results of the evaluated agent. We will examine several scenarios, some with competition and some with cooperation.

The paper is organised as follows. Section 2 reviews the notion of universal intelligence tests and how the Darwin-Wallace distribution can be useful to turn them into social intelligence tests. Section 3 makes the extension by modifying the environments and the reward system for competing and cooperating scenarios. The following sections 4, 5, 6 and 7 perform and discuss the experiments for the different scenarios. Finally, a more comprehensive discussion of results and implications is found in section 8.

2 Universal Tests and Social Intelligence

One approach for measuring intelligence is to take a diverse selection of tasks of different complexity and to measure agent performance over this selection. However, several issues arise here. For instance, the selection of tasks must be unbiased. One approach is to consider all possible (computable) tasks, as defined by a universal Turing machine. In order to link performance to any possible task we can use the notion of rewards. This leads to interactive scenarios, which can be well represented by (discrete) environments, very much like the typical observaton-action-reward environments in reinforcement learning (RL) [12]. Finally, we need to assess the complexity of each task in order to make a proper choice of tasks which capture a wide range of difficulty and, therefore, can suit the agent's level of intelligence. These issues have been addressed in [1,6,2,9,4].

In this context, [4] introduces the idea of universal test, a test which is conceived to be feasibly applicable to any kind of agent: humans, non-human animals, artificial agents, including hybrids and communities, of any degree of intelligence and speed. The test is based on a set of environments as in [9].

In [3], a hopefully unbiased environment class (called Λ) is introduced, which is composed of spaces and agents with universal descriptive (Turing-complete) power. Originally, only two agents (apart from the evaluated agent) were used in the environments. Since their behaviour is generated by a universal distribution using Markov algorithms (a Turing-complete rewriting language), their *sophistication* is really low. Hence it was very difficult to find any social behaviour originating from them, and, therefore, any social behaviour in the environments.

The first evaluations using these tests [7] show that they work well at evaluating very different agents (humans and RL algorithms), but they do not properly reflect their supposed difference in intelligence. Many possible explanations are suggested in [7], with incremental knowledge acquisition and social intelligence being two of the abilities which this test is not giving enough importance.

In order to address the second issue we must define environments which are more social. The question of which agents are introduced becomes crucial, since the results of the evaluated agent will depend on the abilities of the other agents. This is illustrated in Fig. 1. The question is what criteria we can use to introduce the other agents and how we can measure their (social) intelligence in advance. In [5], instead of incorporating other agents in an ad-hoc way, they look for a formal way to determine which agents must be introduced in a social environment. As [5] states: "intelligence is the result of evolution through millions of generations interacting with other live beings. Thus we define intelligence in this context, interacting with other agents of similar intelligence". From here they formalise the so-called Darwin-Wallace distribution for agents and environments.

Briefly and informally, the Darwin-Wallace distribution requires a multiagent environment which has its rewards, actions and observations as usual, but allows the 'introduction' of any number of agents, whose distribution evolves (by properly sampling agents according to their degree of intelligence) and can lead to higher degrees of (social) intelligence. From here, the Darwin-Wallace distribution is defined recursively according to a level or generation.

The use of this distribution has many issues. First, it is a theoretical construct which might be useful for understanding the kind of environments where (social) intelligence is needed. Second, this distribution could be used for the construction of social intelligence tests, just sampling from the distribution. Third, and recursively, the way in which this distribution can be approximated is precisely by the use of intelligence tests, where human-made agents can be inserted into the environments, provided we have been able to assess their intelligence first.

Following this last issue, we need to develop intelligence tests in multiagent scenarios. In particular, we need to adapt the existing intelligence test proposals to a multiagent setting. This is what we do below.

3 Intelligence Tests Considering Several Agents

The first intelligence tests based on the theory developed in [4] were based on the environment class Λ, introduced in [3]. This environment class considers a space which is composed of a directed labelled graph, where vertices are cells and arrows are actions. The graph is selected to be strongly connected (all cells are reachable from any other cell). Cells can contain agents. Every environment must include at least three agents: the evaluated agent, and two special agents *Good* and *Evil*. *Good* and *Evil* are not generally reactive, so, if no further agent is included, the environment cannot be considered a proper multiagent system. Actions allow the evaluated agent (and other agents) to move in the space. Observations show the cell contents. Rewards are rational numbers in the interval $[-1, 1]$ and are generated by the agents *Good* and *Evil*, which leave rewards in the cells they visit. Rewards do not stay unaltered in the cell forever. If a reward in a cell is eaten by any agent (including *Good* and *Evil*) because the agent steps into or stays in the cell, the reward disappears. While rewards are not eaten, their value is divided by 2 for each iteration. This has the effect of seeing *Good* and *Evil* as agents which leave a reward wake as they move. *Good* and *Evil* have the same behaviour (they follow the same pattern) except for the sign of the reward (+ for *Good*, − for *Evil*). This makes *Good* and *Evil* symmetric, which ensures that the environment is balanced (random agents score 0 on average) [4]. For more details of the environment class Λ, see [3].

Environments are composed of a space of cells (a graph of nodes) and the patterns for Good and Evil (a simplified adaptation of [7]). Once an environment has been constructed, evaluation is performed in the following way. Initially, each agent is randomly (using a uniform distribution) placed in a cell. Then, we let *Good*, *Evil* and the evaluated agent interact for a certain number of steps, i.e. a session. The final score is the average of rewards in the session.

This configuration can be easily extended to a multiagent setting, by including more agents in the environment. Agents can move freely to other cells independently of whether they are occupied or not by other agents. In other words, agents can share a cell. A competitive, individualistic scenario is set by each agent trying to improve its own rewards. If two or more agents share a cell, the reward is just divided by the number of agents in the cell. *Good* and *Evil* cannot share a cell with other agents. This re-introduces some degree of reactivity (with respect to the prototype in [7]), even in the single agent case.

There are many possible ways of introducing cooperation and competition, which may lead to different experimental results, some of them similar to what has been previously studied in the AI literature. In this paper we do not want to evaluate these choices, but to analyse *how the degree of intelligence of the agents in a social environment affects the role of cooperation and competition*. The ultimate goal is to shed some light on whether environments become difficult when many agents are introduced (independently of their intelligence) or become difficult (and socially challenging) when other intelligent agents are introduced. These findings are necessary if we aim at measuring social intelligence.

The first question when several agents are introduced in the space is how the rewards are shared among them. A second, relatively more difficult, question is how we can deal with cooperation. The easiest way of making this setting purely cooperative is by just putting all the rewards in the same bag. With this, one should not be concerned about not getting some reward itself if some other agent is able to get it instead. What matters is the overall result. We can of course move between competition and cooperation by using the notion of team. All the members of a team put their rewards in the same bag, and each team should compete against the others as usual in games and economics.

Now we are ready to see what happens with a single agent intelligence test when we turn it into a multiagent test. But, before that, we need to determine the agents that we will use for the experiment. The agents are:

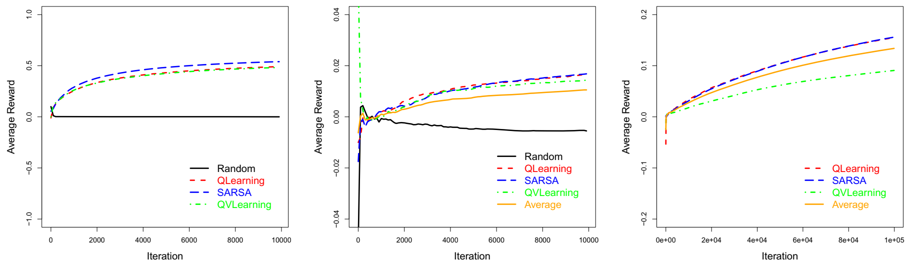

Fig. 2. Left: Isolated scenario. The 4 evaluated agents are evaluated separately. Middle: Competition scenario. The 3 RL agents along with the random agent. Right: Competition scenario. The 3 RL agents without the random agent. 100 environments each.

- Random: an agent which chooses randomly among the available actions using a uniform distribution.
- Q-learning [13]: the most common reinforcement learning algorithm. We use the description of cell contents as a state.
- SARSA [11]: a well-known variant of Q-learning which also takes the future action into account.
- QV-learning [15] (without eligibility trace): a variant of Q-learning which partially resembles ActorCritic methods.

The three latter algorithms will be referred to as RL agents. In order to have a consistent view of the experiments, the parameters for all the RL agents algorithms (*learning rate* α, *discount factor* γ, etc.) were fine-tuned on the single agent scenarios, by using 1,000 sessions for each parameter variation, totalling a huge number of experiments to set the optimal parameters.

4 Evaluating Agents in Isolation

We start our experiments with the scenario where agents are just taken and evaluated isolatedly. This is the same setting as in [7], with the only (minor) difference that *Good* and *Evil* are slightly reactive because they try to avoid sharing a cell with other agents. In addition, we will just restrict the evaluation to environments with nine cells.

The result of Fig. 2 (left) is clear (and consistent with the results in [7]). The random agent has an average reward of 0, as predicted by the theory. The three RL agents are very slow learners and only get closer to 0.5 after 10,000 iterations. Their behaviour is similar and the differences are small.

5 Competitive Scenario

More interesting things can be observed when we switch to the competitive scenario. Here all the agents are located in the environment at the same time competing for rewards. As we can see in Fig. 2 (middle) the random agent gets

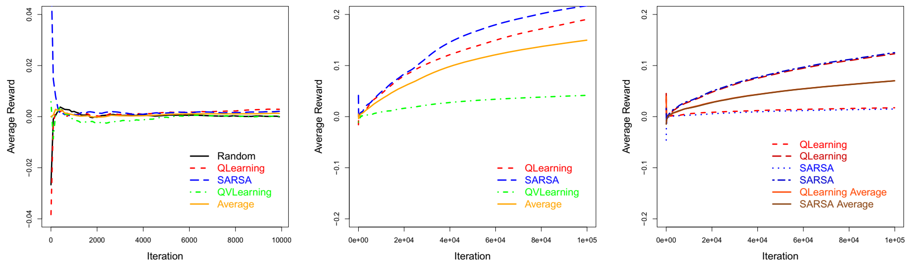

Fig. 3. Left: Cooperative scenario. The 3 RL agents along with the random agent. Middle: Cooperative scenario. The 3 RL agents without the random agent. Right: Two teams scenario. One team with two Q-learning agents against another team with two SARSA agents. 100 environments each.

a value which is even lower than 0, since most of the positive rewards are eaten by the other agents, leaving the negative rewards for the random agent. RL algorithms have a very poor result (not reaching 0.02 in 10,000 iterations). This is explained by the presence of the random agent, which makes the state tables of the RL algorithms grow considerably.

Finally, in order to further confirm that the problem is the state space, we remove the random agent (which can be considered noise), and we only leave the RL agents. We also increase the number of iterations to 100,000. This is shown in Fig. 2 (right). Things improve slightly and, in the very long term, Q-learning and SARSA get close to 0.2, while QV-learning lags behind around 0.1. We see that just the presence of only two other agents makes their matrices so big that they require more than 100,000 iterations to derive their Q-values accurately.

Apart from the comparative results, we see that performance depends on the other agents' policies, but especially on the ability of digesting the state space, and how much noise (e.g., from the random agent) can be handled.

6 Cooperative Scenario

The next scenario we want to explore is when the four agents are prompted to cooperate. This is done by putting all the rewards in the same bag, so the agents just see the reward as the average reward of all the agents.

Fig. 3 (left) changes from Fig. 2 (middle) very significantly. How can it be that moving form a competitive to a cooperative case, we get worse average results? The explanation is a little bit more convoluted. The problem of cooperation is the way we assign rewards. Since the reward they receive is the average of the rewards of all the agents, it is much more difficult for them to determine the goodness of the actions, since rewards are affected by other agents' movements. In other words, they lose 'individuality'.

This explanation is only part of the story if we compare Fig. 3 (middle) with Fig. 3 (left) and Fig. 2 (right). In this case, where the random agent has been removed, the results are slightly better than in the competitive case. However, this improvement is not uniform for the three RL agents.

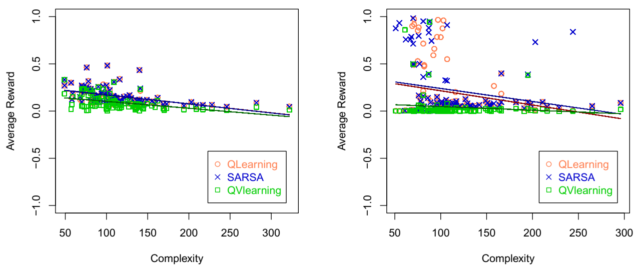

Fig. 4. Relation between environment complexity and results of the three RL agents for the competitive case (left) and the cooperative case (right). Linear regression is also shown for each agent.

7 Scenario Measuring Both Competition and Cooperation

Finally, we examine another scenario where we now have competition and cooperation at the same time, using the notion of 'team'. We define two teams, one with two Q-learning agents and the other one with two SARSA agents. Inside each team the rewards go to the same bag, but different teams compete for the rewards. This is shown in Fig. 3 (right). In general, the results are poorer than with three agents in the cooperative case (Fig. 3, middle). This can be explained because here we have four agents instead of three, but also because having two teams is a more complex scenario than having just one.

The results show that there are no significant differences between both teams. However, there are important differences between the components of each team. This can be observed in Fig. 3 (right), where we assign the best results in the team to the first entry and the worst results to the second entry. So, the plot just shows the difference in (average) performance between the best and the worst component in the team. We see that this difference is very significant. While usually an agent in the team performs around 0.1, the other agent stays at a very low result close to 0. It is not clear which role this second agent takes.

8 Discussion

In the previous sections, we have analysed several scenarios. A test which was originally designed to measure the intelligence of an agent against an environment without other agents is adapted to other scenarios where other agents are introduced in the environments.We see that performance can be seriously degraded by the inclusion of other agents with null intelligence, as a random agent. This is surprising if we look at this from the point of view of game theory (two-player games, in particular), but it is much more natural if we realise that it is more difficult to attain a goal if there is another agent bugging around (even randomly). This is extreme in the case of RL agents, because they cannot *learn* that random agents are just noise, and stick to the original huge state space.

All this means that the difficulty of a task is no longer related to the complexity of the environment in a tight way, as it was for the single agent situation. We can see this by comparing the complexity of the environment (excluding the evaluated agents) and the results for the scenarios where only the three RL agents are used, i.e. Fig. 2 (right) and Fig. 3 (middle). In order to approximate the environment complexity, we use the size of a compressed coding of the concatenation of the space description S and the description of the pattern for *Good* and *Evil*, denoted by P. More formally, we calculate an approximation to its (Kolmogorov) complexity as $K^{approx} = LZ(S, P)$ where LZ is just the 'gzip' method given by the *memCompress* function in R, a GNU project implementation of Lempel-Ziv coding. This comparison is shown in Fig. 4. We see that there is still a relation between the complexity of the environment and the result, while this relation is stronger for Qlearning and SARSA in the cooperative case. In fact, the results for Qlearning and SARSA are very good when the complexity is very low. This means that in very simple cases RL agents are able to perform well, even in social scenarios. This seems to suggest that the difficulty of a social environment is a cumulative issue, which adds the complexity of the environment and the complexity/performance/noise of other agents.

Some lessons can be learnt from these results in the context of the Darwin-Wallace distribution. One of the purported problems of this distribution is that many iterations might be needed to reach a level where some social behaviour can be evaluated. We see that this may not be the case. For instance, the mere introduction of very simple agents in an environment makes that the performance of other agents plunge. This suggests that the evaluation of social intelligence could possibly be performed against other agents of inferior degree of intelligence.

In fact, it would be extremely informative to repeat the experiment performed with humans and RL agents in [7] by using one of these simple multiagent environments. We guess that humans will still be able to manage, mostly because they handle noise much better. Naturally, many other experiments must follow. For instance, for the RL agents, we only consider model-free techniques whose search space grows geometrically as more agents are there. It would be interesting to see the results for model-based algorithms using function approximations, as well as other RL algorithms which work better when the Markov property does not hold (which is the general case in multiagent systems). Also, some other RL algorithms which are specialised for multiagent settings, such as Frequently Adjusted Q-learning [8] might give different results. Other issues which could be reconsidered is the way we modify the reward system to make the test competitive or cooperative.

Summing up, from the notion of Darwin-Wallace distribution, we have pushed forward the idea of 'multiagent intelligence test', which is an intelligence test where there are other agents in the environments. This is a new notion, since the kind of intelligence tests we are used to are typically those where the evaluated agent has to solve some tasks or where it has to be interrogated by other agents (interviews, Turing test, etc.), but the other agents are not *inside* the test. The closest notion is an old companion of artificial intelligence, *games*, especially multiplayer games, but it has only been recently proposed as a testbed for measuring intelligence [4]. However, the role of the opponent and its intelligence has not been clarified to date, especially if we want a test to give an absolute result, not only comparing a pair of agents.

Acknowledgements. This work was supported by the MEC/MINECO projects CONSOLIDER-INGENIO CSD2007-00022, EXPLORA-INGENIO TIN 2009-06078-E, TIN 2010-21062-C02-02 and GVA project PROMETEO/2008/051. Javier Insa-Cabrera was sponsored by Spanish MEC-FPU grant AP2010-4389.

References

1. Dowe, D.L., Hajek, A.R.: A computational extension to the Turing Test. In: Proceedings of the 4th Conference of the Australasian Cognitive Science Society, University of Newcastle, NSW, Australia (September 1997)
2. Hernández-Orallo, J.: Beyond the Turing Test. J. Logic, Language & Information 9(4), 447–466 (2000)
3. Hernández-Orallo, J.: A (hopefully) non-biased universal environment class for measuring intelligence of biological and artificial systems. In: Hutter, M., et al. (eds.) 3rd Intl Conf. on Artificial General Intelligence, pp. 182–183. Atlantis Press (2010)
4. Hernández-Orallo, J., Dowe, D.L.: Measuring universal intelligence: Towards an anytime intelligence test. Artificial Intelligence 174(18), 1508–1539 (2010)
5. Hernández-Orallo, J., Dowe, D.L., España-Cubillo, S., Hernández-Lloreda, M.V., Insa-Cabrera, J.: On More Realistic Environment Distributions for Defining, Evaluating and Developing Intelligence. In: Schmidhuber, J., Thórisson, K.R., Looks, M. (eds.) AGI 2011. LNCS (LNAI), vol. 6830, pp. 82–91. Springer, Heidelberg (2011)
6. Hernández-Orallo, J., Minaya-Collado, N.: A formal definition of intelligence based on an intensional variant of Kolmogorov complexity. In: Proc. Intl Symposium of Engineering of Intelligent Systems, EIS 1998, La Laguna, Spain, pp. 146–163. ICSC Press (February 1998)
7. Insa-Cabrera, J., Dowe, D.L., España-Cubillo, S., Hernández-Lloreda, M.V., Hernández-Orallo, J.: Comparing Humans and AI Agents. In: Schmidhuber, J., Thórisson, K.R., Looks, M. (eds.) AGI 2011. LNCS (LNAI), vol. 6830, pp. 122–132. Springer, Heidelberg (2011)
8. Kaisers, M., Tuyls, K.: Frequency adjusted multi-agent Q-learning. In: Proc. 9th Intl Conf. on Autonomous Agents and Multiagent Systems, pp. 309–316 (2010)
9. Legg, S., Hutter, M.: Universal intelligence: A definition of machine intelligence. Minds and Machines 17(4), 391–444 (2007)
10. Li, M., Vitányi, P.: An introduction to Kolmogorov complexity and its applications, 3rd edn. Springer (2008)
11. Rummery, G., Niranjan, M.: On-line Q-learning using connectionist systems. Cambridge University Engineering Department, TR 166 (1994)
12. Sutton, R.S., Barto, A.G.: Reinforcement learning: An introduction. The MIT Press (1998)
13. Watkins, C., Dayan, P.: Q-learning. Machine Learning 8(3), 279–292 (1992)
14. Weibull, J.: Evolutionary game theory. The MIT Press (1997)
15. Wiering, M.: QV (λ)-learning: A new on-policy reinforcement learning algorithm. In: 7th European Ws. on Reinforcement Learning, pp. 17–18 (2005)