Making a Reinforcement Learning Agent Believe

Klaus Häming and Gabriele Peters

University of Hagen - Human-Computer-Interaction
Universitätsstr. 1, 58097 Hagen - Germany
{klaus.haeming,gabriele.peters}@fernuni-hagen.de

Abstract. We recently explored the benefits of a reinforcement learning agent which is supplemented by a symbolic learning level. This second level is represented in the symbolic form of Spohn's ranking functions. Given this context, we discuss in this paper the creation of symbolic rules from a Q-function. We explore several alternatives and show that the rule generation greatly influences the performance of the agent. We provide empirical evidence about which approach to favor. Additionally, the rules created by the considered application are shown to be plausible and understandable.

Keywords: ranking functions, belief revision, reinforcement learning, hybrid learning architecture.

1 Introduction

It is often desirable to be able to tell what an agent has been learned. To achieve this, one idea is to take a numerical learning scheme and extract rules afterwards, as has been done, e.g, for neural networks [7]. A second approach is to combine a symbolic representation with a numerical one. This has also been done for neural networks [11]. And finally one can imagine to completely omit the numerical representation and use a symbolical representation only. In the context of reinforcement learning [12], the latter approach can be found in the context of relational reinforcement learning [13].

Besides getting insight into the belief base of an agent, an additional benefit is that learning on a numerical level can profit from the belief represented on a symbolic level. This is comparable to the top-down and bottom-up learning capabilities of the human brain [2].

In particular, using a ranking function [10] as the symbolic representation to augment a reinforcement learning agent can lead to a vastly improved performance. We have shown this for small grid-worlds [4] as well as for large and noisy state spaces [5] such as those occurring in object recognition [3].

Ranking functions belong to the field of belief revision [1] and are a deterministic belief representation [9]. They were initially introduced by Wolfgang Spohn under the name of ordinal conditional functions [8]. Each ranking function expresses the belief of an agent in each particular instance of its world. In a reinforcement learning set-up, these instances are given by the variables describing the environment in which the agent learns.

A.E.P. Villa et al. (Eds.): ICANN 2012, Part I, LNCS 7552, pp. 435–443, 2012.

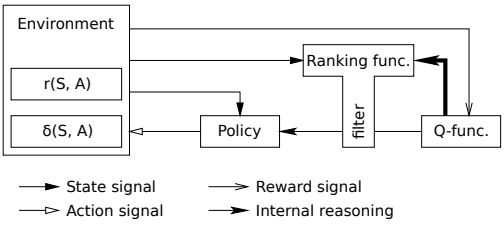

Fig. 1. The agent's architecture. It differs from a classic reinforcement learning agent in the presence of a ranking function which filters the actions. The ranking function itself is revised with information from the Q-function (**bold** arrow).

This paper clarifies how we use the numeric learning level of the reinforcement learning agent to generate a revision which alters the ranking function and hence the symbolic learning level. Given a number of variables which describe the environment, ranking functions map each possible realization of these variables to a non-negative number. A particular realization is called *world model* and the number it is mapped to is called *rank*. The rank represents the disbelief the agent currently has concerning the associated world model. Let κ be a ranking function. Altering κ and thereby creating a new ranking function is called revision. For instance, $\kappa' = \kappa * (A|S)$ creates a new ranking function κ' which expresses the belief that in state S action A is the most preferable. Hence, our main concern is the following: How to choose an action A in each state S such that a subsequent revision with $(A|S)$ is most beneficial for an agents learning progress?

This question is addressed by discussing four candidate algorithms that are assessed in a small application. Additionally, the state space of this application is chosen such that the symbolic rules we expect the agent to learn are obvious. This allows us to verify the agents success by simple inspection.

2 The Candidate Algorithms

Fig. 1 shows the general architecture of the agent. The bold arrow highlights the main concern of this paper. There are four candidate algorithms we evaluate. These are presented in Fig. 2 to 5, where we use the following notation:

Input	S: the last state
	A: the action taken
	r: the reward gained
	κ: a ranking function
	c_t: a threshold (Fig. 4 and 5 only)
Output	κ': a modified ranking function
Other	q_m: the Q-value of the best action
	a_m: the number of actions whose Q-value is q_m
	$\tilde{A}$: the single best action
	$\mathfrak{A}$: all possible actions available in S
	$\texttt{cnt}(S, A)$: count of A being best in S (Fig. 4 and 5 only)
	$\texttt{incCnt}(S, A)$: increments $\texttt{cnt}(S, A)$ (Fig. 4 and 5 only)

We now summarize the general idea behind each algorithm.

- RF1: This is the algorithm of Fig. 2 which revises κ with $(A|S)$ if A is the single best action in S.
- RF2: The algorithm of Fig. 3 is similar to RF1, but only revises κ if the best action is also the action taken, i.e. the agent did not chose an exploratory action.
- RF3: The algorithm of Fig. 4 revises κ with $(A|S)$ if A was the single best action for a pre-defined number of visits. This number is named c_t.
- RF4: Finally, the algorithm of Fig. 5 extends RF3. If A was the single best action most often (with absolute frequency c_m) and A' second most often (with absolute frequency c_n), then κ is revised with $(A|S)$ if $c_m - c_n$ exceeds the threshold c_t.

3 The Example Application

As mentioned before, the application follows the general architecture shown in Fig. 1. With this picture in mind, we first specify the environment. The agent has to learn its way from a starting location to a goal location. These locations are situated on a spherical grid with 128 approximately evenly spaced nodes. The radius of the sphere is 1. The observed states come from the domain:

$$\mathfrak{S} = \mathfrak{D} \times \mathfrak{C}, \text{ with } \mathfrak{D} = \{\texttt{far}, \texttt{middle}, \texttt{close}\}$$
$$\mathfrak{C} = \{\texttt{black}, \texttt{white}, \texttt{red}, \texttt{green}, \texttt{blue}, \texttt{yellow}, \dots\}$$

The number of colors is varied to test the agent in state spaces of different sizes. The data presented in this work was generated using state spaces where $|\mathfrak{C}| \in \{6, 25, 225\}$. While the $\mathfrak{C}$-part of the state description is generated randomly at each step, the $\mathfrak{D}$-part of the signal is chosen according to the distance d of the agent to the goal state. If $d^2 < 1.2$ then element `close` will be chosen. A value in the range of $1.2 \leq d^2 < 2.4$ activates `middle` and `far` stands for $2.4 \leq d^2$.

The set of actions is $\mathfrak{A} = \{\texttt{best}, \texttt{good}, \texttt{bad}, \texttt{worst}\}$. Each of these symbolic values in itself represents a set of "real" actions. This is clarified in Fig. 6. Essentially, `best` chooses a real action that minimizes the distance to the goal while `worst` maximizes it. Whenever the symbolic action chosen corresponds to a set of real actions with more than one element, a particular one is chosen randomly. In such a setting we essentially expect the agent to learn that regardless of the perceived color or distance, the action `best` is always preferable. Hence, the actual d-values used to select on the elements of $\mathfrak{D}$ are rather unimportant.

The reinforcement learning part of the agent uses Q-learning [12] and the update equation

$$Q_{t+1}(S, A) = Q_t(S, A) + \alpha[r(S, A) + \gamma \max_a(Q_t(\delta(S, A), A)) - Q_t(S, A)],$$

where the discount factor γ is set to 0.6, while α is initially 1 and then gradually reduced to 0.25. The reward $r(S, A)$ is always 0 with the exception of A leading

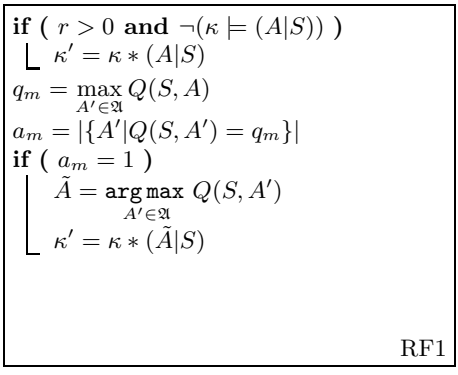

Fig. 2. Revision with $(A|S)$ if A is the single best action at the current state S.

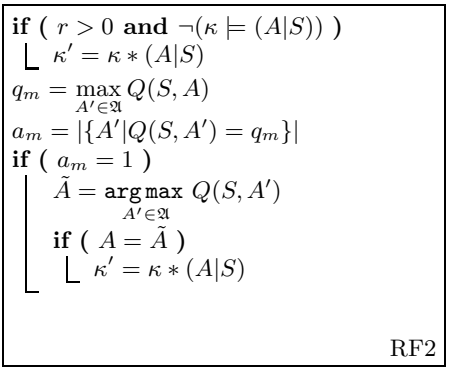

Fig. 3. Revision with $(A|S)$ if A is the single best action at the current state S and has also been chosen as the current action.

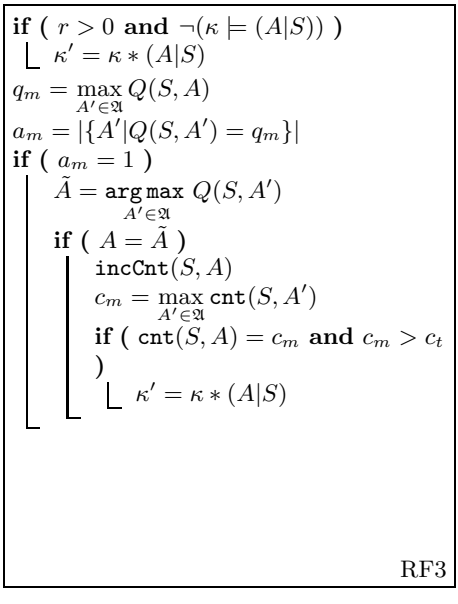

Fig. 4. Revision with $(A|S)$ if A has been the single best available action often enough.

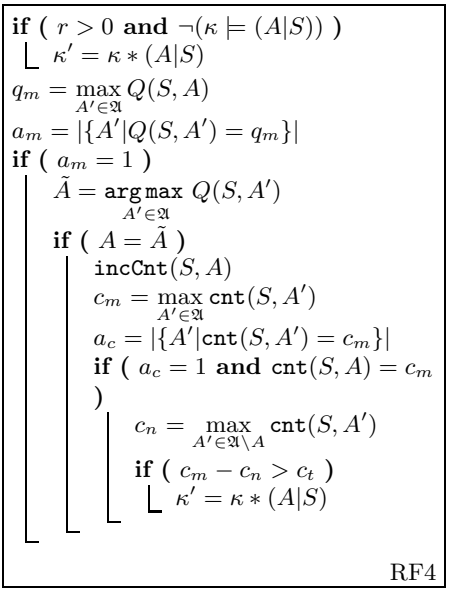

Fig. 5. Revision with $(A|S)$ if A has been the single best available action more often than the second most often action.

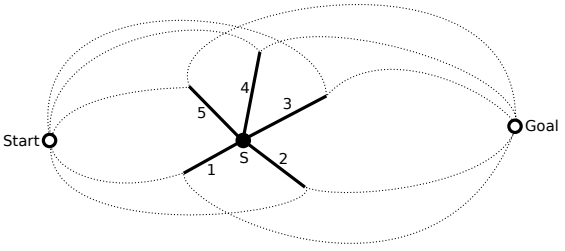

Fig. 6. A mapping from the symbolic actions to actual movements. Consider the state S on a grid from which several paths lead to the start and goal state. The actions available in S are $\mathfrak{A} = \{1, 2, 3, 4, 5\}$. The symbolic actions `best`, `good`, `bad`, and `worst` correspond to subsets of $\mathfrak{A}$. In this example these are `best` $= \{3\}$, `good` $= \{2, 4\}$, `bad` $= \{5\}$, and `worst` $= \{1\}$.

to the goal state in which case a reward of 100 is given. The agent prefers the shortest path towards the goal despite the fact that only the goal state transitions are rewarded. This is due to the discount factor γ which makes longer paths less attractive. The transition function $\delta(S, A)$ is implicitly given through the connections of the spherical grid. The actions are chosen by a modified ϵ-greedy policy [12] with $\epsilon = 0.1$. The modification is necessary to make the policy aware of the presence of the symbolic learning level. It first uses ϵ to decide whether or not to use the ranking function and then, again, to actually decide on the action. This way the presence of a ranking function does not cripple the agents ability to explore its environment once a revision has taken place.

4 Results

Fig. 7 shows the result for a state space of size 3×25, i.e. a configuration which uses 25 colors to confuse the agent. In this, an application of algorithm RF1 yields the fastest learning agent. Also, RF1 is the only algorithm that actually learns faster than a plain Q-learner, i.e. one without the second learning level. Algorithm RF2 ranked second. Worst are algorithms RF3 and RF4. For the results in Fig. 7, we set the threshold $c_t = 2$. We performed various runs with different values for c_t and the results were all quite similar. Therefore only one curve for each algorithm is included. In any case, their performance does not justify the added complexity. To show the aforementioned preference of the agent for short paths, we also include a plot of the episode lengths against the episode number in Fig. 7. An episode was forced to terminate if it took 200 steps.

The advantage of the RF1 algorithm over the plain Q-learner grows with the number of dimensions. This is shown in Fig. 8, where we enlarged the state space to include 225 colors. There, the maximum episode length was set to 300.

Since a ranking function allows us to query its learned rules, Fig. 9 shows all the rules learned by the RF1 agent in a state space which uses just 6 colors. This table was computed by keeping track of the rules the agent was revised with and

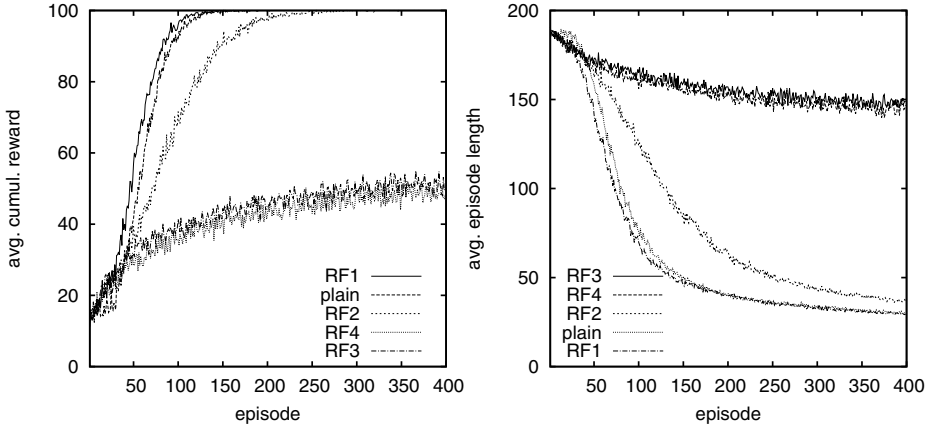

Fig. 7. Learning progress of the RF1 to RF4 agents and a plain Q-learner. The state space had a size of 3×25, an episode at most 200 steps. The left diagram shows the cumulated rewards, the right the episode length, both plotted against the episodes. The averages were computed from 500 runs. The labels are arranged to match the curves' order at episode 100.

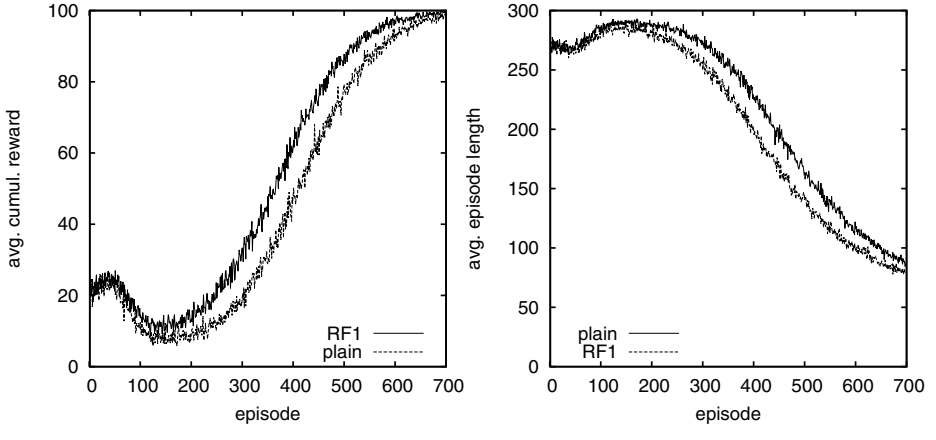

Fig. 8. Learning progress of the RF1 agent and a plain Q-learner. The state space had a size of 3×225, an episode at most 300 steps. The left diagram shows the cumulated rewards, the right the episode length, both plotted against the episodes. The averages were computed from 500 runs. The labels are arranged to match the curves' order at episode 300.

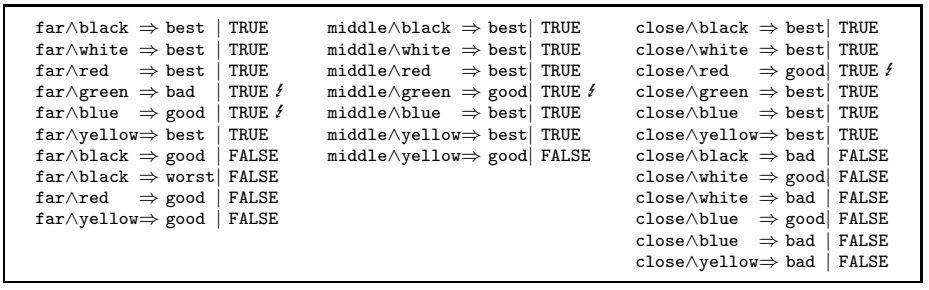

Fig. 9. Rules learned by an agent using the RF1 algorithm. We expect the agent to learn that it is always best to choose the action `best`. *All* rules with which its ranking function was revised at some point during learning are shown. The ones that were still believed after 100 episodes are marked `TRUE` , the ones discarded `FALSE` . The ↯ marks the four sub-optimal rules which happened to be still believed.

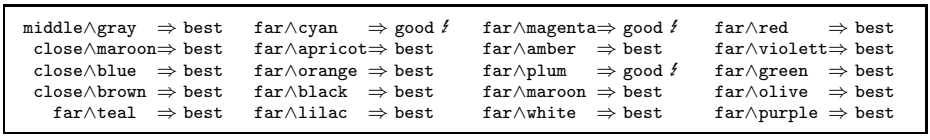

Fig. 10. Rules learned by an agent in a state space of size 3×25 after 400 episodes. We only show the 20 most strongly believed actions. Among these are three sub-optimal ones.

asking the ranking function after 100 episodes whether it still believes them or not. As one can see, the agent mostly believes that the `best` action should be chosen regardless of the $\mathfrak{C}$-part of the state signal.

Because the same table for an agent which had to cope with a state space of size 3×25 already contains more than 200 rules, we present as a second example the 20 most believed ones in Table 9. There, the agent was allowed to learn for 400 episodes and the maximum episode length was set to 200.

Despite the fact, that the learned rules are not perfect, one can say that in both experiments the agent has learned that the `best` action is preferable.

5 Conclusion

Summarizing, we have assessed four candidate algorithms on their impact on the learning progress of an agent with two learning levels. We found that the most simple one not only performed best, but is the only algorithm that actually surpasses a plain Q-learner. This highlights the fact that the actual method of rule extraction is crucial for the applicability of the two-level-architecture.

Additionally, the presented application allowed us to show that the rules learned by such an agent are plausible and match expectations.

In this work we observed that the advantage of the RF1-algorithm over the plain Q-learner increases with the complexity of the state space. In an earlier

work [5] we observed that additional clues provided during an episode also matter. The experiment in this paper does not provide any clues except a reward of 100 for a goal state transition. In [5], however, we made the notion of similarity available to the ranking function which effected an enormous difference between the two-level learner and a plain Q-learner.

In our opinion, future directions of research should investigate the possible benefits of adding further reasoning capabilities to the agent. For instance, the symbolic rules provide a basis for standard reasoning and inference algorithms [6].

Acknowledgments. This research was funded by the German Research Association (DFG) under Grant PE 887/3-3.

References

1. Alchourron, C.E., Gardenfors, P., Makinson, D.: On the logic of theory change: Partial meet contraction and revision functions. J. Symbolic Logic 50(2), 510–530 (1985)
2. Gombert, J.E.: Implicit and explicit learning to read: Implication as for subtypes of dyslexia. Current Psychology Letters 1(10) (2003)
3. Häming, K., Peters, G.: A hybrid learning system for object recognition. In: 8th International Conference on Informatics in Control, Automation, and Robotics (ICINCO 2011), Noordwijkerhout, The Netherlands, July 28-31 (2011)
4. Häming, K., Peters, G.: Improved revision of ranking functions for the generalization of belief in the context of unobserved variables. In: International Conference on Neural Computation Theory and Applications (NCTA 2011), October 24-26 (2011)
5. Häming, K., Peters, G.: Ranking Functions in Large State Spaces. In: Iliadis, L., Maglogiannis, I., Papadopoulos, H. (eds.) EANN/AIAI 2011. IFIP AICT, vol. 364, pp. 219–228. Springer, Heidelberg (2011)
6. Robinson, J.A., Voronkov, A. (eds.): Handbook of Automated Reasoning (in 2 volumes). Elsevier and MIT Press (2001)
7. Ryman-Tubb, N.F., Krause, P.: Neural Network Rule Extraction to Detect Credit Card Fraud. In: Iliadis, L., Jayne, C. (eds.) EANN/AIAI 2011. IFIP AICT, vol. 363, pp. 101–110. Springer, Heidelberg (2011)
8. Spohn, W.: Ordinal conditional functions: A dynamic theory of epistemic states. In: Causation in Decision, Belief Change and Statistics, pp. 105–134 (August 1988)
9. Spohn, W.: Ranking functions, agm style. Internet Festschrift for Peter Gärdenfors (1999)
10. Spohn, W.: A survey of ranking theory. In: Degrees of Belief. Springer (2009)
11. Sun, R., Terry, C., Slusarz, P.: The interaction of the explicit and the implicit in skill learning: A dual-process approach. Psychological Review 112, 159–192 (2005)
12. Sutton, R.S., Barto, A.G.: Reinforcement Learning: An Introduction. MIT Press, Cambridge (1998)
13. Tadepalli, P., Givan, R., Driessens, K.: Relational reinforcement learning: An overview. In: Proceedings of the ICML 2004 Workshop on Relational Reinforcement Learning (2004)