Author prepared draft.
D. Vincze: Fuzzy Rule Interpolation and Reinforcement Learning
SAMI2017, Proceedings of the IEEE 15th International Symposium on
Applied Machine Intelligence and Informatics (SAMI), 2017, Herlany, Slovakia, pp. 173-178.

Fuzzy Rule Interpolation and Reinforcement Learning

Dávid Vincze
Department of Information Technology, University of Miskolc, Miskolc, Hungary
david.vincze@iit.uni-miskolc.hu

Abstract—**Reinforcement Learning (RL) methods became popular decades ago and still maintain to be one of the mainstream topics in computational intelligence. Countless different RL methods and variants can be found in the literature, each one having its own advantages and disadvantages in a specific application domain. Representation of the revealed knowledge can be realized in several ways depending on the exact RL method, including e.g. simple discrete Q-tables, fuzzy rule-bases, artificial neural networks. Introducing interpolation within the knowledge-base allows the omission of less important, redundant information, while still keeping the system functional. A Fuzzy Rule Interpolation-based (FRI) RL method called FRIQ-learning is a method which possesses this feature. By omitting the unimportant, dependent fuzzy rules - emphasizing the cardinal entries of the knowledge representation - FRIQ-learning is also suitable for knowledge extraction. In this paper the fundamental concepts of FRIQ-learning and associated extensions of the method along with benchmarks will be discussed.**

I. Introduction

Many machine learning methods which fall into the category of reinforcement learning (RL) were developed and implemented in the past decades. These methods can be used to find out what to do in certain situations in a given system. Everything an agent does in a given situation is rewarded by the system. What the agent does is called the action and the actual situation is described by the state variables. Feedback on the performed action is given by the reward function in the form of rewards or punishments. Based on the rewards, the quality value of a performed action in a given state can be determined. Therefore RL methods can give possible solutions to problems where priory knowledge can be expressed in a form that describes what is needed to be achieved, not in how to solve the problem imperatively. Generally a reinforcement learning problem is defined by the state values describing the possible states of the agent and environment, a set of actions the agent can execute, and the reward function.

The relationship between the state-action pairs and how good they are (quality level) is represented by the state-action-value function, and the task for certain reinforcement learning algorithms is to approximate this function. Several methods exist following different strategies for the discovery of these mappings, e.g. Temporal Difference learning (TD) [22], Q-learning [30], SARSA [20] and their variants are the most common ones. The goal of all these methods is to find an optimal policy to solve the given problem. Here, policy means a strategy for getting the most out of the environment and using the already explored knowledge.

Knowledge representations used in RL vary from simple discrete Q-tables (lookup tables) through artificial neural networks (ANN) to rule-bases, etc. The drawback of neural networks in this case is that ANNs have a black-box nature. Explaining the knowledge embedded in trained neural networks is not straightforward or even impossible [1], [6]. Although various methods were developed for extracting symbolic knowledge from trained ANNs [1], even in the form of fuzzy rules ([12], [19]), the extracted rules cannot be directly used for control or fed back into the ANN with modifications.

In contrast, using a white-box system like a fuzzy system yields in self explaining, close to natural language knowledge representation. Furthermore the rule-base can be examined even during the learning phase. Application of fuzzy rule-bases as knowledge representation in RL has been already presented: Fuzzy Q-learning [4], [5]. This extends certain RL methods to be used with continuous state and action spaces and store the knowledge in the form of fuzzy rules. Therefore the state-action-value function is approximated by fuzzy inference, with dynamically updated consequent in the rule-base. In this case the fuzzy rule-base should be a complete rule-base having a rule for every possible state-action combination.

Fuzzy Rule Interpolation (FRI) allows using sparse fuzzy rule-bases, meaning that rules can be intentionally missing from the rule-base, and redundant rules can be omitted. Also a sparse fuzzy rule-base as knowledge representation can be easily interpreted directly by human experts. This paper discusses the FRIQ-learning method which incorporates FRI in the approximation process of the state-action-value function. Therefore by using FRIQ-learning the fuzzy rule-base can be significantly smaller compared to Fuzzy Q-learning, in certain cases only a handful of rules is shown to be sufficient. Moreover by using FRIQ-learning, it is possible to automatically discover the most important rules, therefore only the effectively needed rules are kept in the rule-base.

First a brief overview is given on discrete and fuzzy Q-learning in chapter II. Chapter III. is about fuzzy rule interpolation, especially the FRI method FIVE, which was chosen to be incorporated in FRIQ-learning. After introducing the two main techniques FRIQ-learning is based on, Chapter IV. presents FRIQ-learning along with its incremental rule-base construction and decremental rule-base reduction possibilities. Finally, the last chapter discusses application examples (common RL benchmarks) for FRIQ-learning-based knowledge extraction.

II. Q-learning and FQ-learning

As mentioned, the goal of reinforcement learning algorithms is to find an optimal policy with the approximation of the state-action-value function for the given problem. Thus for estimating the optimal policy, the state-action-value function $Q^{\pi}(s,a)$ is needed to be approximated. $Q^{\pi}(s,a)$ is a function of the expected return, in case of taking action $a \in A_s$ in state s while following policy π. As stated before, many algorithms were developed for estimating the state-action-value function. A common algorithm, Watkins' Q-learning [30] is presented here as it was the method chosen for the base of FRIQ-learning. The update function for Q-learning is the following:

$$Q_{i,u} \approx \widetilde{Q}_{i,u}^{k+1} = \widetilde{Q}_{i,u}^{k} + \Delta\widetilde{Q}_{i,u}^{k+1} = \widetilde{Q}_{i,u}^{k} + \alpha_{i,u}^{k} \cdot \left(g_{i,u,j} + \gamma \cdot \max_{v \in U} \widetilde{Q}_{j,v}^{k+1} - \widetilde{Q}_{i,u}^{k}\right) \quad (1)$$

$\forall i \in I$, $\forall u \in U$, where $\widetilde{Q}_{i,u}^{k+1}$ is the $(k+1)^{th}$ iteration of the action-value taking the u^{th} action A_u in the i^{th} state S_i, S_j is the new (j^{th}) observed state, $g_{i,u,j}$ is the observed reward completing the $S_i \rightarrow S_j$ state-transition, γ is the discount factor and $\alpha^{k}_{i,u} \in [0,1]$ is the step size parameter, I is the set of the discrete possible states and U is the set of the discrete possible actions. By applying (1), the state-action-value function can be approximated in a discrete environment.

Several variations of Fuzzy Q-learning applying this iteration (1) to continuous environment have been developed, see e.g. [4], [5] and [7] for details. The main difference between the original Q-learning and Fuzzy Q-learning can be found in the way of the knowledge representation in the model. Fuzzy Q-learning uses fuzzy rules for storing the explored knowledge, while Q-learning keeps this in a simple lookup table as discrete values. Using the 0-order Takagi-Sugeno Fuzzy Inference (TSFI) model for characterizing the value function $Q(s,a)$ in continuous state-action space, the approximation of $\widetilde{Q}(s,a)$ is adapted in the following manner:

$$\textbf{If } s \textbf{ is } S_i \textbf{ And } a \textbf{ is } A_u \textbf{ Then } \widetilde{Q}(s,a) = Q_{i,u}, \; i \in I, u \in U \quad (2)$$

where S_i is the label of the i^{th} membership function of the n dimensional state space, A_u is the label of the u^{th} membership function of the one dimensional action space, $Q_{i,u}$ is the singleton conclusion and $\widetilde{Q}(s,a)$ is the approximated continuous state-action-value function. Setting up the antecedent fuzzy partitions to be Ruspini partitions, TSFI yields in the following function for the approximation of $\widetilde{Q}(s,a)$:

$$\widetilde{Q}(s,a) = \sum_{i_1, i_2, \cdots, i_N, u}^{I_1, I_2, \ldots, I_N, U} \prod_{n=1}^{N} \mu_{i_n, n}(s_n) \cdot \mu_u(a) \cdot q_{i_1 i_2 \ldots i_N u} \quad (3)$$

where $\mu_{i_n,n}(s_n)$ is the membership value of the i_n^{th} state antecedent fuzzy set at the n^{th} dimension of the N dimensional state antecedent universe at the state observation s_n, $\mu_u(a)$ is the membership value of the u^{th} action antecedent fuzzy set at the action selection a, $q_{i_1 i_2 \ldots i_N u}$ is the value of the singleton conclusion of the $i_1, i_2, \ldots, i_N, u^{th}$ fuzzy rule. Applying the function of Q-learning (1) for adjusting the singleton conclusions in (3) yields the function:

$$\begin{aligned} q_{i_1 i_2 \ldots i_N u}^{k+1} &= q_{i_1 i_2 \ldots i_N u}^{k} + \prod_{n=1}^{N} \mu_{i_n, n}(s_n) \cdot \mu_u(a) \cdot \Delta\widetilde{Q}_{i,u}^{k+1} = \\ &= q_{i_1 i_2 \ldots i_N u}^{k} + \prod_{n=1}^{N} \mu_{i_n, n}(s_n) \cdot \mu_u(a) \cdot \alpha_{i,u}^{k} \cdot \left(g_{i,u,j} + \gamma \cdot \max_{v \in U} \widetilde{Q}_{j,v}^{k+1} - \widetilde{Q}_{i,u}^{k}\right) \end{aligned} \quad (4)$$

where $q_{i_1 i_2 \ldots i_N u}^{k+1}$ is the $(k+1)^{th}$ iteration of the singleton conclusion of the $i_1, i_2, \ldots, i_N, u^{th}$ fuzzy rule taking action A_u in state S_i, S_j is the new observed state, $g_{i,u,j}$ is the observed reward completing the $S_i \rightarrow S_j$ state-transition, γ is the discount factor and $\alpha^{k}_{i,u} \in [0,1]$ is the step size parameter. The $\mu_{i_n,n}(s_n) \cdot \mu_u(a)$ is the partial derivative of the conclusion of TKSI with respect to the fuzzy rule consequents $q_{u,i}$ according to (3), required for the applied steepest-descent optimization method. Having (4), the state-value-function can be approximated using a fuzzy rule-base as knowledge representation.

III. Fuzzy Rule Interpolation

When using classical fuzzy reasoning (non-FRI), for every possible observation there should be a rule present in the rule-base, otherwise the system cannot determine a valid a conclusion (output). Therefore many additional rules should be inserted into the rule-base. Basically these fuzzy rules are mostly redundant rules having no or little effect on the reasoning process. To ease the process of adding these supplementary, but necessary rules, methods exists for automatically generating these rules with their corresponding conclusions [8].

Although this can be sufficient when constructing a fuzzy controller with a static embedded rule-base, however in the case when the rule-base is updated frequently this approach can have drawbacks. By using fuzzy rule interpolation this drawback can be easily solved. Furthermore it is possible to construct sparse rule-bases based on sample input-output data (see [8], [9]), even with embedding some minimal a priori knowledge as an initial rule-base [18].

From the various FRI techniques [2], the method called Fuzzy rule Interpolation in Vague Environment (FIVE) serves as the base of FRIQ-learning. FIVE, introduced in [14], [15] and [16] was chosen because it is an application oriented FRI method, and well suits the needs of RL problems.

The main idea of the FIVE is based on the fact that most of the control applications serves crisp observations and requires crisp conclusions from the controller. FIVE works by adopting the idea of the vague environment [11]. By handling the antecedent and consequent fuzzy partitions of the fuzzy rule-base as scaling functions (weighting factors), FIVE turns fuzzy interpolation to crisp interpolation. Therefore any crisp interpolation method can be adapted for FRI. The Shepard operator based interpolation [21] is adapted for FIVE because of its simple multidimensional applicability.

The implementation of the FIVE FRI is available as part of the FRI Toolbox [10].

IV. FRIQ-learning

As its name suggests, FRIQ-learning (introduced in [25]) is based on Fuzzy Q-learning preserving the state-action-value update function of Watkins' Q-learning [30]. Compared to FQ-learning, FRIQ-learning advances further by incorporating fuzzy rule interpolation into the system, hence allowing the omission of those fuzzy rules which have small importance or can be calculated based on the surrounding rules. Standard Fuzzy Q-learning requires a complete fuzzy rule-base, which contains a rule for every possible case. FRIQ-learning in contrast does not need a complete fuzzy rule-base, only the cardinal rules should be built into the rule-base.

Basically FRIQ-learning extends Fuzzy Q-learning by replacing the underlying fuzzy inference method. Fuzzy Q-learning uses the zero-order Takagi-Sugeno inference method and exchanging it with the FIVE model leads to FRIQ-learning.

The FIVE FRI based fuzzy model in case of singleton rule consequents [13] can be expressed by the following formula (similarly to (3)):

$$\widetilde{Q}(s,a)=\begin{cases} q_{i_1 i_2 .. i_N u} & \text{if } \mathbf{x}=\mathbf{a}_l \text{ for some } l, \\ \left(\sum_{l=1}^{r} q_{i_1 i_2 .. i_N u} / \delta_{s,l}^{\lambda}\right) \Big/ \left(\sum_{l=1}^{r} 1/\delta_{s,l}^{\lambda}\right) & \text{otherwise.} \end{cases} \tag{5}$$

where $q_{i_1 i_2 .. i_N u}$ is the singleton conclusion of the rule for states $i_1 ... i_N$ (N is the number of state dimensions) and action u, r is the number of rules and $\delta_{s,l}$ is the scaled distance:

$$\delta_{s,l}=\delta_s(\mathbf{a}_l,\mathbf{x})=\left[\sum_{i=1}^{m}\left(\int_{\mathrm{a}_{l,i}}^{\mathrm{x}_i} s_{\mathrm{X}_i}(\mathrm{x}_i)d\mathrm{x}_i\right)^2\right]^{1/2}, \tag{6}$$

and s_{X_i} is the i^{th} scaling function of the m dimensional antecedent universe, $\mathbf{x}$ is the m dimensional crisp observation and $\mathbf{a}_l$ are the cores of the m dimensional fuzzy rule antecedents $\boldsymbol{A}_l$.

Incorporating (5) into the state-action-value $\widetilde{Q}(s,a)$ approximation function results in the following equation:

$$\widetilde{Q}(s,a)=\begin{cases} q_{i_1 i_2 .. i_N u} & \text{if } \mathbf{x}=\mathbf{a}_k \text{ forsome} k, \\ \sum_{i_1,i_2,\cdot\cdot,i_N,u}^{I_1,I_2,\ldots,I_N,U} \prod_{n=1}^{N}\left(1/\delta_{s,k}^{\lambda}\right)\Big/\left(\sum_{l=1}^{r} 1/\delta_{s,l}^{\lambda}\right)\cdot q_{i_1 i_2 .. i_N u} & \text{otherwis} \end{cases} \tag{7}$$

The partial derivative of the model consequent $\widetilde{Q}(s,a)$ with respect to the fuzzy rule consequents $q_{u,i}$, required (as in (4)) in case of the FIVE fuzzy rule interpolation FRI model (see [17]) from (7) can be expressed by the following formula:

$$\frac{\partial \widetilde{Q}(s,a)}{\partial q_{i_1 i_2 \ldots i_N u}}=\begin{cases} 1 & \text{if x=a}_k \text{ for some } k, \\ \left(1/\delta_{s,k}^{\lambda}\right)\Big/\left(\sum_{l=1}^{r} 1/\delta_{s,l}^{\lambda}\right) & \text{otherwise} \end{cases} \tag{8}$$

where $q_{u,i}$ is the constant rule consequent of the k^{th} fuzzy rule, $\delta_{s,k}$ is the scaled distance in the vague environment of the observation, and the k^{th} fuzzy rule antecedent, λ is a parameter of Shepard interpolation [21], x is the actual observation, and r is the number of the rules.

Replacing the partial derivative of the conclusion of the 0-order Takagi-Sugeno fuzzy inference (5) with the partial derivative of the conclusion of FIVE (8) with respect to the fuzzy rule consequents $q_{u,i}$ leads to the following equation for the Q-Learning action-value-function iteration:

$$\begin{aligned} &\text{if } \mathbf{x}=\mathbf{a}_k \text{ for some } k: \\ &q_{i_1 i_2 \ldots i_N u}^{k+1}=q_{i_1 i_2 \ldots i_N u}^{k}+\Delta\widetilde{Q}_{i,u}^{k+1}= \\ &=q_{i_1 i_2 \ldots i_N u}^{k}+\alpha_{i,u}^{k}\cdot\left(g_{i,u,j}+\gamma\cdot\max_{v\in U}\widetilde{Q}_{j,v}^{k+1}-\widetilde{Q}_{i,u}^{k}\right) \\ &\text{otherwise}: \\ &q_{i_1 i_2 \ldots i_N u}^{k+1}=q_{i_1 i_2 \ldots i_N u}^{k}+\prod_{n=1}^{N}\left(1/\delta_{s,k}^{\lambda}\right)\Big/\left(\sum_{l=1}^{r} 1/\delta_{s,l}^{\lambda}\right)\cdot\Delta\widetilde{Q}_{i,u}^{k+1}= \\ &=q_{i_1 i_2 \ldots i_N u}^{k}+\prod_{n=1}^{N}\left(1/\delta_{s,k}^{\lambda}\right)\Big/\left(\sum_{l=1}^{r} 1/\delta_{s,l}^{\lambda}\right)\cdot\alpha_{i,u}^{k}\cdot\left(g_{i,u,j}+\gamma\cdot\max_{v\in U}\widetilde{Q}_{j,v}^{k+1}-\widetilde{Q}_{i,u}^{k}\right) \end{aligned} \tag{9}$$

where variables are the same as for (4), (7) and (8).

Similarly to (4), for FQ-learning, the state-action-value function can be approximated by (9) conforming to the FIVE FRI model, creating FRIQ-learning.

Following (9), not only one rule is updated, but many rules are updated with different weights depending on the distance between the observation and the given rule.

V. Knowledge Extraction with FRIQ-Learning

As described in the previous chapters, FRIQ-learning opens up the possibility to operate with sparse fuzzy rule-bases in RL. In [25], FRIQ-learning was demonstrated with an initial rule-base transformed from the original discrete Q-table, practically resulting in a covering rule-base, but with interpolation capabilities.

In order to exploit the real advantage of using FRI, an incremental rule-base construction strategy was suggested in [27] to extend the base method of FRIQ-learning. This method can automatically construct a fuzzy rule-base from scratch, only inserting new rules when required, possibly resulting in a significantly smaller rule-base compared to the covering rule-base. Additional methods [29] for further reducing the size of the incrementally constructed rule-base while keeping it to be able to operate the system correctly were developed. Applying these reduction methods, knowledge extraction becomes possible. Relatively small sparse fuzzy rule-bases able to solve RL problems can result in forming a human readable, natural language like knowledge representation.

The next two subchapters give an overview of the incremental rule-base construction and the decremental rule-base reduction strategies.

A. Incremental rule-base construction

This method starts the FRIQ-learning process with an empty fuzzy rule-base. Empty, considered from the viewpoint of incorporated knowledge, but still having initial fuzzy rules, which have the purpose of defining the boundaries (n-dimensional hypercube, where n is the number of states) of the space of the given problem. Then, in every iteration, the method checks whether the current rule-base is sufficient or a new rule has to be inserted into the rule-base, based on the reward gathered in the current state for the chosen action. In the case when the rule to be inserted is in the vicinity of an existing rule, then only the conclusion (Q value) is updated of the surrounding rules. Otherwise a new rule is inserted to the closest possible rule position. These possible rule positions are gained by inserting a new state among the existing ones. The terminal condition of this extension is having a rule-base which is sufficient to solve the problem without adding new or updating existing rules. For more details and examples, see [26] and [27].

B. *Decremental rule-base reduction*

The rule-base created by the previously presented incremental process might contain rules which are redundant or have no significant effect in the final rule-base. The reason for this is the dynamically changing rule-base during the construction process. The importance of a rule can change in each iteration. One rule can have a significant role during the construction phase, but not in the final rule-base: rules can override each other, or rules can be merged thanks to interpolation. To identify and filter these rules, different decremental rule-base reduction strategies were developed [29] especially for rule-bases constructed with the previous incremental method.

The first strategy eliminates rules one by one, and reruns the simulation episode to see whether the omission of that rule made any difference in the control process. If there is no or slight difference in the results, then the rule is considered unimportant and it is permanently dropped from the rule-base, otherwise the rule remains untouched. Repeating these steps until every rule is checked for the possibility of removal gives the final reduced rule-base. The selection of the rule to be tested is based on the absolute Q value (consequent) of the rule. FRIQ-learning uses a greedy policy for action selection, which suggest that rules with high absolute Q values (expected reward is high) probably are the most important ones. Accordingly this strategy begins with rules which have low assumed importance (low Q values).

The second strategy is very similar to the first strategy, with the difference, that it starts the elimination of rules which posses the highest absolute consequents (Q values). Therefore it tests the probably most important rules first.

The third strategy works with several rules in one removal step. Based on low consequent Q values, it forms a group of rules, hence allowing mass rule removal, which could significantly speed up the reduction process. Also this way different (than the previous two strategies) final rule-bases can be found.

A fourth reduction strategy with promising results is presented in [24]. This strategy selects the rules for possible removal with clustering methods also based on Q values.

VI. Implementation and Examples

The prototype implementation of the FRIQ-learning method was written in MATLAB. It was chosen because the implementation of the underlying FRI method FIVE was already implemented in MATLAB, and MATLAB is also suitable for rapid prototyping and for easy visualization. Since the development of the first versions of the implementation, FRIQ-learning was ported to the ANSI C language, which allows the programmer to be more close to the hardware level. This way a much more efficient implementation using less memory and considerably less CPU time is possible. Experiments conducted with this new implementation, depending on the problem, proved to be approx. 100-400 times faster than the original MATLAB implementation running on the same hardware. Some other optimizations were done within the FIVE method especially to be used in conjunction with FRIQ-learning, see [28] for details. Investigations for an embedded implementation of the FIVE FRI, which could support FRIQ-learning in dedicated hardware, were presented in [3].

The following subchapters present three common reinforcement learning benchmarks, namely the cart-pole, the mountain-car, and the acrobot problem. FRIQ-learning was applied to these benchmarks with the presented extensions for knowledge extraction by using rule-base reduction methods. Furthermore, a clone of Pong controlled by FRIQ-learning was developed; details and results can be found in [23].

A. *Cart-pole problem*

One of the most common test bed environments for RL methods is the cart-pole problem. A pole is attached to a self-propelled cart, which can move only in one dimension, to the left or to the right, but with different speeds. By controlling the direction and the speed of the cart, the pole should maintain its standing position. A simulation episode fails when the pole becomes out of balance or the cart crashes into the walls (boundaries of the simulation environment). Therefore the goal is to keep the pole in the upright position and to stay away from the walls. Thus the reward function is constructed based on these circumstances. If the pole falls a huge negative reward is given. Positive reward is given if the pole stays up for a whole simulation episode, without hitting the walls.

A MATLAB based implementation of the cart-pole problem written by José Antonio Martin H. [31] was taken as a base for the FRIQ-learning version of the simulation. In fact the state and action definitions, the reward function, the procedure for updating the state of the system, and the visualization part are used. The part in charge of the learning process itself was rewritten conforming to FRIQ-learning.

The variables describing the current state are: s_1 – shift of the pendulum, s_2 – velocity of the pendulum, s_3 – angular offset of the pole, s_4 – angular velocity of the pole. The size of the discrete Q-table in the original implementation is 2268 entries (4 states dimensions with possible values of, respectively: 2, 3, 9, 2, and 21 possible actions). Transforming this into a fuzzy rule-base for FRIQ-learning would result in the same number of rules as the number of entries in the discrete Q-table. Although one advantage of FRIQ-learning over the original version is that the simulation can now work in continuous space instead of discrete space; the real advantage is the introduction of interpolation, which allows having a working system with a sparse fuzzy rule-base.

When using the incremental rule-base construction procedure starting from scratch, the result of the construction process is a rule-base consisting of only 182 rules. With these rules the system is capable of achieving the same results as with the original 2268 rules. Moreover the decremental reduction process can be applied on the previous incrementally constructed rule-base, resulting in a rule-base which is significantly smaller than the original.

Table I. shows the final rule-base after the reduction process following the first reduction strategy (see previous chapter). Only five fuzzy rules are enough to control the cart in the same way as it was with 182 or 2286 rules originally. The terms used in Table I. are: P – Positive, N – Negative, Z – Zero, N3 – Negative Level 3, P12 – Positive Level 12, AP – Action Positive with different levels, and finally AN – Action Negative with different levels.

TABLE I.
RULE-BASE AFTER REDUCTION FOR THE CART-POLE PROBLEM

R#	s_1	s_2	s_3	s_4	a	Q
1	P	Z	Z	P	AP10	1325.1
2	P	Z	N3	N	AN10	1316.5
3	P	Z	Z	N	AN8	1322
4	P	P	Z	N	AN8	-3100.5
5	P	Z	P12	P	AP6	-6446.9

TABLE II.
RULE-BASE FROM TABLE I. AFTER TRANSFORMING TERMS

R#	*shift*	*speed*	*pole pos*	*pole fall*	*action*	*quality*
1	right	stands	stands	right	full right	good
2	right	stands	little left	left	full left	good
3	right	stands	stands	left	high left	good
4	right	right	stands	left	high left	bad
5	right	stands	fall right	right	mid right	very bad

The form of fuzzy rules and the few number of rules result in a knowledge-base which can be easily transformed to be human-readable with very little effort. Converting the above described terms into more easily interpretable terms results in Table II.

This way the knowledge-base becomes directly interpretable by human operators, without explicit knowledge about the learning method. One possible interpretation of the rules could be the following:

1. If the cart is to the right from the center and it is not moving and the pole is standing but falling to the right then full throttle to the right.
2. If the cart is to the right from the center and it is not moving and the pole is a little bit to the left and falling to the left then full throttle to the left.
3. If the cart is to the right from the center and it is not moving and the pole stands but falls to the left then go to the left with high speed.
4. If the cart is to the right from the center and it is moving to the right and the pole stands but falls to the left then going to the left with high speed is a bad idea (do not go to left with high speed – do something else.)
5. If the cart is to the right from the center and it is not moving and the pole is to the right and falls to the right then going to the right with medium speed is a very bad idea (do not go to right with medium speed – do something else.)

Although this example shows how effective the knowledge extraction can be in certain cases, but in fact the effectiveness heavily depends on the definition of the exact problem as can be seen in the following cases.

B. *Mountain-car problem*

The mountain-car problem is also a widely used benchmark for reinforcement learning algorithms [22], [31]. A car is placed in a valley between two hills with equal heights. By itself the car is not powerful enough to climb the hills, so it has to reach the top by gaining momentum while swinging between the two hill sides. In the simulation the hills and the valley is described by a simple cosine function. The goal to be reached is located on the top of the right hill, and the car starts from the bottom of the valley.

The problem space is defined with two state variables: s_1 – position of the car, s_2 – velocity of the car. The position state variable (s_1) can possess 10 different values, while the velocity state variables can posses 6 values, ranging from N3 (Negative, level 3) to P3 (Positive, level 3). Three actions are possible: go left, go right, and no propulsion (let the car roll back into the valley). Rewards are given for every step: -10 for every step which results in not reaching the target, and 1000 when the target is reached at the top of the hill.

Considering the possible state values, a covering fuzzy rule-base would have the size of 180 rules. The incremental construction process results in 110 rules, then using the first reduction strategy on this rule-base, the final rule-base will consist of 28 rules. This can be also considered as an effective rule-base size reduction, because only approx. 15% of the rules are kept. Also 28 rules could be still considered to be human readable knowledge-base after transformation of the terms. Not as spectacular as in the case of the cart-pole example, where only 5 rules are kept out of 182/2286, but still a good result.

C. *Acrobot problem*

The third example presented in this paper is the acrobot problem ([22], [31]), where the goal is to control a mechanical arm. This arm consists of two rods; the upper rod is fixed to an axle at one end, on the other end it is joined with the second part. Starting from the hanging position, the goal is to swing the arm to stand in the upright position. However, the arm can only be controlled by applying torque to the middle joint.

Four state variables are used in this problem, an angle and an angular velocity for both rods. Angles can posses five possible values, while angular velocities can have three different values. Three actions can be performed: move clockwise, move counter-clockwise, or do not apply torque (let gravity do its job).

According to the above specified possible state values and actions a covering rule-base would consist of 675 rules. Incremental construction gives 367 rules for this problem. When further reducing the rule-base using the first reduction strategy, rule count stops decreasing at 191. Hence approximately 28% of the full rule-base is kept for solving the same problem.

VII. CONCLUSIONS

As presented, FRIQ-learning can be used to perform knowledge extraction in the form of fuzzy rules. This is achieved by combining fuzzy rule interpolation-based inference with reinforcement learning and applying various techniques resulting in a reduced rule-base. FRIQ-learning allows the handling of sparse fuzzy rule-bases in RL. Furthermore it can automatically construct a sufficient sparse rule-base with its incremental rule-base construction and its decremental reduction extensions.

By using sparse fuzzy rule-bases as knowledge representation, the extracted knowledge can be easily read by humans as its form is very close natural language. Hence modifications and tuning of the knowledge-base in this form is straightforward. Also the resulting rule-base can be the basis of knowledge transfer into other systems.

While the effectiveness of the reduction can be high, it varies depending on the exact problem to be solved, and the definition of the problem (properly chosen states, actions and reward function). Three common RL benchmarks modified to incorporate FRIQ-learning were presented, showing the usability and effectiveness of FRIQ-learning and the associated rule-base construction and reduction techniques.

Acknowledgment

This work was partially supported by the Terplán Zénó TETRA program of the University of Miskolc. Also this research was partially realized in the frames of TÁMOP 4.2.4. A/2-11-1-2012-0001 „National Excellence Program – Elaborating and operating an inland student and researcher personal support system convergence program". The project was subsidized by the European Union and co-financed by the European Social Fund.

References

[1] R. Andrews, J. Diederich and A. B. Tickle, "Survey and critique of techniques for extracting rules from trained artificial neural networks" in Knowledge-Based Systems Vol. 8, Issue 6, 1995, pp. 297-396.

[2] P. Baranyi, L. T. Kóczy, and T. D. Gedeon, "A Generalized Concept for Fuzzy Rule Interpolation" in IEEE Trans. on Fuzzy Systems, vol. 12, No. 6, 2004, pp 820-837.

[3] R. Bartók and J. Vásárhelyi, "A fuzzy rule interpolation base algorithm implementation on different platforms" in Proceedings of the 16th International Carpathian Control Conference (ICCC), Szilvásvárad, Hungary, May 27-30, 2015, pp. 37-40.

[4] H. R. Berenji, "Fuzzy Q-Learning for Generalization of Reinforcement Learning" in Proc. of the 5th IEEE International Conference on Fuzzy Systems, 1996, pp. 2208-2214.

[5] A. Bonarini, "Delayed Reinforcement, Fuzzy Q-Learning and Fuzzy Logic Controllers" in Herrera, F., Verdegay, J. L. (Eds.) Genetic Algorithms and Soft Computing, (Studies in Fuzziness, 8), Physica-Verlag, Berlin, D, 1996, pp. 447-466.

[6] K. Dixon, R. J. Malak and P. K. Khosla, "Incorporating prior knowledge and previously learned information into reinforcement learning agents." Technical Report, Carnegie Mellon University, Institute for Complex Engineered Systems, 2000.

[7] T. Horiuchi, A. Fujino, O. Katai and T. Sawaragi, "Fuzzy Interpolation-Based Q-learning with Continuous States and Actions" in Proc. of the 5th IEEE International Conference on Fuzzy Systems, Vol.1, 1996, pp. 594-600.

[8] Zs. Cs. Johanyák, "Sparse Fuzzy Model Identification Matlab Toolbox - RuleMaker Toolbox" in Proceedings of the 6th IEEE International Conference on Computational Cybernetics, November 27-29, 2008, Stara Lesná, Slovakia, pp. 69-74.

[9] Zs. Cs. Johanyák, "New Initial Fuzzy System Generation Features in the SFMI Toolbox" in Proceedings of the 5th IEEE International Symposium on Logistics and Industrial Informatics (LINDI 2013), Wildau, Germany, September 5-7, 2013, pp. 29-34.

[10] Zs. Cs. Johanyák, D. Tikk, Sz. Kovács and K. W. Wong: Fuzzy Rule Interpolation Matlab Toolbox – FRI Toolbox, Proc. of the IEEE World Congress on Computational Intelligence (WCCI'06), 15th Int. Conf. on Fuzzy Systems (FUZZ-IEEE'06), July 16-21, Vancouver, BC, Canada, Omnipress. ISBN 0-7803-9489-5, 2006, pp. 1427-1433.

[11] F. Klawonn, "Fuzzy Sets and Vague Environments" in Fuzzy Sets and Systems, 66, 1994, pp. 207-221.

[12] E. Kolman and M. Margaliot, "Extracting symbolic knowledge from recurrent neural networks—A fuzzy logic approach" in Fuzzy Sets and Systems, Vol. 160, Issue 2, 2009, pp. 145-161.

[13] Sz. Kovács, "Extending the Fuzzy Rule Interpolation "FIVE" by Fuzzy Observation" in Advances in Soft Computing, Computational Intelligence, Theory and Applications, Bernd Reusch (Ed.), Springer, 2006, Germany, pp. 485-497.

[14] Sz. Kovács and L.T. Kóczy, "Approximate Fuzzy Reasoning Based on Interpolation in the Vague Environment of the Fuzzy Rule base as a Practical Alternative of the Classical CRI", Proceedings of the 7th International Fuzzy Systems Association World Congress, Prague, Czech Republic, 1997, 144-149.

[15] Sz. Kovács, "New Aspects of Interpolative Reasoning" in Proceedings of the 6th. International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems, Granada, Spain, 1996, pp. 477-482.

[16] Sz. Kovács and L.T. Kóczy, "The use of the concept of vague environment in approximate fuzzy reasoning" in Fuzzy Set Theory and Applications, Tatra Mountains Mathematical Publications, Mathematical Institute Slovak Academy of Sciences, Bratislava, Slovak Republic, vol.12, 1997, pp. 169-181.

[17] Z. Krizsán and Sz. Kovács, "Gradient based parameter optimisation of FRI "FIVE"" in Proceedings of the 9th International Symposium of Hungarian Researchers on Computational Intelligence and Informatics, Budapest, Hungary, Nov. 6-8, 2008, pp. 531-538.

[18] J. Li, H. P. H. Shum, X. Fu, G. Sexton and L. Yang, "Experience-based rule base generation and adaptation for fuzzy interpolation", 2016 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE), Vancouver, BC, 2016, pp. 102-109.

[19] C.J. Mantas, J.M. Puche and J.M. Mantas, "Extraction of similarity based fuzzy rules from artificial neural networks" in International Journal of Approximate Reasoning, Vol. 43, Issue 2, 2006, pp. 202-221.

[20] G. A. Rummery and M. Niranjan, "On-line Q-learning using connectionist systems" in CUED/F-INFENG/TR 166, 1994, Cambridge University, UK.

[21] D. Shepard, "A two dimensional interpolation function for irregularly spaced data", Proc. 23rd ACM Internat. Conf., 1968, pp. 517-524.

[22] R. S. Sutton and A. G. Barto: *Reinforcement Learning: An Introduction,* MIT Press, Cambridge, 1998.

[23] T. Tompa, D. Vincze and Sz. Kovács: "The Pong game implementation with the FRIQ-learning reinforcement learning algorithm" in Proceedings of the 16th International Carpathian Control Conference (ICCC), Szilvásvárad, Hungary, May 27-30, 2015, pp. 542-547.

[24] T. Tompa and Sz. Kovács, "Clustering-based fuzzy knowledgebase reduction in the FRIQ-learning" in Proceedings of the 15th International Symposium on Applied Machine Intelligence and Informatics (SAMI 2017), January 26-28, 2017, Herl'any, Slovakia, in press.

[25] D. Vincze and Sz. Kovács, "Fuzzy Rule Interpolation-based Q-learning" in Proceedings of the 5th International Symposium on Applied Computational Intelligence and Informatics (SACI 2009), Timisoara, Romania, May 28-29, 2009, pp. 55-59.

[26] D. Vincze and Sz. Kovács, "Reduced Rule Base in Fuzzy Rule Interpolation-based Q-learning" in Proceedings of the 10th International Symposium of Hungarian Researchers on Computational Intelligence and Informatics, CINTI 2009, November 12-14, 2009, Budapest Tech, Budapest, pp. 533-544.

[27] D. Vincze and Sz. Kovács, "Incremental Rule Base Creation with Fuzzy Rule Interpolation-Based Q-Learning" in I. J. Rudas et al. (Eds.), Computational Intelligence in Engineering, Studies in Computational Intelligence, Vol. 313/2010, Springer-Verlag, Berlin Heidelberg, 2010, pp. 191-203.

[28] D. Vincze and Sz. Kovács, "Performance Optimization of the Fuzzy Rule Interpolation Method 'FIVE'" in Journal of Advanced Computational Intelligence and Intelligent Informatics (JACIII), Vol.15 No.3, Special issue on Fuzzy Rule Interpolation, 2011, Fuji Technology Press, Tokyo, Japan, pp. 313-320.

[29] D. Vincze and Sz. Kovács, "Rule-Base Reduction in Fuzzy Rule Interpolation-Based Q-Learning" in Recent Innovations in Mechatronics (RIiM) Vol. 2. (2015) No. 1-2.

[30] C. J. C. H. Watkins, "Learning from Delayed Rewards", Ph.D. thesis, Cambridge University, 1989, Cambridge, England

[31] The cart-pole example for discrete space can be found at: http://www.dia.fi.upm.es/~jamartin/download.htm