Knowledge Base Optimization of the HFRIQ-Learning

Tamás Tompa, Szilveszter Kovács

Department of Information Technology, University of Miskolc,
Miskolc-Egyetemváros, H-3515 Miskolc, Hungary
e-mail: tompa@iit.uni-miskolc.hu, szkovacs@iit.uni-miskolc.hu

Abstract: The learning process of conventional reinforcement learning methods, such as Q-learning and SARSA typically start with an empty knowledge base. In each iteration step, the initial empty knowledge base is gradually constructed by reinforcement signals obtained from the environment. Even only if a fragment of knowledge is available regarding the system behavior which can be injected into the learning process, the learning performance can be improved. In Heuristically Accelerated Fuzzy Rule Interpolation-based Q-learning (HFRIQ-learning), the external knowledge can be represented in the form of human experts defined state-action fuzzy rules. If the expert knowledge base contains inaccuracies, i.e., incorrect state-action rules, it can negatively impact the learning performance. The main goal of this paper is to introduce a methodology for correcting (optimizing) the inaccurate a priori expert knowledge and as an additional benefit of optimization, to reduce the size of the Q-function representation fuzzy rule-base during the learning phase. The paper also introduces some examples how the quality of expert knowledge influences the HFRIQ-learning performance on a well-known reinforcement learning benchmark problem.

Keywords: Reinforcement Learning; Heuristically Accelerated Reinforcement Learning; Expert knowledge representation; Fuzzy Rule Interpolation; Q-Learning; Expert rule validation

1 Introduction

Due to the increasing computational capacity and its integration into everyday devices, the fields of artificial intelligence [25], machine learning [10] [23], and the exploration of opportunities offered by these methods are becoming increasingly relevant and significant. Machine learning (ML) is a collection of methods that learn through experience, gradually building the knowledge base of the system over iterations.

In the case of reinforcement learning (RL) [26], the system shapes its behavior based on only reinforcement (reward or punishment) information. The RL methods receive feedback (reinforcement) for the execution of certain decisions or sequences

of decisions. The main idea of the RL methods is not only to utilize feedback for shaping the current actions of the agent but also to improve the quality of the future decisions. The knowledge base representation (Q-function) can be different, in the case of the traditional Q-learning [42] and the Deep Q-learning [15] the Q-function is represented by a Q-table and in the case of the fuzzy (and fuzzy rule-interpolation [17] [22]) model based RL methods [2] [6] [11] the Q-function is described by a fuzzy rule-base. In these methods, the learning process begins with an empty knowledge base and the system incrementally fills it according to the feedback (reinforcements) from the environment.

The "Heuristically Accelerated Reinforcement Learning" (HARL) methods [7] such as HA-Q(λ), HA-SARSA(λ) and HA-TD(λ) [8] [9] provide ways for injecting external knowledge into the learning system. In these cases, the preliminary knowledge is defined by a $H_t(s_t, a_t)$ heuristic function. This heuristic function can be considered a policy modifier ($H: S \times A \rightarrow R$) because it determines which action a_t is preferred to be executed in the state s_t at the given time t [9]. There are also methods that reuse (after the learning) the knowledge created during the learning phase, such as "Transfer Learning" [36] and multi-agent systems [27]. In [1], a policy modifier for action selection is proposed that incorporates human knowledge (by a user-tunable parameter) into the RL system, enhancing the learning performance through the expert knowledge. Furthermore, other authors also recommend incorporating expert knowledge into machine learning methods, such as Deep Neural Networks (DNN), RL [44] and Deep RL systems [3] [14] [28]. The role of the human (expert) factor in machine learning ("human-in-the-loop" ML) methods are detailed in [43]. Generally, RL systems are used to tune external knowledge bases, but the RL system injected knowledge itself is not the subject of the optimization. The paper [45] introduces an automatic cloud-based database (CDB) tuning system (CDBTune) which optimizes the high-dimensional configuration space using deep RL. In [13] the "XTuning" system is introduced, which is an expert database tuning system that applies reinforcement learning (RL) techniques. XTuning incorporates a correlation knowledge model to eliminate unnecessary training costs and employs a multi-instance mechanism (MIM) for fine-grained tuning across diverse workloads.

The HFRIQ-learning (Heuristically Accelerated Fuzzy Rule-Interpolation based Q-learning) system [34] is a HARL method in which the external, a priori expert knowledge is given by fuzzy production rules having state-action format [30]. The quality of the initial expert rules can influence the efficiency of the learning process. If the expert knowledge base contains inaccuracies (or incorrect information), it can adversely affect the efficiency of the learning [29] [31].

The main goal of this paper is to improve the HFRIQ-learning (introduced in [34]) by introducing a methodology, which is suitable for locating and fixing (tuning) the inaccurate expert rules during the learning process and also able to validate the initial expert rules by comparing them to the tuned final expert rules after the learning phase. For demonstrating the effect of the expert knowledge quality on the

improved HFRIQ-learning performance some examples of a well-known reinforcement learning benchmark problem will be also discussed in the paper.

2 The Heuristically Accelerated Fuzzy Rule Interpolation based Q-learning

The "Heuristically Accelerated Fuzzy Rule-Interpolation based Q-learning" (HFRIQ-learning) [34] is an extension of the "Fuzzy Rule-Interpolation based Q-learning" (FRIQ-learning) [39] by the capability of embedding [34] external expert knowledge into the system. This method applies the "FIVE" (Fuzzy Rule Interpolation based on Vague Environment) [20] Fuzzy Rule Interpolation (FRI) method for representing the Q-function describing the state-action space as a continuous function. The "FIVE" [20] is an application-oriented FRI method, its low computational demands [4], [5], makes it suitable for real-time applications and robotic control. Additionally, being an FRI method it also allows sparse fuzzy rule base as a knowledge representation, further reducing the system complexity compared to other classical fuzzy inference methods (e.g. the Zadeh-Mamdani CRI) [21].

The knowledge base of the HFRIQ-learning system is represented by a sparse fuzzy rule-base. The form of a rule r_i ($i \in [1, m]$) in the rule-base R of size m is as follows [39]:

$$r_i\text{: } \boldsymbol{If}\ s_1 \text{ is } S_1^i \ \boldsymbol{And}\ s_2 \text{ is } S_2^i \ \boldsymbol{And} \ldots \boldsymbol{And}\ s_n \text{ is } S_n^i \ \boldsymbol{And}\ a \text{ is } A^i \ \boldsymbol{Then}\ \tilde{Q}(\boldsymbol{s}, a) = q^i \qquad (1)$$

where S_j^i is the fuzzy set of the i-th ($i \in [1, m]$) rule in the j-th ($j \in [1, n]$) state dimension in the n-dimensional state space $\boldsymbol{S}$, $\boldsymbol{s} \in \boldsymbol{S}$ is the n-dimensional state observation, s_j is the j-th dimension of the state observation $\boldsymbol{s}$, A^i is the fuzzy set of the one-dimensional action universe (U) for the i-th rule, $a \in U$ is the action, $\tilde{Q}(\boldsymbol{s}, a)$ is the approximated Q-function according to the FIVE FRI [19], and q^i is the consequent (Q-value) of the i-th rule. The Q-function is obtained by the "FIVE" FRI method from the Q-function fuzzy rule-base.

The rule format of the expert knowledge base R_{expert} is similar to the Q-function fuzzy rules (1), with the difference that the antecedent of the $\hat{r}$ expert rules is the state, and the consequent is the state related preferred action [30]:

$$\hat{r}_i\text{: } \textbf{If } s_1 \boldsymbol{is}\ \hat{S}_1^i \ \boldsymbol{And}\ s_2 \boldsymbol{is}\ \hat{S}_2^i \ \boldsymbol{And} \ldots \boldsymbol{And}\ s_n \boldsymbol{is}\ \hat{S}_n^i \ \boldsymbol{Then}\ \text{a} = \hat{A}^i \qquad (2)$$

where $\hat{r}_i$ is the i-th ($i \in [1, \hat{m}]$) expert rule in the R_{expert} rule-base, $\hat{S}_n^i = [\hat{S}_1^i, \hat{S}_2^i, \ldots \hat{S}_n^i]$ is the n-dimensional state observation for the i-th expert rule, $\hat{A}^i$ is the action related to the $\hat{S}_n^i$ state observation, and i ($i \in [1, \hat{m}]$) is the index of the rule in the $\hat{m}$ expert rule-base.

Incorporation of the expert rule-base in the FRIQ-learning requires transforming the state-action expert rules into a state-action-Q-value (1) format. This transformation is a necessary step to ensure the compatibility of external knowledge with the fuzzy Q-function rule-base, allowing the integration of the external knowledge within the FRIQ-learning environment. The transformed rules will have the state and action of the expert rule as the antecedent and an estimated $\tilde{Q}_{init}$ value as the consequent. The estimated initial $\tilde{Q}_{init}$ value can be determined based on the maximum possible reinforcement that the environment can provide, denoted by g_{max} [30]. The learning process (the incremental rule-base construction) of the HFRIQ-learning system begins with the merging of the R_{expert} expert rule-base (2) and the initial (empty) rule-base of the FRIQ-learning, which is the $R^{\square}$ corner rule-base (3). Each $r_i^{\square}$ corner rule having 0 consequent value [30] [39]:

$$r_i^{\square}: \textbf{\textit{If}}\ s_1\ \textit{is}\ S_1^{\square i}\ \textbf{\textit{And}}\ s_2\ \textit{is}\ S_2^{\square i}\ \textbf{\textit{And}}\ \ldots\ \textbf{\textit{And}}\ s_n\ \textit{is}\ S_n^{\square i}\ \ \textbf{\textit{And}}\ a\ \textit{is}\ A^{\square i}\ \textbf{\textit{Then}}\ \tilde{Q}(\boldsymbol{s},a)=0 \qquad (3)$$

where $S_l^{\square i} \in [\min(S_l), \max(S_l)]$ $(\forall i \in [1, 2^{n+1}], \forall l \in [1,n])$ and $A^{\square i} \in [\min(A), \max(A)]$ $(\forall i \in [1, 2^{n+1}])$ are the corner state, $r_i^{\square} \in R$ $(i \in [1, 2^{n+1}])$ is the i-th corner rule and n is the number of the state dimension.

In cases if where is a contradicting situation arises, i.e. if an expert rule antecedent matches a corner rule antecedent, but their consequents are different, a single rule with the consequent of the transformed expert rule is inserted [34]. The initial expert rule-base, created previously, grows incrementally during the learning process by adding new rules that are generated by the system [39]. A new rule is inserted into the rule-base (in the state-action observation position as antecedent) if the Q-update value ($\Delta\tilde{Q}$) is greater than ε_Q ($\Delta\tilde{Q} > \varepsilon_Q$) [33], [34] and the closest rule is considered to be distant. The determination of rule proximity is based on distances calculated for each (antecedent) dimension between the rules [33], [34]. In cases if the $\Delta\tilde{Q}$ value is small ($\Delta\tilde{Q} < \varepsilon_Q$), the entire rule-base consequent is updated in the following manner [40]:

$$q_i^{k+1} = \begin{cases} q_i^k + \Delta\tilde{Q}^{k+1}(\boldsymbol{s},a) & \text{if } (\boldsymbol{s},a) = (\boldsymbol{s}^i, a^i) \text{ for some } i \\ q_i^k + \Delta\tilde{Q}^{k+1}(\boldsymbol{s},a) * \left(1/\delta_{v,i}^{\lambda}\right) \Big/ \left(\sum_{i=1}^{m} 1/\delta_{v,i}^{\lambda}\right) & \text{otherwise} \end{cases} \qquad (4)$$

where the $\tilde{Q}^{k+1}(\boldsymbol{s},a)$:

$$\tilde{Q}^{k+1}(\boldsymbol{s},a) = \tilde{Q}^{k}(\boldsymbol{s},a) + \Delta\tilde{Q}^{k+1}(\boldsymbol{s},a) \qquad (5)$$

$$\Delta\tilde{Q}^{k+1}(\boldsymbol{s},a) = \alpha * \left(g(\boldsymbol{s},a,\boldsymbol{s}') + \gamma * \max_{a' \in U} \tilde{Q}^{k}(\boldsymbol{s}', a') - \tilde{Q}^{k}(\boldsymbol{s},a)\right) \qquad (6)$$

where $\gamma \in [0,1]$ is the discount factor, $\alpha \in [0,1]$ is the learning rate, q_i^{k+1} is the i-th rule conclusion in the $(k+1)$-th iteration, a is the action taken in state $\boldsymbol{s}$, $\boldsymbol{s}'$ is the newly observed state, $g(\boldsymbol{s},a,\boldsymbol{s}')$ is the observed reward for the $\boldsymbol{s} \rightarrow \boldsymbol{s}'$ state transition,

$\tilde{Q}^k$ and $\tilde{Q}^{k+1}$ are the Q-values approximated by the FIVE FRI in the k-th and $(k+1)$-th iteration, respectively [39]:

$$\tilde{Q}(\boldsymbol{s},a)=\begin{cases} q^i & \text{if } (\boldsymbol{s},a)=(\boldsymbol{s}^i,a^i) \text{ for some } i, \\ \sum_{i=1}^{m}\left(\left({q^i}/{(\delta_v^i)^\lambda}\right)\Big/\left(\sum_{j=1}^{m}{1}/{(\delta_v^j)^\lambda}\right)\right) & \text{otherwise} \end{cases} \tag{7}$$

where q^i is the conclusion of the i-th ($i \in [1,m]$) rule, $(\boldsymbol{s},a)$ is the observation, λ is the Shepard parameter, the m is the number of rules and the δ_v^i is the scaled distance [20] between the $(\mathbf{s},\mathrm{a})$ observation and the antecedent of the i-th rule $(\boldsymbol{s^i},a^i)$:

$$\delta_v^i=\delta_v\left((\boldsymbol{s},a),(\boldsymbol{s^i},a^i)\right)=\left[\sum_{j=1}^{n}\left(\int_{s_j^i}^{s_j}v_j(s_j)ds_j\right)^2+\left(\int_{a^i}^{a}v(a)da\right)^2\right]^{1/2} \tag{8}$$

where $(\boldsymbol{s},a)$ is the state-action observation, $(\boldsymbol{s^i},a^i)$ state-action antecedent of the i-th rule, s_j is the j-th $(j \in [1,n])$ dimension of the n-dimensional state space universe, s_j^i is the j-th state dimension of the i-th rule, a^i is the action universe of the i-th rule, $v_j(s_j)$ is the scaling function of the s_j state universe, and the $v(a)$ is the scaling function of the U action universe. The fuzzy sets are described by the scaling functions of the corresponding universes [18], [20].

The incremental rule-base construction phase (the learning process) ends when no new incremental rules are added to the rule-base and the Q-update value becomes relatively low [30], [40]. After the learning phase, the size of the incrementally constructed rule-base can be reduced using offline rule-base reduction strategies [32], [37], [38]. These reduction methods [32], [37], [38] aim to improve the efficiency of the FRIQ-learning (and HFRIQ-learning) by eliminating the redundant rules from the gained Q-function rule-base.

3 The Knowledge Base Optimization

The HFRIQ-learning (and the FRIQ-learning) tunes (updates) the consequents (Q-values) of the fuzzy rules (knowledge base) based on the (4) update rule. However, the antecedent part of the newly added rules remain unchanged throughout the entire learning process. The antecedents of the newly added rules are created at the state-action point of the current observation. Consequently, the rules can be located at any state-action point. In case if any of the expert-defined production rules assign an incorrect action value to a state (i.e. the expert rule-base is only partially correct), the antecedent of the transformed expert rules will also be placed to an incorrect state-action antecedent position.

The partially correct (or completely incorrect) expert rule-base can have a negative impact on the efficiency of the learning process [29], [31]. For correcting the incorrectly defined rules, in this paper a tuning method is suggested, which can optimize the rules by aligning the antecedents to the correct state-action positions. This way, both the consequent part (as introduced in Chapter II.) and the antecedent part of the rules can be adjusted and corrected.

The improved HFRIQ-learning [34] algorithm, suggested in this paper, built upon the following steps:

- During the learning phase (incremental rule base construction), the initial rule-base (the corner and the transformed expert rules) is expanded with newly added rules by the system
- If there is no existing rule at the given state-action rule point, and even the closest rule is considered to be far then the system places a new rule at the current observation position.
    - The closeness is determined based on the proximity measure between rules and an allowed minimum rule distance [33] (which defines when two rules are considered close to each other)
    - The calculation of the distance between rules is based on the distance in each antecedent dimension [33]
- If there is an existing rule close to the observed state-action point, the rules in the rule-base are fine-tuned. The tuning is based on a gradient-based optimization of the rule position (antecedent and consequent) update according to the gradient of the Q-function
- In case if two rules are getting close to each other (due to the rule position change), they are merged into a single rule, to reduce the size of the rule-base during the learning phase

3.1 The Gradient Descent-based Rule-base Optimization

The main idea of the suggested rule-base optimization is that, in cases if an existing rule is found close to the observation, instead of a new rule insertion, the antecedent and consequent parts of the close rules are tuned. The applied optimization technique is the gradient descent (GD) method. The gradient descent method iteratively updates the antecedents and consequents of the fuzzy rules based on the gradient of the approximated Q-function with respect to partial derivatives. By iteratively updating the parameters according to the gradient of the Q-function, the algorithm can fine-tune the rules to align the state-action antecedent positions. For each rule, the gradient descent method calculates the partial derivatives of the Q-function with respect to the antecedent (states and action) and consequent (Q-value). These derivatives provide the direction of the changes for each parameter to minimize the error. In this case, the error can be considered the TD-error (Temporal

Difference error), like in case of the „Deep Q-learning Network" (DQN) method [12]:

$$TDerror = g(\boldsymbol{s}, a, \boldsymbol{s}') + \gamma * \max_{a' \epsilon U} \tilde{Q}^k(\boldsymbol{s}', a') - \tilde{Q}^k(\boldsymbol{s}, a) \tag{9}$$

The MSE (Mean Squared Error, which aims to minimize) applied in the gradient descent can be calculated as follows:

$$MSE = \frac{1}{\hat{m}} \sum_{i=1}^{\hat{m}} (TDerror)^2 \tag{10}$$

The tuning of the antecedents and the consequents of the fuzzy rules based on the gradient descent updating rule:

$$x_{k+1} = x_k - \nabla F(x_k) * \alpha \tag{11}$$

where x_{k+1} is the new value in the $k+1$-th iteration, x_k is the old value in the k-th iteration, α is the learning rate, F is the function and $\nabla F(x_k)$ is the gradient of the F function at the x_k. The $\nabla F(x_k)$ partial derivatives can be determined according to the chain rule, as it can be expressed as follows:

$$\nabla F(x_k) = \frac{\partial MSE(x_k)}{\partial x_k} = \frac{\partial (TDerror)^2}{\partial x_k} = 2 * TDerror * \frac{\partial \tilde{Q}^k(\boldsymbol{s}, a)}{\partial x_k} \tag{12}$$

Substituting $\nabla F(x_k)$ into (11) according to (12), the x_{k+1} can be expressed as follows:

$$x_{k+1} = x_k - \left(2 * TDerror * \frac{\partial \tilde{Q}^k(\boldsymbol{s}, a)}{\partial x_k}\right) * \alpha \tag{13}$$

The update rule (13), applied to the partial derivatives of the $\tilde{Q}^k(\boldsymbol{s}, a)$ for each $\boldsymbol{s}, a$ and q variables, results in the new $\boldsymbol{s}_{k+1}$ state, the new a_{k+1} action, and the new q_{k+1} consequent values, which can be calculated as follows:

$$\boldsymbol{s}_{k+1} = \boldsymbol{s}_k - \left(2 * TDerror * \frac{\partial \tilde{Q}(\boldsymbol{s}, a)}{\partial \boldsymbol{s}}\right) * \alpha \tag{14}$$

$$a_{k+1} = a_k - \left(2 * TDerror * \frac{\partial \tilde{Q}(\boldsymbol{s}, a)}{\partial a}\right) * \alpha \tag{15}$$

$$q_{k+1} = q_k - \left(2 * TDerror * \frac{\partial \tilde{Q}(\boldsymbol{s}, a)}{\partial q}\right) * \alpha \tag{16}$$

3.1.1 Selecting the Rules to be Tuned

In the suggested optimization process, it is necessary to identify which rules of the fuzzy rule-base should be selected for optimization. During the learning process, the rules are tuned if the Q-update value is considered significant and if there is an existing rule already close to the state-action observation. For rule base tuning different versions of the gradient method [16] can be applied, one well-known

approach is the gradient descent method, which considers all data points during each iteration to calculate the error function and derivatives. Another variation, called Stochastic Gradient Descent (SGD), randomly selects data points for determining gradients and error functions in each iteration. In case of fuzzy rule base optimization, the sample points can be considered fuzzy rule points and the goal of the minimization of the TD-error (9). In the case if all the fuzzy rule points are tuned in each iteration step, the convergence of the approximated Q-function could be instable. The reason of this instability is inherited from the exploration-exploitation manner as the HFRIQ-learning discovers the Q values the state-action space. The state-action values have higher approximated Q value gets more trials, than which have lower Q values. Hence, those fuzzy rule points in the state-action space, which are rarely explored and updated are deteriorated by the updates of the frequently explored regions, if all the fuzzy rule points are updated in each iteration step. Consequently, simultaneous tuning of all the fuzzy rule points is not feasible. A possible solution is that in each iteration step only the fuzzy rules close to the observed state-action point are selected for tuning [33]. This method stabilizes the Q-function, updating the fuzzy rule points of the Q-function in the correct direction, without changing the rules that are far from the state-action observation point, where the update is required.

3.2 The Distance-based Rule Base Reduction

During the learning phase, as a result of the previously introduced rule-base tuning, the state-action positions (antecedents) of the rules can move in the state-action space (not only their consequents are changing). This can lead to situations, when two or more rules become close to each other. The close rules have similar antecedents and consequents, thus they describe similar information. By merging similar rules, the size of the rule-base can be reduced. To prevent redundancy and avoid close rules, a distance-based rule-base reduction technique is proposed for identifying and merging similar rules during the learning phase.

The merging of the rules is based on a distance threshold called $\boldsymbol{dtr}$ [33]. The $\boldsymbol{dtr}$ is a vector that contains the distance thresholds for each state, action, and Q-value dimensions:

$$\boldsymbol{dtr} = [dtr_1, dtr_2, \dots, dtr_n, dtrU, dtrQ] \tag{17}$$

Where $dtr_1, dtr_2, \dots, dtr_n$ are the distance thresholds for the states (n is number of the state variables), $dtrU$ is the distance thresholds for the action and $dtrQ$ is the distance thresholds for the Q-value dimension. To determine the distance threshold values for states, action, and Q-value dimensions, a fraction value of their universes, referred as the dR distance rate, is applied. The dR is a constant, determined by an expert and can be different for each antecedent (states, actions) and consequent (Q-value) universes. In other words, the distance threshold values represent a portion of the length of each dimension:

$$dtr_j = \frac{length(S_j)}{\mathrm{dR}_{\boldsymbol{S}}}, \qquad j \in [1, n] \tag{18}$$

$$dtrU = \frac{length(U)}{\mathrm{dR}_U}, \qquad j = n + 1 \tag{19}$$

$$dtrQ = \frac{length(Q)}{dR_q} \tag{20}$$

Where $\mathrm{dR}_{\boldsymbol{S}}$, dR_U and dR_q are the distance rates for the states ($\boldsymbol{S}$), action (a) and Q-value (q) universe, n is the number of the state dimensions and j is the running variable of the n. The length of the universes is determined as the follows:

$$length(S_j) = \left|\max(S_j) - \min(S_j)\right|, \qquad j \in [1, n] \tag{21}$$

$$length(U) = |\max(U) - \min(U)|, \tag{22}$$

$$length(Q) = |\max(Q) - \min(Q)| \tag{23}$$

Where $\max(S_j)$ is the maximum (largest) and $\min(S_j)$ is the minimum (smallest) element of the j-th ($j \in [1, n]$) S_j state and U action universe.

The $\boldsymbol{dtr}$ distance thresholds serve as criteria for determining when rules are close enough to be considered similar and eligible for merging during the learning phase. Two rules can be considered close if distance between them calculated in each universe is less than the corresponding distance thresholds in all universes:

$$\underset{t,p\in[1,m+\hat{m}]}{\exists}\, t,p \;\; \underset{j\in[1,n+1]}{\forall} \left(d_j(t,p) < dtr_j\right) \text{ and } \left(d_Q(t,p) < dtrQ\right) \tag{24}$$

where $dtr_j, dtrQ$ ($j \in [1, n+1]$) are the distance thresholds, n is the number of the antecedent universes, $t, p \in [1, m + \hat{m}]$ are the indexes of two rules in the $m + \hat{m}$ sized rule-base, $d_j(t,p)$ is the distance between the rules indexed by t and p in the j-th antecedent universe, and $d_Q(t,p)$ is the distance between the rules indexed by t and p in the consequent (Q-value) universe. The distance between two rules indexed by t and p in the rule-base can be determined as follows:

$$d_j(t,p) = \left|s_j^t - s_j^p\right|, \qquad j \in [1, n] \tag{25}$$

$$d_j(t,p) = |a^t - a^p|, \qquad j = n + 1 \tag{26}$$

$$d_Q(t,p) = |Q^t - Q^p| \tag{27}$$

where s_j^t is the j-th ($j \in [1, n]$) antecedent fuzzy set of the rule indexed by t, s_j^p is the j-th antecedent fuzzy set of the rule indexed by p, a^t is the action of the rule index by t, a^p is the action of the rule indexed by p, Q^t is the consequent of the rule indexed by t, Q^p is the consequent of the rule indexed by p, and $||$ denotes the absolute value. Consequently, the distance between the t-th and p-th indexed rules is not represented by a single value but rather as a vector with multiple elements. This vector includes distance values for each antecedent and consequent dimension. The following figure illustrates the distances between the r_t and the r_p rules:

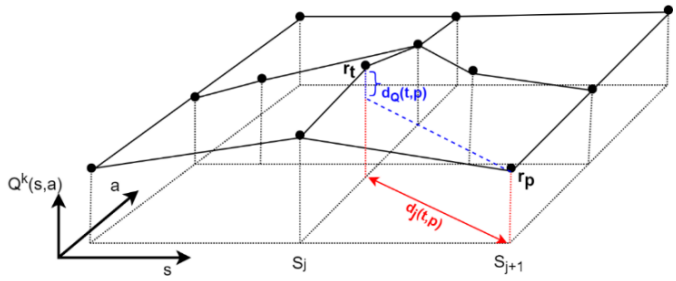

Figure 1
The distances between the r_t and the r_p rules:
The $d_j(t,p)$ are the distances in the antecedent universes and $d_Q(t,p)$ is the distance in the consequent universe between the p and t indexed r rules

3.2.1 Determination the Type of the New Merged Rule

In the HFRIQ-learning system 3 types of fuzzy rules are distinguished: the expert rules $(\hat{r}_i \in R_{expert})$, the rules crated by the system $(r_i \in R)$ and the corner rules $(r^{\sqcap} \in R)$. Based on the type of the source rules (expert, corner or new) the type of the new merged rule must be determined. Since the rules (knowledge) defined by the expert presumably describe correct knowledge about final solution, they are considered with greater importance. This importance determines the type of the new merged rule and also controls if the source rules are merged or not.

In the case if two rules are close to each other, the rule merging, suggested in this paper, is performed as follows:

- If one of the two source rules is an expert type and the other is a system-generated new rule, then the merged rule will be considered an expert type rule.
- If both of the two source rules are expert type rules, the new merged rule will also be an expert type rule.
- If both of the two source rules are newly inserted rules by the system, then the new merged rule will be marked as a newly inserted rule.
- It is a special case, when one of the two source rules is corner type rule. The importance of the corner rules is greater in the process of rule base construction due to the interpolation manner of the applied FRI "FIVE" method. Therefore, in this case, no rule merging is performed, and the antecedent of the corner rule is not modified.

Equation (28) summarizes the type of the r_{red} merged rule that is created after merging the source rules. The "⊔" operator denotes the merging of the two source rules, and the "→" operator indicates the result of the rule merging:

$$\underline{r_t \qquad r_p \qquad r_{red}} \tag{28}$$

x	$\|\|$	$\hat{x}$	$\rightarrow$	$\hat{x}$
$\hat{x}$	$\|\|$	$\hat{x}$	$\rightarrow$	$\hat{x}$
x	$\|\|$	x	$\rightarrow$	x
$x^{\square}$	$\|\|$	x	$\rightarrow$	$x^{\square}$
$x^{\square}$	$\|\|$	$\hat{x}$	$\rightarrow$	$x^{\square}\ \hat{x}$

All the rules in the system are marked by a unique identifier for tracking the movement of the rules during the optimization. This enables the comparison between the expert knowledge base obtained after the tuning process and the expert knowledge base defined before the learning process.

4 Application Example

The efficiency of the proposed improved HFRIQ-learning system is investigated using a classic reinforcement learning benchmark example, the "Mountain car" simulation scenario.

In this application example, the agent is a car, and its environment is a steep valley. The car is positioned in the middle of the steep valley at the start of the learning process. The goal of the agent is to navigate from the middle of the steep valley to the hilltop located at the top of the valley. In this example, the state space is described by two variables and the action space has one variable:

- position of the car: $s_1 \in [-1.5,0.5]$
- velocity of the car: $s_2 \in [-0.07,0.07]$
- movement (right, left, neutral) of the car: $a \in [-1,0,1]$

The reward function was kept simple, the system gives 1000 immediate reward if the agent reaches the hilltop, otherwise the immediate reward is -10. The parameters of the improved HFRIQ-learning are the following:

- $\alpha = 0.5$
- $\gamma = 0.99$
- Number of iterations in an episode: 2000
- Gradient method $\alpha = 0.01$
- Insertion of a new rule, the values of the dR parameters, which determine the minimum distance between rules, are defined:
    - $\mathrm{dR}_S = \mathrm{dR}_U = 40$
- The values of the dR parameters for the distance-based rule-base reduction method applied during the learning process:
    - $\mathrm{dR}_S = 15, \mathrm{dR}_U = 15, \mathrm{dR}_q = 100$

To investigate the effectiveness of the proposed knowledge base optimization in the improved HFRIQ-learning, three different run cases were defined:

I. Original FRIQ-learning [39], without expert knowledge base

II. The previous (I. case), enhanced with the expert knowledge base, without the proposed knowledge base optimization method (original HFRIQ-learning, introduced in [34]). This run case contains four additional cases depending on the type of the injected expert rule-base:

   a) correctly defined expert rule base

   b) fragment of the correctly defined (a.) expert rule base

   c) partially incorrect defined expert rule base

   d) "randomly" generated expert rule base

III. The improved HFRIQ-learning case, enhanced with the expert knowledge base applied the proposed knowledge base optimization (3.1 chapter) and the distance-based rule base reduction (3.2 chapter) methods. This run still contains four additional cases (a.-d.) depending on the type of expert rule-base introduced in the II. case.

The correctly defined expert rule-base (III.a.) is obtained after applying the rule-base reduction method III. introduced in [38], to the complete rule base with 110 rules. This correct expert rule-base (after the reduction) contains 17 state-action rules, having correct action in their states. The fragment of the correct expert rules (required for II.b. and III.b.) were randomly selected 10 rules from the 17 correct expert rules. The resulting expert rule-base is still correct but smaller (10 rules) than the previous case (17 rules). The partially incorrect expert rule-base (II.c. and III.c.) is obtained by modifying the action of 6 rules from the correct (17 rules) expert rule-base. The modified 6 rules can be considered incorrect, because they involve modified and thus incorrect actions for their state points. In the last case (II.d. and III.d.), the expert rule-base contains 17 rules having randomly generated states and actions.

The comparison of the different run cases (I.-III.) is based on the convergence speed (the number of episodes required for learning) and the size of the knowledge base (number of the fuzzy rules in the rule base).

The results are summarized in following table:

Table 1

The average results in different run cases

Run case	Convergence speed (number of episodes)	Size of the rule-base (number of rules)
I.	29	110
II.a.	10	124.3
II.b.	10.4	114.3

II.c.	11.7	120.1
II.d.	26.6	124.4
III.a.	1	79
III.b.	9	81
III.c.	20	88
III.d.	37	86

In the case (III.a.), where the simulation ran with the correctly defined expert rule-base, the system significantly converged faster (within a single episode) than the original FRIQ-learning version, and the number of rules decreased from 110 to 79. This is because the system did not need to tune the expert knowledge base (it was correct), and the number of rules was also reduced during the learning process by the suggested distance-based rule-base reduction method. In case having the fragment of the correctly defined expert rules (III.b.), the system converged in 9 episodes with 81 rules, requiring slightly more episodes than the previous run. When the simulation ran with partially incorrect expert rules (III.c. having some incorrect rules in the expert knowledge base) the proposed method needed 20 episodes to correct the incorrectly defined expert rule-base. In the last case, when a completely incorrect initial knowledge base was injected into the learning system (III.d.), the system still converged, but it needed 37 episodes (with 2000 iterations per episode) to tune (modify) the incorrect expert rules.

Based on the results, it can be concluded that the convergence speed of the learning process (and the size of the final rule-base) is significantly influenced by the quality of the injected expert defined state-action rules. The reason for this is that in the case of an incorrect (or partially incorrect) expert rule-base, the incorrect rules are also needed to be corrected (tuned).

The expert defined dR distance thresholds applied for rule creation and rule base reduction have also effect on the performance. Lower values increase the number of the rules, higher values can ruin the model convergence.

Before learning and after the learning phase results for the worst-case scenario (III.d. when a randomly generated expert rule-base is injected) of the proposed improved HFRIQ-learning are introduced on (Table 2) and (Table 3).

Table 2

The “randomly” generated (incorrect) expert rules before the learning

R#	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
s1	-0.475	-0.5	-0.475	-0.475	-0.27	-0.27	-0.27	-0.475	-0.475
s2	0	0	-0.014	0.014	0	-0.014	0	-0.042	0
a	1	-1	-1	0	-1	0	-1	1	1

R#	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**
s1	-0.475	-0.065	0.14	-0.27	-0.885	0.885	-0.065	-1.09

s2	0	0	-0.014	-0.042	0.042	0.042	0.042	0.042
a	-1	0	1	-1	-1	1	0	-1

Table 3

The "randomly" generated (incorrect) expert rules after the learning: the tuned expert rule-base

R#	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
s1							-0.52		-0.39
s2	x	x	x	x	x	x	-0.04	x	0.05
a							-1		1

R#	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**
s1		-0.21	-0.47	-0.31		0.885		-0.81
s2	x	-0.03	-0.016	-0.03	x	0.042	x	0.03
a		0	1	-1		1		-1

Applying the proposed knowledge base optimization (3.1) and the distance-based rule-base reduction (3.2) methods, only 7 (see Table 3) out of the 17 randomly generated expert rules (see Table 2) are retained and 6 of them were significantly tuned. Only expert rule (rule no 15th) remained unchanged. Therefore, it can be concluded that from the randomly generated expert rule-base, only the 15th rule could be considered correctly defined. The expert rules marked by "x" on Table 2 were removed during the learning, by merging them to another expert rules during the proposed rule base reduction. These removed rules can be considered redundant rules. Based on simulation, it can be concluded that the proposed rule base tuning and the rule base reduction methods in the improved HFRIQ-learning allows the tuning (fixing) of the injected expert knowledge base in cases where it contains incorrect or inaccurate information about the final solution.

Conclusions

For improving the performance of the HFRIQ-learning (Heuristically Accelerated Fuzzy Rule-Interpolation based Q-learning), in this paper a gradient descent-based rule base optimization and a distance-based rule base reduction method is suggested. The proposed methods are capable of tuning (optimizing) the injected external expert knowledge base even if they are describing inaccurate information. Furthermore, the convergence speed of the HFRIQ-learning system can be improved, but only in cases where the expert heuristic is at least partially correct. Otherwise, when incorrect expert rules are injected, the system still converges, but more episodes (and thus more iterations) are needed for tuning (fixing) the rules of the rule-base. The HFRIQ-learning can also be valuable for optimizing models with a sparse fuzzy rule-base and FRI reasoning, such as ethologically inspired robot behavior models [19] or other FRIQ-learning benchmark applications [35] [41].

The further research will focus on developing an expert knowledge base validation method, which can be suitable for comparing the initial expert rules with the optimized expert rules obtained after the learning process, providing information

about the accuracy of the initial expert knowledge base. For formalized representation of the initial state-action expert fuzzy rule base, the Fuzzy Behavior Description Language (FBDL) [24] could be also applied to define the expert rules in a human-readable form.

Acknowledgement

SUPPORTED BY THE ÚNKP-23-4-I-ME/5 NEW NATIONAL EXCELLENCE PROGRAM OF THE MINISTRY FOR CULTURE AND INNOVATION FROM THE SOURCE OF THE NATIONAL RESEARCH, DEVELOPMENT AND INNOVATION FUND.

The authors wish to thank the support of the Hungarian Research Fund (OTKA K143595).

References

[1] Annabestani, M., Abedi, A., Nematollahi, M. R., & Sis-tani, M. B. N. (2021) A new soft computing method for integration of expert's knowledge in reinforcement learning problems. arXiv preprint arXiv:2106.07088

[2] Appl, M.: Model-based Reinforcement Learning in Continuous Environments. Ph.D. thesis, Technical University of München, München, Germany, dissertation.de, Verlag im Internet (2000)

[3] Arumugam, D., Lee, J. K., Saskin, S., & Littman, M. L. (2019) Deep reinforcement learning from policy-dependent human feedback. arXiv preprint arXiv:1902.04257

[4] Bartók, Roland, and József Vásárhelyi. "Design of a FPGA accelerator for the FIVE fuzzy interpolation method." International Journal of Computer Applications in Technology 68.4 (2022): 321-331

[5] Bartók, Roland, and József Vásárhelyi. "Examining Cache Handling of the FIVE Method on Multicore Systems." 2019 IEEE 17th World Symposium on Applied Machine Intelligence and Informatics (SAMI) IEEE, 2019

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

[7] Bianchi, Reinaldo AC, Carlos HC Ribeiro, and Anna Helena Reali Costa. "Heuristically Accelerated Reinforcement Learning: Theoretical and Experimental Results." ECAI. 2012

[8] Bianchi, Reinaldo AC, Carlos HC Ribeiro, and Anna Helena Reali Costa. "Heuristically Accelerated Reinforcement Learning: Theoretical and Experimental Results." ECAI. 2012

[9] Bianchi, Reinaldo AC, Carlos HC Ribeiro, and Anna HR Costa. "Accelerating autonomous learning by using heuristic selection of actions." Journal of Heuristics 14.2 (2008): 135-168

[10] Bonaccorso, Giuseppe. Machine learning algorithms. Packt Publishing Ltd, 2017

[11] Bonarini, A.: 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

[12] Brunton, Steven L., and J. Nathan Kutz. Data-driven science and engineering: Machine learning, dynamical systems, and control. Cambridge University Press, 2019

[13] Chai, Y., Ge, J., Chai, Y., Wang, X., & Zhao, B. (2021, October) Xtuning: Expert database tuning system based on reinforcement learning. In International Conference on Web Information Systems Engineering (pp. 101-110) Cham: Springer International Publishing

[14] Deng, C., Ji, X., Rainey, C., Zhang, J., & Lu, W. (2020) Integrating machine learning with human knowledge. Iscience, 23(11)

[15] Fan, Jianqing, et al. "A theoretical analysis of deep Q-learning." Learning for Dynamics and Control. PMLR, 2020

[16] Haji, Saad Hikmat, and Adnan Mohsin Abdulazeez. "Comparison of optimization techniques based on gradient descent algorithm: A review." PalArch's Journal of Archaeology of Egypt/Egyptology 18.4 (2021): 2715-2743

[17] Johanyák, Z. C., & Kovács, S. (2006) A brief survey and comparison on various interpolation based fuzzy reasoning methods. Acta Polytechnica Hungarica, 3(1) 91-105

[18] Klawonn, F.: Fuzzy Sets and Vague Environments, in Fuzzy Sets and Systems, Vol. 66, 1994, pp. 207-221

[19] Kovács, S., Vincze, D., Gácsi, M., Miklósi, Á., & Korondi, P. (2011, May) Ethologically inspired robot behavior implementation. In 2011 4th International Conference on Human System Interactions, HSI 2011 (pp. 64-69) IEEE

[20] Kovács, Szilveszter. "Extending the fuzzy rule interpolation" FIVE" by fuzzy observation." Computational Intelligence, Theory and Applications. Springer, Berlin, Heidelberg, 2006. 485-497

[21] Kovacs, Szilveszter. "Fuzzy Rule Interpolation in Practice." SCIS & ISIS SCIS & ISIS 2006, Japan Society for Fuzzy Theory and Intelligent Informatics, 2006

[22] Li, F., Shang, C., Li, Y., Yang, J., & Shen, Q. (2021) Approximate reasoning with fuzzy rule interpolation: background and recent advances. Artificial Intelligence Review, 54(6), 4543-4590

[23] Mitchell, T. M., & Mitchell, T. M. (1997) Machine learning (Vol. 1, No. 9) New York: McGraw-hill

[24] Piller, Imre, and Szilveszter Kovács. "FBDL: A Declarative Language for Interpolative Fuzzy Behavior Modeling." 2019 IEEE 23rd International Conference on Intelligent Engineering Systems (INES) IEEE, 2019

[25] Russell Stuart, J., & Norvig, P. (2009) Artificial intelligence: a modern approach. Prentice Hall

[26] Sutton, R. S., Barto, A. G.: Reinforcement Learning: An Introduction, MIT Press, Cambridge (1998)

[27] Tan, Ming. "Multi-agent reinforcement learning: Independent vs. cooperative agents." Proceedings of the tenth international conference on machine learning. 1993

[28] Tato, A., & Nkambou, R. (2022) Infusing Expert Knowledge Into a Deep Neural Network Using Attention Mechanism for Personalized Learning Environments. Frontiers in Artificial Intelligence, 5, 921476

[29] Tompa, T., Kovács, S., Vincze, D., & Niitsuma, M. (2021, January) Demonstration of expert knowledge injection in Fuzzy Rule Interpolation based Q-learning. In 2021 IEEE/SICE International Symposium on System Integration (SII) (pp. 843-844) IEEE

[30] Tompa, Tamás, and Szilveszter Kovács. "Applying Expert Heuristic as an a Priori Knowledge for FRIQ-Learning." Acta Polytechnica Hungarica 17.4 (2020)

[31] Tompa, Tamás, and Szilveszter Kovács. "Benchmark example for the Heuristically accelerated FRIQ-learning." 2023 24th International Carpathian Control Conference (ICCC) IEEE, 2023

[32] Tompa, Tamás, and Szilveszter Kovács. "Clustering-based fuzzy knowledgebase reduction in the FRIQ-learning." 2017 IEEE 15th International Symposium on Applied Machine Intelligence and Informatics (SAMI) IEEE, 2017

[33] Tompa, Tamás, and Szilveszter Kovács. "Determining the minimally allowed rule-distance for the incremental rule-base contruction phase of the FRIQ-learning." 2018 19th International Carpathian Control Conference (ICCC) IEEE, 2018

[34] Tompa, Tamás, and Szilveszter Kovács. "Heuristically accelerated FRIQ-learning." 20th Jubilee International Symposium on Intelligent Systems and Informatics (SISY 2022) IEEE, 2022

[35] Tompa, Tamás, Dávid Vincze, and Szilveszter Kovács. "The Pong game implementation with the FRIQ-learning reinforcement learning algorithm." Proceedings of the 2015 16th International Carpathian Control Conference (ICCC) IEEE, 2015

[36] Torrey, Lisa, and Jude Shavlik. "Transfer learning." Handbook of research on machine learning applications and trends: algorithms, methods, and techniques. IGI global, 2010, 242-264

[37] Vincze, D., Kovács, Sz.: Reduced Rule Base in Fuzzy Rule Interpolation-based Q-learning, 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

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

[39] Vincze, Dávid, and Szilveszter Kovács. "Fuzzy rule interpolation-based Q-learning." 2009 5th International Symposium on Applied Computational Intelligence and Informatics. IEEE, 2009

[40] Vincze, Dávid, and Szilveszter Kovács. "Incremental rule base creation with fuzzy rule interpolation-based Q-learning." Computational Intelligence in Engineering. Springer, Berlin, Heidelberg, 2010, 191-203

[41] Vincze, Dávid. "Fuzzy rule interpolation and reinforcement learning." 2017 IEEE 15th International Symposium on Applied Machine Intelligence and Informatics (SAMI) IEEE, 2017

[42] Watkins, C. J. C. H., Dayan, P.: Q-learning, in Machine Learning, Vol. 8 (3/4) 1992, pp. 279-292

[43] Wu, X., Xiao, L., Sun, Y., Zhang, J., Ma, T., & He, L. (2022) A survey of human-in-the-loop for machine learning. Future Generation Computer Systems, 135, 364-381

[44] Z. Wei, W. Zhang, J. Chen and Z. Yang, "Expert knowledge based multi-agent reinforcement learning and its application in multi-robot hunting problem," 2018 Chinese Control And Decision Conference (CCDC), Shenyang, China, 2018, pp. 2687-2692, doi: 10.1109/CCDC.2018.8407581

[45] Zhang, J., Liu, Y., Zhou, K., Li, G., Xiao, Z., Cheng, B., ... & Li, Z. (2019, June) An end-to-end automatic cloud database tuning system using deep reinforcement learning. In Proceedings of the 2019 International Conference on Management of Data (pp. 415-432)