Fuzzy Behavior Description Language: A Declarative Language for Interpolative Behavior Modeling

Imre Piller, Szilveszter Kovács

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

Abstract: The behavior-based system (BBS) is a hierarchical structure built upon behavior components, behavior coordination and behavior fusion. The goal of this paper, is to recall the concept of the interpolative fuzzy behavior-based system and to introduce a declarative language especially designed for supporting its implementation and configuration into embedded applications. The suggested Fuzzy Behavior Description Language (FBDL) aids the definition of fuzzy rule-based systems and their connections to form behavior components and behavior coordination as fuzzy state-machines. The suggested language also assists the fuzzy rule definition with variable consequent, to help the creation of behavior fusion functions. For simplifying the definition of hierarchical rule-bases, the structure of rule-base dominancy is also introduced in the FBDL. According to the suggested embedded application concept, the FBDL code, as a parameter configuration, can directly "run" on a built in fuzzy state machine controller, called "FRI Behavior Engine". This case the behavior of the agent controlled by the FRI Behavior Engine, can be directly modified by changing the FBDL code, without reprogramming other parts of the agent controller software.

Keywords: Behavior Based Control; Fuzzy Rule Interpolation; Fuzzy State Machine; Declarative Language; FBDL: Fuzzy Behavior Description Language

1 Introduction

The behavior-based system (BBS) is a hierarchical structure built upon independent and parallel behavior components, and a behavior coordination, which can determine the usefulness (weights) of the behavior components in handling a given situation. The task of the behavior fusion is the combination of the behavior component actions to form the control action of the BBS. A behavior component could be a reactive (stateless) function, e.g. in case of a fuzzy BBS, a fuzzy rule-based system, or a compound behavior (a whole BBS itself). In common sense the function of the behavior fusion could be a convex combination

of the behavior component actions according to the corresponding component weights, but practically to be able to handle alternative, or contradictive component actions, it could be a (fuzzy) rule-based system again. The role of the behavior coordination is the situation awareness, the determination, which behavior component action with what level is required in handling a given situation. Considering the BBS to be a model based reflexive agent, the behavior coordination should have states, i.e. in case of an interpolative fuzzy BBS, it must be an interpolative fuzzy state machine.

There are numerous adaptations of the BBS concept [1] [2]. The suggested Fuzzy Behavior Description Language (FBDL) follows the fuzzy rule-based systems, fuzzy rule interpolation (FRI) [12] and fuzzy state machines based adaptation [3] [4] [5] [6] [7] [8]. See for example Fig. 1 (see Section 3 for notation in details).

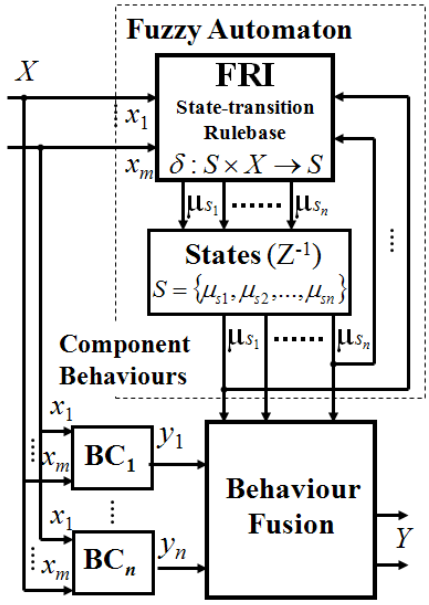

Figure 1

The adapted Fuzzy Rule Interpolation (FRI) based BBS structure with Fuzzy Automaton acting as Behavior Coordination, Behavior Components (BC) and Behavior Fusion

The goal of this paper is to introduce a declarative language especially designed for supporting embedded behavior-based applications, by offering a common framework for defining a BBS (the behavior components, the behavior coordination and the behavior fusion) as an interpolative fuzzy rule-based system, as Fuzzy Rule Interpolation based BBS (FRI BBS).

The interpolative fuzzy rule-based knowledge representation is an important issue in the suggested methodology. The rule-based structure makes the knowledge representation human readable and self-explanatory. The fuzziness, and its

"Linguistic Term" fuzzy set concept even strengthen the human readability in case of variables defined on continuous universes. The applied Fuzzy Rule Interpolation (FRI) reasoning methodology, simplifies the fuzzy rule-base definition by relaxing some constraints of the fuzzy rule-base, the FRI can handle sparse fuzzy rule-bases too [32].

The unified rule-based human readable system construction FRI BBS framework makes the suggested FBDL to be a suitable platform to support the implementation of human (expert) knowledge to a working system. The FBDL code can "run" on a system directly or, having some additional observed data, can serve as an object for machine learning parameter optimization methods. Some examples for FRI BBS applications are appearing in ethological model based human-robot interaction applications [9] [10] [11] and Ethorobotics [33], e.g. for expressing human readable emotions [34] for robots. In case of ethological models, the expert's knowledge is based on real animal observations and represented as a descriptive verbal model built upon a series of facts and action-reaction rules. Moreover, the verbal models can be incomplete, or contain some expert domain specific implicit knowledge, which is missing from the verbal description, e.g. as "well-known" facts. These requirements fit well the fuzzy rule-based knowledge representation and the FRI reasoning of the suggested FBDL description. See the concept of FRI with some application examples more detailed in [12].

For creating the FBDL the goal was forming a language which supports function definition by linguistic rules similar like fuzzy systems. This case the fuzzy rules act like "fuzzy points" of a fuzzy function and the fuzzy reasoning method acts as a fuzzy function definition. In classical fuzzy reasoning, the fuzzy rule-base has to be complete (i.e. they need fully defined rule-base (e.g. the Zadeh-Mamdani-Larsen Compositional Rule of Inference (CRI) (Zadeh [13]) (Mamdani [14]) (Larsen [15]) or the Takagi-Sugeno fuzzy inference (Sugeno [16], Takagi-Sugeno [17])). For releasing this condition, to be able to handle sparse fuzzy rule-base (where not all the possible rules are defined), the concept of the FRI was adapted. In case of FRI, the fuzzy reasoning method is a fuzzy interpolation, where the fuzzy rules are the fuzzy node points of the fuzzy interpolating function. (An axiomatic approach of the fuzzy interpolation can be found in [18]).

There are numerous FRI methods that exist in the literature. For the FRI Behavior Engine implementation any of them is adaptable, which can handle multidimensional antecedent spaces. For the current implementation [31], because of its simplicity, the FRI "FIVE" [19] [20] [21] [22] were adapted.

2 Related Works

The current work is the continuation of [23]. The Fuzzy Behavior Description Language and the FRI Behavior Engine have been improved but the aims remained unchanged. A framework was developed, where an expert could define the behavior of an agent. The applied FBDL is technically a declarative programming language. Therefore, it is necessary to make some comparison with the available agent based modeling methods too. This section briefly summarizes some works related to other declarative behavior definition languages and modeling methods.

For dynamic declarative agent configuration in [24], the authors show a plan generation mechanism in a declarative manner. They emphasize the importance of adaptation of the agent. The approach is the same as the role of fuzzy interpolation in the suggested FRI BBS concept, i.e. the agent should be able to succeed in unseen situations.

Other authors recognize that the usage of a simplified programming toolchain for non-technical users only viable solution for a short term [25]. For avoiding the fast increment of software complexity they introduced a new programming paradigm (*Targets-Drives-Means*). Their construction is similar to the proposed FBDL behavior description method. The main difference is the usage of fuzzy reasoning and a dedicated description language which is interpreted by the FRI behavior engine directly.

Authors in [26] also suggests that the simplified imperative way is also viable for simple behaviors. They choose a component centric approach where the reasoning system tries to find a solution which match with the requirements of the designer and the runtime system.

For agent based modeling and simulation, the SESAM [27] provides a useful visual programming environment, as UML-like activity diagrams. On the other hand, for larger models, and parameter optimization, the textual description could be more concise and practical.

The comparison of the previously mentioned behavior description methods can be seen in the Table 1.

Table 1

Comparison of AgentSpeak [24], Target-Drives-Means (TDM) [25], Agent Based Modeling (ABM) [26] and SeSAmUML [27] description languages

	AgentSpeak	**TDM**	**ABM**	**SeSAmUML**	**FBDL**
representation	textual	textual	textual	visual	textual
base language	STRIPS	-	JSON	UML	-
paradigm	declarative	TDM	declarative	procedural	declarative
inference	layered planning	score calculation	backtracking	transition rules	fuzzy interpolation
hierarchy	STRIPS operators	priority control	subcomponents	rules	dominancy

The first aspect of the comparison is the representation method of the behavior. Most of the considered languages use textual representation. It is important to note that, the textual and visual representation are always interchangeable. This comparison focuses on the primary usage of the behavior description method. It means that the choice of visual and textual representation is depends on the preferred usage regardless the implementation difficulties.

The behavior description is a special kind of knowledge representation. Therefore, in many cases the behavior description language is a refined version of a general purpose language. Three of the mentioned languages are based on STRIPS, JSON and UML languages. The TDM has an own semi-graphical definition language, while the proposed FBDL language uses a simple, natural language-like syntax.

The paradigms behind the behavior description languages show many differences. The AgentSpeak is a declarative language where the user can define operators and their pre- and post-conditions. The TDM itself is a new programming paradigm, which organizes the behaviors to small, trigger activated behavior components. The ABM proposes a fully declarative model in the sense that the JSON-like description defines the required, higher level actions instead of low-level commands. The SeSAmUML follows the standard UML method for defining a final state machine. The FBDL defines the set of production rules which have organized to rule-bases. It is also a declarative approach which is similar to the AgentSpeak because the antecedent parts of the rules are similar to the preconditions of the standard STRIPS language.

A common difficulty of the behavior description systems is how they can resolve the conflicts and contradictions of the behavior descriptions. They have to use some kind of inference for calculating the most suitable action for the given situation. The AgentSpeak solves the problem by a layered planning approach. It tries to dynamically evaluate the lower and higher level plans. The TDM calculates score for any action of its behavior components. After, it can choose the appropriate action according to these values. The ABM tries to find a proper solution which fulfils all constraints by backtracking. For the unspecified properties, it uses interpolation. The SeSAmUML uses different abstraction for the actions and for the resolution of conflicting cases. It has specific rules for selecting and terminating activities. The FBDL solves the conflicting situations by fuzzy rule interpolation. It obtains the required action by interpolating the consequent values.

For simplification purposes, it is reasonable hierarchically organize or prioritize the actions of the behavior descriptions. The last aspect of the comparison considers the preferred way of this kind of hierarchy. In the AgentSpeak language, the domain expert can use STRIPS operators to express higher priority plans. The TDM contains priority management in its model. It assigns priorities for the different behaviors and prioritize the actions after the calculation of behavior scores. The ABM organizes the behavior description to a tree of components.

Each component has constraints, therefore, the component preference can be coded into this strict hierarchical structure. The SeSAmUML solves the conflict resolution problem on the level of rules. After an action has activated it does not terminate until the terminating rule has not fired. The FBDL organizes the rule-bases into a hierarchy which automatically denotes their priorities.

3 Fuzzy State Machine Model

Having FRI models in Behavior Component definitions and FRI state transition-rules in Behavior Coordination, most part of the BBS model forms a Fuzzy State Machine [8]. There are various understandings of the fuzzy state machine can be found in the literature (for summary see [28] [29]). Most of them are extending the classical finite state automaton by applying fuzziness for the state transitions, while the state remains discrete (crisp, one of the predefined ones). On the other hand, the fuzzy model suggested for the FRI Behavior Engine adapts the concept of "fuzzy state", where the state is a vector of membership values. According to the fuzzy state concept, the system could be in all its states in the same time, but with different membership levels. This view fits well the FRI BBS concept, as the actual fuzzy state can be easily interpreted as the usefulness (weights) of the behavior components in handling a given situation. For example, the fuzzy state of the behavior coordination can directly control the behavior fusion.

The fuzzy state machine adapted for the FRI Behavior Engine is an extended version of the Fuzzy Finite-state Automaton. It extends the finite set of input symbols to finite dimensional input values and the finite set of states to finite dimensional state values [8]. This case the fuzzy state machine can be defined by a tuple:

$$\tilde{F} = (S, X, \delta, P, Y, \omega) \tag{1}$$

where S is a finite n length of fuzzy states, $S = \{\mu_{s_1}, \mu_{s_2}, \dots, \mu_{s_n}\}$, μ_{s_i} is the membership value of the i^{th} dimension of the n dimensional fuzzy state. X is a finite m dimensional input vector, $X = \{x_1, x_2, \dots, x_m\}$. $P \in S$ is the fuzzy initial state of $\tilde{F}$. Y is a finite l dimensional output vector, $Y = \{y_1, y_2, \dots, y_l\}$. $\delta: S \times X \to S$ is the fuzzy state-transition function which is used to map the current fuzzy state into the next fuzzy state upon an input value. $\omega: S \times X \to Y$ is the output function which is used to map the fuzzy state and input to the output value. See e.g. on Fig. 2.

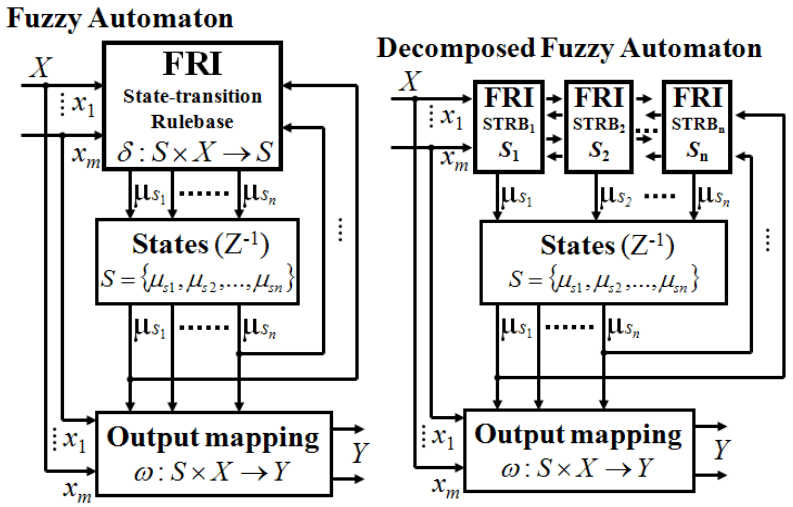

Figure 2
FRI based Fuzzy Automaton and its decomposition

In case of fuzzy rule-based representation of the state-transition function $\delta: S \times X \to S$, the rules have $n + m$ dimensional antecedent space, and n dimensional consequent space.

It is important to note, that the state S is a vector of membership values, the actual state is a point in the n dimensional unit hypercube. The state-transition rule-base moves this point in each discrete time step.

The FRI state-transition rule-base defines the state-transition function, which is a $R^{n+m} \to R^n$ mapping, having $n + m$ rule antecedent and n rule consequent dimensions. For simplifying the rule-base definition, this rule-base is decomposed to n pieces of single consequent rule-bases $R_i^{n+m} \to R_i$, $i \in [1, \ldots, n]$. See e.g. in Fig. 2.

For evaluating the FRI state-transition rule-base the FRI FIVE [22] was applied. The main idea of the FIVE FRI is based on the fact that most of the control applications serves crisp observations and requires crisp conclusions from the controller. Adopting the concept of the Vague Environment (VE) [30], FIVE can handle the antecedent and consequent fuzzy partitions of the fuzzy rule-base by scaling functions [30] turning the fuzzy interpolation to crisp interpolation. The idea of a VE is based on the indistinguishability of elements. In VE the fuzzy membership function $\mu_A(x)$ is indicating level of similarity of x to a specific element a which is a representative or prototypical element of the fuzzy set $\mu_A(x)$, or, equivalently, as the degree to which x is indistinguishable from a (see e.g. on Fig. 3) [30]. Two values in a VE are ε-indistinguishable if their distance is less or equal than ε. The distances in a VE are weighted distances (Eq. 2). The weighting factor or function is called scaling function (factor) [30]:

$$\delta_s(a,b) = \left| \int_a^b s(x)dx \right| \leq \varepsilon \tag{2}$$

where $\delta_s(a,b)$ is the scaled distance of the values a, b and $s(x)$ is the scaling function on X.

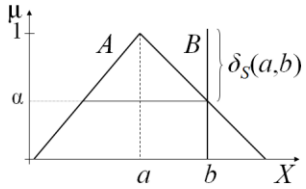

Figure 3
The α-cuts of $\mu_A(x)$ contain the elements that are $(1-\alpha)$-indistinguishable from a

If the VE of a fuzzy partition (the scaling function or at least the approximate scaling function [19] [20] [21]) exists, the member sets of the fuzzy partition can be characterized by points in that VE (see e.g. scaling function s in Fig. 4).

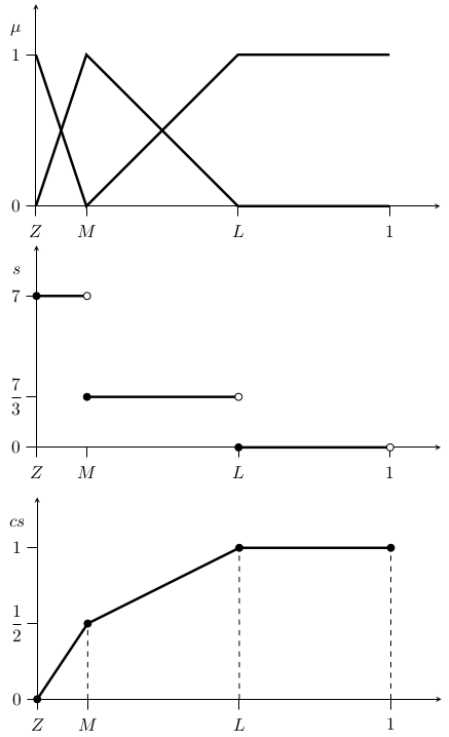

Figure 4
A "Ruspini" (0.5-covering) fuzzy partition built upon three linguistic terms,
namely the *Z* (Zero), *M* (Middle), *L* (Large) fuzzy sets,
its scaling function s, and its normalized cumulative scaling function cs

Having the VE concept and the scaling function based similarity calculation, any crisp interpolation, extrapolation, or regression method can be adapted very simply for FRI [7] [8] [9]. Because of its simple multidimensional applicability, in FIVE the Shepard operator based interpolation (first introduced in [35]) is adapted.

In case of singleton rule consequents (c_k) the fuzzy rule R_k has the following form:

$$\text{If } x_1 = A_{k,1} \text{ And } x_2 = A_{k,2} \text{ And ... And } x_m = A_{k,m} \text{ Then } y = c_k \tag{3}$$

Adapting the VE concept and the scaling function based similarity calculation to the Shepard operator based interpolation, the conclusion of the FRI can be obtained as:

$$y(x) = \begin{cases} c_k & if x = a_k \text{ for some } k, \\ \left(\sum_{k=1}^{r} \frac{c_k}{\delta_{s,k}^{\lambda}}\right)\left(\sum_{k=1}^{r} \frac{1}{\delta_{s,k}^{\lambda}}\right) & otherwise, \end{cases} \tag{4}$$

where $\delta_{s,k}$ are normalized scaled distances:

$$\delta_{s,k} = \delta_s(a_k, x) = \sqrt{\sum_{i=1}^{m} \left(cs_i(x_i) - cs_i(a_{k,i})\right)^2 / m} \tag{5}$$

$cs_i(x_i)$ is the normalized cumulative scaling function (see e.g. on Fig. 4) of s_i:

$$cs_i(x_i) = \int_0^{x_i} s_i(x_i)dx_i \Big/ \int_0^1 s_i(x_i)dx_i \tag{6}$$

and s_i is the i^{th} scaling function of the m dimensional antecedent universe, x is the m dimensional crisp observation and a_k are the cores of the m dimensional fuzzy rule antecedents A_k.

According to Eq. 4 the proposed interpolation method calculates the conclusions in the following steps.

1) Determine the normalized Euclidean distances of the observations from the rules on all the antecedent universes according to Eq. 5.
2) The rule weights are calculated as the reciprocal value of the distances corrected by the Shepard power p.
3) The consequent of the rule-base is calculated as the convex combination of the rule consequences weighted by the rule weights according to Eq. 4.

Considering the implemented fuzzy state machine model to be a discrete time system, the connections between the decomposed state-transition rule-bases (see Fig. 2) can serve as the time delays (Z^{-1}) temporarily storing the state variables. This case the fuzzy state machine can be defined as a recurrent network of multiple input single output state-transition rule-bases (see e.g. on Fig. 5), where the state variables are the values of the connections between them. On Fig. 5 the OBS prefix denotes observations and the RB denotes rule-bases. The number of the input is equivalent with the number of incoming edges.

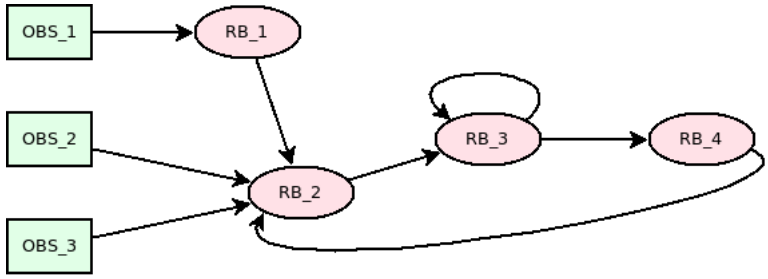

Figure 5
Example of a behavior description with recurrent connections
(OBS denotes observations, RB denotes rule-bases)

4 Behavior Fusion

Comparing the suggested FRI BBS structure (see Fig. 1) with the FRI state machine (see Fig. 2), the network of the Behavior Components and the Behavior Fusion is corresponding to the Output mapping of the Fuzzy Automaton.

A behavior component could be a reactive (stateless) function, e.g. an FRI rule-based system, or a compound FRI BBS. The problematic part is the Behavior Fusion. The function of the behavior fusion could be a convex combination of the behavior component actions according to the corresponding component weights, but to be able to handle alternative, or contradictive behavior component actions, it should be a FRI rule-based system again.

For simplifying the fusion, the continuous conclusions of the behavior components, for the FRI BBS, the application of a special fuzzy rule format is suggested. The basic rule structure is similar to the rule format applied in the Takagi-Sugeno fuzzy inference (Sugeno [16], Takagi-Sugeno [17]). In Takagi-Sugeno fuzzy inference, the consequent of a fuzzy rule is a function of the input variables. In the suggested FRI BBS, a rule of the Behavior Fusion rule-base has a consequence, which is a conclusion of another (behavior component) rule-base. This case the conclusions of the behavior component rule-bases are appearing as consequences of the behavior fusion rules.

From the viewpoint of the FBDL these are fuzzy rules with variable rule consequents, where the variable could be an observation, or a conclusion of a rule-base. The variable consequent fuzzy rule R_k has the following suggested form:

$$\text{If } x_1 = A_{k,1} \text{ And } x_2 = A_{k,2} \text{ And ... And } x_m = A_{k,m} \text{ Then } y = y_{V_k} \tag{7}$$

where $y_{V_k} = x_i$ is an observation, or $y_{V_k} = y_{RB_i}$ the conclusion of the i^{th} rule-base RB_i.

5 The Syntax of the Behavior Description Language

In this section the proposed behavior description language will be introduced. It is a declarative language which has a SQL-like syntax. (The assumption behind is that the verbosity of the language and the avoidance of special characters makes easier to learn its usage for non-technical users.)

The formal specification of the language is provided by extended Backus-Naur form. By using its common notation, the following language definition has obtained.

```
behavior ::= universe+ rulebase+ [init]
universe ::= 'universe' string ['description' string]
symbol+ 'end'
symbol ::= string number number
rulebase ::= 'rulebase' string ['description' string] rules
'end'
rules ::= rule+
rule ::= 'rule' ['description' string] ['use'] string
['when' predicates] 'end'
predicates ::= predicate ('and' predicate)*
predicate ::= string 'is' string
init ::= 'init' ['description' string] (string (string |
number))+ 'end'
```

In this formalism the `string` as a terminal symbol is a string literal with quotas, for example `string literal example`. The `number` is also a literal, which describes a floating point value, for instance 12.34.

The behavior description is a `behavior` non-terminal symbol. It contains at least one universe definition and a rule-base definition. After them, the expert can define an initialization block, where the initial state of the state machine can be set.

The definition of the universe uses the `universe` keyword. It followed by the name of the universe. Its name must be unique in a behavior description. The definition (similarly to the `rulebase`, `rule` and `init` definition) can be contains an optional description. It is introduced by the `description` keyword.

The `symbol` is a named point in the enclosing universe. The expert defines all of the language variables in this way. The symbol name is followed by the associated points of the antecedent and consequent side.

As an example, let define a universe called `distance`:

```
universe "distance"
description "The distance from the target."
    "close" 0 0
```

```
    "middle" 10 0.8
    "far" 50 1
end
```

As it can be seen in the description part of the definition block, the purpose of this universe is to represent the distance from a target position. There are three symbols: close, middle and far in the defined universe. The measurement unit of the distance is not necessary for the universe. According to the definition, the 0 distance is close, the 10 unit distance, is considered as middle and 50 units is seen as far. The second parameters after the symbol names are required for scaling. It means that the distance on interval [0, 10] changes quickly, while there is only a slight difference on the interval [10, 50].

For defining rules, it is necessary to define a further universe:

```
universe "tiredness"
description "The measure of tiredness."
    "low" 0 0
    "small" 0.3 0.5
    "middle" 0.7 0.5
    "high" 1 1
end
```

The universes of distance and tiredness are sufficient for the antecedent space. For the output side let define the universe `approach`:

```
universe "approach"
description "The speed how the agent approaching the
target."
    "low" 0 0
    "high" 1 1
end
```

At this point there are two universes: one for the observations and one for the consequence. The rule-bases defines the connection between the inputs and the output of the defined behavior. The rule-base encloses its rules. The general form of a rule-base is the following:

```
rule <consequent-name> when <predicates> end
```

The name of the consequent must match with the name of the universe. Let assume that we would like to define the behavior which fulfils the following rules:

1) The agent approaches the target, when the target is close and the agent is not tired.
2) The agent does not approach the target, when the target is far.
3) The agent does not approach the target, when the agent is tired.

These rules can be formalized via the proposed description language by the following way:

```
rulebase "approach"
    rule "high" when
        "distance" is "close" and "tiredness" is "low"
    end
    rule "low" when
        "distance" is "far"
    end
    rule "low" when
        "tiredness" is "high"
    end
end
```

5.1 Variable Valued Consequent

The language makes possible to use a recently calculated variable value as a consequent instead of a symbol of the universe. It is notated by the `use` keyword and the rule-base name which represents the variable, for example:

```
rulebase "first" ... end

rulebase "second" ... end

rulebase "output"
    rule use "first" when "need_first" is "true" end
    rule use "second" when "need_first" is "false" end
end
```

In the suggested FRI BBS, the variable valued consequents are the tools of the behavior fusion. The behavior component rule-bases are calculating the conclusions of the behavior components as variables. Then a rule-base with the corresponding variable valued consequents fuses them as behavior fusion.

The variable valued consequent has improved the flexibility of the language significantly. In fact, it makes available the calculation of weighted summation of rule-base outputs, where the weights are also calculated according to the user defined rules.

5.2 Initialization

The behavior model is iterative; therefore it requires an initial state. The initialization is compulsory if the variable (a value of a rule-base conclusion) is used as an input. The observations are also needed to be initialized, but it must be done by the environment. For variable initialization the language uses the `init`

keyword. It contains a pair of variables (rule-base names) and values. It is possible to use both symbols and values for setting the initial values.

```
init
    "value" "low"
    "result" 20
end
```

The integrity of the initial values are also checked. It means, that the following statements must be fulfilled:

- All universe names must have corresponding universe definitions.
- The symbol values of a given universe must be in the discourse bounds of the universe.
- All input values must be defined.

The initialization is an essential part of the behavior model. The initial values, as initial state is an integral part of the fuzzy state machine model.

5.3 Evaluation of the Consequences

If we consider our fuzzy state machine to be a connectionist structure, a rule-base is a node which represents a $R^n \rightarrow R$ function.

At first the behavior engine must calculate the distance of the observations from the symbols on all the antecedent universes. Let see the following example:

```
universe "distance"
    "zero" 0 0
    "close" 1 0.1
    "far" 5 1
    "max" 10 1
end
```

In this example the distance is given directly in meter. The consequent value is on the $[0,1]$ interval. (The normalization on consequent side is optional but a good practice in most cases.)

Above 5 meter the rule-base does not distinguish the values. All values are “far” up to 10 meter.

Let see the following rule-base:

```
rulebase "speed"
    rule "high" when "distance" is "far" and "curiosity" is
"high" end
    rule "low" when "distance" is "close" end
```

```
    rule "low" when "curiosity" is "low" end
end
```

where the appropriate universes are:

```
universe "curiosity"
    "low" 0 0
    "high" 1 1
end

universe "speed"
    "low" 0 0
    "high" 1 100
end
```

It is necessary to initialize the observations and calculate the distance of the observations from the symbols on all the antecedent universes. For instance, the values are initialized as the followings:

```
init
    "distance" 3
    "curiosity" 0.4
end
```

Consider the first rule and calculate the distance of 3 from “far” on the “distance” universe. The distance is determined from the difference of the cumulative scaling function values of the two points.

The value of the cumulative scaling function at 3 is 0.55, at “far” is 1, therefore their cumulative scaled distance is 0.45 (see Fig. 6). Similarly we can calculate the distance of 0.4 and “high” on “curiosity” interval. The result is 0.6.

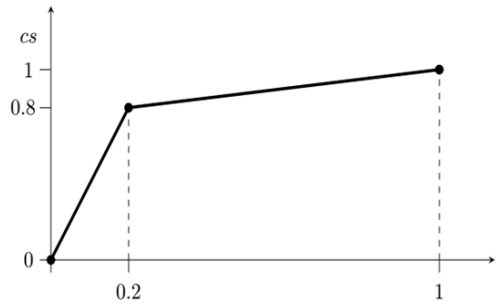

Figure 6
The non-linear scaling of the “distance” universe

The distance of the rule is given by the Euclidean norm of the distances by dimensions divided by the square of the number of antecedents. (It is necessary

because this way the distance is on $[0,1]$ is independent from the scaling of the universe.) The rule distance for the first rule is calculated as:

$$d_{rule1} = \frac{\sqrt{0.45^2+0.6^2}}{\sqrt{2}} \approx 0.5303 \tag{8}$$

Similarly, the engine calculates the distance of the second and the third rule:

$$d_{rule2} = \frac{\sqrt{0.45^2}}{\sqrt{2}} \approx 0.3182, d_{rule3} = \frac{\sqrt{0.4^2}}{\sqrt{2}} \approx 0.2828 \tag{9}$$

If there is a 0 distance from some rules, then the consequent is calculated as the mean value of the corresponding consequent symbol values, or variable values.

The weights are the reciprocals of the distances on the p Shepard-power, $w_i = \frac{1}{d_i^p}$

If $p = 1$ then the weights of the rules are:

$$w = [1.8856,3.1427,3.5355] \tag{10}$$

The consequent values of the rules are:

$$c = [1,0,0] \tag{11}$$

The consequence can be obtained as the sum of rule consequents weighted by the rule weights:

$$C = \frac{\sum_i w_i \cdot c_i}{\sum_i w_i} \approx \frac{1.8856}{8.5638} \approx 0.2202 \tag{12}$$

The consequent value of the "speed" rule-base is approximately 22.0183.

6 Dominancy

In many cases the domain expert's heuristically descriptive verbal model can be incomplete, or can contain some expert domain specific implicit knowledge. This domain specific implicit knowledge is inherently missing from the verbal description, e.g. as "well-known" facts, supposing that these are already known (explicitly given earlier).

On the other hand, having a rule-based knowledge representation, the lack of information means some missing cardinal rules from the rule-base. Let see a simple example. We want to describe the connection between the distance and the interest. We give a single expert's rule, that the interest is high when the distance is close. This case the natural assumption is that the interest is low when the distance is far. However, this assumption is not explicitly stated as a rule. It can be deduced, as a straightforward fact, only in the case, if we take into consideration, the domain knowledge. Otherwise, this rule is meaningless. But the rule-base is formally incomplete, because we do not say anything what should happen, when

the distance is far. Hence the previously considered model with one rule results high interest value independently from the distance. The FRI (fuzzy interpolation based) behavior model designed for handling undefined situation only in case if it can be deduced from the existing knowledge. A single rule means that only one output value is given, therefore it should be chosen independently from the input value. The other example is a kind of contradiction because of the unequal priorities of the rules. The first rule is "Stay home", the second is "Go to the garden if the sunshine is moderate". This case the "Stay home" is a kind of "default" rule requesting staying home if nothing special is happening. The second rule has more specialty (priority), requesting "Go to the garden" in a special case of "the sunshine is moderate". These two rules have contradictive conclusions in the case if "the sunshine is moderate". One solution could be the extension of the contradictive rules to make them to be the same specialty e.g. "Stay home if the sunshine is not moderate". On the other hand, this solution could be problematic, if the linguistic term "not moderate" is not defined (or it is impossible to define).

As a possible heuristic to solve the above problems, we suggest the concept of rule dominancy. The suggested rule dominancy heuristic is implemented as an extension of the FBDL, which enables the expression of rule relations in the form of rule dominancy.

6.1 Dominancy of the Rules

The suggested language extension provides elements for expressing hierarchical relation between rule-bases and between rules. We can express hierarchical rule relation without language extension, only by modifying the suppressed rule antecedents in the FBDL, but usually this is a tedious task. The suggested language extension automatizes this burden. Instead of try to cover the antecedent space by hand written rules, we can save this task by defining dominancy on the rule level. For supporting this, the grammar is slightly modified in the following way:

```
rules ::= rule+ ['dominates' rules 'end']
```

The dominancy means that the user can select dominant and dominated rules. The dominant rule consequent suppresses the conclusion of the dominated rules according to the fulfilment of the dominant rule antecedent. The consequent of the dominated rule is viable in a high level only when the fulfilment of the dominant rule antecedent is low. The dominancy can be applied in a hierarchical manner.

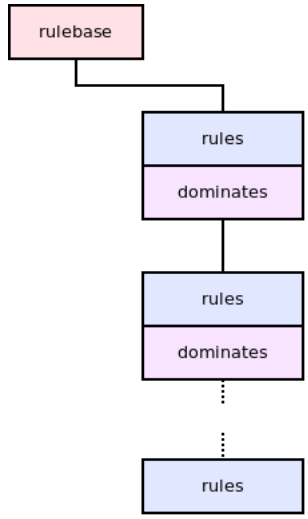

Figure 7
The semantical hierarchy of dominated rules

Let see the following example for multilevel dominancies:

```
rulebase "sample"
description "rule-base dominancy with examples"
   rule
      "low" when "stay_at_home" is "high"
   end
   dominates
      rule
         "high" when "playmates_exist" is "true"
      end
      dominates
         rule # Default rule
            "low"
         end
      end
   end
end
```

The dominancy also gives sense for defining default rules (see e.g. last rule of the rule-base "sample" above). It has no antecedent part; its conclusion is always selected. Having a default rule dominated by the others, all the undefined situations are automatically covered.

7 Sample Behavior

The following behavior shows a complete example of the previously mentioned features.

The model describes a one dimensional world, where an agent can walk between the target and the rest position. The target can make noise. The agent can move forward (to the target) and backward (to the rest position). The *interest* is the internal state of the agent. When the target makes noise, the agent interest is increasing. While the agent stays near the target, the interest value is decreasing. The measure of approaching mood to the target is also depending on the distance from the target. If the level of tiredness is high, the agent goes to the rest position. The FBDL language definition of this sample behavior is the following.

```
universe "has_noise"
description "Has the target noise?"
    "false" 0 0
    "true" 1 1
end

universe "distance"
description "The distance from the target."
    "close" 0 0
    "far" 1 1
end

universe "tiredness"
description "The measure of tiredness."
    "low" 0 0
    "high" 1 1
end

universe "speed"
description "The speed of the agent."
    "forward" 1 1
    "stop" 0 0
    "backward" -1 -1
end

universe "approach"
description "The speed how the agent approaching the
target."
    "low" 0 0
    "high" 1 1
end
```

```
universe "go_to_rest"
description "The speed how the agent go to the rest
position."
    "low" 0 0
    "high" -1 -1
end

universe "interest"
description "The measure how the agent interested about the
target."
    "low" 0 0
    "high" 1 1
end

rulebase "interest"
description "The agent is interested when the target has
noise."
    rule "high" when "has_noise" is "true" end
    rule "high" when "interest" is "high" end
    rule "low" when "distance" is "close" end
    dominates
        rule "low" end
    end
end

rulebase "approach"
description "Approach when there is a closer interesting
target."
    rule "high" when "distance" is "far" end
    dominates
        rule "low" end
    end
end

rulebase "go_to_rest"
description "Go to rest position when the target is tired."
    rule "high" when "tiredness" is "high" end
    dominates
        rule "low" end
    end
end

rulebase "speed"
```

```
description "The speed fusioned from approach and go_to_rest
values."
    rule use "approach" when "interest" is "high" end
    rule use "go_to_rest" when "tiredness" is "high" end
    dominates
        rule "stop" end
    end
end
```

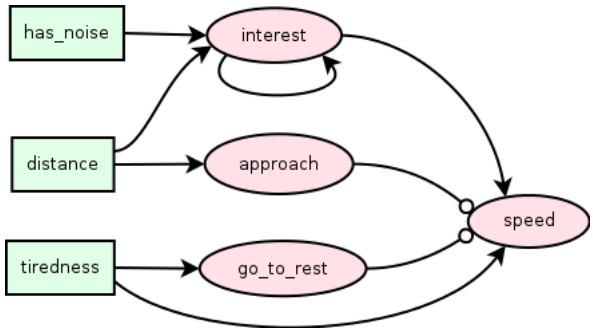

Figure 8
The behavior description of the agent

The behavior description can be checked by testing the following scenario in the simulation framework [31]:

1) Move the agent to the rest position by setting the *distance* value to 1.
2) Make noise by setting the *has_noise* observation to 1 and after back to 0. As a result, the interest value remains high, however the noise already stopped.
3) The agent speed is 1 which means that the agent wants to go to the target. Simulate the motion by setting the *distance* value to 0. At that point the interest level has decreased and the speed is near 0.
4) Set the tiredness to 0.5 - The agent now wants to move to the rest position. Start to increase the *distance* value, supposing the agent is moving away from the target. Near 0.55 the agent stops, because the interest level and the tiredness are close to each other.
5) Set the tiredness value to 0.8 - The *speed* value becomes negative, therefore the agent wants to go to the rest position. By increasing the *distance* further, the direction of speed remains, which means that the agent goes back to the rest position.

8 JavaScript Library and Framework

According to the suggested concept, the FBDL code is "running", as a parameter configuration, directly on a FRI state machine called "FRI Behavior Engine". In the current implementation, the FRI Behavior Engine was written in JavaScript. The instantiated FRI Behavior Engine loads and interprets the FBDL code and according to the fetched parameters provides interface for accessing its functions. For the implementation of the FRI Behavior Engine the JavaScript version was chosen. It makes testing easier and can run in a browser without installation. Moreover, it can be easily embedded to an online simulation environment too.

The FBDL web page, the FBDL library, the online FBDL simulator and their sources are available at [31].

The behavior engine is a software component which is able to evaluate the inputs based on the defined rules and calculates the internal state and the output, which will be available via its interface.

The engine instance can be applied according to the following sample:

```
engine.set("observation", 10);
engine.step();
var distance = engine.get("result");
```

The *set* method in the first line sets the *observation* value to 10. After, the *step* method calculates the new state. The variable values in the new state are accessible via the *get* method.

Conclusion and Future Work

The suggested Fuzzy Behavior Description Language (FBDL) is an easy-to-learn declarative language for non-programmer users, to define Fuzzy Rule Interpolation based Behavior-based System (FRI BBS) applications. Its inference model is based on the Fuzzy Rule Interpolation. Among the basic parameter and topology configuration the suggested FBDL also supports the concept of variable valued consequent and rule dominancy definitions.

The FBDL describes a fuzzy state machine having a massively parallel structure. The rule-bases and even the rules of the rule-bases can be evaluated parallel independently from each other. This parallel structure can be implemented on a massively parallel architecture speeding up the evaluation time and supporting the implementation of more complex models.

The FBDL is also serves embedded applications, where the FRI Behavior Engine is implemented on an embedded system. Instead of storage consuming look-up tables, or time consuming calculation of complex models, the FBDL and the FRI Behavior Engine can be an effective way for implementing complex behavior models. Moreover, the modification of the agent behavior can be simply achieved by modifying the FBDL description.

The proposed behavior description method, basically targets the non-programmer domain experts, where the definition of complex behaviors is a requirement from behavioral related studies or agent development reasons. The proposed Behavior Description Language and its Behavior Engine can be control physical and virtual robots, where there is a feedback from the environment and the actions are available. The FBDL description is in fact a special kind of configuration. It can be applied where the user has predefined inputs and outputs for the system. For instance, it is appropriate for the high level configuration, of IoT devices. The presented FBDL language can be regarded as a structured, semi-natural way of automatized robot control.

Future research will focus on the complexity of the defined behaviors. There is an assumption that the quantitative metrics of the behavior description, shows a strong correlation with the complexity of the modeled behavior. Its validation requires more experimentation, within the various types of behaviors.

Acknowledgement

The project acknowledges the financial support of Hungarian Research Fund (OTKA K120501).

References

[1] Arkin, R.C. et al.: *Behavior-based robotics*. MIT press, 1998

[2] P. Pirjanian, P.: *Behavior Coordination Mechanisms - State-of-the-art.* Techreport IRIS-99-375, Institute for Robotics and Intelligent Systems, School of Engineering, University of Southern California, October, 1999, 49 p.

[3] Kovács, Sz. et al.: *Behaviour based techniques in user adaptive kansei technology*. In: Proceedings of the VSMM2000, 6th international conference on virtual systems and multimedia, october. 2000, pp. 3-6

[4] Kovács, S.: *Interpolative fuzzy reasoning and fuzzy automata in adaptive system applications*. In: Proceedings of the IIZUKA. 2000, pp. 1-4

[5] Kovács, Sz.: *Fuzzy behaviour-based control techniques in adaptive system applications*. In: Proceedings of the ieee international conference on computational cybernetics, ICCC, 2003, pp. 29-31

[6] Kovács, Sz., Kóczy, L. T.: *Application of interpolation-based fuzzy logic reasoning in behaviour-based control structures*. In: 2004 ieee international conference on fuzzy systems (ieee cat. no. 04CH37542) IEEE, 2004, pp. 1543-1548

[7] Kovács, Sz.: *Interpolative fuzzy reasoning in behaviour-based control*. In: Computational intelligence, theory and applications. Springer, 2005, pp. 159-170

[8] Kovács, Sz.: *Fuzzy rule interpolation in embedded behaviour-based control*. In: 2011 IEEE international conference on fuzzy systems (Fuzz-IEEE 2011) IEEE, 2011, pp. 436-441

[9] Kovács, Sz. et al.: Interpolation based fuzzy automaton for human-robot interaction. *IFAC Proceedings Volumes*, **42** (16) 2009, pp. 317-322

[10] Kovács, Sz. et al.: *Ethologically inspired robot behavior implementation*. In: 2011 4th international conference on human system interactions, hsi 2011. IEEE, 2011, pp. 64-69

[11] Vincze, D. et al.: A novel application of the 3d virca environment: Modeling a standard ethological test of dog-human interactions. *Acta Polytechnica Hungarica*, **9** (1), 2012, pp. 107-120

[12] The Fuzzy Rule Interpolation (FRI) web page, the FRI-Toolbox, the Sparse Fuzzy Model Identification Toolbox and their sources can be found at: http://fri.uni-miskolc.hu/

[13] Zadeh, L. A.: Outline of a new approach to the analysis of complex systems and decision processes. *IEEE Transactions on systems, Man, and Cybernetics*, (1), 1973, pp. 28-44

[14] Mamdani, E., Assilian, S.: An experiment in linguistic synthesis with a fuzzy logic controller. *International journal of human-computer studies*, **51** (2), 1999, pp. 135-147

[15] Larsen, P. M.: Industrial applications of fuzzy logic control. *International Journal of Man-Machine Studies*, **12** (1), 1980, pp. 3-10

[16] Sugeno, M.: An introductory survey of fuzzy control. *Information sciences*, **36** (1-2), 1985, pp. 59-83

[17] Takagi, T., Sugeno, M.: *Fuzzy identification of systems and its applications to modeling and control*. In: Readings in fuzzy sets for intelligent systems. Elsevier, 1993, pp. 387-403

[18] Perfilieva, I.: Fuzzy function as an approximate solution to a system of fuzzy relation equations. *Fuzzy sets and systems*, **147** (3), 2004, pp. 363-383

[19] Kovács, Sz.: *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

[20] Kovacs, Sz., Koczy, L. T.: *Approximate fuzzy reasoning based on interpolation in the vague environment of the fuzzy rulebase*. In: Proceedings of IEEE international conference on intelligent engineering systems. IEEE, 1997, pp. 63-68

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

[22] Kovács, Sz.: *Extending the fuzzy rule interpolation‘ five’ by fuzzy observation*. In: Computational intelligence, theory and applications. Springer, 2006, pp. 485-497

[23] Piller, I. et al.: *Declarative language for behaviour description*. In: Emergent trends in robotics and intelligent systems. Springer, 2015, pp. 103-112

[24] Meneguzzi, F., Luck, M.: Declarative planning in procedural agent architectures. *Expert Systems with Applications*, **40** (16), 2013, pp. 6508-6520

[25] Berenz, V., Suzuki, K.: Targets-drives-means: A declarative approach to dynamic behavior specification with higher usability. *Robotics and Autonomous Systems*, **62** (4), 2014, pp. 545-555

[26] Borenstein, D. B.: A composite constraints approach to declarative agent-based modeling. *arXiv preprint arXiv:1503.08880*, 2015

[27] Klügl, F. et al.: *From simulated to real environments: How to use sesam for software development*. In: German conference on multiagent system technologies. Springer, 2003, pp. 13-24

[28] Doostfatemeh, M., Kremer, S. C.: New directions in fuzzy automata. *International Journal of Approximate Reasoning*, **38** (2), 2005, pp. 175-214

[29] Stamenković, A. et al.: Different models of automata with fuzzy states. *Facta Universitatis, Series: Mathematics and Informatics*, **30** (3), 2015, pp. 235-253

[30] Klawonn, F.: Fuzzy sets and vague environments. *Fuzzy Sets and Systems*, **66** (2), 1994, pp. 207-221

[31] The FBDL web page, the FBDL library, the online FBDL simulator and their sources can be found at: http://fbdl.iit.uni-miskolc.hu/

[32] Kóczy, L. T., Hirota, K.: *Interpolative reasoning with insufficient evidence in sparse fuzzy rule bases*, Information Sciences, **71**, 1992, pp. 169-201

[33] Miklosi, A., Korondi, P., Matellan, V., Gacsi, M.: *Ethorobotics: A New Approach to Human-Robot Relationship*, Frontiers in Psychology, **8** Paper: 958, 2017, p. 8

[34] Lakatos, G., Gácsi, M., Konok, V., Brúder, I., Bereczky, B., Korondi, P., Miklósi, Á.: Emotion Attribution to a Non-Humanoid Robot in Different Social Situations, PLOS ONE, **9**, 12, Paper: e114207, 2014, p. 32

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