ния документов на академический отпуск, смена студенческого билета, зачетной книжки – для оформления документов на смену студенческого билета, либо зачетной книжки, начисление стипендии – для оформления документов на начисление стипендии, смена фамилии – для оформления документов на смену фамилии, выпуск – для оформления документов на получение квалификации, наименование учреждения образования – для хранения всех видов учреждений образования, информация о родственниках – для хранения информации о каждом родственнике студента, город, деревня – для хранения информации о городах и деревнях, район – для хранения информации о районах, область – для хранения информации об областях, страна – для хранения информации о странах, состав группы – функциональная таблица для формирования групп.

Для организации хранения данных была выбрана система управления базами данных (СУБД) MySQL Server 5.5, так как она является удобной в использовании, многопоточной, свободно распространяемой и мультиплатформенной [6].

Для добавления, изменения и удаления данных написаны пользовательские функции и процедуры, многие из которых организуют поддержание целостности системы. Самое важное ограничение целостности на уровне отдельных полей – это тип данных. Механизм баз данных MySQL обеспечивает богатый спектр типов данных.

Целостность системы поддерживается также при помощи триггеров и ограничений в виде Check [5, 7].Созданы триггеры, которые проверяют регистр букв в именах, автоматически формируют название группы, полное и сокращенное имя студента, проверяют отметки студента (помимо цифр от одного до десяти, можно вводить «зачтено»/«не зачтено») и т.д. Для безошибочной работы с данными вводим следующие ограничения: проверка форм обучения на имеющиеся в базе, проверка семестров (нельзя ввести число меньше единицы и больше 12), проверка всевозможных дат (например, чтобы дата рождения не была больше текущей или, чтобы студенту не превышало 60), проверка среднего балла на вхождение в отрезок [4.0, 10.0], проверка типа учреждения образования (выбор только из существующих в базе) и т.д.

В результате проделанной работы была спроектирована база данных, при помощи которой можно создать автоматизированную информационную систему направленную на сокращение временных затрат работника деканата, занимающегося ведением различной документации и учетом контингента, минимизировать появление ошибок при составлении документов и автоматизировать формирование отчетов.

ЛИТЕРАТУРА

1. Деканат [Электронный ресурс] // Википедия. – Режим доступа: https://ru.wikipedia.org/wiki/Декан/. – Дата доступа: 10.04.2015.
2. Что такое деканат в университете? [Электронный ресурс] // Студенческая жизнь. – Режим доступа: http://life-students.ru/chto-takoe-dekanat-v-universitete/. – Дата доступа: 10.04.2015.
3. Малыхина, М. Базы данных: основы, проектирование, использование / М. Малыхина. – М. : BHV, 2004. – 512 с.
4. Дейт, К.Д. Введение в системы баз данных / К.Д. Дейт. – М. : Вильямс, 2005. – 1328 с.
5. Боуман, Дж. Практическое руководство по SQL / Дж. Боуман, С. Эмерсон, М. Дарновски. – Изд. 4-е. – М. : Вильямс, 2002. – 352 с.
6. Кузнецов, М.В. MySQL 5 / М.В. Кузнецов, И.В. Симдянов. – СПб. : БХВ-Петербург, 2010. – 1024 с.
7. Грофф, Дж. Полное руководство по SQL / Дж. Грофф, П. Вайнберг. – М. : BHV, 2001. – 816 с.

УДК 004

РАЗРАБОТКА ФОРМАЛЬНОГО ЯЗЫКА ОПИСАНИЯ ПОВЕДЕНИЯ АГЕНТА НА БАЗЕ НЕЧЕТКОЙ ЛОГИКИ

В.А. ПЛЯСОВ
(Представлено: канд. техн. наук, доц. Д.О. ГЛУХОВ)

Рассмотрены основные моменты создания формального языка для искусственного интеллекта на базе нечеткой логики.

Нечеткая логика – это логика, в которой мы можем оперировать не только с лог. «0» или лог. «1», а так же со значениями в интервале [0;1]. Рассмотрим классический пример: расстояние от машины до препятствия в обыкновенной логике: мы бы оперировали такими значениями: 1(150м) – далеко, 0(0м) – близко, а нечеткая логика подразумевает нечеткие понятия, такие как очень близко, близко, средняя, далеко и т.д., такие понятия характерны для мышления человека (рис. 1.) [1].

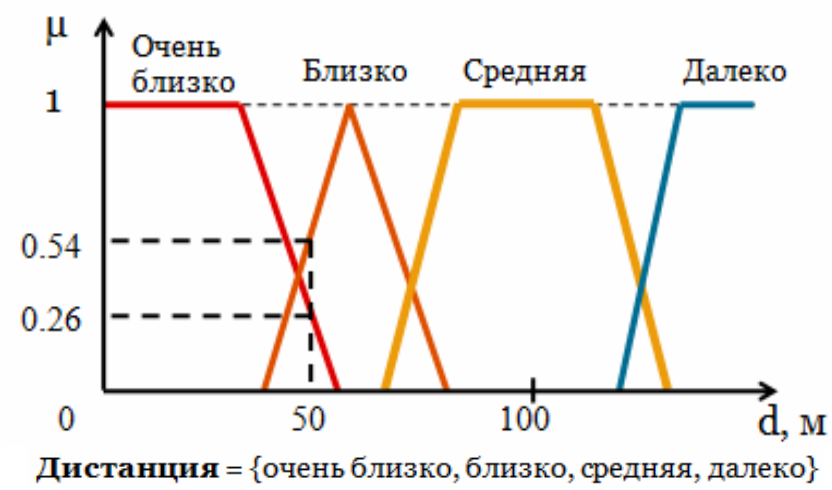

Рис. 1. Графическое изображение дистанции от машины до препятствия

Все основные понятия (например, скорость, дистанция, уровень жизней и т.д.) в нечеткой логике называются лингвистическими переменными, а их возможные значения (например: скорость (маленькая, большая, ...), дистанция (близко, далеко, ...) и т.д.) называются лингвистическими термами. Каждый терм охватывает какой-либо промежуток из линг. переменной, которая называется функция линг. терма. На представленном графике ось y(μ) показывает степень принадлежности для каждого значения дистанции.

Расширим предыдущий пример. Добавим еще 2 переменных: «направление», которая отвечает, в какую сторону движется машина, и «рулевой угол» – куда поворачивать в зависимости от дистанции и направления. Данные переменные будут иметь следующие термы:

Направление = {левое, прямое, правое}

Рулевой угол = {резко влево, влево, прямо, вправо, резко вправо}

Чтобы определить в какую сторону необходимо повернуть «Рулевой угол», воспользуемся различными условиями, представленными в таблице.

Таблица

Условия для определения «рулевой угол»

	Дистанция				
Направление		Очень близко	Близко	Средняя	Далеко
	Правое	Резко влево	Резко влево	Влево	Прямо
	Прямо	Резко влево	Влево	Влево	Прямо
	Левое	Резко вправо	Резко вправо	Вправо	Прямо

$\mu 1 = 0.89$ $\mu 2 = 0.60$

Допустим: if «направление» = «правое» and «дистанция» = «средняя» then «рулевой угол» = «влево». В нечеткой логике в условиях нет понятия истинно или ложно, здесь оперируют понятием вероятность срабатывания. Причем она для этого условия будет определяться по формуле [2]

$$\min(\mu 1, \mu 2) \quad (1)$$

для операции «И».

Для операции «ИЛИ» по формуле:

$$\max(\mu 1, \mu 2), \quad (2)$$

для отрицания по формулам:

$$1 - \mu 1, \quad (3)$$
$$1 - \mu 2. \quad (4)$$

Выходной параметр для данного примера будет вычисляться по формуле (введя следующие замены A = «рулевой угол»; T1 = «влево»):

$$A = \min(\mu 1, \mu 2) \cdot T1. \quad (5)$$

Если рассматривать более сложные конструкции (вложенные условия), то выходной параметр будет находиться следующим образом:

Допустим, у нас имеется конструкция следующего вида:

```
if (cond1)
{
   A is T1
   if (cond2) A is T2
}
if (cond3) A is T3
```

где cond1, cond2, cond3 – какие-либо условия, у которых степени принадлежности $\mu 1$, $\mu 2$, $\mu 3$соответственно.

A – лингвистическая переменная.

T1, T2, T3 – лингвистические термы переменной *A*.

‘if’ – условный оператор.

‘is’ – оператор присвоения.

Из данного выражения выходной параметр *A* будет находится по следующей формуле:

$$A = \frac{T1 \cdot \mu_1 + T2 \cdot \min(\mu_1, \mu_2) + T3 \cdot \mu_3}{\mu_1 + \min(\mu_1, \mu_2) + \mu_3}. \quad (6)$$

Пройдя введение можно рассмотреть основные моменты для создания ИИ в играх на основе данной логики.

Основная проблема, которая возникает на первом этапе разработки это: как сделать так, чтобы машина понимала нечеткие понятия и интерпретировала в нужный для нее язык. Чтобы решить данную проблему необходимо простроить следующий механизм, который изображен на рисунке 2 [3].

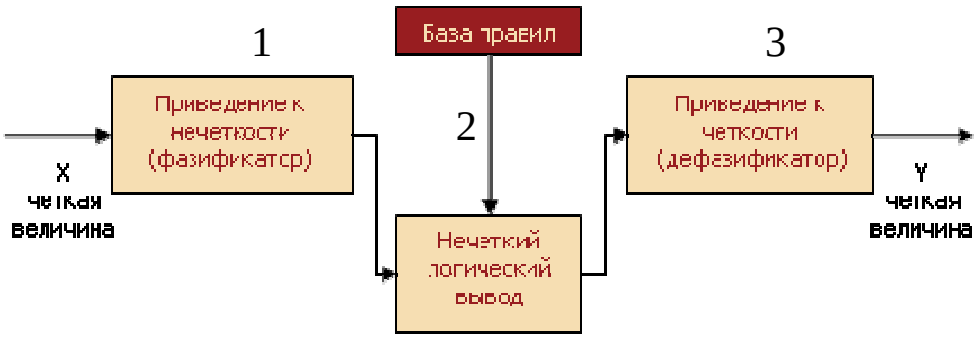

Рис. 2. Механизм для использования нечеткой логики

Суть его такова: разберем тот же пример с дистанцией от машины до препятствия. На вход данного «черного ящика» поступает четкая дистанция, допустим 100 м. Затем элемент ящика (приведение к нечеткости) интерпретирует данное значение в нечеткий терм (из примера «средняя»). Полученный терм переходит на следующий блок (нечеткий логический вывод) где располагаются различные условия, которые оперируют с данной лингвистической переменной (в данном примере «Дистанция»). После выполнения условий полученное действие так же является еще нечетким (к примеру выполнилось условие: если «направление» = «правое» и «дистанция» = «средняя», то «рулевой угол» = «влево») «рулевой угол» = «влево». Данное действие переходит к блоку (приведение к четкости) и получает какой-то четкий угол поворота (к примеру, 15о).

Чтобы воспользоваться данным механизмом в языках программирования необходимо написать свой язык, который будет, поступившее на вход четкое значение преобразовывать в нечеткое, затем с помощью нечетких условий определяет действие и обратно на выход отдавать четкое значение полученного действия.

Пример разработанного формального языка описания поведения агента (юнита) на базе нечетких логических условий и установок:

```
set "R" (100, 1) // Радиус объекта
//Нечеткие множества параметров
set "все слева" (-0.1, 0;1.75, 0.5;3, 1;3.3, 0)
set "более слева" (-0.1, 0;0.8, 1;1.2, 0)
set "чуть слева" (-0.01, 0;0, 1;0.03, 1;1, 0)
//Нечеткие множества действий
set "чуть повернуть направо" (-0.02, 1)
set "резко повернуть направо" (-0.2, 1)
set "чуть повернуть налево" (0.01, 1)
set "резко повернуть налево" (0.2, 1)

//Правила
if "угол на цель" is not "впереди" {
        "скорость корпуса" ~ "стоп"
        if "угол на цель" is "все справа" { "поворот корпуса" ~ "резко повернуть направо" }
        if "угол на цель" is "все слева" { "поворот корпуса" ~ "резко повернуть налево" }
}
```

Данный формальный язык реализует 2 блока (см. рис. 2): приведение к нечеткости и нечеткий логический вывод с базой правил. 3 блок уже реализуется непосредственно с помощью языков программирования получением данных и данного языка (см. рис. 2).

Описание формального языка будет иметь вид (в форме Бэкуса-Наура)[4, 5]:

```
<prog> ::= <def> <block>
<def> ::= set <term> (fp,fp; fp, fp; fp, fp) | <def> set <term> (fp,fp; fp, fp; fp, fp).
<block> ::= <operator> | <prog>
<operator> ::= <ifoperator> | <setoperator>
<ifoperator> ::= if <cond> { <block>} | if <cond> { <block>} else {<block>}
<cond> ::= <term> | <cond> and <cond> | <cond> or <cond> | (<cond>) | not <cond>
<term> ::= name
<setoperator> ::= <lparam> is <term>
<lparam> ::= name
```

Вероятность срабатывания того или иного правила будет высчитываться по тем же формулам, о которых говорилось ранее.

ЛИТЕРАТУРА

1. Шеври, Ф. Нечеткая логика / Ф. Шеври, Ф. Гели. – Вып. 31.
2. ЭР – Нечеткая логика [Электронный ресурс]. – Режим доступа: http://fuzzy-group.narod.ru/files/Fuzzy_Modeling/Lection07.Fuzzy.logic.pdf. – Дата доступа: 20.08.2015.
3. ЭР – Математические основы нечеткой логики [Электронный ресурс]. – Режим доступа: http://bourabai.ru/tpoi/fuzzy.htm. – Дата доступа: 20.08.2015.
4. ЭР – Формальный язык [Электронный ресурс]. – Режим доступа: https://ru.wikipedia.org/wiki/%D0%A4%D0%BE%D1%80%D0%BC%D0%B0%D0%BB%D1%8C%D0%BD%D1%8B%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA. – Дата доступа: 20.08.2015.
5. ЭР – форма Бэкуса-Наура [Электронный ресурс]. – Режим доступа: https://ru.wikipedia.org/wiki/%D0%A4%D0%BE%D1%80%D0%BC%D0%B0_%D0%91%D1%8D%D0%BA%D1%83%D1%81%D0%B0_%E2%80%94_%D0%9D%D0%B0%D1%83%D1%80%D0%B0. – Дата доступа: 20.08.2015.

УДК 004

ОСОБЕННОСТИ МОДЕЛИРОВАНИЯ ПОВЕДЕНИЯ АГЕНТА ПРИ ОБХОДЕ ПРЕПЯТСТВИЙ С ИСПОЛЬЗОВАНИЕМ НЕЧЕТКОЙ ЛОГИКИ

В.А. ПЛЯСОВ
(Представлено: канд. техн. наук, доц. Д.О. ГЛУХОВ)

Рассматриваются проблемы при обходе препятствий агентами с использованием нечеткой логики и дальнейшие способы их решения, которые максимально приближают поведение агента к идеальным случаям.