#### Лабораторная работа № 2-1.

**Деревья решений и системы нечеткого вывода**

1. Сгенерировать случайные числа по указанному закону распределения. Из них создается матрица из M строк (наблюдений) и K столбцов (признаков). Сгенерированные строки должны принадлежать одному из N классов за счет прибавления к сгенерированным числам положительной или отрицательной добавки (функция)
2. Все признаки должны быть числовыми и нормализованными (находиться в [0,1]). Дополнительно создается классификационный признак. Нормализация выполняется с помощью функции normr(X).
3. Разделить данные варианта на обучающую и тестирующую выборки
4. По обучающей выборке построить классификационное дерево решений
5. Построить дерево минимальной стоимости
6. По дереву решений минимальной стоимости сформировать систему нечеткого вывода (fis) типа Сугено с треугольными функциями принадлежности термов, входные переменные которой – дуги дерева решений, выходная переменная – номер класса.
7. Сохранить fis в файле и выдать ее на экран
8. В редакторе систем нечеткого вывода выполнить пункт меню View, Rules, с помощью которого проверить работоспособность созданной системы нечеткого вывода
9. Отдельно данные обучающей выборки и тестирующие данные классифицировать с помощью дерева решений минимальной стоимости и с помощью системы нечеткого вывода.
10. Результаты разместить в таблице, в которой записаны проценты совпадений по каждому классу отдельно по обучающей и по контролирующей выборке

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| N класса | % верно классифицированных – дерево решений, обучающая выборка | % верно классифицированных – дерево решений, контролирующая выборка | % верно классифицированных – fis, обучающая выборка | % верно классифицированных – fis, тестирующая выборка |
| 1 |  |  |  |  |
| … |  |  |  |  |
| M |  |  |  |  |

1. Создать GUI-приложение, реализующее экспертную консультацию в соответствии с системой нечеткого вывода

Вариант работы имеет вид:ZPM, где

Z=1-6 – вид закона распределения генерируемых признаков

1. Биномиальное
2. Геометрическое
3. Гипергеометрическое
4. Пуассона
5. Равномерное
6. Нормальное

K=3,4,5,6 – число признаков

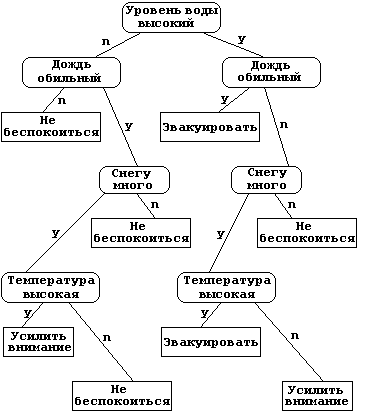
N=3,4,5 – число классов

**Деревья решений**

Деревья решений – это способ представления правил в иерархической, последовательной структуре, где каждому объекту соответствует единственный узел, дающий решение.

Под правилом понимается логическая конструкция, представленная в виде "если ... то ...".

Пример дерева решений.



Деревья решений отлично справляются с задачами классификации. Целевая переменная должна иметь дискретные значения.

## Как построить дерево решений?

Пусть нам задано некоторое обучающее множество T, содержащее объекты (примеры), каждый из которых характеризуется m признаками, причем один из них указывает на принадлежность объекта к определенному классу.

Пусть через {C1, C2, ... Ck} обозначены классы, тогда существуют 3 ситуации:

1. множество T содержит один или более примеров, относящихся к одному классу Ck. Тогда дерево решений для Т – это лист, определяющий класс Ck;
2. множество T не содержит ни одного примера, т.е. пустое множество. Тогда это снова лист, и класс, ассоциированный с листом, выбирается из другого множества, отличного от T, скажем, из множества, связанного с родителем;
3. множество T содержит примеры, относящиеся к разным классам. В этом случае следует разбить множество T на некоторые подмножества. Для этого выбирается один из признаков, имеющий два и более отличных друг от друга значений O1, O2, ... On. T разбивается на подмножества T1, T2, ... Tn, где каждое подмножество Ti содержит все примеры, имеющие значение Oi для выбранного признака. Это процедура будет [рекурсивно](http://www.basegroup.ru/glossary_ajax/definitions/recursion) продолжаться до тех пор, пока конечное множество не будет состоять из примеров, относящихся к одному и тому же классу.

Вышеописанная процедура лежит в основе многих современных алгоритмов построения деревьев решений, этот метод известен еще под названием «разделяй и властвуй» (divide and conquer). Очевидно, что при использовании данной методики построение дерева решений будет происходит сверху вниз.

Поскольку все объекты были заранее отнесены к известным нам классам, такой процесс построения дерева решений называется обучением с учителем ([supervised learning](http://www.basegroup.ru/glossary_ajax/definitions/supervised_learning)). На сегодняшний день существует значительное число алгоритмов, реализующих деревья решений, но наибольшее распространение и популярность получил алгоритм

**CART (Classification and** Regression Tree**)** – это алгоритм построения бинарного дерева решений – дихотомической классификационной модели. Каждый узел дерева при разбиении имеет только двух потомков.

Большинство из известных алгоритмов являются "жадными алгоритмами". Если один раз был выбран признак, и по нему было произведено разбиение на подмножества, то алгоритм не может вернуться назад и выбрать другой признак, который дал бы лучшее разбиение. Поэтому на этапе построения нельзя сказать, даст ли выбранный признак, в конечном итоге, оптимальное разбиение.

## Этапы построения деревьев решений

При построении деревьев решений особое внимание уделяется следующим вопросам: выбору признака, по которому пойдет разбиение и остановки обучения.

### Правило разбиения. Каким образом следует выбрать признак?

Для построения дерева на каждом внутреннем узле необходимо найти такое условие, которое бы разбивало множество, связанное с этим узлом, на подмножества. В качестве такой проверки должен быть выбран один из признаков. Общее правило для выбора атрибута можно сформулировать следующим образом: выбранный признак должен разбить множество так, чтобы получаемые в итоге подмножества состояли из объектов, принадлежащих к одному классу, или были максимально [приближены](http://www.basegroup.ru/glossary_ajax/definitions/approximation) к этому, т.е. количество объектов из других классов ("примесей") в каждом из этих множеств было как можно меньше.

Были разработаны различные критерии, рассмотрим один из них:

#### Статистический критерий

[**Алгоритм CART**](http://www.basegroup.ru/glossary_ajax/definitions/cart) использует так называемый индекс Gini (в честь итальянского экономиста Corrado Gini), который оценивает "расстояние" между распределениями классов.

|  |  |
| --- | --- |
| http://www.basegroup.ru/files/image/library/analysis/tree/description/gini.gif |  |

где c – текущий узел, а pj – вероятность класса j в узле c.

**Функции Matlab для работы с деревьями решений**

**Деление выборки на обучающую и тестирующую**

**Функция cvpartition**

**c = cvpartition(n,'holdout',p)** создает случайное разделение n наблюдений на обучающую выборку и тестирующую выборку. Параметр p должен быть 0 < p < 1, в этом случае функция cvpartition случайно выбирает примерно p\*n наблюдений для тестирующей выборки. Значение p по умолчанию равно 0.1.

Объект c имеет свойства:

N – число наблюдений;

TestSize – размер тестирующей выборки;

TrainSize – размер обучающей выборки

и методы:

idx=test(c) – возвращает логический вектор idx для отличия номеров тестирующей выборки (1) от номеров обучающей выборки (0).

idx=training(c) – возвращает логический вектор idx для отличия номеров обучающей выборки (1) от номеров тестирующей выборки (0).

Поэтому выражение X(training(cv),:) формирует подматрицу X из строк обучающей выборки.

**Построение таблицы частот**

**tabulate(x)** создаеттаблицу частот данных из вектора x. Информация в таблице организована следующим образом:

1 столбец — уникальное значение x

2 столбец — число повторений каждого значения

3 столбец — процент каждого значения

**Формирование дерева классификации**

**tree = fitctree(x,y)** возвращает бинарное дерево классификации tree, основанное на входных переменных x и выходной переменной y. В бинарном дереве каждый узел имеет не более двух потомков по значениям колонки матрицы x.

**Вывод дерева классификации на экран в графическом виде**

**view(tree,'mode','graph')**

Промежуточные узлы дерева решений отмечены метками http://matlab.exponenta.ru/statist/book2/17/images_treedisp/image001.jpg с условиями выбора относительно значения какой либо независимой переменной. Каждый из конечных узлов дерева решений имеет метку http://matlab.exponenta.ru/statist/book2/17/images_treedisp/image002.jpg со значением зависимой переменной. Для выбора одной из двух возможных ветвей дерева решений исходящих из промежуточного узла действует правило: левая ветвь соответствует выполнению условия относительно независимой переменной, правая - невыполнению этого условия. Определение значения зависимой переменной начинается с первого промежуточного узла, далее, следуя заданному условию, выбирается правая или левая исходящая ветвь. Аналогично просматриваются последующие узлы до тех пор, пока не будет достигнут последний конечный узел. Символьная метка, соответствующая последнему узлу, является значением зависимой переменной.

Пример

load fisheriris

% Классифицирующие значения

for i=1:50

Y(i,1)=1;

end

for i=51:100

Y(i,1)=2;

end

for i=101:150

Y(i,1)=3;

end

X = double(meas);

% В примере 40% данных случайным образом включаются в тестовый набор

cv = cvpartition(nrows,'holdout',0.40);

% Обучающее множество

Xtrain = X(training(cv),:);

Ytrain = Y(training(cv),:);

% Тестовое множество

Xtest = X(test(cv),:);

Ytest = Y(test(cv),:);

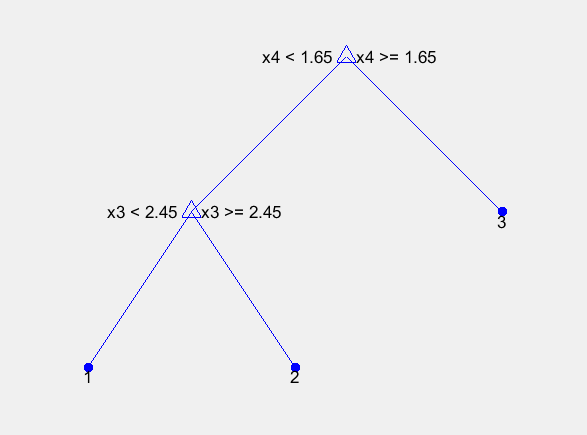
tabulate(Ytrain);

tabulate(Ytest);

% t1= fitctree(Xtrain,Ytrain);

view(t1,'mode','graph')

Результат вывода дерева:



**Предсказание значений классифицирующего признака по дереву**

**Ynew = predict(tree,Xnew)**

Для каждой строки данных матрицы Xnew выдается классифицирующий признак в соответствии с деревом; все значения объединяются в вектор Ynew.

**Нахождение оптимального уровня сокращения дерева за счет минимизации потерь**

**[~,~,~,bestlevel] = cvLoss(tree, 'SubTrees','All','TreeSize','min')** функция возвращает наилучший уровень сокращения дерева; все остальные выходные параметры не используются

**Сокращение дерева до оптимального уровня**

**tree1 = prune(tree,'Level', bestlevel)** получение сокращенного дерева (дерева оптимального уровня)

При построении системы нечеткого вывода нужно учитывать, что каждая ветвь дерева от корня до листа – это правило; каждое условие для входной переменной – это ее терм; термы одной переменной не должны дублировать; система нечеткого вывода включает только те входные переменные, которые отражены в дереве. Система нечеткого вывода содержит одну выходную переменную с термами, соответствующими листьям дерева, то есть значениям классификационного признака.

Кроме того, надо учитывать, что при многократном запуске скрипта генерируются разные данные, а для того, чтобы дерево решений и система нечеткого вывода соответствовали друг другу, надо сохранить переменную-идентификатор дерева в файле и при последующих запусках скрипта загружать из файла.

Сохранение переменных рабочей области в файле выполняется по команде

**save имя файла имена переменных через запятую**

Файл имеет расширение .mat.

Загрузка переменных из файла в рабочую память происходит по команде

**load имя файла имена переменных через запятую**

**Системы нечеткого вывода в Matlab**

Для реализации процесса нечеткого моделирования в среде MATLAB предна­значен специальный пакет расширения Fuzzy Logic Toolbox. В рамках этого па­кета пользователь может выпол­нять необходимые действия по разработке и использованию нечетких моделей в одном из следующих режимов:

* в интерактивном режиме с помощью графических средств редактирования и визуализации всех компонентов систем нечеткого вывода;
* в режиме выполнения сценария (m-файла) с помощью ввода его имени непосредственно в окно команд системы MATLAB
* в режиме выполнения графического приложения (GUI)

Мы будем сочетать все три режима:

* по дереву решений построим систему нечеткого вывода с помощью команд, затем просмотрим созданную систему в интерактивном режиме
* в заключение сформируем экспертную систему, выполняющую ввод входных переменных и определение выходных переменных по системе нечеткого вывода с помощью GUI-приложения.

**Нечеткое множество и функция принадлежности**

Нечеткое множество (fuzzy set) представляет со­бой совокупность элементов произвольной природы, относительно которых нельзя с полной определенностью утверждать, принадлежит ли тот или иной элемент рассматриваемой совокупности данному множеству или нет. Другими словами, нечеткое множество отличается от обычного множества тем, что для всех или части его элементов не существует однозначного ответа на вопрос: «Принадлежит или не принадлежит тот или иной элемент рассматриваемому не­четкому множеству?» Можно этот вопрос задать и по-другому: «Обладают или нет его элементы некоторым характеристическим свойством, которое может быть использовано для задания этого нечеткого множества?»

Наиболее естественным и интуитивно понятным является задание функции, определяющей степень уверенности положительного ответа на подобный вопрос. Такая функция в качестве области допустимых значений чаще всего имеет действительные числа, заключенные между 0 и 1 (включая и сами эти значения).

Формально **нечеткое множество** определяется как множество упорядоченных пар вида: A=<х, μ(x)> или μ(x)/x, где x является элементом некоторого множества X, а μ(x) — функция принадлежности, ко­торая ставит в соответствие каждому из элементов хХ некоторое действитель­ное число из интервала [0,1].

**Лингвистической переменной**(linguistic variable) называется переменная, значениями которой могут быть слова или словосочетания некоторого естественного или искусственного языка.

**Терм–множеством** (term set)называется множество всех возможных значений лингвистической переменной.

**Термом** (term)называется любой элемент терм–множества. В теории нечетких множеств терм формализуется нечетким множеством с помощью функции принадлежности.

Рассмотрим некоторые конкретные примеры нечетких множеств.

**Пример 1**. Рассмотрим типичную бытовую си­туацию, с которой сталкиваются многие из нас при попытке дать характеристи­ку температуры того или иного напитка. Подобная характеристика обычно ос­новывается исключительно на субъективных ощущениях, например, горячий кофе или чай, холодный квас или кола. Хотя в этом случае неявно используется некоторая шкала температуры, при этом, как правило, не применяется никаких измерительных инструментов.

Применительно к данной ситуации рассмотрим нечеткое множество В, которое будет характеризовать «горячий кофе». В этом случае в качестве множества X есте­ственно взять шкалу температуры, измеренной в градусах Цельсия и заключен­ной в открытом интервале (0 °С, 100 °С). Выбор этого интервала вполне оправдан с физической точки зрения, поскольку именно в этом диапазоне температур кофе потенциально может существовать как напи­ток. Очевидно, что отдельная чашка кофе, скажем х1 с температурой 10°С не может быть признана горячей, поэтому для нее значение функции принадлежно­сти рассматриваемому множеству В будет равно нулю. С другой стороны, другая чашка кофе х2 с температурой 90°С вполне может быть призна­на горячей, поэтому для нее значение функции принадлежности рассматривае­мому множеству В будет равно 1.

Что касается значений температур, заключенных между этими крайними значе­ниями, то ситуация представляется уже не столь однозначной. Более того, она по своей сути является исключительно субъективной и неопределенной, поскольку чашка кофе с температурой 55°С для одного человека может оказаться го­рячей, а для другого - не слишком горячей. Именно в этом и проявляется не­четкость задания соответствующего множества.

**Пример 2.**Представить в виде нечеткого множества понятие “мужчина среднего роста”.

*Решение:*http://matlab.exponenta.ru/fuzzylogic/book1/image5691.gif = 0/155+0.1/160+0.3/165 + 0.8/170 +1/175 +1/180 + 0.5/185 +0/180.

**Пример 3.**Рассмотрим переменную “*скорость автомобиля*”, которая оценивается по шкале “*низкая*", "*средняя*", "*высокая*” и “*очень высокая*".

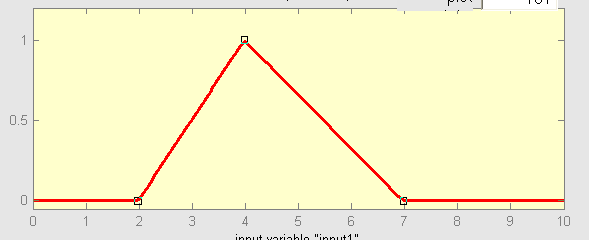
В этом примере лингвистической переменной является “*скорость автомобиля*”, термами - лингвистические оценки “*низкая*", "*средняя*", "*высокая*” и “*очень* *высокая*”, которые и составляют терм–множество.

**Типы функции принадлежности**

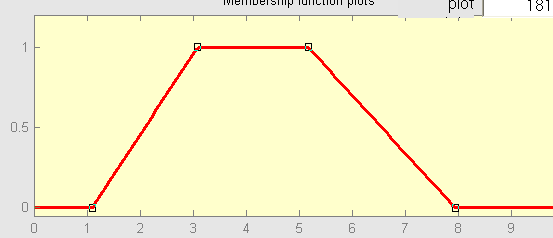
Формальное определение нечеткого множества не накладывает никаких ограничений на выбор конкретной функции принадлежности для его представ­ления. Однако на практике удобно использовать те из них, которые допускают аналитическое представление в виде некоторой простой математической функ­ции. Это упрощает соответствующие численные расчеты.

**Кусочно-линейные функции принадлежности**

В качестве первого типа функций принадлежности рассмотрим функции, кото­рые, как следует из их названия, состоят из отрезков прямых линий, образуя не­прерывную или кусочно-непрерывную функцию. Наиболее характерным приме­ром таких функций являются "треугольная" и "трапециевидная" функции принадлежности. В примере каждая из этих функций задана на множестве Х=[0, 10] (рис.).



**Рис. Треугольная функция принадлежности**



**Рис. Трапециевидная функция принадлежности**

Первая из этих функций принадлежности в общем случае может быть задана аналитически следующим выражением:



где a, b, с — некоторые числовые параметры, принимающие произвольные дей­ствительные значения и упорядоченные отношением: а< b< с.

Применительно к конкретной функции, изображенной на рис.19, значения параметров равны: а=2, b=4, с=7. Параметры a и c характеризуют основание треугольника, а параметр b — его вершину.

Трапециевидная функция принадлежности в общем случае может быть задана аналитически следующим выражением:



где а, b, с, d — некоторые числовые параметры, принимающие произвольные действительные значения и упорядоченные отношением: а< b< с< d.

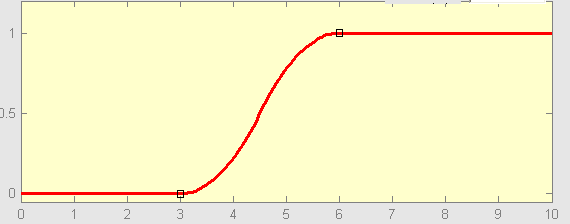
Применительно к конкретной функции, изображенной на рис. 20, значения параметров равны: а=1, b=3, с=5, d=8. Параметры а и d характеризуют нижнее основание трапеции, параметры b и с — верхнее осно­вание трапеции.

**Z-образные и S-образные функции принадлежности**

Эти функции принадлежности также получили свое название по виду кривых, которые представляют их графики. Первая из функций этой группы называется Z-образной кривой и в общем случае может быть задана аналитически следующим выражением:



где а, b — некоторые числовые параметры, принимающие произвольные дейст­вительные значения и упорядоченные отношением: а<b. График этой функции для некоторого нечеткого множества на множестве ¥=[0, 10] изображен на рис. , при этом значения параметров соответственно равны а=3, b=6.



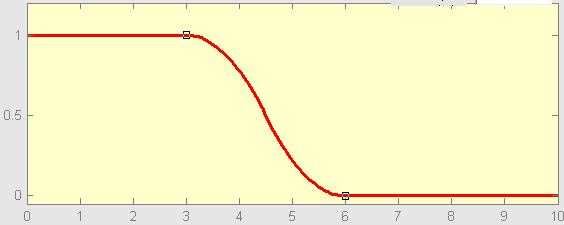
**Рис. Z-образная функция принадлежности**

Эта функция используются для представления таких свойств нечетких множеств, которые характеризуются неопределенностью типа: «малое количество», «небольшое значение», «незначительная величина». Общим для всех таких ситуаций является слабая степень проявления того или иного качественного или количественного признака. Особенность нечеткого моделирования при этом заключается в представлении соответствующих нечет­ких множеств с помощью невозрастающих (монотонно убывающих) функций принадлежности.

Вторая из функций рассматриваемой группы называется S-образной кривой и в общем случае может быть задана аналитически следующим выражением:



На рис. 22 показан график этой функции при a=3, b=6.



**Рис. S-образная функция принадлежности**

**Нечеткая логика**

В четком логическом выводе действуют правила:

ЕСЛИ булево условие, ТО действие или

ЕСЛИ посылка, ТО заключение

Здесь посылка и заключение – булевские величины, принимающие значения true или false.

Нечеткая логика предназначена для формализации человеческих способностей к приближенным рассуждениям, которые позволяют более адекват­но описывать ситуации с неопределенностью. В отличие от классической логики в нечеткой логике истин­ность рассуждений оценивается в некоторой степени, которая может принимать различные значения.

Чтобы иметь возможность выражать неопределенные знания, необходима такая логическая система, которая позволяет некоторой величине иметь истин­ностное значение, отличающееся от бинарного. Для этого необходимо расширить множество истинностных значений {true, false} и позволить величинам принимать некоторые дополнительные значения истинности. Одним из пер­вых логиков, предложивших в 1930 г. вариант многозначной логической систе­мы, отличающийся от классической бинарной логики, был польский математик Ян Лукасевич (1878—1956). В трехзначной логике Лукасевича используется три истинностных значения: {0, 0.5, 1}, где значение 0 интерпретируется как ложь, 1 — как истина, а число 0.5— как возможно. Так, например, высказывание «Сборная России по футболу выйдет в 1/4 финала на предстоящем чемпионате Европы» до начала чемпионата не может быть оцене­но ни как истинное, ни как ложное. Именно по этой причине более адекватным ответом на вопрос об его истинности будет использование трехзначной логики с соответствующей интерпретацией истинности в форме значения возможно.

В предложенном Л. Заде варианте нечеткой логики множество истинностных значений величин обобщается до интервала действительных значений [0, 1], что позволяет величине принимать любое значение истинности из этого интервала. Это численное значение является количественной оценкой сте­пени истинности величины, относительно которой нельзя с полной уверен­ностью заключить о ее истинности или ложности. Использование в качестве множества истинностных значений интервала [0, 1] позволяет построить логиче­скую систему, в рамках которой оказалось возможным выполнять рассуждения с неопределенностью и оценивать истинность высказываний типа: «Скорость ав­томобиля довольно высокая», «Давление в системе весьма значительное», «Высота полета самолета предельно низкая».

Исходным понятием нечеткой логики является понятие нечеткого высказывания.

В общем случае нечетким высказыванием называется повествовательное предложе­ние, выражающее законченную мысль, относительно которой можно судить об ее истинности или ложности только с некоторой степенью уверенности.

Нечеткие высказывания для удобства будем обозначать теми же буквами, что и нечеткие множества: A, В, С.

Ниже приводится несколько примеров нечетких высказываний:

1. О. Бендер имеет довольно высокий рост.
2. Завтра будет пасмурная погода.
3. 3.3 — малое число.
4. Возможно, нам подадут горячий кофе.

Содержательно неопределенность нечетких высказываний может иметь различную природу. Так, например, неопределенность оценки истинности в высказы­вании (1) связана с нечеткостью определения понятия «высокий рост», которое является нечеткой переменной. Аналогичный характер неопреде­ленности имеет нечеткое высказывание (3), связанное с определением не­четкой переменной «малое число». Что касается высказываний (2) и (4), то здесь кроме определения нечетких пере­менных «пасмурная погода» и «горячий кофе» следует оценить их истинность от­носительного некоторого момента времени в будущем. Общим для всех этих вы­сказываний является то обстоятельство, что относительно их истинности мы можем судить лишь с некоторой степенью, количественно оцениваемой действи­тельным числом из интервала [0, 1].

**Нечеткие логические операции**

Напомним основные положения обычной (булевой) логики. Рассмотрим два утверждения A и B, каждое из которых может быть истинным или ложным, т.е. принимать значения "1" или "0" (true или false). Для этих двух утверждений всего существует http://matlab.exponenta.ru/fuzzylogic/book1/images_1_7_3/image001.gifразличных логических операций, из которых рассмотрим: И (http://matlab.exponenta.ru/fuzzylogic/book1/images_1_7_3/image002.gif), ИЛИ (http://matlab.exponenta.ru/fuzzylogic/book1/images_1_7_3/image003.gif), импликация (). Таблицы истинности для этих операций приведены в табл.

Таблицы истинности булевой логики

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| A | B | AB | AB | AB |
| 0 | 0 | 0 | 0 | 1 |
| 0 | 1 | 1 | 0 | 1 |
| 1 | 0 | 1 | 0 | 0 |
| 1 | 1 | 1 | 1 | 1 |

Правила выполнения нечетких логических операций получают из булевых логических операций с помощью принципа обобщения.

Обозначим нечеткие логические переменные через http://matlab.exponenta.ru/fuzzylogic/book1/images_1_7_3/image013.gifи http://matlab.exponenta.ru/fuzzylogic/book1/images_1_7_3/image014.gif, а функции принадлежности, задающие истинностные значения этих переменных через http://matlab.exponenta.ru/fuzzylogic/book1/images_1_7_3/image015.gifи http://matlab.exponenta.ru/fuzzylogic/book1/images_1_7_3/image016.gif, http://matlab.exponenta.ru/fuzzylogic/book1/images_1_7_3/image017.gif.

**Нечеткие логические операции** И (http://matlab.exponenta.ru/fuzzylogic/book1/images_1_7_3/image002.gif), ИЛИ (http://matlab.exponenta.ru/fuzzylogic/book1/images_1_7_3/image003.gif), НЕ (http://matlab.exponenta.ru/fuzzylogic/book1/images_1_7_3/image018.gif) и импликация () выполняются по таким правилам:

http://matlab.exponenta.ru/fuzzylogic/book1/images_1_7_3/image019.gif;

http://matlab.exponenta.ru/fuzzylogic/book1/images_1_7_3/image020.gif;

http://matlab.exponenta.ru/fuzzylogic/book1/images_1_7_3/image021.gif;

http://matlab.exponenta.ru/fuzzylogic/book1/images_1_7_3/image022.gif.

Нечеткой базой знаний называется совокупность нечетких правил "Если - то", определяющих взаимосвязь между входами и выходами исследуемого объекта. Обобщенный формат нечетких правил такой:

ЕСЛИ посылка правила, ТО заключение правила.

При этом посылка и заключение – термы нечетких переменных

Посылка правила представляет собой утверждение типа "x есть низкий", где "низкий"  - это терм (лингвистическое значение), заданный нечетким множеством на универсальном множестве лингвистической переменной x.

Заключение правила представляет собой утверждение типа "y есть d", в котором значение выходной переменной (d) может задаваться:

1. нечетким термом: "y есть высокий";
2. классом решений: "y есть бронхит"
3. четкой константой: "y=5";
4. четкой функцией от входных переменных: "y=5+4\*x".

Пример Следующая нечеткая база знаний описывает зависимость между возрастом водителя (x) и возможностью дорожно-транспортного происшествия (y):

Если x = Молодой, то y = Высокая;

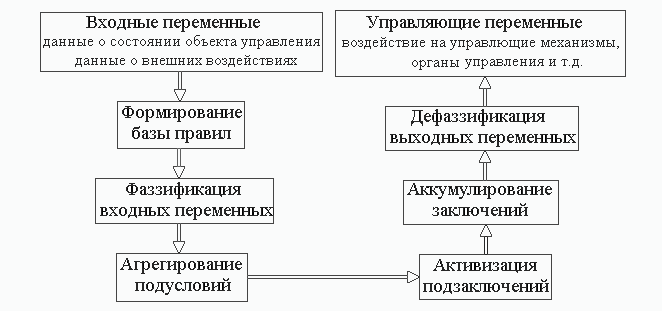
Если x = Средний, то y = Низкая;

Если x = Очень старый, то y = Высокая

**Этапы нечеткого вывода**

**Система нечеткого вывода** – это процесс получения нечетких заключений о требуемом управлении объектом на основе нечетких условий или предпосылок, представляющих собой информацию о текущем состоянии объекта.

Этот процесс соединяет в себе все основные концепции теории нечетких множеств: функции принадлежности, лингвистические переменные, методы нечеткой импликации. Разработка и применение систем нечеткого вывода включает в себя ряд этапов, реализация которых выполняется на основе рассмотренных ранее положений нечеткой логики (рис.).



**Рис. Этапы нечеткого вывода**

**Фаззификация**(введение нечеткости) – это установка соответствия между численным значением входной переменной системы нечеткого вывода и значением функции принадлежности соответствующего ей терма лингвистической переменной. На этапе фаззификации значениям всех входным переменным системы нечеткого вывода, полученным внешним по отношению к системе нечеткого вывода способом ставятся в соответствие конкретные значения функций принадлежности соответствующих лингвистических термов, которые используются в посылках нечетких продукционных правил.

Пусть входная лингвистическая переменная – температура пара на выходе из котла. Рассмотрим процесс фаззификации следующих трех нечетких высказываний:

- температура пара низкая;

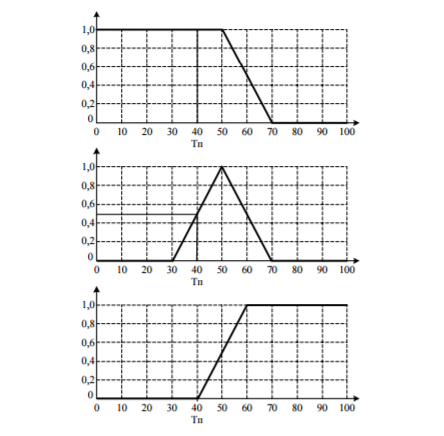
- температура пара средняя;

- температура пара высокая. Пусть температура пара равна Тп = 40°С. Таким образом, при фаззификации получаем степени истинности элементарных нечетких высказываний: - температура пара низкая – 0;

- температура пара средняя – 0,47;

- температура пара высокая – 0.

На рис. показан пример фаззификации входной лингвистической переменной «температура пара на выходе из котла»



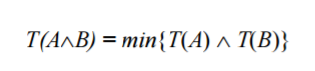
**Рис. Пример фаззификации входной лингвистической переменной «температура пара на выходе из котла» для трех нечетких высказываний**

**Агрегирование**– это процедура определения степени истинности условий по каждому из правил системы нечеткого вывода. При этом используются полученные на этапе фаззификации значения функций принадлежности термов лингвистических переменных, составляющих посылки нечетких продукционных правил.

Если условие нечеткого продукционного правила является простым нечетким высказыванием, то степень его истинности соответствует значению функции принадлежности соответствующего терма лингвистической переменной.

Если условие представляет составное высказывание, то степень истинности сложного высказывания определяется на основе известных значений истинности составляющих его элементарных высказываний при помощи введенных ранее нечетких логических операций.

При этом для определения результата нечеткого логического «И» может быть использована следующая формула:



а для определения результата нечеткой дизъюнкции или логического «ИЛИ» может быть использована формула:



Для иллюстрации этапа агрегирования рассмотрим следующий пример. Первое нечеткое высказывание: «температура пара средняя» и «расход пара низкий». Второе нечеткое высказывание: «температура пара средняя» или «расход пара низкий». Тогда агрегирование первого нечеткого высказывания дает в результате число 0,47, а агрегирование второго нечеткого высказывания дает в результате число 0,78.

**Активизация** – это процедура нахождения степени истинности каждого из элементарных логических высказываний, составляющих заключения всех нечетких продукционных правил. Поскольку заключения делаются относительно выходных лингвистических переменных, то степеням истинности элементарных подзаключений при активизации ставятся в соответствие элементарные функции принадлежности.

Если заключение нечеткого продукционного правила является простым нечетким высказыванием, то степень его истинности равна алгебраическому произведению весового коэффициента и степени истинности посылки нечеткого продукционного правила.

Если заключение представляет составное высказывание, то степень истинности каждого из элементарных высказываний равна алгебраическому произведению весового коэффициента и степени истинности посылки нечеткого продукционного правила.

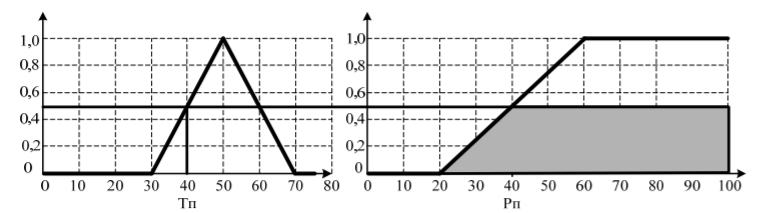
Если весовые коэффициенты продукционных правил не указаны явно на этапе формирования базы правил, то их значения по умолчанию равны единице.

Функции принадлежности μ (y) каждого из элементарных подзаключений всех продукционных правил находятся при помощи одного из методов нечеткой композиции:

* min–активизация – μ( y )=min{ c,μ( x ) } ;
* prod-активизация – μ (y) =c μ (x) ;
* average-активизация – μ (y) =0.5(c + μ (x) ) ;

где μ (x) и c – соответственно функции принадлежности термов лингвистических переменных и степени истинности нечетких высказываний, образующих соответствующие заключения нечетких продукционных правил.

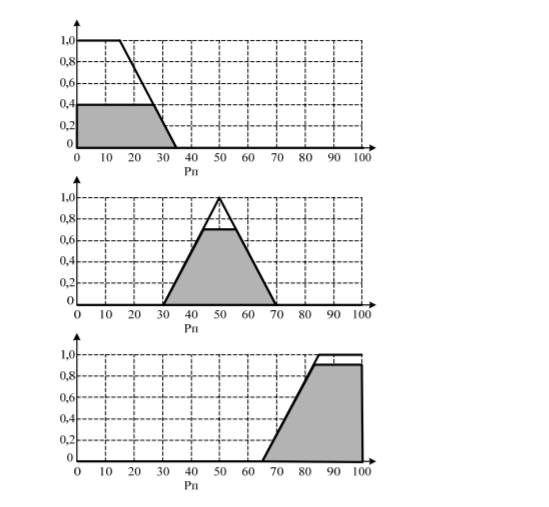
На рис. приведен пример процесса активизации заключения в правиле нечеткой продукции: ЕСЛИ «температура пара средняя» ТО «расход пара низкий». В данном случае входной лингвистической переменной является нечеткое высказывание: «температура пара средняя», а выходной лингвистической переменной: «расход пара низкий».



**Рис. Пример активизации заключени**я

Пусть температура пара равна Тп = 40°С, тогда результат (вид функции принадлежности), полученный методом min-активизации, изображен на рис. 25 темным цветом.

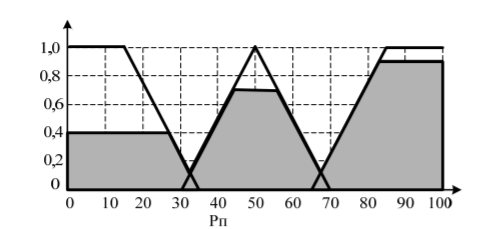
**Аккумуляция** – это процесс нахождения функции принадлежности для каждой из выходных лингвистических переменных. Цель аккумуляции состоит в объединении всех степеней истинности подзаключений для получения функции принадлежности каждой из выходных переменных. Результат аккумуляции для каждой выходной лингвистической переменной определяется как объединение нечетких множеств всех подзаключений нечеткой базы правил относительно соответствующей лингвистической переменной. Для иллюстрации данного этапа рассмотрим пример аккумуляции заключения для выходной лингвистической переменной «расход пара». Пусть функции принадлежности нечетких множеств имеют следующий вид, показанный на рис.



**Рис. Вид функций принадлежности подзаключений**

**выходной переменной**

Следовательно, результирующая функция принадлежности для выходной лингвистической переменной «расход пара» имеет вид, представленный на рис.



**Рис. Результирующая функция принадлежности для выходной лингвистической переменной «расход пара»**

**Дефаззификация** – это процесс перехода от функции принадлежности выходной лингвистической переменной к её четкому (числовому) значению.

Цель дефаззификации заключается в том, чтобы, используя результаты

аккумуляции всех выходных лингвистических переменных, получить количественное значение каждой из выходных переменных, которое может быть использовано в дальнейшем.

Этап дефаззификации считается законченным, когда для каждой из выходных лингвистических переменных будут определены итоговые количественные значения в виде некоторого действительного числа, т.е. в виде: у1, у2, …, уs, где s - общее количество выходных лингвистических переменных в базе правил системы нечеткого вывода.

Для выполнения численных расчетов на этапе дефаззификации могут быть использованы следующие методы.

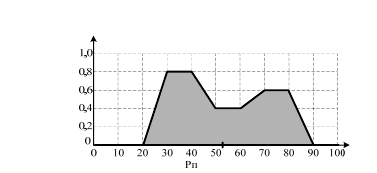
**Метод центра тяжести**. Этот метод приведения к чёткости называется

центроидным и рассчитывается по следующей формуле:



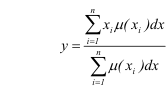
где переменная y – результат дефаззификации; x – переменная, соответствующая выходной лингвистической переменной и принимающая значения от x = min до x = max; min и max – левая и правая точки интервала носителя нечёткого множества; µ(x) – функция принадлежности нечёткого множества.

Пример дефаззификации методом центра тяжести функции принадлежности выходной лингвистической переменной «расход пара» изображен на рис.



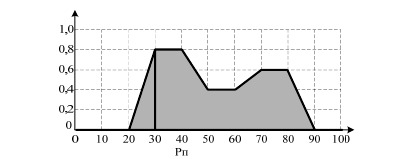
**Рис. Дефаззификация методом центра тяжести**

Результат дефаззификации в этом случае уi = 53кг/ч (приближенное значение). Для одноточечных множеств метод центра тяжести рассчитывается по формуле:



где n – число одноточечных нечётких множеств, каждое из которых характеризует единственное значение рассматриваемой выходной лингвистической переменной.

Пример дефаззификации методом центра тяжести для одноточечных множеств функции принадлежности выходной лингвистической переменной «расход пара» изображен на рис.



**Рис. Дефаззификация методом центра тяжести для одноточечных множеств функции принадлежности**

Результат дефаззификации в этом случае уi = 62кг/ч (приближенное значение).

**Метод левого модального значения**. Левое модальное значение определяется по формуле:



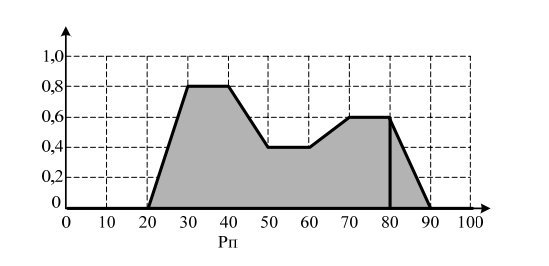
где *xm* – модальное значение (наименьшая из мод, самая левая) значений нечёткого множества.

**Метод правого модального значения**. Правое модальное значение  
рассчитывается по формуле:



где *xm* – модальное значение (наибольшая из мод, самая правая) значений нечёткого множества.

Пример дефаззификации методом правого модального значения функции  
принадлежности выходной лингвистической переменной «расход пара» изображен на рис.

****

**Рис. Дефаззификация методом правого модального значения функции принадлежности**

Результат дефаззификации в этом случае *уi* = 80кг/ч (приближенное значение).  
К настоящему времени наибольшее применение получили   
алгоритмы нечеткого вывода Мамдани и Сугено. Первый из них чаще всего используется для реализации экспертных систем на нечеткой системе правил, второй – для решения задач классификации.

**Функции Fuzzy Logic Toolbox для работы с системой нечеткого вывода**

|  |  |
| --- | --- |
|  | **Создание новой системы нечеткого логического вывода** |

**fis = newfis(fis\_name, fis\_type)**

Создает в рабочей области новую систему нечеткого логического вывода. Функция **newfis** может иметь до двух входных аргументов:

1. fis\_name - наименование системы нечеткого логического вывода;
2. fis\_type - тип системы нечеткого логического вывода. Допустимые значения:'mamdani' - система типа Мамдани (значение по умолчанию);'Sugeno' - система типа Сугено;

Пример:

a=newfis('new\_fuzzy\_system')

В рабочей области создается структура a, содержащая систему нечеткого логического вывода с именем

|  |  |
| --- | --- |
|  | **Загрузка системы нечеткого логического вывода с диска** |

**fis = readfis(‘имя файла’)**

Загрузка в рабочую область системы нечеткого логического вывода из файла с именем имя файла.fis . Выходным аргументом функции **readfis**является структура системы нечеткого логического вывода fis.

Пример

a=readfis(‘tipper’)

Загружается в рабочую область демонстрационная система нечеткого логического вывода “Tipper”.

|  |  |
| --- | --- |
|  | **Запись системы нечеткого логического вывода на диск** |

**writefis (fis, имя файла)**

Функция **writefis** сохраняет систему нечеткого логического вывода на диске. Она имеет два входных аргумента:

1. fis – обязательный аргумент, задающий систему нечеткого логического вывода;
2. filename – имя файла, в котором будет сохранена система нечеткого логического вывода;

|  |  |
| --- | --- |
|  | **Добавление переменной в систему нечеткого логического вывода** |

**FIS= addvar (FIS, varType, varName, varBound)**

Переменную можно добавить только к существующей в рабочей области MatLab системе нечеткого логического вывода.

Функция **addrvar** имеет четыре входных аргумента:

1. FIS – идентификатор системы нечеткого логического вывода в рабочей области MatLab;
2. varType – тип добавляемой переменной. Допустимые значения - ‘input’  - входная переменная и ‘output’ – выходная переменная;
3. varName – наименование добавляемой переменной. Задается в виде строки символов;
4. varBound – вектор, задающий диапазон изменения добавляемой переменной.

Порядковый номер переменной в системе нечеткого логического вывода соответствует порядку добавления с помощью функции **addvar**, т.е. первая добавленная переменная будет иметь порядковый номер 1. Входные и выходные переменные нумеруются независимо.

Пример

FIS=addvar(FIS, ‘input’, ‘Рост’, [155 205])

Строка добавляет в систему нечеткого логического вывода FIS входную переменную ‘Рост’, заданную на интервале [155 205].

|  |  |
| --- | --- |
| **Добавление функции принадлежности к системе нечеткого вывода** |  |

**FIS=addmf(FIS, varType, varIndex, mfName, mfType, mfParams)**

Функцию принадлежности можно добавить только к существующей в рабочей области MatLab системе нечеткого логического вывода. Другими словами, система нечеткого логического вывода должна быть каким-то образом загружена в рабочую область или создана с помощью функции **newfis**. Функция **addmf** имеет шесть входных аргументов:

1. FIS – идентификатор системы нечеткого логического вывода в рабочей области MatLab;
2. varType – тип переменной, к которой добавляется функция принадлежности. Допустимые значения - ‘input’  - входная переменная и ‘output’ – выходная переменная;
3. varIndex – порядковый номер переменной, к которой добавляется функция принадлежности;
4. mfName – наименование добавляемой функции принадлежности (терм). Задается в виде строки символов;
5. mfType – тип добавляемой функции принадлежности. Задается в виде строки символов;
6. mfParams – вектор параметров добавляемой функции принадлежности.

Порядковый номер функции принадлежности в системе нечеткого логического вывода соответствует порядку добавления с помощью функции **addmf**, т.е. первая добавленная функция принадлежности всегда будет иметь порядковый номер 1. С помощью функции **addmf** невозможно добавить функцию принадлежности к несуществующей переменной. В этом случае необходимо вначале добавить переменную к системе нечеткого логического вывода с помощью функции addvar.

Пример.

FIS=addmf(FIS, ‘input’, 1, ‘низкий’, ‘trimf’, [150, 155, 170])

Строка добавляет в терм-множество первой входной переменной нечеткой системы FIS терм ‘низкий’ с треугольной функцией принадлежности с параметрами [150, 155, 170].

|  |
| --- |
| **Добавление правил к нечеткой системе вывода в текстовом виде** |

**fis = parsrule (fis, inrulelist)**

Функция **parsrule** предназначена для ввода правил в нечеткую базу знаний. При этом удаляются ранее существующие в базе знаний правила. Функция **parsrule** имеет два входные аргумента:

1. fis - идентификатор исходной системы нечеткого логического вывода;
2. inrulelist - список правил "если - то". Правила можно задавать в виде предложений на английском языке. При задании правил на естественном языке необходимо использовать следующие ключевые слова: "if", "and", "or", "then", "is", "not",которые эквиваленты русским словам "если", "и", "или", "то", "есть", "не", соответственно. Весовой коэффициент можно указать в конце правила. По умолчанию значение весового коэффициента равно 1. Список правил задается в виде матрицы, каждая строчка которой определяет одно правило;

Пример

infis=readfis('tipper');  
r1='if service is good then tip is average ';

r2='if service is poor and food is rancid then tip is cheap ';

r3='if service is excellent and food is delicious then tip is generous';

inrulelist=[r1; r2; r3];

fis=parsrule(fis,inrulelist)

Загружается в рабочую область демонстрационная система нечеткого логического вывода "Tipper", задающая зависимость размера чаевых от качества пищи и уровня сервиса в ресторане. Затем формируется новая база знаний, содержащая следующие правила:

"if service is good then tip is average";

"if service is poor and food is rancid then tip is cheap";

"if service is excellent and food is delicious then tip is generous'".

|  |  |  |
| --- | --- | --- |
| |  |  | | --- | --- | | **Выполнение нечеткого логического вывода** |  |   **output = evalfis(input, fis)**  Выполняет нечеткий логический вывод. Функция **evalfis**имеет два входных аргумента:   1. input – матрица значений входных переменных, для которых необходимо выполнить нечеткий логический вывод. Матрица должна иметь размер M x N, где N – количество входных переменных; M – количество входных данных. Каждая строчка матрицы представляет один вектор значений входных переменных; 2. fis – идентификатор системы нечеткого логического вывода;   Функция **evalfis**имеет один выходной аргумент:   1. output – матрица значений выходных переменных, получаемая в результате нечеткого логического вывода для матрицы входных значений input. Матрица имеет размер M x L, где M – количество входных данных; L – количество выходных переменных в fis;   Пример  Первая строчка загружает систему нечеткого логического вывода tipper, предназначенную для определения процента чаевых в ресторане.  Вторая строчка рассчитывает размер чаевых, в случае если service=3 и food=8.  fis = readfis('tipper');tip = evalfis([3 8], fis)  **Изображение структуры системы нечеткого логического вывода** |

**plotfis (fis)** функция выводит в графическом окне структуру системы нечеткого логического вывода fis.

Входные переменные системы изображаются в левой части графического окна, выходные переменные – в правой части, в центре – база знаний.

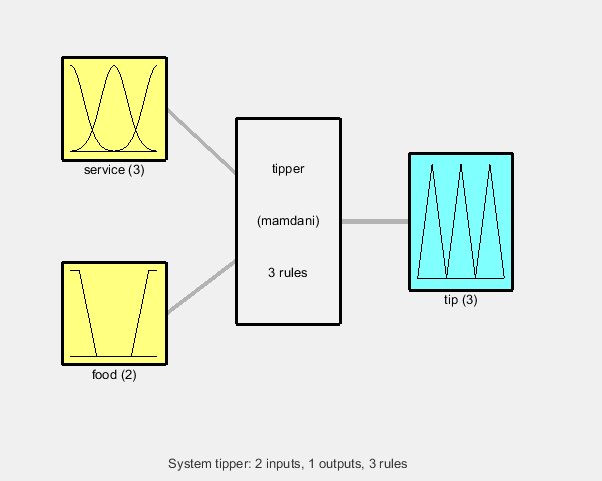
В графическом окне выводится наименование и тип системы нечеткого логического вывода, количество термов для каждой переменной и количество правил. Также изображаются графики функций принадлежности всех термов.

Пример

a=readfis(‘tipper’);  
plotfis(a)

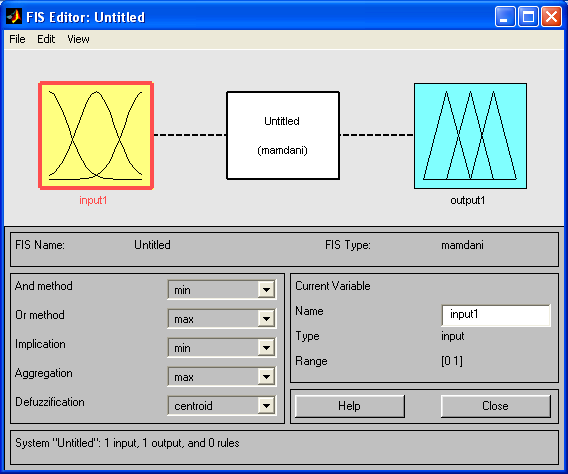
Вывод структуры демонстрационной системы нечеткого логического вывода

“Tipper”

****

**Редактор систем нечеткого вывода**

Редактор систем нечеткого вывода FIS является ос­новным средством, которое используется для создания или редактирования сис­тем нечеткого вывода в графическом режиме. Редактор FIS может быть открыт с помощью ввода функции **fuzzy** в окне команд. Эта функция предоставляет пользователю возможность задавать и редактировать на высоком уровне свойства системы нечеткого вывода, такие как число входных и выходных переменных, тип системы нечеткого вывода, используемый метод дефаззификации и т. д. Редактор FIS обладает графическим интерфейсом и позволяет вызывать все дру­гие редакторы и программы просмотра систем нечеткого вывода. В верхней части рабочего интерфейса редактора FIS изображается диаграмма, представляющая в визуальной форме входы и выходы системы нечеткого выво­да, в центре которых находится так называемый процессор нечетких правил. Щелчок на прямоугольнике с изображением входа или выхода выделяет соот­ветствующую переменную и делает ее текущей. Прямоугольник текущей пере­менной при этом выделяется красным цветом.



Двойной щелчок на прямоугольнике с изображением входной или выходной пе­ременной вызывает редактор функций принадлежности с загруженной в него соответствующей переменной.

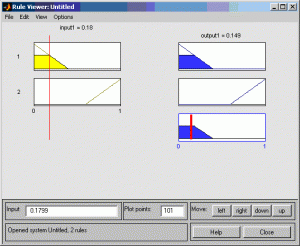
Редактор FIS имеет главное меню, которое позволяет пользователю вызывать другие графические средства работы с системой нечеткого вывода FIS, загру­жать и сохранять структуру FIS во внешних файлах и т. д. Рассмотрим назначе­ние пунктов меню редактора FIS.

* Пункт меню File редактора FIS содержит следующие операции:
* New FIS... — позволяет выбрать тип задаваемой новой системы нечеткого вывода: Mamdani — типа Мамдани или Sugeno — типа Сугено. При этом задаваемая система нечеткого вывода не имеет ни входных, ни выходных переменных, а ее имя задается по умолчанию как Untitled;
* Import — позволяет загрузить в редактор FIS существующую систему не­четкого вывода From Disk... из внеш­него файла. В этом случае вызывается стандартное диалоговое окно открытия внешнего файла с диска;
* Export — позволяет сохранить редактируемую систему нечеткого вывода То Disk... во внешнем файле. В этом случае вызывается стандартное диалоговое окно сохранения файла на диске;

### Просмотр правил

Пункт меню **View,Rules** обеспечивает ввод входных переменных и получение выходных для проверки работоспособности разработанной системы нечеткого вывода.

Значения переменных, используемые в правилах, изображаются в окне (рис. ) в виде прямоугольников с графиком их функций принадлежности.

[](http://fiseudocs.ru/wp-content/uploads/2010/10/pic_7_3.gif)

**Рис. Окно Rule Viewer**

Прямоугольники, относящиеся к одной переменной, размещаются в столбце с её именем.

Число прямоугольников в столбце определяется числом правил в базе знаний. Столбец выходной переменной дополняется прямоугольником  с результатом дефаззификации выбранным методом.

Вертикальная линия, пересекающая прямоугольники столбца входной переменной, показывает введённое значение числового значения нечёткой переменной, которое определяется заданием диапазона числовой шкалы (“Range”) и его функции принадлежности на ней.

Числовое значение может быть введено как в виде числа в поле ввода Input в нижней части окна, либо перетаскиванием вертикальной линии внутри столбца, после его выделения щелчком. Вводимое значение может быть проконтролировано по значению в цифрах вверху столбца.  
Результат можно увидеть вверху столбца выходной переменной.

### Просмотр поверхности нечёткого вывода

Для оценки взаимовлияния значений переменных на результаты работы системы нечеткого вывода можно воспользоваться пунктом меню **View, Surface.**

**Пример разработки системы нечеткого вывода**

В качестве примера разработки системы нечеткого вывода рассмотрим следующую нечеткую модель.

"Чаевые в ресторане". Рассмотрим ситуацию в ресторане, при которой после окончания обслуживания посетителя принято оставлять официанту чаевые. Основываясь на устоявшихся обычаях и интуитивных представлениях посетителей ресторанов, величина суммы чаевых не является постоянной и зависит, например, от качест­ва обслуживания и качества приготовления заказанных блюд.

Задача состоит в том, чтобы разработать некоторую экспертную систему, кото­рая была бы реализована в виде системы нечеткого вывода и позволяла бы оп­ределять величину чаевых на основе субъективных оценок посетителей качества обслуживания и качества приготовления заказанных блюд.

Эмпирические знания о рассматриваемой проблемной области могут быть пред­ставлены в форме следующих эвристических правил продукций:

1. Если обслуживание плохое или ужин подгоревший, то чаевые — малые.
2. Если обслуживание хорошее, то чаевые — средние.
3. Если обслуживание отличное или ужин превосходный, то чаевые — щедрые.

В качестве входных параметров системы нечеткого вывода будем рассматривать 2 нечеткие переменные: "качество обслуживания" (service) и "качество ужина" (supper), а в ка­честве выходного параметра — нечеткую переменную "величи­на чаевых" (tip).

В качестве множества первой переменной "качество обслу­живания" будем использовать множество Т1={"плохое", "хорошее", "отличное"}, а в качестве множества второй переменной "качество ужина" будем использовать множество Т2={"подгоревший"', "превосходный"}. В качестве множества выходной переменной "величина чаевых" будем использовать множество Т3={"малые", "средние", "щедрые"}. При этом каждый из термов первой и второй входной переменной будем оценивать по 10-балльной порядковой шкале, при которой числу 0 соответствует наихудшая оценка, а числу 10 — наилучшая оценка. Что касается термов выходной переменной, то будем предполагать, что малые чаевые составляют около 5% от стоимости заказанных блюд, средние чае­вые — около 15%, а щедрые чаевые — около 25%.

С учетом сделанных уточнений, рассмотренная субъективная информация о ве­личине чаевых может быть представлена в форме 3-х нечетких правил следующего вида (система нечеткого вывода типа Мамдани):

ПРАВИЛО\_1: ЕСЛИ "качество обслуживания плохое" ИЛИ "ужин подгоревший" ТО "величина чаевых малая"

ПРАВИЛО\_2: ЕСЛИ "качество обслуживания хорошее" ТО "величина чаевых средняя"

ПРАВИЛО\_3: ЕСЛИ "качество обслуживания отличное" ИЛИ "ужин превосход­ный" ТО "величина чаевых щедрая"

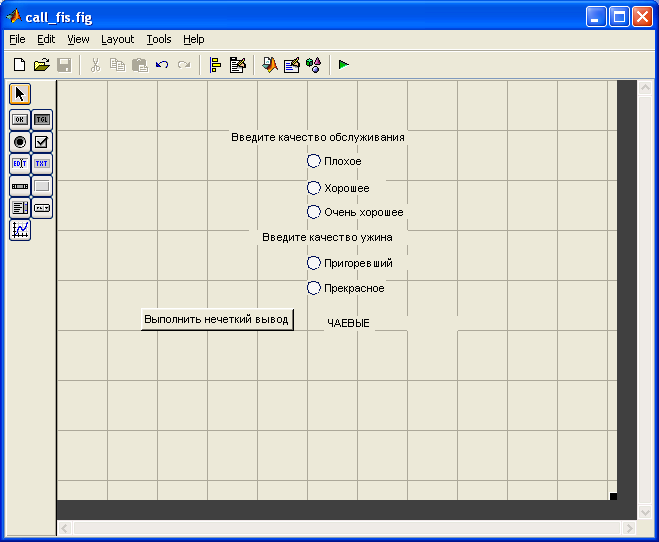
**Разработка графического приложения для пользователя**

Создадим графическое приложение, в котором предусмотрим ввод входных переменных в символьном виде, преобразование их в числовые значения (для каждого символьного значения берем среднее значение соответствующего интервала), выполнение системы нечеткого вывода для этих значений и получение числового значения выходной переменной.

Затем значение выходной переменной анализируется и переводится в один из осмысленных термов, который выводится на экран. Так проводится экспертная консультация

Разместим на форме 4 поля StaticText (две подсказки для двух вводимых величин, одну подсказку для результата и одно поле для выводимого результата консультации), 3 поля RadioButton (для выбора значения первой входной величины ), 2 поля RadioButton (для выбора значения второй входной величины ) и кнопку Push Button, по нажатии которой происходит обращение к функции отклика (Callback).

Окно примет вид:



При нажатии правой кнопкой мыши по любому элементу на форме или по самой форме появляется контекстное меню

Для нашего примера функция CallBack кнопки имеет вид:

%Прием выбранных значений входных параметров

r11=get(handles.radiobutton1,'Value');

r12=get(handles.radiobutton2,'Value');

r13=get(handles.radiobutton3,'Value');

r21=get(handles.radiobutton4,'Value');

r22=get(handles.radiobutton5,'Value');

%Фаззификация

if (r11==1)

r1=1

end

if (r12==1)

r1=5.5

end

if (r13==1)

r1=8.5

end

if (r21==1)

r2=1.5

end

if (r22==1)

r2=6.5

end

%загрузка системы нечеткого вывода из файла

fis=readfis('qualnew');

%выполнение нечеткого вывода для заданных значений входных переменных

y=evalfis([r1 r2],fis)

%Дефаззификация

if (y<=10)

ys='малые';

elseif (y<=20)

ys='средние';

else

ys='щедрые';

end

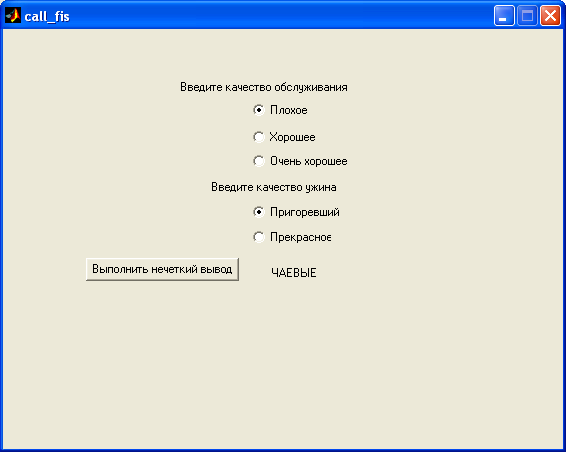
%Запись значения на экран

set(handles.text3,'String',ys);

Выполнение диалоговой программы начинается при нажатии кнопки  на форме

Пример диалога:

До нажатия кнопки:



После нажатия кнопки:

