Министерство науки и высшего образования Российской Федерации федеральное государственное бюджетное образовательное учреждение высшего образования

«Курганский государственный университет» Кафедра «Организация и безопасность движения»

Информационные технологии на автомобильном транспорте

МЕТОДИЧЕСКИЕ УКАЗАНИЯ к выполнению курсовой работы для студентов всех форм обучения направления 23.03.01 «Технология транспортных процессов»



Кафедра: «Организация и безопасность движения»

Дисциплина: «Информационные технологии на автомобильном транспорте» (направление 23.03.01 «Технология транспортных процессов»)

Составил: старший преподаватель Н.С. Безотеческих

Утверждены на заседании кафедры

«20» января 2018 г.

Рекомендованы методическим советом университета «20» декабря 2017 г.

Введение

Курсовая работа по дисциплине «Информационные технологии на автомобильном транспорте» предусматривает углубление теоретических знаний и приобретения практических навыков функционального проектирование автоматизированных систем управления для автотранспортных организаций, а также построение экспертных систем управления в области транспортных процессов.

Выполнение курсовой работы позволит студенту овладеть профессиональными компетенциями в части повышения эффективности деятельности субъектов автотранспортной деятельности на основе применения современных информационных технологий.

Решение поставленных задач в курсовом проектировании обеспечит формирование межпредметных связей профессиональных дисциплин, в том числе «Организация и безопасность движения», «Технические средства организации дорожного движения», а также возможность эффективных решений на основе информационных технологий в ходе итоговой аттестации бакалавра – выпускной квалификационной работе.

1 Общие указания

1.1 Цель курсовой работы

Курсовая работа по дисциплине «Информационные технологии на автомобильном транспорте» выполняется с целью закрепления и углубления теоретических знаний, полученных при изучении курса, приобретения практических навыков функционального проектирования автоматизированных систем управления для автотранспортных организаций, а также построения экспертных систем управления в области транспортных процессов.

Курсовая работавыполняется студентом согласно индивидуальному заданию. В курсовой работе студенту необходимо разработать информационные системыавтотранспортной организации, на основе нечеткой логики, а также основе автоматизированной обработке данных.

Студенты получают практические навыки проектирования информационных систем (ИС), а также разработки экспертных систем на основе интеллектуальных технологий.

1.2 Задание на курсовую работу

Задание на курсовую работу выдается руководителем каждому студенту индивидуально на специальном бланке и содержит:

- сведения о студенте, руководителе и сроках проектирования;
- основные разделы курсовой работы и необходимые дополнительные сведения;
 - наименование теоретического вопроса и практической задачи;

- условия решения практической задачи.

1.3 Объем, содержание и оформление

Курсовая работа состоит из расчетно-пояснительной записки формата А4 (210х297 мм).

Порядок расположения материала в расчетно-пояснительной записке:

Титульный лист

Задание на курсовую работу

Содержание

Введение

1 Разработка базы данных

2 Проектирования экспертной системы на основе нечеткой логики

Заключение

Список литературы

Приложения

Требования к содержанию разделов

Содержание

Включает введение, наименование всех разделов, подразделов, пунктов (если они имеют название), заключение, список используемых источников, приложение с указанием страниц, с которых начинаются эти элементы работы.

Введение

В данном разделе рассматривается актуальность решаемых вопросов в курсовой работе, в части роли специальных информационных систем, экспертных систем и автоматизированных систем управления (АСУ) в сфере автомобильного транспорта, рекомендуется завершать данный раздел целью работы.

Заключение

В этом разделе формулируются все основные результаты и выводы по рассматриваемым вопросам в работе, обычно заключение не превышает одной страницы.

Список использованных источников

В список использованной литературы включаются лишь источники, которые были использованы студентом, с обязательным указанием ссылок в тексте.

Основные разделы

Текст излагается от третьего лица, грамотно, кратко и четко. Необходимо обращать внимание на логическую последовательность изложения, правильность построения предложений, выделение абзацев.

Каждый раздел необходимо заканчивать краткими, четко сформулированными выводами, согласованными с задачами, поставленными в начале раздела; после чего нужен переход к следующему. В тексте нежелательно применять для одного и того же понятия различные технические термины, близкие по смыслу (синонимы), а также иностранные слова и термины при наличии равнозначных в русском языке.

Требования к оформлению

Курсовая работа оформляется в соответствии с методическими рекомендациями по подготовке рукописей к изданию [1].

Более подробно требования к оформлению изложены в методических указаниях к оформлению текстовой части курсовых и дипломных проектов для студентов направления (специальностей) 190600 (190601, 190603) [2] и ГОСТ 2.105-95 ЕСКД. Общие требования к текстовым документам [3].

Список использованных источников оформляется по ГОСТ Р 7.0.5-2008 СИ-БИД. Библиографическая ссылка. Общие требования и правила составления [4].

Рекомендуется использовать машинописный метод исполнения в программах текстовых редакторов, например, Microsoft Office или Libre Office. Рукописное исполнение пояснительной записки не рекомендуется. При необходимости допускается использование цветной печати отдельных листов работы.

2 Рекомендации по выполнению основных разделов

2.1 Разработка базы данных для автотранспортного предприятия

В первой части курсовой работы студенту необходимо разработать базу данных для автоматизированного рабочего места сотрудника автотранспортного предприятия или организации транспортной сферы с целью повышения эффективности его работы.

Перед разработкой студенту необходимо описать СУБД, в которой будет реализована база данных. Для выполнения задания рекомендуется использовать СУБД Microsoft Office Access или Libre Office Base. После описания СУБД необходимо описать особенности работы APMa.

Выбор задания осуществляется по списочному составу группы:

- 0 АРМ отдела кадров;
- 1 АРМ технического отдела;
- 2 АРМ диспетчера АТП;
- 3 АРМ таксировщика;
- 4 АРМ бухгалтерии;
- 5 АРМ планового отдела;
- 6 АРМ техника по учету транспорта АТП;
- 7 АРМ техника по учету ходимости шин;
- 8 АРМ ремонтной службы;
- 9 АРМ склада;
- 10 АРМ автосервиса;
- 11 АРМ автомагазина;
- 12 АРМ магазина запасных частей;
- 13 АРМ автомойки;
- 14 АРМ диспетчера такси;
- 15 АРМ пункта технического осмотра ТС;
- 16 АРМ страховой компании.

Основные понятия, используемые при разработке БД.

База данных— это совокупность данных и объектов(т.е. форм, отчетов и т.п.), относящихся к определенной задаче ипредставляющих законченную систему. Базу данных Microsoft Office Access (Libre Office Base) составляют таблицы, запросы, формы, отчеты, страницы доступа, макросы и модули. Кроме того, приложение содержит некоторые другие объекты, в том числе связи, свойства базы данных и спецификации импорта и экспорта.

Таблица – объект базы данных, в котором данные хранятся в виде записей (строк) и полей (столбцов). Является основным структурным элементом системы управления реляционной базой данных.

Запрос – объект базы данных, позволяющий осуществлять поиск и вывод данных, хранящихся в таблицах, удовлетворяющих заданным условиям (в том числе из нескольких таблиц). С помощью запроса можно модифицировать и удалять записи таблиц, а также выполнять различные вычисления.

 Φ орма — объект базы данных, являющийся элементом пользовательского интерфейса, предназначенный для просмотра, ввода и модификации данных в одной или более таблицах.

Ответ — объект базы данных, предназначенный для анализа и вывода на печать данных, организованных и отформатированных в соответствии с требованиями пользователя.

Макрос — макрокоманда или набор макрокоманд, используемый для автоматизации задач.

Модуль — объект базы данных, который позволяет создавать библиотекиподпрограмм и функций, используемых во всем приложении. Используякоды модулей можно решать такие задачи, как обработка ошибок ввода,объявление и применение переменных, организация циклов и т. п.

Проектирование базы данных (БД) состоит из двух основных фаз:логического и физического моделирования. Во время фазы логическогомоделирования конструктор собирает требования и разрабатывает модель, не зависящую от конкретной системы управления базами данных (СУБД).

Во время фазы физического моделирования конструктор создает модель, оптимизированную для конкретного приложения СУБД; именно этамодель реализуется на практике.

Процесс проектирования БД состоит из следующих этапов:

- 1) сбор информации;
- 2) идентификация объектов;
- 3) моделирование объектов;
- 4) идентификация типов информации для каждого объекта;
- 5) идентификация отношений;
- 6) нормализация;
- 7) преобразование к физической модели;
- 8) создание базы данных.

Этапы 1-6 образуют фазу логического моделирования. Этапы 7-8 представляют собой фазу физического моделирования.

На первом этапе проектирования базы данных необходимоопределить назначение базы данных, режимы ее использования иосновные алгоритмы, реализующие реальные бизнес-процессы — т. е. изучить предметную область ее использования с целью создания модели.

Идентификации подлежат все сущности, относящиеся к поставленнойзадаче (в данном случае — данные и объекты базы), а также связи междуданными. В ходе идентификации определяются атрибуты (свойства)сущностей, и для этого необходимо принять решения по следующимвопросам:

- какие значения должны содержаться в поле;
- сколько места необходимо для хранения значений в поле;
- какие операции должны производиться со значениями в поле;
- нужна ли сортировка данных поля;
- необходимо ли группировать данные.

Один из способов изучить определенную базу данных — воспользоваться архивариусом (средством документирования базыданных). Архивариус используется для построения отчета, содержащегоподробные сведения об объектах в базе данных. Сначала следует выбрать, какие объекты будут подробно рассмотрены в отчете. Когда запускается архивариус, его отчет содержит все данные о выбранных объектах базыданных.

В теории реляционных баз данных таблица представляет собойизначально неупорядоченный набор записей. Единственный способидентифицировать определённую запись в этой таблице — это указатьнабор атрибутов, который был бы уникальным для этой записи.

Ключом называется набор атрибутов, однозначно определяющийзапись. Существуют несколько видов ключей:

• Первичный ключ — представляет собой одно или несколько полей (столбцов), значения которых однозначно определяют каждую запись втаблице. Первичный ключ не допускает значений Null и всегда должен иметь уникальный индекс. Первичный ключ используется для связываниятаблицы с внешними ключами в других таблицах. Первичный ключ можетбыть естественным или искусственным. Ключ, состоящий изинформационных полей таблицы (т. е. полей, содержащих полезнуюинформацию об описываемых объектах) называется естественнымключом. Теоретически, естественный ключ всегда можно сформировать, вэтом случае он носит название «интеллектуальный ключ». Искусственный ключ — это дополнительное служебное поле, единственное предназначение которого — служить первичным ключом. Значения этого поля не образуетсяна основе каких-либо других данных из БД, а генерируются искусственно.

Как правило, суррогатный ключ — это просто числовое поле, в котороезаносятся значения из возрастающей числовой последовательности.

• Внешний ключ - представляет собой одно или несколько полей(столбцов), содержащих ссылку на поле или поля первичного ключа вдругой таблице. Внешний ключ определяет способ связи таблиц.

Ключи также делятся на два класса: простые и составные.

Простой ключ состоит из одного атрибута, составной ключ состоит изнескольких атрибутов. Применение составных ключей усложняетобъединение таблиц.

Индекс — средство, ускоряющее поиск и сортировку данных в таблице. Существенное повышение скорости выполнения запросов приноситиндексирование полей, расположенных по обе стороны отношения, илисоздание связи между этими полями, а также индексирование всех полей, используемых для задания условий отбора в запросе. Индекс может бытьпростым (состоять из одного атрибута) или составным (состоять изнескольких атрибутов). Индекс может быть уникальным или неуникальным.

Реляционные базы данных позволяют объединять информацию, принадлежащую разным сущностям базы данных. Отношения определяются в процессе проектирования базы; для этого следуетпроанализировать разные

таблицы, выявить логические связи, существующие между ними, и охарактеризовать выявленные связи.

После создания таблицы для каждой темы в базе данных нужнопредоставить СУБД средства, с помощьюкоторых можно будет вновь объединять сведения при необходимости. Этоделается путем помещения общих полей в связанные таблицы иопределения связей между таблицами. После этого можно создавать запросы, формы и отчеты, одновременно отображающие сведения изнескольких таблиц.

Межтабличные связи могут объединять две и более сущности. Какправило, они соответствуют некоторому взаимодействию междусущностями и описывают связь, возникающую между ними. Во времялогического проектирования связи между таблицами могут обладатьсобственными атрибутами. Такое отношение выделяется в отдельнуюсущность типа связь.

Отношения делятся на три основных типа, в зависимости отколичества записей сущности, связанных с записью другой сущности.

1*Один к одному*: каждой записи первой сущности соответствует толькоодна запись второй сущности, а каждой записи второй сущности — толькоодна запись первой сущности. Пример — автор, у которого в данныймомент имеется лишь одна незавершенная книга.

2*Один ко многим*: каждой записи первой сущности могутсоответствовать несколько записей второй сущности, однако каждойзаписи второй сущности соответствует только одна запись первойсущности. Пример – издательство, выпустившее несколько книг.

3 *Многие ко многим*: каждой записи первой сущности могутсоответствовать несколько записей второй сущности, а каждой записивторой сущности соответствуют несколько записей первой сущности. Пример — один автор может написать несколько книг, а у одной книги может быть несколько авторов.

В реляционных базах данных этот тип отношений не реализуем,поэтому создается дополнительная сущность, ассоциирующая даннуюсвязь (ассоциация).

Связи между сущностями устанавливаются по равенству значенийпервичного и внешнего ключей.

В СУБД межтабличные связи можно создатьнепосредственно с помощью окна «Схема данных» или путемперетаскивания поля из области «Список полей». Межтабличные связи используются для того, чтобы продемонстрировать,как связать таблицы для использования их в объекте базы данных.

Существует несколько причин для создания межтабличных связей передсозданием других объектов базы данных (форм, запросов и отчетов):

- межтабличные связи предоставляют сведения для структуры запросов;
- межтабличные связи предоставляют сведения для структуры форм иотчетов;
- межтабличные связи являются основой, с помощью которой можнообеспечить целостность данных.

Целостность данных означает систему правил, используемых вСУБД для поддержания связей между записями в связанных таблицах, а также обеспечивающих защиту от случайного удаления илиизменения связанных данных.

Установить целостность данных можно, если выполнены следующие условия:

- связанное поле главной таблицы является первичным ключом илиальтернативным ключом (имеет уникальный индекс);
- связанные поля (первичный ключ главной таблицы и внешний ключподчиненной таблицы) имеют один тип данных. Здесь существуетисключение: поле счетчика может быть связано с числовым полем,свойство которого «Размер поля (FieldSize)» имеет значение «Длинное целое»;
- обе таблицы принадлежат одной базе данных. Дляустановки целостности данных база данных, в которой находятся таблицы, должна быть открыта.

При установке целостности данных, необходимо помнить следующиеправила:

1 невозможно ввести в поле внешнего ключа связанной таблицы значение, не содержащееся в ключевом поле главной таблицы;

2 не допускается удаление записи из главной таблицы, если существуютсвязанные с ней записи в подчиненной таблице;

3 невозможно изменить значение первичного ключа в главной таблице, еслисуществуют записи, связанные с данной записью.

Чтобы задать правила целостности данных для конкретной связи, приее создании в СУБД следует установить флажок.

Обеспечение целостности данных. Если данный флажок установлен, толюбая попытка выполнить действие, нарушающее одно из перечисленныхвыше правил, приведет к выводу на экран предупреждения, а самодействие будет отменено.

Чтобы преодолеть ограничения на удаление или изменение связанных аписей, сохраняя при этом целостность данных, следует установить флажки «Каскадное обновление связанных полей» и «Каскадное удаление связанных полей». Если установлен флажок Каскадное обновление связанных полей, то при изменении ключевого поля главной таблицы автоматически изменяются и соответствующие значения связанных записей.

Если установлен флажок «Каскадное удаление связанных полей», то приудалении записи в главной таблице удаляются и все связанные записи вподчиненной таблице.

После создания необходимых таблиц, полей и связей необходимо ещераз просмотреть структуру базы данных и выявить возможные недочеты.

Желательно это сделать на данном этапе, пока таблицы не заполненыданными.

В СУБД существует два инструмента, помогающих вусовершенствовании структуры баз данных:

• «Мастер анализа таблиц» может проанализировать структуру таблицы,предложить подходящие новые структуры и связи, а также разделитьтаблицу на новые связанные таблицы, если это имеет смысл.

• «Анализатор быстродействия» исследует всю базу данных, даетрекомендации по ее улучшению, а также осуществляет их.

В курсовой работе студенту необходимо описать последовательность выполнения задания, путем указания скриншотов рабочих окон СУБД.

Готовая БД должна содержать следующее:

- кнопочная форма;
- таблицы, не менее 3;
- многотабличная форма ввода данных;
- форма отчетов, не менее, чем по 2 критериям (таблицам);
- отчет на основе запроса.

2.2 Проектирования экспертной системы на основе нечеткой логики

Во второй части курсовой работы студенту необходимо разработать экспертную систему, основанную на нечеткой логике, для решения поставленного вопроса.

Перед разработкой системы студенту необходимо кратко рассмотреть предметную область изучаемого вопроса, а также сформулировать главный вопрос для решения системы. После этого студент обосновывает возможность применения нечеткой логики.

После рассмотрения предметной области студент определяет факторы, определяющие решение поставленного вопроса, делает их описание. После этого определяется формат вывода решения – количественный или качественный.

В рамках курсовой работы студент должен выступить как эксперт, определяющий базу знаний.

После разработки экспертной системы необходимо произвести тестирование ее работы путем изменения исходных данных.

Ход выполнения может быть оформлен в виде скриншотов рабочих окон с пояснениями производимых действий.

Решение качественных или смешанных задач в настоящее время является наиболее сложным, т. к. необходимо решать задачи, не поддающиеся полностью или частично формализации для применения стандартной логики. Поэтому качественные задачи пока в основном решаются человеком, но есть уже и опыт решения их нейронными сетями или экспертными системами.

Управление техническими или иными объектами в большинстве своем являются задачами количественными. Аналитически управление объектом сводятся к решению одного или системы дифференциальных уравнений.

При достаточно точном описании процесса системой уравнений ее целесообразно решать аналитическим или численным методом. Эффективно такие вычисления производятся в среде MathCAD, MATLAB.

Однако зачастую аналитическое описание объекта (математическая модель) или недостаточно верно, или решение носит сложный характер. В этих случаях прибегают к методам моделирования, анализа, построенным

на основе нейросетей, нечеткой логики, генетических алгоритмах, то есть методам искусственного интеллекта (ИИ).

Нечеткая логика возникла как наиболее удобный способ построения систем управления метрополитенами и сложными технологическими процессами, а также нашла применение в бытовой электронике, диагностических и других экспертных системах. Несмотря на то, что математический аппарат нечеткой логики впервые был разработан в США, активное развитие данного метода началось в Японии, и новая волна вновь достигла США и Европы.

Термин fuzzy (англ. «нечеткий, размытый», произносится «фаззи») является ключевым понятием. Нечеткая логика является многозначной логикой, что позволяет определить промежуточные значения для таких общепринятых оценок, как да/нет, истинно/ложно, черное/белое и т.п. Выражения, подобные таким, как «слегка тепло» или «довольно холодно» становится возможно формулировать математически и обрабатывать на компьютерах. Нечеткая логика появилась в 1965 в работах Лотфи А. Задэ (Lotfi A. Zadeh), профессора технических наук Калифорнийского университета в Беркли.

Рассмотрим базовые понятия «нечеткой логики». Самым главным понятием систем, основанных на нечеткой логике, является понятие нечеткого (под)множества.

Нечеткое множество — это такое множество, которое образуется путем введения обобщенного понятия принадлежности, т.е. расширения двухэлементного множества значений функции принадлежности {0,1} до отрезка [0,1]. Это означает, что переход от полной принадлежности объекта множества к его полной непринадлежности происходит не скачком, как в обычных «четких» множествах, а плавно, постепенно, причем степень принадлежности элемента множеству выражается числом из интервала [0,1].

Таким образом, нечеткое множество $A = \{(x, \stackrel{\mu}{}_A(x))\}$ определяется математически как совокупность упорядоченных пар, составленных из элементов х множества X и соответствующих им степеней принадлежности

 $\mu_{A}(x)$ или непосредственно в виде функции $\mu_{A}: X \to [0,1]$.

Функцией принадлежности (membership function) называется функция, которая позволяет вычислить степень принадлежности произвольного элемента универсального множества к нечеткому множеству. Графическое представление функции принадлежности называется термом.

Нечеткое число — это нечеткое подмножество универсального множества действительных чисел, имеющее нормальную и выпуклую функции принадлежности, то есть такую, что а) существует такое значение носителя, в котором функция принадлежности равна единице, а также б) при отступлении от своего максимума влево или вправо функция принадлежности убывает.

Понятие нечеткого множества — это попытка математической формализации нечеткой информации для построения математических моделей. В основе этого понятия лежит представление о том, что составляющие данное множество элементы, обладающие общим свойством, могут обладать этим свойством в различной степени и, следовательно, принадлежать к данному множеству с различной степенью.

Из вышесказанного можно сделать следующие выводы:

- 1) нечеткие множества описывают неопределенные понятия (быстрый бегун, горячая вода, жаркая погода);
- 2) нечеткие множества допускают возможность частичной принадлежности к ним (пятница частично выходной день (укороченный), погода скорее жаркая)
- 3) степень принадлежности объекта к нечеткому множеству определяется соответствующим значением функции принадлежности на интервале [0,1];
- 4) функция принадлежности ставит в соответствие объекту (или логической переменной) значение степени его принадлежности к нечеткому множеству.

Для наглядности приведем функцию принадлежности (терм) множества молодых людей.

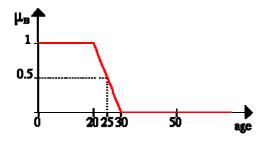


Рисунок 1 – Функция принадлежности μ_{B} множества В

Из рисунка 1 следует, что 25-летние все еще молоды со степенью принадлежности 0,5.

Определим базовые операции (действия) над нечеткими множествами (числами). Аналогично действиям с обычными множествами нам потребуется определить пересечение, объединение и отрицание нечетких множеств. В своей самой первой работе по нечетким множествам Л. А. Задэ предложил оператор минимума для пересечения и оператор максимума для объединения двух нечетких множеств. Легко видеть, что эти операторы совпадают с обычными (четкими) объединением и пересечением, только рассматриваются степени принадлежности 0 и 1. Чтобы пояснить это, приведем несколько примеров. Пусть А – нечеткое число (интервал) от 5 до 8 и В – нечеткое число около 4, как показано на рисунке 2.

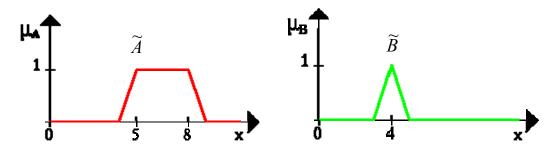
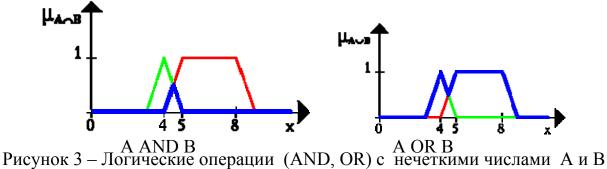


Рисунок 2 – Графическое представление двух нечетких чисел А и В



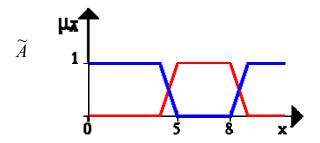


Рисунок 4 – Логическое отрицание (NOT) нечеткого числа A

Фаззификацией (fuzzification) называется процедура преобразования «четкой» точки $\overline{X} = (x_1, ... x_n) \in X$ в нечеткое множество число \widetilde{A} из X. Другими словами, нахождение значений функции принадлежности нечеткого числа в заданной точке. На этом этапе происходит установление соответствия между численным значением входной переменной и значением функции принадлежности соответствующего ей терма входной лингвистической переменной. Так, фаззификация числа 25 (рисунок 1) дает нам значение 0,5.

Дефаззификацией (defuzzification) называется процедура преобразования нечеткого множества в четкое число.

В теории нечетких множеств процедура дефаззификации аналогична нахождению характеристик положения (математического ожидания, моды, медианы) случайных величин в теории вероятности. Простейшим способом выполнения процедуры дефаззификации является выбор четкого числа, соответствующего максимуму функции принадлежности. Однако пригодность этого способа ограничивается лишь одноэкстремальными функциями принадлежности. Для многоэкстремальных функций принадлежности в Fuzzy Logic Toolbox запрограммированы такие методы дефаззификации:

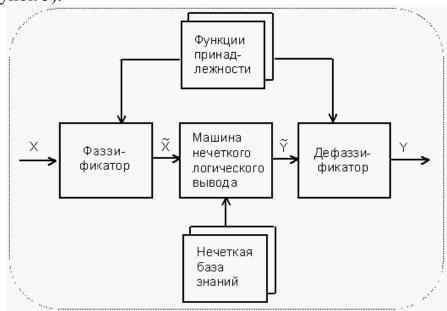
- *Centroid центр тяжести;
- *Bisector медиана;
- *LOM (Largest Of Maximums) наибольший из максимумов;
- *SOM (Smalles tOf Maximums) наименьший из максимумов;
- *Mom (Mean Of Maximums) центр максимумов.

Нечеткой базой знаний (fuzzy knowledge base) о влиянии факторов X=x₁...x_n на значение параметра Y называется совокупность логических высказываний типа:

ЕСЛИ $(x1=\mu 1)$ И $(x2=\mu 2)...$ И $(xn=\mu n)$ ИЛИ $(x1=\mu 1)$ ИЛИ $(x2=\mu 2)...$ ИЛИ $(xk=\mu k)$, ТО $yi=\mu j$,

где $\mu 1...\mu n$ — нечеткий терм, которым оцениваются входные переменные x1...xn;

μj – нечеткий терм, которым оцениваются выходные переменные у1... уп. Графически систему на основе нечеткой логики можно представит в виде блоков (рисунок 5).



X — входной четкий вектор; \widetilde{X} — вектор нечетких множеств, соответствующий входному вектору X; \widetilde{Y} — результат логического вывода в виде вектора нечетких множеств; Y — выходной четкий вектор.

Рисунок 5 – Структурная схема системы, использующей «нечеткую логику»

МАТLAВ – матричная лаборатория – наиболее развитая система программирования для научно-технических расчетов, дополненная к настоящему времени несколькими десятками более частных приложений, относящихся к вычислительной математике, обработке информации, конструированию электронных приборов, экономике и ряду других разделов прикладной науки.

Fuzzy Logic Toolbox – это пакет прикладных программ, входящих в состав среды MATLAB. Он позволяет создавать системы нечеткого логического вывода и нечеткой классификации в рамках среды MATLAB, с возможностью их интегрирования в Simulink.

Базовым понятием Fuzzy Logic Toolbox является FIS-структура — система нечеткого вывода (Fuzzy Inferen ceS ystem). FIS-структура содержит все необходимые данные для реализации функционального отображения «входывыходы» на основе нечеткого логического вывода согласно схеме, приведенной на рисунке 5.

Fuzzy Logic Toolbox содержит следующие категории программных инструментов:

• функции;

- интерактивные модули с графическим пользовательским интерфейсом (с GUI);
 - блоки для пакета Simulink;
 - демонстрационные примеры.

Модуль fuzzy позволяет строить нечеткие системы двух типов – Мамдани и Сугэно. В системах типа Мамдани база знаний состоит из правил вида: «Если х1=низкий и х2=средний, то у=высокий». В системах типа Сугэно база знаний состоит из правил вида: «Если х1=низкий и х2=средний, то у=а0+a1x1+a2x2». Таким образом, основное отличие между системами Мамдани и Сугэно заключается в разных способах задания значений выходной переменной в правилах, образующих базу знаний. В системах типа Мамдани значения выходной переменной задаются нечеткими термами, в системах типа Сугэно – как линейная комбинация входных переменных.

Проектирование нечетких систем имеет ряд шагов. Рассмотрим в качестве примера последовательность создания системы управления технического объекта по зависимости, представленной на рисунке 6. Обычно для таких задач выбирают систему типа Мамдани.

Шаг1. Для загрузки основного fis-редактора напечатаем слова fuzzy в командной строке. После этого откроется новое графическое окно, показанное на рисунке 7.

Шаг 2. Добавим вторую входную переменную. Для этого в меню Edit выбираем команду Add input.

Шаг 3. Переименуем первую входную переменную. Для этого сделаем один щелчок левой кнопкой мыши на блоке input1, введем новое обозначение х1 в поле редактирования имени текущей переменной и нажмем <Enter>.

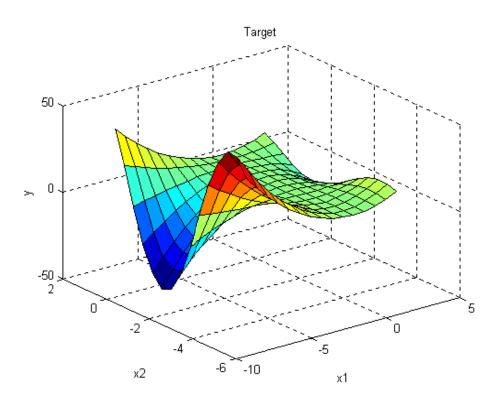


Рисунок 6 – Требуемая закономерность управления объектом

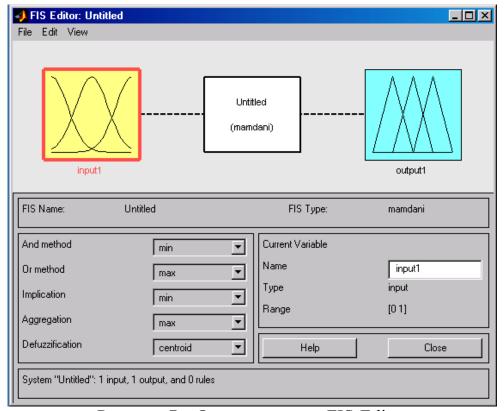


Рисунок 7 – Окно редактора FIS-Editor

- *Шаг 4*. Переименуем вторую входную переменную. Для этого сделаем один щелчок левой кнопкой мыши на блоке input2, введем новое обозначение x2 в поле редактирования имени текущей переменной и нажмем <Enter>.
- *Шаг* 5. Переименуем выходную переменную. Для этого сделаем один щелчок левой кнопкой мыши на блоке output1, введем новое обозначение у в поле редактирования имени текущей переменной и нажмем <Enter>.
- *Шаг* 6. Зададим имя системы. Для этого в меню File выбираем в подменю Export команду To disk и вводим имя файла, например, first.
- *Шаг* 7. Перейдем в редактор функций принадлежности. Для этого сделаем двойной щелчок левой кнопкой мыши на блоке x1.
- *Шаг* 8. Зададим диапазон изменения переменной х1. Для этого напечатаем -73 в поле Range (рисунок 8) и нажмем <Enter>.

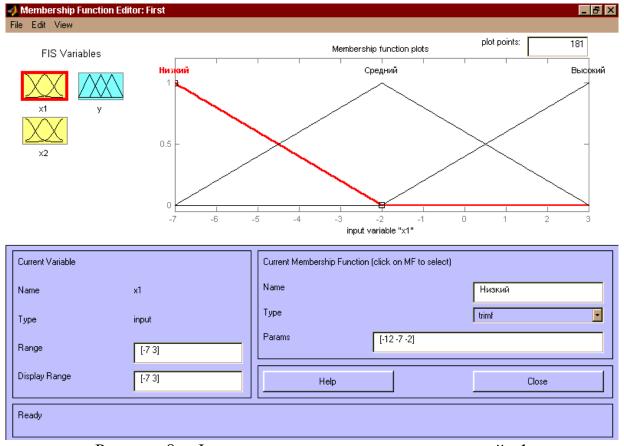


Рисунок 8 – Функции принадлежности переменной х1

Шаг 9. Зададим функции принадлежности переменной х1. Для лингвистической оценки этой переменной будем использовать 3 терма с треугольными функциями принадлежности. Для этого в меню Edit выберем команду Add MFs... В результате появится диалоговое окно выбора типа и количества функций принадлежностей. По умолчанию это 3 терма с треугольными функциями принадлежности, поэтому просто нажимаем <Enter>.

Шаг 10. Зададим наименования термов переменной х1. Для этого делаем один щелчок левой кнопкой мыши по графику первой функции принадлежности (рисунок 8). Затем вводим наименование терма, например, Низкий, в поле Name и нажмем <Enter>. Затем делаем один щелчок левой кнопкой мыши по графику второй функции принадлежности и вводим наименование терма, например, Средний, в поле Name и нажмем <Enter>. Еще раз делаем один щелчок левой кнопкой мыши по графику третьей функции принадлежности и вводим наименование терма, например, Высокий, в поле Name и нажмем <Enter>. В результате получим графическое окно, изображенное на рисунке 8.

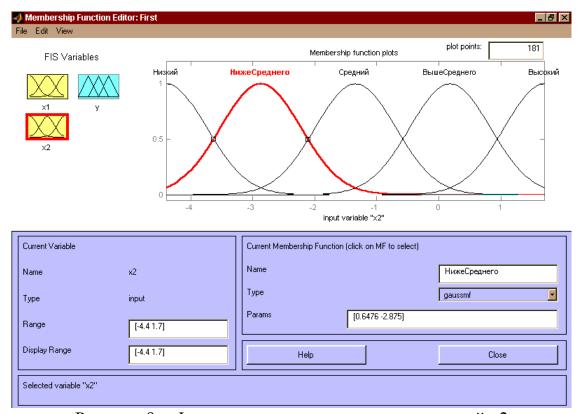


Рисунок 9 – Функции принадлежности переменной х2

Шаг 11. Зададим функции принадлежности переменной x2. Для лингвистической оценки этой переменной будем использовать 5 термов с гауссовскими функциями принадлежности. Для этого активизируем переменную x2 с помощью щелчка левой кнопки мыши на блоке x2. Зададим диапазон изменения переменной x2. Для этого напечатаем -4.4 1.7 в поле Range (рисунок 9) и нажмем <Enter>. Затем в меню Edit выберем команду Add MFs.... В появившемся диалоговом окне выбираем тип функции принадлежности gaussmf в поле MF type и 5 термов в поле Number of MFs. После этого нажимаем <Enter>.

Шаг 12. По аналогии с шагом 10 зададим следующие наименования термов переменной х2: Низкий, Ниже среднего, Средний, Выше среднего, Высокий. В результате получим графическое окно, изображенное на рисунке 9.

Шаг 13. Зададим функции принадлежности переменной у. Для лингвистической оценки этой переменной будем использовать 5 термов с треугольными функциями принадлежности. Для этого активизируем переменную у с помощью щелчка левой кнопки мыши на блоке у. Зададим диапазон изменения переменной у. Для этого напечатаем [-50 50] в поле Range (рисунок 10) и нажмем <Enter>. Затем в меню Edit выберем команду Add MFs... В появившимся диалоговом окне выбираем 5 термов в поле Number of MFs. После этого нажимаем <Enter>.

Шаг 14. По аналогии с шагом 10 зададим следующие наименования термов переменной у: Низкий, Ниже среднего, Средний, Выше среднего, Высокий. В результате получим графическое окно, изображенное на рисунке 10.

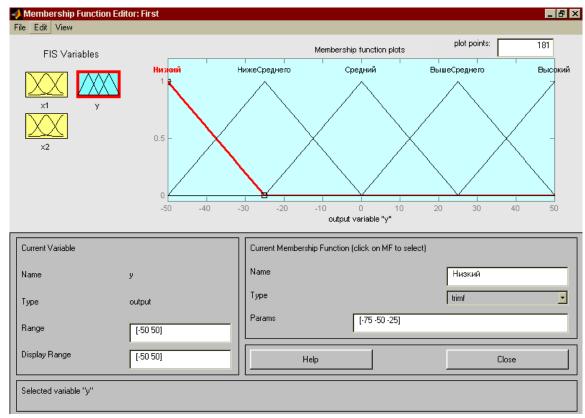


Рисунок 10 – Функции принадлежности переменной у

Шаг 15. Перейдем в редактор базы знаний RuleEditor. Для этого выберем в меню Edit выберем команду Edit rules....

Шаг 16. На основе визуального наблюдения за графиком, изображенным на рисунке 6, сформулируем следующие девять правил.

Если х1=Средний, то у=Средний.

Если х1=Низкий и х2=Низкий, то у=Высокий.

Если х1=Низкий и х2=Высокий, то у=Высокий.

Если х1=Высокий и х2=Высокий, то у=Выше Среднего.

Если х1=Высокий и х2=Низкий, то у=Выше Среднего.

Если х1=Высокий и х2=Средний, то у=Средний.

Если х1=Низкий и х2=Средний, то у=Низкий.

Если х1=Высокий и х2=Выше Среднего, то у=Средний.

Если х1=Высокий и х2=Ниже Среднего, то у=Средний.

Для ввода правила необходимо выбрать в меню соответствующую комбинацию термов и нажать кнопку Add rule. На рисунке 11 изображено окно редактора базы знаний после ввода всех девяти правил. Число, приведенное в скобках в конце каждого правила, представляет собой весовой коэффициент соответствующего правила.

Шаг 17. Сохраним созданную систему. Для этого в меню File выбираем в подменю Export команду To disk.

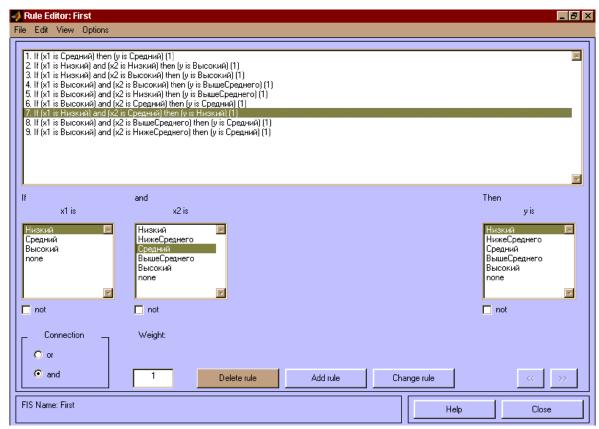


Рисунок 11 – База знаний в RuleEditor

На рисунке 12 приведено окно визуализации нечеткого логического вывода. Это окно активизируется командой View rules... меню View.

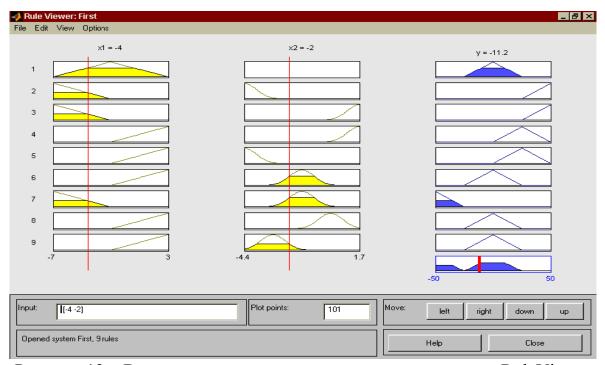


Рисунок 12 – Визуализация нечеткого логического вывода в RuleViewer

В поле Input указываются значения входных переменных, для которых выполняется логический вывод. На рисунке 13 приведена поверхность «входы-

выход», соответствующая синтезированной нечеткой системе. Для вывода этого окна необходимо использовать команду View surface... меню View.

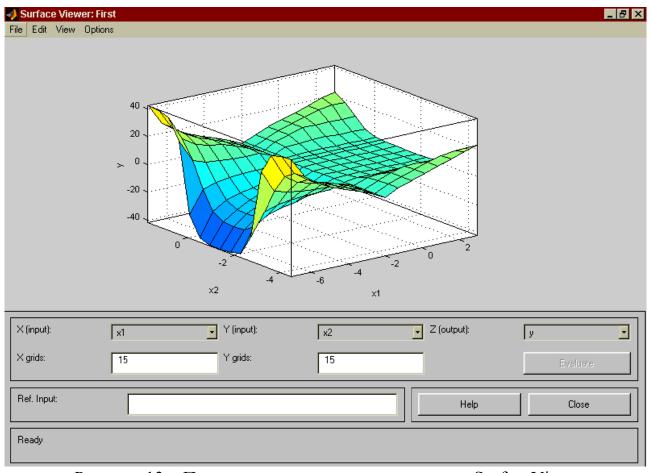


Рисунок 13 – Поверхность «входы-выход» в окне SurfaceViwer

Сравнивая поверхности на рисунке 6 и рисунке 13, можно сделать вывод, что нечеткие правила достаточно хорошо описывают сложную нелинейную зависимость.

Список литературы

- 1 Учебная, научная и методическая литература методические рекомендации по подготовке рукописей к изданию / сост. А. В. Зайцев, Я. А. Борщенко, О. Г. Арефьева, Н. М. Быкова. Курган : Изд-во КГУ, 2012. 26 с.
- 2 Шабуров В. Н. Требования к оформлению учебных документов: метод. указания к оформлению текстовой части курсовых и дипломных проектов для студентов направления (специальностей) 190600 (190601, 190603). Курган: РИЦ КГУ, 2007 32 с.
- 3 ГОСТ 2.105-95 ЕСКД. Общие требования к текстовым документам (с Изменением №1). URL: http://docs.cntd.ru/document/gost-2-105-95-eskd (дата обращения: 01.12.2015).
- 4 ГОСТ Р 7.0.5-2008 СИБИД. Библиографическая ссылка. Общие требования и правила составления. URL: http://docs.cntd.ru/document/1200063713 (дата обращения: 01.12.2015).
- 5 Нечеткое моделирование и управление / А. Пегас ; пер. с англ. 2-е изд.— Москва : БИНОМ. Лаборатория знаний, 2013.— 798 с. : ил.
- 6 Леоненков А. В. Нечеткое моделирование в среде MATLAB и fuzzy-TECH. – Санкт Петербург : БХВ – Петербург, 2005. – 736 с.
- 7 Карчевский, Е. М., Филиппова, И.А. Access 2010 в примерах : И.Е. Филиппов, учеб. пособие. Казань : Казанский университет, 2012.

Безотеческих Николай Сергеевич

Информационные технологии на автомобильном транспорте

МЕТОДИЧЕСКИЕ УКАЗАНИЯ к выполнению курсовой работы для студентов всех форм обучения направления 23.03.01 «Технология транспортных процессов»

Редактор Л. П. Чукомина

Подписано в печать 28.01.19	Формат 60х84 / 16	Бумага тип. 65 г/м ²
Печать цифровая	Усл. печ.л.	Учизд. л. 1,5
Заказ №34	Тираж	Не для продажи

Библиотечно-издательский центр КГУ. 640020 г. Курган, ул. Советская 63/4. Курганский государственный университет.