Н. П. Садовникова, Д. С. Парыгин,
Т. В. Ерещенко, Н. М. Рашевский

Имитационное моделирование

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ
ВОЛГОГРАДСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

На правах рукописи

Н. П. Садовникова, Д. С. Парыгин,
Т. В. Ерещенко, Н. М. Рашевский

Имитационное моделирование

Учебное пособие

Волгоград
2021

УДК 004.414.23(075)

Рецензенты:

Печатается по решению редакционно-издательского совета
Волгоградского государственного технического университета

Садовникова, Н. П.

Имитационное моделирование : учеб. пособие / Н. П. Садовникова, Д. С. Парыгин, Т.В. Ерещенко, Н. М. Рашевский ; ВолгГТУ. – Волгоград, 2021. – 132 с.

ISBN 978–5–9948–0000–0

В учебном пособии представлен всесторонний анализ вопросов, связанных с разработкой математического обеспечения в процессе создания программных средств для проведения аналитических исследований. Рассматриваются наиболее востребованные модели и методы. Определяются основные этапы построения и использования математических моделей для анализа объектов и процессов, разбираются примеры построения моделей для решения практических задач. Предназначено для студентов очной и заочной форм обучения по направлениям 09.03.02 «Информационные системы и технологии», 09.04.02 «Информационные системы и технологии» (профиль «Искусственный интеллект в проектировании городской среды»), 08.04.01 «Строительство» (профиль «Организация информационного моделирования в строительстве»)..

Ил. 23. Табл. 4. Библиогр.: 47 назв.

ISBN 978-5-9948-0000-0

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

Аналитические исследования используются для выяснения причин, лежащих в основе изучаемого явления и анализа свойств объектов и процессов. Специфика аналитических исследований определяет сложность создания специализированного программного обеспечения. С одной стороны, это развитие математического моделирования, появление новых методов и подходов. С другой стороны, это создание новых информационных технологий, позволяющих решать задачи исключительной сложности. Изменения в процессе исследований требуют адекватной инструментальной поддержки и, соответственно, создания специальных программных средств для решения задач связанных с анализом ситуаций, прогнозированием, принятием решений.

Подходы к проведению аналитических исследований в разных сферах человеческой деятельности являются междисциплинарными и базируются на знаниях математики, прикладной статистики, искусственного интеллекта, машинного обучения, теории оптимизации, баз данных и пр.

Современные тенденции развития аналитических исследований связаны с появлением нового научного направления связанного с анализом данных. Появились новые подходы, ориентированные на работу с большими массивами разнородных данных и извлечение знаний.

Умение работать с разными математическими методами и моделями, извлекать нужную информацию является одной из наиболее перспективных и востребованных компетенций ближайшего времени.

Целью создания данного пособия является ознакомление студентов и других заинтересованных лиц с методами моделирования, программными системами, позволяющими решать задачи связанные с анализом ситуаций, прогнозированием, принятием решений.

ГЛАВА 1 АНАЛИТИЧЕСКИЕ ИССЛЕДОВАНИЯ И МОДЕЛИРОВАНИЕ

1.1 Основные положения теории моделирования

Моделирование это универсальный метод получения, описания и использования знаний. Оно используется в любой профессиональной деятельности. Исследование любого явления, объекта или процесса сводится, как правило, к созданию ее модели. Модель в общем смысле есть создаваемый с целью получения и (или) хранения информации специфический объект (в форме мысленного образа, описания знаковыми средствами либо материальной системы), отражающий свойства, характеристики и связи объекта – оригинала произвольной природы, существенные для задачи, решаемой субъектом [1].

Моделирование базируется на математической теории подобия, согласно которой абсолютное подобие может иметь место лишь при замене одного объекта другим, точно таким же. При моделировании большинства систем (за исключением, возможно, моделирования одних математических структур другими) абсолютное подобие невозможно [2].

В большинстве случаев в качестве объекта-оригинала рассматривается некоторая система. Под системой понимается совокупность объектов, компонентов или элементов произвольной природы, образующая некоторую целостность в том или ином контексте. Определяющим принципом рассмотрения некоторой совокупности объектов как системы является появление у нее новых свойств, которые не имеют составляющие ее элементы.

Модель всегда строится с определенной целью. Цель моделирования может быть явно или неявно выражена, однако она всегда существует.

Целями моделирования могут являться:

– осмысление действительности (познание и разработка теории исследуемых систем);

– постановка над моделью экспериментов с последующей интерпретацией их результатов применительно к моделируемой системе;

– прогнозирование будущего поведения системы (выходных данных, ситуаций, состояний системы);

– проектирование систем;

– управление системами;

– общение с другими лицами, общественными группами, устройствами обработки информации;

– обучение и тренаж специалистов.

В рамках каждой научной дисциплины разрабатывается совокупность приемов и правил, следование которым позволяет создавать отвечающее исходным гипотезам описание и получать предварительную оценку пригодности модели для проведения исследований. Окончательный анализ данной оценки осуществляется на этапе проверки модели, на котором устанавливается правомерность исходных посылок в соответствии с целью исследования реального явления и определяется степень соответствия ему полученной модели.

Можно выделить несколько общих критериев для оценки качества модели. Наиболее важным является *адекватность модели* – способность отображать заданные свойства объекта с погрешностью не выше заданной.

Неадекватность модели может быть связана со следующими причинами:

1. Модель может неправильно отражать действительную зависимость, которая существует между результатом операции и переменными.

2. В модели могут не учитываться переменные, которые в действительности влияют на результат.

3. Значения переменных, входящих в модель, могут быть оценены неправильно.

Существует два подхода к оценки адекватности моделей. Первый способ основан на процедуре сравнение значений изучаемых параметров реального объекта и модели. Модель считается адекватной, если отражает исследуемые свойства с приемлемой точностью, где под точностью модели понимается количественный показатель, характеризующий степень различия модели и изучаемого явления.

Второй способ применяют в случае отсутствия возможности сравнить модель и объект (например, объект находится в стадии проектирования, либо эксперименты с объектом невозможны). В данном случае осуществляется проверка соблюдения физических законов, размерности и знаков, предельных соответствий, проверка тенденции изменения выходных переменных в зависимости от внутренних и внешних переменных и т.п.

Еще один критерий связан с полезностью модели. Для того чтобы его оценить обычно рассчитывают соотношение средств, затрачиваемых на построение модели и выгод от ее использования. Этот критерий называют *эффективностью* модели.

Устойчивость модели – это ее способность сохранять адекватность при исследовании системы на всем возможном диапазоне рабочей нагрузки, а также при внесении изменений в конфигурацию системы.

В общем случае можно утверждать, что чем ближе структура модели структуре системы и чем выше степень детализации, тем устойчивее модель.

Очевидно, что устойчивость является положительным свойством модели. Однако если изменение входных воздействий или параметром моде-

ли (в некотором заданном диапазоне) не отражается на значениях выходных переменных, то польза от такой модели невелика. В связи с этим возникает задача оценивания чувствительности модели к изменениям параметров рабочей нагрузки и внутренних параметров самой системы.

Чувствительность модели – способность модели реагировать определённым образом на определённое малое воздействие, а также количественная характеристика этой способности.

Обычно такую оценку проводят по каждому параметру отдельно. Основана она на том, что диапазон возможных изменений параметра известен. Данные, полученные при оценке чувствительности модели, могут быть использованы, в частности, при планировании экспериментов: большее внимание должно уделяться тем параметрам, по которым модель является более чувствительной.

В общем случае процесс моделирования включает ряд этапов (рисунок 1.1).

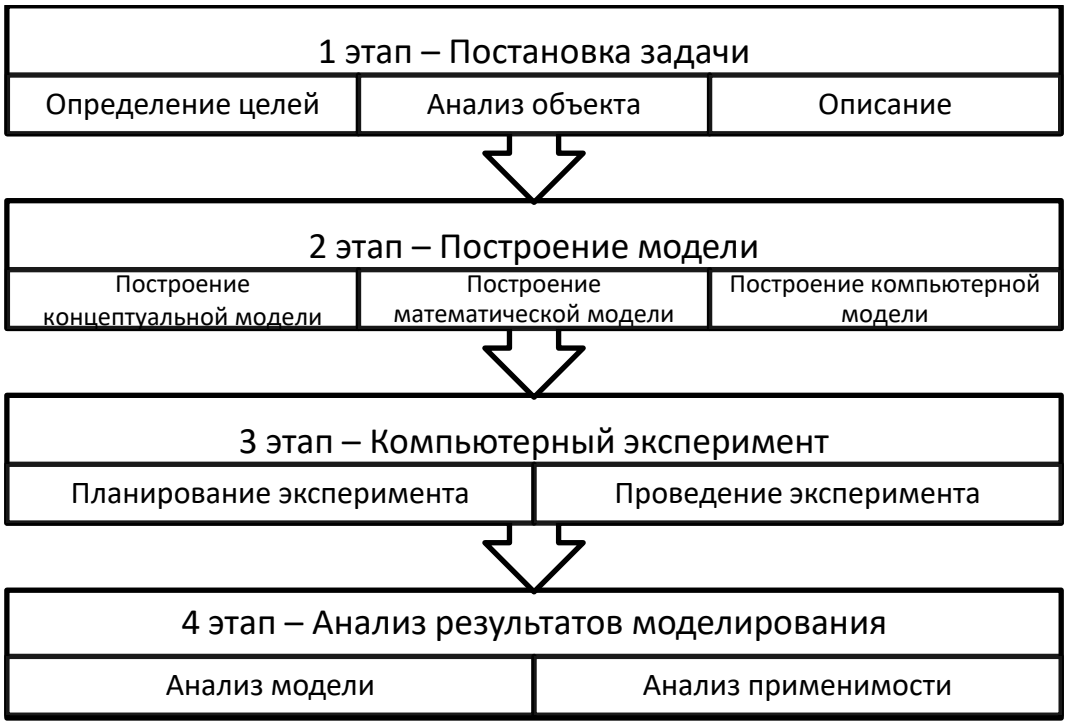


Рис. 1.1. Этапы моделирования

1.2 Классификация математических моделей

В качестве одного из первых признаков классификации видов моделирования можно выбрать степень полноты модели. На основе этого можно выделить следующие группы моделей [2]:

– полные;

– неполные;

– приближенные.

В основе *полного моделирования* лежит полное подобие, которое проявляется как во времени, так и в пространстве. Для *неполного моделирования* характерно неполное подобие модели изучаемому объекту. В основе *приближенного моделирования* лежит приближенное подобие, при котором некоторые стороны функционирования реального объекта не моделируются совсем.

В зависимости от характера изучаемых процессов в системе все виды моделирования могут быть разделены:

– детерминированные;

– стохастические;

– статические и динамические;

– дискретные;

– непрерывные;

– дискретно-непрерывные.

Детерминированное моделирование отображает детерминированные процессы, т.е. процессы, в которых предполагается отсутствие всяких случайных воздействий.

Стохастическое моделирование отображает вероятностные процессы и события. В этом случае анализируется ряд реализаций случайного процесса, и оцениваются средние характеристики, т. е. набор однородных реализаций.

Статическое моделирование служит для описания поведения объекта в какой-либо момент времени, а динамическое моделирование отражает поведение объекта во времени.

Дискретное моделирование служит для описания процессов, которые предполагаются дискретными, соответственно непрерывное моделирование позволяет отразить непрерывные процессы в системах, а дискретно-непрерывное моделирование используется для случаев, когда хотят выделить наличие как дискретных, так и непрерывных процессов.

Под *математическим моделированием* будем понимать процесс установления соответствия данному реальному объекту некоторого математического объекта, называемого *математической моделью*, и исследование этой модели, позволяющее получать характеристики рассматриваемого реального объекта. Вид математической модели зависит как от природы реального объекта, так и задач исследования объекта и требуемой достоверности и точности решения этой задачи. Любая математическая модель, как и всякая другая, описывает реальный объект лишь с некоторой степенью приближения к действительности. Математическое моделирование для исследования характеристик процесса функционирования систем можно разделить на:

– аналитическое;

– имитационное;

– комбинированное.

Для *аналитического моделирования* характерно то, что процессы функционирования элементов системы записываются в виде некоторых функциональных соотношений или логических условий. Аналитическая модель может быть исследована следующими методами:

а) аналитическим, когда стремятся получить в общем виде явные зависимости для искомых характеристик;

б) численным, когда, не умея решать уравнений в общем виде, стремятся получить числовые результаты при конкретных начальных данных;

в) качественным, когда, не имея решения в явном виде, можно найти некоторые свойства решения.

При *имитационном моделировании* реализующий модель алгоритм воспроизводит процесс функционирования системы во времени, причем имитируются элементарные явления, составляющие процесс, с сохранением их логической структуры и последовательности протекания во времени, что позволяет по исходным данным получить сведения о состояниях процесса в определенные моменты времени, дающие возможность оценить характеристики системы.

Основным преимуществом имитационного моделирования по сравнению с аналитическим является возможность решения более сложных задач. Имитационные модели позволяют достаточно просто учитывать такие факторы, как наличие дискретных и непрерывных элементов, нелинейные характеристики элементов системы, многочисленные случайные воздействия и др., которые часто создают трудности при аналитических исследованиях. В настоящее время имитационное моделирование — наиболее эффективный метод исследования больших систем, а часто и единственный практически доступный метод получения информации о поведении системы, особенно на этапе ее проектирования.

Комбинированное (аналитико-имитационное) моделирование при анализе и синтезе систем позволяет объединить достоинства аналитического и имитационного моделирования. При построении комбинированных моделей проводится предварительная декомпозиция процесса функционирования объекта на составляющие подпроцессы, и для тех из них, где это возможно, используются аналитические модели, а для остальных подпроцессов строятся имитационные модели. Такой комбинированный

подход позволяет охватить качественно новые классы систем, которые не могут быть исследованы с использованием только аналитического и имитационного моделирования в отдельности.

Вопросы к главе 1

1. Сколько моделей может быть у одного объекта?
2. Как можно оценить качество модели?
3. Какими могут быть цели моделирования?
4. Что такое концептуальная модель?
5. Что такое математическая модель?
6. Что означает слово формализация?
7. Назовите признаки, по которым могут быть классифицированы математические модели.
8. Что именно зависит от времени в динамических моделях?
9. При каких условиях можно перейти от стохастической модели к детерминированной?
10. В чем заключаются основные преимущества моделирования?

ГЛАВА 2 МЕТОДЫ АНАЛИЗА ДИНАМИКИ ПОВЕДЕНИЯ СИСТЕМ

Как правило, для изучения динамики поведения систем используют различные имитационные модели. Имитационное моделирование предполагает создание логико-математической модели сложной системы. Логическая структура моделируемой системы адекватно отображается в модели, а процессы ее функционирования, динамика взаимодействия ее элементов воспроизводятся (имитируются) на модели. Поэтому построение имитационной модели включает структурный анализ моделируемой системы и разработку функциональной модели, отражающей динамику поведения системы. Другой важной специфической особенностью имитационного моделирования как вида моделирования является то, что методом исследования компьютерной модели здесь является направленный вычислительный эксперимент, содержание которого определяется проведенными аналитическими исследованиями и соответствующими вычислительными процедурами, реализуемыми как на стадии стратегического планирования эксперимента, так и на стадии обработки, интерпретации его результатов.

Имитационное моделирование основано на применении логико-математической модели сложной системы со всеми вытекающими особенностями и осложнениям [3,4]. Во-первых, построение математической модели в отличие от структурно-функционального моделирования требует большого объема детальной информации о системе, включая всевозможные логические и количественные соотношения. Во-вторых, выбор математического аппарата существенно сказывается на самой имитационной модели и на выборе инструментальных средств. Ясно, что выбор излишне сложного математического аппарата (скажем, систем дифференциальных уравнений в частных производных) или привлечение большого числа методов из различных разделов математики значительно усложнит задачу

имитационного моделирования. В-третьих, при построении логико-математической модели всегда приходится решать проблему выбора между сложностью модели и ее точностью, удобством использования и ее универсальностью, поскольку эти критерии, как правило, противоречивы.

Имитационное моделирование представляет собой экспериментальный метод исследования реальной системы по ее имитационной модели, который сочетает особенности экспериментального подхода и специфические условия использования вычислительной техники.

Имитационную модель можно рассматривать как множество правил (дифференциальных уравнений, карт состояний, автоматов, сетей и т.п.), которые определяют, в какое состояние система перейдёт в будущем из заданного текущего состояния.

Применение имитационного моделирования целесообразно при наличии определенного условия. Эти условия определяет Р. Шеннон [5]:

1. Не существует законченной математической постановки данной задачи, либо еще не разработаны аналитические методы решения сформулированной математической модели. К этой категории относятся многие модели массового обслуживания, связанные с рассмотрением очередей.

2. Аналитические методы имеются, но математические процедуры столь сложны и трудоемки, что имитационное моделирование дает более простой способ решения задачи.

3. Кроме оценки определенных параметров, желательно осуществить на имитационной модели наблюдение за ходом процесса в течение определенного периода.

При имитационном моделировании структура моделируемой системы адекватно отображается в модели, а процессы ее функционирования проигрываются (имитируются) на построенной модели. Поэтому построение имитационной модели заключается в описании структуры и процессов

функционирования моделируемого объекта или системы. *В описании имитационной модели выделяют две составляющие* [6]:

– *статическое описание системы*, которое по-существу является описанием ее структуры. При разработке имитационной модели необходимо применять структурный анализ моделируемых процессов;

– *динамическое описание системы*, или описание динамики взаимодействий ее элементов. При его составлении фактически требуется построение *функциональной модели* моделируемых динамических процессов.

Идея метода, с точки зрения его программной реализации, состоит в следующем. Что, если элементам системы поставить в соответствие некоторые программные компоненты, а состояния этих элементов описывать с помощью переменных состояния. Элементы, по определению, взаимодействуют (или обмениваются информацией), значит, может быть реализован алгоритм функционирования отдельных элементов, т.е., моделирующий алгоритм. Кроме того, элементы существуют во времени, значит надо задать алгоритм изменения переменных состояний.

Отличительной особенностью метода имитационного моделирования является возможность описания и воспроизведения взаимодействия между различными элементами системы. Таким образом, чтобы составить имитационную модель, надо:

– представить реальную систему (процесс), как совокупность взаимодействующих элементов;

– алгоритмически описать функционирование отдельных элементов;

– описать процесс взаимодействия различных элементов между собой и с внешней средой.

Ключевым моментом в имитационном моделировании является выделение и описание *состояний* системы. Система характеризуется *набором*

переменных состояний, каждая комбинация которых описывает конкретное состояние. Следовательно, путем изменения значений этих переменных можно имитировать переход системы из одного состояния в другое.

Можно выделить следующие обобщенные задачи, решаемые средствами имитационного моделирования:

– прогнозирование развития системы;

– выявление наиболее эффективных механизмов реорганизации систем;

– анализ адаптивных свойств системы;

– оценка параметров надежности функционирования систем;

– анализ эксплуатационных параметров функционирования систем;

– анализ взаимодействий отдельных компонент системы.

2.1 Этапы имитационного моделирования

Этап 1. Формулировка проблемы и определение целей имитационного исследования.

На этом этапе определяется и детально изучается объект моделирования, те стороны его функционирования, которые представляют интерес для исследователя. заказчика. Результатом работ на данном этапе является содержательное описание объекта моделирования с указанием целей имитационного моделирования и тех аспектов функционирования объекта моделирования, которые необходимо изучить на имитационной модели. Содержательное описание составляется в терминологии реальной системы, как правило, на языке предметной области.

Общая последовательность действий на этом этапе следующая:

– сбор данных об объекте моделирования;

– составление содержательного описания объекта моделирования;

– изучение проблемной ситуации и постановка задачи;

– выбор метода моделирования.

Цели моделирования определяют определяют все последующие этапы имитационного моделирования. Примером целей имитационного моделирования могут служить оценка, прогнозирование, оптимизация, сравнение альтернатив и др.

Оценка – определение, насколько хорошо система предлагаемой структуры будет соответствовать некоторым конкретным критериям.

Сравнение альтернатив – сопоставление конкурирующих систем, рассчитанных на выполнение определенной функции, или же на сопоставление нескольких предлагаемых рабочих принципов или методик.

Прогноз – оценка поведения системы при некотором предполагаемом сочетании рабочих условий.

Анализ чувствительности – выявление из большого числа действующих факторов тех, которые в наибольшей степени влияют на общее поведение системы.

Выявление функциональных соотношений – определение природы зависимости между двумя или несколькими действующими факторами, с одной стороны, и откликом системы с другой.

Оптимизация – точное определение такого сочетания действующих факторов и их величин, при котором обеспечивается наилучший отклик всей системы в целом.

Этап 2. Разработка концептуального описания.

На этом этапе составляется концептуальная модель, т.е. логико-математическое описание моделируемой системы в соответствии с формулировкой проблемы. Основное содержание этого этапа – формулировка общей цели модели, переход от реальной системы к логической схеме ее функционирования. Здесь приводится описание объекта в терминах математических понятий и алгоритмизация функционирования ее компонент.

Концептуальное описание представляет собой упрощенное алгоритмическое отображение реальной системы.

При разработке концептуальной модели определяются границы системы, приводится описание внешней среды, выделяются существенные элементы и дается их описание, формируются переменные, параметры, функциональные зависимости как для отдельных элементов и процессов, так и для всей системы, ограничения, целевые функции (критерии). На этом этапе фиксируются все допущения (предположения), которые необходимо принять для построения имитационной модели. Обсуждается уровень детализации моделируемых процессов.

Этап 3. Формализация имитационной модели.

На третьем этапе имитационного исследования осуществляется формализация объекта моделирования.

Цель формализации – получить формальное представление логико-математической модели, т.е. алгоритмов поведения компонент системы и отразить на уровне моделирующего алгоритма взаимодействия между собой этих компонент.

При формализации задачи, прежде всего, необходимо уяснить, что дано, что надо найти, и каковы правила преобразования исходных данных в результат. А для этого, отталкиваясь от общего описания задачи надо выделить прототип моделирования и, опираясь на цели моделирования, решить вопрос, рассматривать его как целостный объект или как систему.

Если моделируется система, то производится ее анализ: выявляются составляющие системы и определяются связи между ними. При анализе необходимо также решить вопрос о степени детализации системы.

После определения прототипа необходимо определить, какие характеристики объекта будут учитываться при моделировании.

Методически формализацию проводят в виде поиска ответов на вопросы, уточняющие общее описание задачи:

– что моделируется;

– какие параметры моделируются;

– какие параметры известны, и какие надо определить;

– возможный диапазон значений;

– отношения и связи (для систем);

– установить правила (формулы) преобразования исходных данных в результат.

Этап 4. Программирование имитационной модели.

На данном этапе формальное описание модели сложной системы преобразуется в модель в соответствии с некоторой методикой (дисциплиной) программирования, с применением языков и систем моделирования. Важным моментом здесь является корректный выбор инструментального средства для реализации имитационной модели.

Этап 5. Сбор и анализ исходных данных.

Исходные данные для модели могут быть получены тремя способами:

1. Экспериментально. На практике, если речь идет о крупной системе (предприятие, крупный проект – множество исполнителей), возможность получения экспериментальных данных весьма ограничена в связи с трудоемкостью данного процесса и сложностью оценки достоверности полученных данных.

2. Эмпирические данные. На сегодняшний день данный способ становится наиболее востребованным и эффективным. Связано это, прежде всего, с массовым распространением информационных сервисов и технологий, которые являются поставщиками огромных массивов разнородных данных.

Здесь существует две альтернативы:

– использование данных непосредственно;

– использование теоретико-вероятностных или частотных распределений. Очевидно, что значительная часть параметров системы – это случайные величины.

3. Экспертные оценки. Когда нет достоверных экспериментальных и эмпирических данных, приходится полагаться на субъективные оценки, которые могут быть получены в процессе анализа мнений специалистов, основанных на профессиональном, научном и практическом опыте. Различают индивидуальные и коллективные экспертные оценки.

В случае участия в опросе нескольких экспертов расхождения в их оценках неизбежны, однако величина этого расхождения имеет важное значение. Групповая оценка может считаться достаточно надежной только при условии хорошей согласованности ответов отдельных специалистов.

Для анализа разброса и согласованности оценок применяются статистические характеристики – меры разброса или статистическая вариация.

Этап 6. Испытание и исследование модели, проверка модели [6].

Валидация модели

В общем случае валидация предполагает проверку соответствия между поведением имитационной модели и исследуемой реальной системы. Валидация модели есть подтверждение того, что модель в пределах рассматриваемой области приложений ведет себя с удовлетворительной точностью в соответствии с целями моделирования.

Верификация модели

Это проверка на соответствие поведения модели замыслу исследователя и моделирования. Т.е. процедуры верификации проводят, чтобы убедиться, что модель ведет себя так, как было задумано. Для этого реализуют формальные и неформальные исследования имитационной модели. Верификация имитационной модели предполагает доказательство возможности

использования создаваемой программной модели в качестве машинного аналога концептуальной модели на основе обеспечения максимального сходства с последней. Цель процедуры верификации – определить уровень, на котором это сходство может быть успешно достигнуто. Валидация и верификация имитационной модели связаны с обоснованием внутренней структуры модели, в ходе этих процедур проводятся испытания внутренней структуры и принятых гипотез, исследуется внутренняя состоятельность модели.

На этапе верификации устанавливается верность логической структуры модели, реализуется комплексная отладка с использованием средств трассировки, ручной имитации, в ходе которой проверяется правильность реализации моделирующего алгоритма. Комплексные процедуры верификации включают неформальные и формальные исследования. Неформальные процедуры могут состоять из серии проверок следующего типа:

– проверка преобразования информации от входа к выходу;

– трассировка модели на реальном потоке данных;

– обязательное масштабирование временных параметров в зависимости от выбранного шага моделирования;

– тестирование модели для критических значений и при наступлении редких событий.

Формальные процедуры связаны с проверкой исходных предположений выдвинутых на основе опыта, теоретических знаний, интуитивных представлений, на основе имеющейся информации.

Валидация данных

Валидация данных направлена на доказательство того, что все используемые в модели данные, в том числе входные, обладают удовлетворительной точностью и не противоречат исследуемой системе, а значения параметров точно определены и корректно используются. С этой целью

проводят исследование свойств имитационной модели: оценивается точность, устойчивость, чувствительность результатов моделирования.

Точность имитационной модели показывает относительную величину разброса данных на выходе модели. Для определения точности результатов имитации оцениваем доверительные интервалы. Если мы имеем оценку истинного среднего μ совокупности, мы определяем верхнюю и нижнюю границы интервала, так, чтобы вероятность попадания истинного среднего в интервал, заключенный между этими границами, равнялась некоторой заданной величине (*a* – доверительная вероятность).

Под *устойчивостью* результатов имитации будем понимать степень нечувствительности ее к изменению условий моделирования. Устойчивость результатов моделирования характеризуется сходимостью контролируемого параметра моделирования к определенной величине при увеличении времени моделирования. На практике, рекомендуется устойчивость результатов моделирования оценивать дисперсией значений отклика (по выбранной компоненте). Если эта дисперсия при увеличении времени моделирования не увеличивается, значит, результаты моделирования устойчивы.

Анализ чувствительности модели определяет оценку влияния колебаний значений входных переменных на отклики (выходные переменные) модели.

Анализ чувствительности обычно заключается в том, что величины параметров систематически варьируются в некоторых представляющих интерес пределах и при этом наблюдается влияние этих вариаций на характеристики модели. Если при незначительных изменениях величин некоторых параметров результаты меняются очень сильно, это может служить основанием для затраты большего количества времени и средств с целью получения более точных оценок.

Этап 7. Планирование и проведение имитационного эксперимента.

Имитационный эксперимент, содержание которого определяется предварительно-проведенным аналитическим исследованием, результаты которого, достоверны и математически обоснованы, называется направленным вычислительным экспериментом.

В зависимости от решаемой задачи имитационный эксперимент может выполняться с целью:

– оценки значений выходных переменных при заданных параметрах системы;

– сравнения альтернативных вариантов функционирования системы;

– получения знаний о влиянии управляемых параметров на результаты эксперимента;

– нахождения оптимальных параметров функционирования системы;

– вариантного синтеза.

В таблице 2.1 приведены типы вычислительных экспериментов в зависимости от целей имитационного моделирования.

Табл. 2.1. Типы направленных вычислительных экспериментов

Цели вычислительного эксперимента	**Тип направленного вычислительного эксперимента**
Оценка выходных переменных при заданных параметрах системы	Оценка и сравнение средних и дисперсий различных альтернатив
Сравнение альтернатив (или выбор на множестве альтернатив)	Оценка и сравнение средних и дисперсий различных альтернатив
Получение знаний о влиянии управляемых параметров на результаты эксперимента	Анализ чувствительности
Определение тех значений входных параметров и переменных, при которых достигается оптимальный выход	Поиск оптимума на множестве возможных значений переменных
Вариантный синтез	Многокритериальная оптимизация, выбор

2.2 Пример постановки задачи имитационного моделирования

Подробное описание примера с решением приведено в [7].

Описание ситуации

Электрик Петров приставил к стене лестницу и, поднявшись вверх, остановился на одной из ступенек. В это время концы лестницы начали скользить вдоль стены и пола.

Цели моделирования

Исследовать движение ступеньки лестницы, на которой стоит электрик.

Объект моделирования – «лестница», который представляет систему, состоящую из ступенек

Имеющиеся данные: длина L, количество ступенек N, угол φ, образуемый лестницей и стеной, ступеньки расположены на одинаковом расстоянии, верхняя совпадает с концом лестницы. Известен номер ступеньки k, на которой стоит электрик Петров Начальное положение можно считать вертикальным угол $\varphi_0=0$. Концы лестницы скользят вдоль стены и пола, угол φ изменяется от 0 до 90°.

Надо определить кривую, по которой движется ступенька (и вместе с ней электрик Петров).

Технология построения кривой. Как строится кривая?

Кривая строится по точкам. Точка кривой характеризуется координатами x и y, которые определяются в некоторой заданной системе координат и связаны с углом отклонения лестницы. Координаты определяются через равные промежутки изменения угла φ ($\Delta\varphi$). h_φ – шаг изменения угла должен быть задан.

Описание задачи в формальной постановке

Верхний конец A лестницы длиной L прислонен к вертикальной стене, а ее нижний конец B расположен на горизонтальной поверхности. На лест-

нице N ступенек, расположенных на одинаковом расстоянии d. Первая ступенька расположена на расстоянии d от точки B, последняя – на верхнем конце A. Трение отсутствует. В начальный момент времени лестница имела угол наклона к стене равный φ_0. Далее она начинает падать, так что верхний конец скользит по стене, а нижний – по горизонтальной поверхности. Построить траекторию движения точки (x,y), соответствующей k-той ступеньке в координатах (X,Y) , где X – горизонтальная ось, направленная вдоль поверхности земли, а Y – вертикальная ось, направленная вдоль стены вверх.

2.3 Классификация имитационных моделей

Простейшая классификация на основные виды имитационных моделей связана с применением двух этих способов продвижения модельного времени. Различают имитационные модели:

– непрерывные;

– дискретные;

– непрерывно-дискретные.

В *непрерывных имитационных моделях* переменные изменяются непрерывно, состояние моделируемой системы меняется как непрерывная функция времени, и, как правило, это изменение описывается системами дифференциальных уравнений. Соответственно продвижение модельного времени зависит от численных методов решения дифференциальных уравнений.

В *дискретных имитационных моделях* переменные изменяются дискретно в определенные моменты имитационного времени (наступления событий). Динамика дискретных моделей представляет собой процесс перехода от момента наступления очередного события к моменту наступления следующего события.

Поскольку в реальных системах непрерывные и дискретные процессы часто невозможно разделить, были разработаны *непрерывно-дискретные модели*, в которых совмещаются механизмы продвижения времени, характерные для этих двух процессов.

2.4 Технологии имитационного моделирования

На рисунке 2.1 показаны основные подходы в имитационном моделировании: системная динамика (СД), дискретно-событийное моделирование (ДС), агентное моделирование (АМ). СД и динамические системы оперируют в основном с непрерывными во времени процессами, тогда как ДС и АМ – в основном дискретными.

Рис. 2.1. Подходы в имитационном моделировании
на шкале уровня абстракции [8]

Динамические системы находятся внизу шкалы. СД, заменяя индивидуальные объекты их агрегатами, наоборот, предполагает наивысший уровень абстракции. ДС-моделирование работает в низком и среднем диапазоне. АМ может применяться практически на любом уровне и в любых

масштабах. Агенты могут представлять пешеходов, автомобили или роботов в физическом пространстве, клиента или продавца на среднем уровне, или же конкурирующие компании на высоком [8].

2.4.1 Дискретно-событийное моделирование

Термин «дискретно-событийное моделирование» исторически возник для описания моделей, описывающих системы обслуживания потоков объектов некоторой природы: клиентов магазина, автомобилей на заправочных станциях, туристов у стойки регистрации на рейс, междугородних переговоров и т.д. В основе данного подхода лежит концепция заявок (транзактов, entities), ресурсов и потоковых диаграмм (flowcharts), определяющих потоки заявок и использование ресурсов. Этот подход восходит к Джеффри Гордону, который в 1960-х придумал язык GPSS. Заявки (транзакты в GPSS) – это пассивные объекты, представляющие людей, детали, документы, задачи, сообщения и т.п. Они перемещаются, ожидают в очередях, обрабатываются, захватывают и освобождают ресурсы и т.д. Считается, что все заявки обладают универсальной логикой поведения и обрабатываются по единому, заранее известному алгоритму. На рисунке 2.2 изображена потоковая диаграмма, моделирующая работу call-центра.

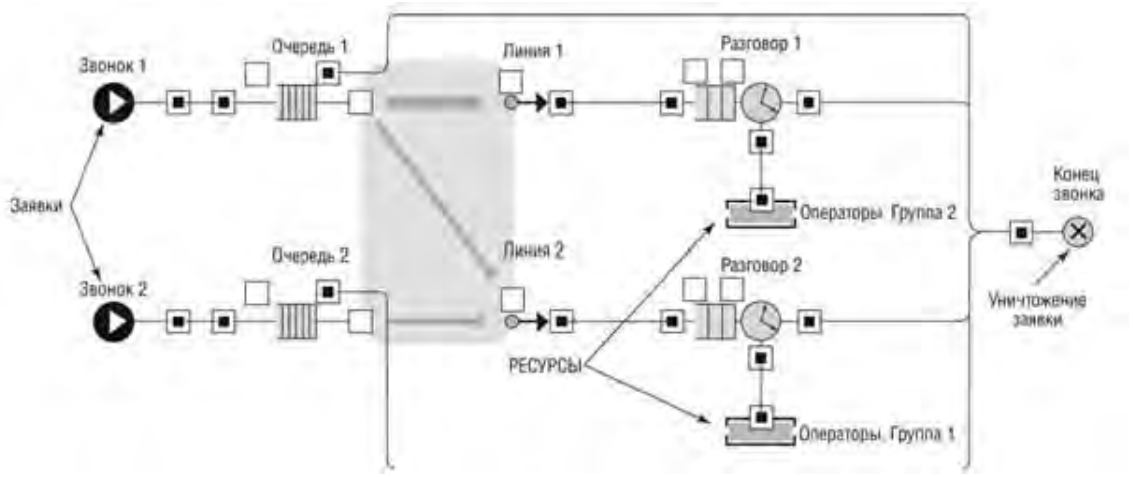


Рис. 2.2. Потоковая диаграмма «обработка звонков в call-центре» [7]

2.4.2 Системная динамика

Этот подход был разработан и предложен Джеем Форрестером в конце 1950-х годов.

Системная динамика позволяет исследовать поведение системы посредством выявления причинно-следственных отношений и взаимодействий контуров обратной связи, проявляющихся в особенностях ее структурной организации. К числу достоинств метода относятся: возможность отражать практически любую причинно-следственную связь; простая математическая форма.

Для выявления причинно-следственных отношений необходимо собрать и структурировать всю информацию, касающуюся изучаемого процесса или явления. После этого строится формальная модель системы, как правило, в виде логических диаграмм, которые впоследствии преобразуются в сетевую модель и соответствующую систему уравнений.

Формальная модель состоит из набора абстрактных элементов, представляющих некие свойства моделируемой системы. Выделяются следующие типы элементов [9]:

– *уровни* – характеризуют накопленные значения величин внутри системы. Уровни применимы не только к физическим величинам. Например, уровень осведомленности существенен при принятии решения. Уровни представляют собой значения переменных, накопленные в результате разности между входящими и исходящими потоками. На диаграммах изображаются прямоугольниками;

– *потоки* – скорости изменения уровней. Например, потоки информации, заказов, денежных средств, рабочей силы. Потоки могут быть направлены к уровню или от него. Изображаются сплошными стрелками;

– *функции решений* (*вентили*) – функции зависимости потоков от уровней. На диаграммах изображаются двумя треугольниками в виде ба-

бочки. Функция решения может иметь форму простого уравнения, определяющего реакцию потока на состояние одного или двух уровней. Функции задают правило выбора решений, связанных с величинами текущих темпов. Все решения касаются предстоящих действий и выражаются в форме темпов потока;

– *каналы информации*, соединяющие вентили с уровнями. Изображаются штриховыми стрелками;

– *линии задержки* (*запаздывания*) – служат для имитации задержки потоков. Характеризуются параметрами среднего запаздывания и типом неустановившейся реакции. Второй параметр характеризует отклик элемента на изменение входного сигнала. Разные типы линий задержки имеют различный динамический отклик;

– *вспомогательные переменные* – располагаются в каналах информации между уровнями и функциями решений и определяют некоторую функцию. Изображаются кружком.

Основой имитации служит численное решение систем дифференциальных уравнений. Информация о модели поступает от различных накопителей об уровнях и от разных каналов связи о темпах. На основе этой информации на каждый шаг имитации вырабатывается управляющее воздействие, т.е. темп управляемого потока. Каждый решающий элемент вырабатывает темп для воздействия на одну связь.

Основные этапы системно-динамического моделирования:

1) определение целей моделирования;
2) концептуальное описание исследуемой системы;
3) определение структуры модели и ее ключевых переменных;
4) описание поведения системы;
5) построение системной диаграммы;
6) формализация зависимостей и построение уравнений;

7) подбор значений параметров;

8) тестирование модели;

9) вычислительный эксперимент.

Пример системно-динамической модели [8,10]

Рассмотрим модель распространения нового продукта, или инновации (рис.2.3).

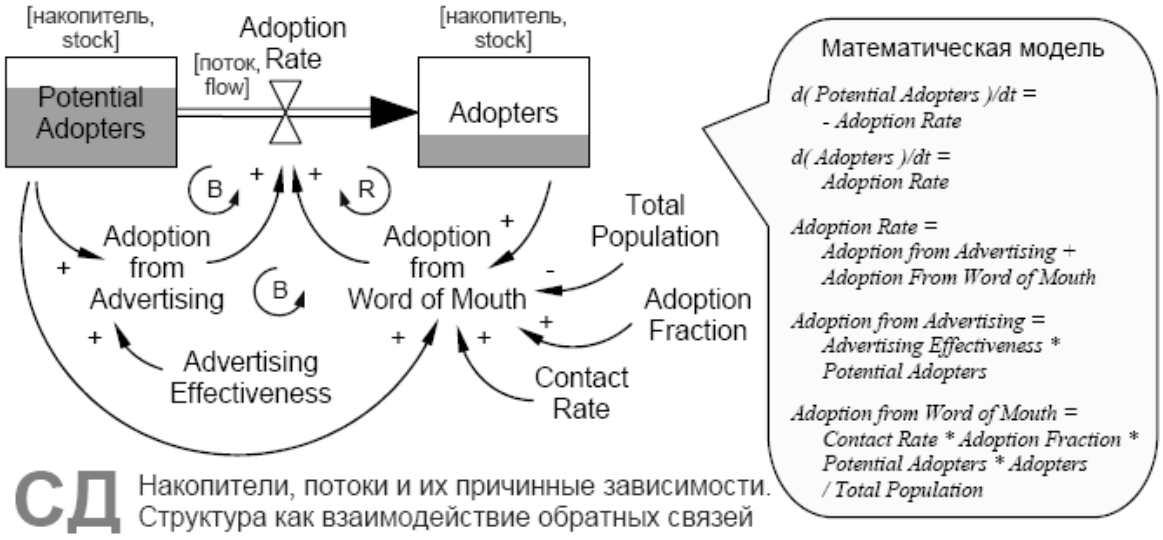


Рис. 2.3. Модель системной динамики: Bass Diffusion в VensimTM

Потенциальные клиенты (Potential Adopters) становятся клиентами (Adopters) со скоростью диффузии, распространения (Adoption Rate), которая зависит от рекламы и “устной рекламы”, т.е. общения клиентов с неклиентами. Влияние рекламы моделируется следущим образом: некий постоянный процент потенциальных клиентов (Advertising Effectiveness = 0.011 в этой статье) всё время становятся клиентами. Их доля в Adoption Rate равна, соответственно, Potential Adopters * Advertising Effectiveness. Что касается устной рекламы, мы делаем предположение, что в нашей группе людей все контактируют со всеми. Количество контактов человека в единицу времени обозначим как Contact Rate (100). В случае если клиент

общался с потенциальным клиентом, последний становится клиентом с вероятностью Adoption Fraction (0.015). Таким образом, в единицу времени все клиенты обратят Adopters * Contact Rate * Adoption Fraction * [Potential Adopters / (Potential Adopters + Adopters)] потенциальных клиентов в клиентов. Выражение в квадратных скобках – вероятность того, что тот, с кем был контакт у клиента, ещё не клиент.

2.4.3 Модели динамических систем

Данная технология моделирования используется в механике, электронике, энергетике, химии как часть стандартного процесса разработки. На рисунке 2.4 показана типичная блок-схема в языке MATLABTM SimulinkTM [8].

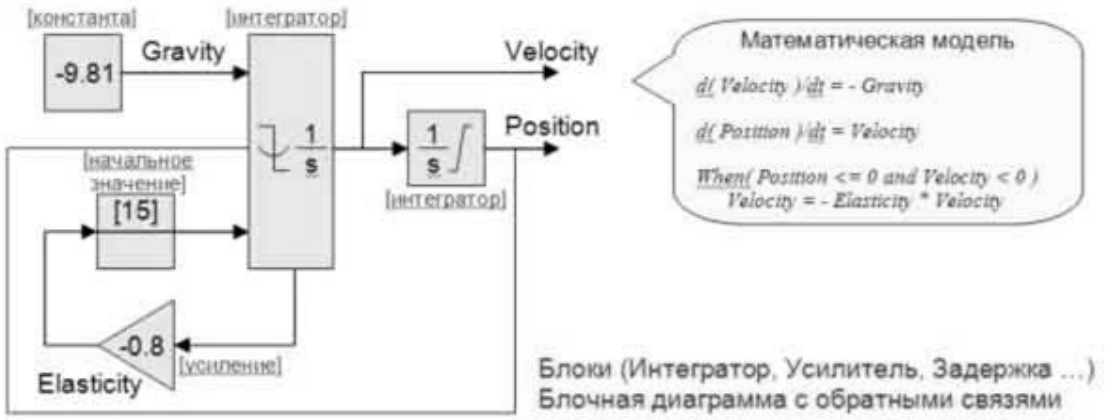


Рис. 2.4. Модель динамической системы: прыгающий мячик в MATLABTM SimulinkTM [8]

Соответствующая математическая модель, как и в случае СД, будет состоять из набора переменных состояния и системы алгебро-дифференциальных уравнений над ними. В отличие от СД, здесь переменные состояния имеют прямой “физический” смысл: координата, скорость, давление, концентрация, и т.д.; они естественно непрерывные и не являются агрегатами (количествами) дискретных объектов. Математическое раз-

нообразие и сложность в динамических системах могут быть значительно выше, чем в системной динамике, так что в принципе любая СД-проблема может быть решена инструментами для моделирования динамических систем, и даже с лучшей точностью (за счёт более совершенных численных методов) [8].

2.4.4 Агентное моделирование

Агент – это развитие известного понятия "объект", представляющего абстракцию множества экземпляров предметов реального мира, имеющих одни и те же свойства и правила поведения. Точное определение агента на сегодняшний день отсутствует. В основном используется определение, принятое на конференции международной ассоциации по лингвистике FIRA (Federation of Intelligent Physical Agents) в октябре 1996 года в Токио: "Агент – это сущность, которая находится в некоторой среде, интерпретирует их и исполняет команды, воздействующие на среду. Агент может содержать программные и аппаратные компоненты. Отсутствие четкого определения мира агентов и присутствие большого количества атрибутов, с ним связанных, а также существование большого разнообразия примеров агентов говорят о том, что агенты – это достаточно общая технология, которая аккумулирует в себе несколько различных областей" [12].

Как правило, используя понятие "агент", каждый исследователь определяют своего агента с конкретным набором свойств в зависимости от целей разработки, решаемых задач, техники реализации, критериев. Как следствие, в рамках данного направления появилось множество типов агентов, например: автономные агенты, персональные ассистенты, интеллектуальные агенты, социальные агенты и т.д. В зависимости от степени возможности внутреннего представления внешнего мира и способа пове-

дения агенты классифицируются как локальные, сетевые, мобильные, интерфейсные, транслирующие, маршрутизации и т.д.

Идея многоагентности предполагает кооперацию агентов при коллективном решении задач. В многоагентной системе агент, который не способен решить некоторую задачу самостоятельно, может обратиться к другим агентам. Другой вариант, когда необходима кооперация- это использование коллектива агентов для решения одной общей трудной задачи. При этом агенты могут строить планы действий, основываясь уже не только на своих возможностях, но и “думать” о планах и намерениях других агентов [12].

Важное преимущество агентного моделирования в том, что разработка модели возможна в отсутствии знания о глобальных зависимостях: вы можете знать очень немного о том, как вещи влияют друг не друга на глобальном уровне, или какова глобальная последовательность операций, и т.п., но, понимая индивидуальную логику поведения участников процесса, вы сможете построить агентную модель и вывести из неё глобальные поведение [8].

Совместное поведение различных объектов изучается в рамках многих научных дисциплин. Выделим среди них те, которые представляются наиболее адекватными идее коллектива интеллектуальных агентов [12].

– *распределенный искусственный интеллект* [13, 14]. Эта область искусственного интеллекта занимается самыми общими аспектами коллективного поведения агентов. Здесь основу составляют результаты, полученные в теории распределенных систем и теории принятия решений;

– *теория игр* [15]. Аппарат теории игр часто используется для исследования коллективов интеллектуальных агентов. Многие ситуации, возникающие в многоагентной системе, находят подходящие аналоги в теории игр. Исследуются кооперативные игры, различные стратегии ведения тор-

гов (переговоров), игры в размещения и др., которые являются аналогами ряда моделей коллективного поведения агентов;

– *теория коллективного поведения автоматов* [16]. Она исследует поведение больших коллективов автоматов с примитивными функциями. Поведение автомата может рассматриваться как недетерминированное, что позволяет строить различные вероятностные модели. Допускается обучение автомата при помощи штрафов и поощрений. Автомат может быть наделен памятью, в которой он в некоторой форме запоминает предыдущие штрафы и поощрения, и может использовать эту информацию для улучшения своего и коллективного поведения в соответствии с некоторой функцией дохода;

– *биологические, экономические и социальные модели.*

Пример агентной модели (Динамика населения страны) [8]

В этой модели поведение агента задано картой состояний (statechart). Модель среды может включать жильё, рабочие места, транспортную инфраструктуру и т.д. (рис. 2.5).

Рис. 2.5. Типичная архитектура агентной модели.

Поведение (карта состояний) в AnyLogicTM

Вопросы к главе 2

1. В чем специфика имитационного моделирования?
2. Недостатки и преимущества имитационного моделирования.
3. Сфера применения имитационных моделей.
4. Основные этапы ИМ.
5. Какие классификационные признаки имитационных моделей вы знаете?
6. Какие свойства имитационной модели, изучаются для оценки ее качества?
7. Перечислите технологии имитационного моделирования.
8. Перечислите основные компоненты системно-динамической модели.
9. Для описания каких процессов используют дискретно-событийное моделирование?
10. Опишите идею агентного моделирования.

ГЛАВА 3 МОДЕЛИ ДЛЯ ИССЛЕДОВАНИЯ ЗАВИСИМОСТЕЙ

Задачи, связанные с установлением зависимостей активно изучаются уже более 200 лет, с момента разработки К. Гауссом в 1794 г. метода наименьших квадратов. На его основе разработано огромное количество методов и инструментов которые успешно используются в настоящее время. Тем не менее, за последнее время появились альтернативные подходы для решения задач анализа зависимостей. К ним можно отнести экспертные методы, аффинитивный анализ и пр.

3.1 Регрессионный анализ

Задачи, связанные с установлением зависимостей активно изучаются уже более 200 лет, с момента разработки К. Гауссом в 1794 г. метода наименьших квадратов. В математической статистике с этого времени было разработано огромное количество методов и инструментов для решения этих задач. Рассмотрим один из наиболее значимых и распространенных подходов – регрессионный анализ.

В общем случае, регрессионный анализ представляет собой задачу о выявлении внутренних свойств объекта по имеющимся данным о входах и выходах. Термин "регрессия" был введён Ф. Гальтоном (1886) и сначала он использовался в биологическом смысле. В статистике термин стал использоваться после опубликования работ К Пирсона (Pearson) в 1908 г.

Регрессионная модель f(w,x) – это параметрическое семейство функций, задающее отображение

$$f\colon W\times X\to Y,$$

где $w\in W$ – пространство параметров, $x\in X$ – пространство свободных переменных, Y – пространство зависимых переменных.

Так как регрессионный анализ предполагает поиск зависимости математического ожидания случайной величины от свободных переменных $E(y|x) = f(x)$, то в её состав входит аддитивная случайная величина ε:

$$y = f(w,x) + \varepsilon. \quad (3.1)$$

Параметры модели настраиваются таким образом, что модель наилучшим образом приближать данные. Критерием качества приближения (целевой функцией) обычно является среднеквадратичная ошибка. Предполагается, что зависимая переменная есть сумма значений некоторой модели и случайной величины. Относительно характера распределения случайной величины *y* делаются предположения, называемые гипотезой порождения данных. Для подтверждения или опровержения этой гипотезы выполняются статистические тесты, такие как, например, *F*-тест, или критерий Фишера.

С помощью регрессионного анализа исследуются внутренние механизмы рассматриваемого явления, и оценивается роль отдельных факторов.

Основные задачи регрессионного анализа следующие:

– определения вида и формы зависимости;

– оценка параметров уравнения регрессии;

– проверка значимости уравнения регрессии;

– проверка значимости отдельных коэффициентов уравнения;

– построение интервальных оценок коэффициентов;

– исследование характеристик точности модели;

– построение точечных и интервальных прогнозов результирующей переменной.

Задача нахождения регрессионной модели ставится следующим образом [17]. Задана выборка – множество $\{x_1, x_2, \dots, x_N | x \in \mathbb{R}\}$ значений свободных переменных и множество $\{y_1, y_2, \dots, y_N | y \in \mathbb{R}\}$ соответствующих

им значений зависимой переменной. Эти множества обозначаются как D, множество исходных данных $\{(x, y)_i\}$. Задана регрессионная модель (5.1) зависящая от параметров $w \in \mathbb{R}$ и свободных переменных x. Требуется найти наиболее вероятные параметры w.

Методы регрессионного анализа делят на *многомерные* и *одномерные* в зависимости от числа независимых переменных, *линейные* и *нелинейные.* Для нахождения параметров нелинейных регрессионных моделей используются методы оптимизации, например метод сопряжённых градиентов, метод Ньютона-Гаусса и т.д.

Для реализации множественной регрессии существует множество подходов. Например, шаговый и ступенчатый метод [18], метод группового учета аргументов (МГУА) [19].

3.1.1 Модель линейной регрессии

Линейная регрессия предполагает, что функция f зависит от параметров w линейно. При этом линейная зависимость от свободной переменной x необязательна.

$$y = f(w, x) + \nu = \sum_{i=1}^{N} w_i g_i(x) + \nu \qquad (3.2)$$

В случае, когда функция $g_i(x) = x_i$, тогда линейная регрессия имеет вид

$$y = \sum_{i=1}^{N} w_i x_i + \nu = \langle w, x \rangle + \nu \qquad (3.3)$$

здесь x_i – компоненты вектора x.

Значения параметров в случае линейной регрессии находят с помощью метода наименьших квадратов.

Разности $y_i - f(x_i)$ между фактическими значениями зависимой переменной и восстановленными называются *регрессионными остатками*. В литературе используются также синонимы: *невязки* и *ошибки*.

3.1.2 Шаговый регрессионный метод

Шаговый регрессионный метод (ШРМ) реализует следующую последовательность действий [18]:

Вычисляются коэффициенты корреляции r_{yx1}, $r_{yx2},\ldots,r_{yxm}$ и первой вводится в уравнение регрессии переменная с номером w_1, для которой выполняется условие

$$v_1 = r^2_{yx_{w_1}} = \max\left\{r^2_{yx_1},\ldots,r^2_{yx_m}\right\}. \qquad (3.4)$$

Строится уравнение

$$\hat{y} = a_0{}^{(1)} + a_{w1}{}^{(1)} \cdot x_{w1} \qquad (3.5)$$

Записывается полная дисперсия как сумма дисперсий, вызванной влиянием фактора x_{w1}, и дисперсии, вызванной влиянием неучтенных факторов $\sigma_\varepsilon{}^2$, т.е.

$$\sigma^2_y = \sigma^2_{w_1} + \sigma^2_\varepsilon \qquad (3.6)$$

Предполагается, что x_{w1} и ε независимы. Тогда

$$\sigma^2_{w_1} = v_1\sigma^2_y;\ \sigma^2_\varepsilon = \sigma^2_y - \sigma^2_{w_1} = (1-v_1)\sigma^2_y \qquad (3.7)$$

Далее вычисляются частные коэффициенты корреляции первого порядка $r_{yxj\cdot xw1}$ для $j \neq w_1$ и $j = 2,\ldots,m$. В уравнение регрессии вводится та переменная, например, x_{w2}, которая имеет наибольший частный коэффициент корреляции первого порядка, т.е.

$$r^2_{yx_{w_2}\cdot x_{w_1}} = \max\left\{r^2_{yx_2\cdot x_{w_1}},\ldots,r^2_{yx_m\cdot x_{w_1}}\right\}. \qquad (3.8)$$

Тогда для полной вариации зависимого фактора, вычисленного для x_{w2}, исключая влияние x_{w1}, будет

$$v_2 = (1 - r_{yxw1}{}^2)\cdot r_{yxw2\cdot xw1} \qquad (3.9)$$

Результирующее уравнение регрессии запишется в виде, соответствующем

$$\hat{y} = a_0{}^{(2)} + a_{w1}{}^{(2)} \cdot x_{w1} + a_{w2}{}^{(2)} \cdot x_{w2} \qquad (3.10)$$

с

$$\sigma^2_{w_2} = (v_1 + v_2)\sigma^2_y, \tag{3.11}$$

а остаточная дисперсия

$$\sigma^2_\varepsilon = (1 - v_1 - v_2)\sigma^2_y. \tag{3.12}$$

Продолжая дальше этот процесс, можно ввести каждую из оставшихся $m - 2$ независимых переменных и так далее анализируя остаточные дисперсии, получаем данные о существенности введенной переменной.

Основная идея шагового метода состоит в нахождении регрессии с несколькими переменными в виде серий линейных регрессионных зависимостей и в преобразовании исходной корреляционной матрицы шаг за шагом. На каждом шаге получают результаты для анализа дисперсий с помощью F-критерия для проверки двух типов гипотез: одна для включения переменной в уравнение регрессии, другая для исключения из него.

Вычислительная процедура состоит в том, чтобы применить линейное преобразование к матрице

$$\mathbf{A} = \begin{vmatrix} \mathbf{R}_{xx} & r_{xy} & \mathbf{E} \\ r_{xy}^{\mathrm{T}} & r_{yy} & \mathbf{D} \\ -\mathbf{E} & \mathbf{B} & \mathbf{C} \end{vmatrix}, \tag{3.13}$$

где

$\mathbf{R}_{xx}$ – корреляционная матрица размера $m \times m$;

r_{xy}^{T} – вектор-строка коэффициентов корреляции;

r_{xy} – вектор-столбец коэффициентов корреляции;

E – единичная матрица размера $m \times m$;

r_{yy} – коэффициент корреляции ($r_{yy} = 1$);

D – вектор строка с m нулевыми элементами;

B – вектор столбец с m нулевыми элементами;

C – матрица размера mxm с нулевыми элементами.

3.1.3 Метод группового учета аргументов (МГУА)

Теория самоорганизации моделей, положенная в основу МГУА [19], отвергает путь расширения и усложнения модели, увеличения объема входных данных, постулируя при этом существование оптимального, ограниченного размера области моделирования и единственной модели оптимальной сложности. Их можно найти при помощи самоорганизации, т.е. перебора многих моделей претендентов (конкурирующих моделей) по целесообразно выбранным критериям. В отличие от традиционных методов регрессионного анализа, использующих критерий среднеквадратической ошибки, который является внутренним (рассчитанным по всем точкам выборки), индуктивный метод самоорганизации основан на применении внешних критериев, для определения которых используют входные данные, не задействованные в процессе структурно-параметрической идентификации модели. Любой внутренний критерий сравнения конкурирующих моделей приводит к ложному правилу: чем сложнее структура модели, тем она точнее.

Согласно МГУА входные данные Z делятся, по крайней мере, на две части, называемые обучающей Z_α и проверочной Z_β последовательностями. При этом

$$\dim Z_\alpha = n_\alpha \times (m + 1),\ \dim Z_\beta = n_\beta \times (m + 1),\ n = n_\alpha + n_\beta.$$

Среднеквадратическая ошибка, определяемая на проверочной последовательности, является одним из критериев выбора структуры модели, получаемой по данным обучающей последовательности. В качестве внешних критериев в алгоритмах МГУА используются критерий регулярности, минимума смещения и др. Критерий регулярности имеет вид:

$$\Delta_\beta{}^2 = \frac{\sum_{i=1}^{n_\beta} (\hat{y}_i - y_i)^2}{\sum_{i=1}^{n_\beta} y_i^2}. \qquad (3.14)$$

Для расчета критерия минимума смещения $N_{..}^2$ сначала идентифицируется модель $\hat{y} = f_\alpha(\hat{a}_i, x)$ при условии, что Z_α – обучающая, Z_β – проверочная последовательности. Затем идентифицируется модель $\hat{y} = f_\beta(\hat{a}_i, x)$, при условии, что Z_β – обучающая, а Z_α – проверочная последовательности.

$$N^2 = \frac{\sum_{i=1}^{n}\left[f_\alpha(\hat{a}_i, x) - f_\beta(\hat{a}_i, x)\right]^2}{\sum_{i=1}^{n} y_i^2} \qquad (3.15)$$

Задача структурно-параметрической идентификации модели оптимальной сложности решается в два этапа. На первом этапе из полного набора моделей различной сложности отбираются по одному из внешних критериев k лучших структур, параметры которых на втором этапе пересчитываются по всей выборке n. На основе сравнительного анализа k структур полученных моделей и соответствующих значений внешних критериев выбирается лучшая модель.

3.1.4 Оценка качества модели регрессии

Проверка качества уравнения регрессии включает проверку ее общей точности, а также точности оценок параметров. Для этого необходимо проверить:

а) является ли модель и ее параметры статистически значимыми (позволяет ли имеющаяся выборка судить о генеральной совокупности);

б) какова практическая ценность модели (насколько информация, полученная по модели, снимает неопределенность в отношении регрессора Y).

Снижение качества модели может быть вызвано:

– наличием выбросов;

– наличием влиятельных наблюдений;

– нарушением основных предположений регрессионного анализа;

– наличие мультиколлинеарности.

Наиболее простыми критериями можно считать *средние абсолютное и относительное отклонения* («mean absolute error» и «mean absolute percentage error»):

$$MAE = \frac{1}{n}\sum_{k=1}^{n}\left|Y_k - Y_k^*\right|, \tag{3.16}$$

$$MAPE = \frac{1}{n}\sum_{k=1}^{n}\left|\frac{Y_k - Y_k^*}{Y_k}\right| \cdot 100\%. \tag{3.17}$$

Первый критерий выражается в тех же единицах измерения, что и моделируемый ряд динамики, поэтому он может быть использован только для сравнения моделей одного и того же ряда динамики. С другой стороны, он позволяет оценить «физическое содержимое» погрешности моделирования.

Среднее относительное отклонение является безразмерной величиной и позволяет судить о точности модели и сравнивать их между собой. Однако ее значение также во многом зависит от уровней ряда динамики. Если значения Y_k велики по сравнению со своим разбросом, то MAPE-оценка будет малой, а если Y_k близки к нулю, то MAPE-оценка будет большой вне зависимости от точности модели. Если же имеются наблюдения, строго равные нулю, то использовать относительные величины вообще невозможно.

Одной из важных оценок критерия качества полученной зависимости является *сумма квадратов остатков*:

$$SSE = \sum_{i=1}^{N}\left(y_i - f(w, x_i)\right)^2. \tag{3.18}$$

Коэффициент корреляции (выборочный) определяет силу линейной зависимости между показателями:

$$r_{YX} = r_{XY} = \frac{m_{YX} - m_Y m_X}{s_Y s_X}, \qquad (3.19)$$

$$-1 \le r_{YX} \le 1.$$

Чем ближе модуль $|r_{YX}|$ к 1, тем точнее модель. При $|r_{YX}| = 1$ между X и Y существует функциональная зависимость, при $|r_{YX}| = 0$ линейная связь полностью отсутствует.

Знак r_{XY} определяет направление зависимости (положительная или отрицательная).

Для нелинейных моделей коэффициент корреляции рассчитывается для их линеаризованной формы, например для логарифмической модели рассчитывается

$$r_{Y\ln X} = \frac{m_{Y\ln X} - m_Y m_{\ln X}}{s_Y s_{\ln X}}. \qquad (3.20)$$

Обычно считают, что если $|r_{YX}| < 0{,}5$, то линейная зависимость отсутствует, если $|r_{YX}| < 0{,}7$ – зависимость слабая, если $|r_{YX}| \ge 0{,}9$ – присутствует сильная линейная зависимость.

Коэффициент детерминации R^2:

$$R^2 = 1 - \frac{\sum_{k=1}^{n}\left(Y_k^* - Y_k\right)^2}{\sum_{k=1}^{n}\left(Y_k - m_Y\right)^2} = 1 - \frac{\sum_{k=1}^{n} e_k^2}{\sum_{k=1}^{n}\left(Y_k - m_Y\right)^2}, \qquad (3.21)$$

$$\sum_{k=1}^{n}\left(Y_k^* - Y_k\right)^2 = \sum_{k=1}^{n}\varepsilon_k^2$$

(сумма квадратов отклонений) – мера остаточного, не объясненного моделью разброса исходных данных.

$$\sum_{k=1}^{n}\left(Y_k - m_Y\right)^2$$

(общая сумма квадратов) – мера общего рассеивания Y_k относительно линии математического ожидания m_Y.

Смысл R^2: какая доля зависимого показателя не является случайной, т.е. описывается моделью.

Для линейных моделей $0 \le R^2 \le 1$. Чем ближе коэффициент детерминации к единице, тем точнее модель. При $R^2 = 1$ модель проходит точно через исходные данные.

Для нелинейных моделей коэффициент детерминации может быть отрицательным $R^2 \in (-\infty;1]$, если модель совершенно не объясняет показатель (даже хуже, чем просто горизонтальная прямая). Причиной может быть, например, вычислительная ошибка, неверный выбор функционального вида модели.

Для линейных моделей $R^2 = r_{XY}{}^2$.

Основным недостатком R^2 является то, что при усложнении модели он возрастает и поэтому не может служить достоверным критерием выбора одной модели из нескольких.

Критерии Фишера (F-тест)

F-критерий Фишера заключается в проверке гипотезы H_o *о статистической незначимости уравнения регрессии.* Для этого выполняется сравнение фактического $F_{факт}$ и критического (табличного) $F_{табл}$ значений *F*-критерия Фишера. $F_{факт}$ определяется из соотношения значений факторной и остаточной дисперсий, рассчитанных на одну степень свободы. Выполняется сравнение $F_{факт}$ и критического (табличного) $F_{табл}$ значений *F*-критерия Фишера. Если табличное значение меньше фактического, то признается статистическая значимость и надежность характеристик, если наоборот, то признается статистическая незначимость, ненадежность уравнения регрессии.

3.2 Пример построения регрессионной модели

Исследуем зависимость посещаемости web-сайта, предоставляющего информацию для поиска работы, от различных факторов. Моделируемым показателем в этом исследовании является количество человек, посетивших сайт за день.

Выбранные для решения данной задачи данные представлены в таблице 3.1.

Табл. 3.1. Данные для построения модели

Кол-во человек в день (Y)	**Загруженность внутренней сети (чел/ден) (P)**	**Кол-во вакансий на текущий день (V)**
11	651	165
18	1046	400
19	944	312
11	1084	341
15	1260	496
10	1212	264
12	254	78
14	1795	599
9	2851	622
15	1156	461

Найти коэффициенты регрессии можно используя один из пакетов статистического анализ библиотеки языков программирования. Искомое уравнение множественной регрессии: $Y = 0.018 \cdot V + 12.567 – 0.005 \cdot P$.

Коэффициент детерминации $R = 0.69$. Это означает, что факторы, вошедшие в модель объясняют изменение количества посетивших сайт людей на 69%. Следовательно, значения, полученные с помощью линейной модели, близки к фактическим.

Можно сделать вывод о том, что увеличение количества вакансий связано с, количеством посещений сайта.

3.3 Аффинитивный анализ

Цель аффинитивного анализа (affinity analysis) – исследование исследования взаимной связи (ассоциаций) между событиями происходящими совместно и их количественная оценка. Результат выполенния аффинитивного анализа – набор ассоциативных правил. (association rules).

Примерами приложения ассоциативных правил могут быть следующие задачи:

– выявление наборов товаров, которые в супермаркетах часто покупаются вместе или никогда не покупаются вместе;

– определение доли клиентов, положительно относящихся к нововведениям в их обслуживании;

– определение профиля посетителей веб-ресурса;

– определение доли случаев, в которых новое лекарство показывает опасный побочный эффект;

– выявление связи между параметрами оборудования и получаемыми качественными характеристиками продукта.

3.3.1 Основные понятия аффинитивного анализа

Базовым понятием аффинитивного анализа является *транзакция* – некоторое множество событий, происходящих совместно.

В результате аффинитивного анализа мы устанавливаем закономерность следующего вида: "Если в транзакции встретился набор элементов A, то можно сделать вывод, что в этой же транзакции должен появиться набор элементов B". Установление таких закономерностей дает нам возможность находить ассоциативные правила.

Ассоциативное правило имеет вид: "Из события A следует событие B": A→B. Основными характеристиками ассоциативного правила являются поддержка и достоверность правила.

Поддержка ассоциативного правила – это число транзакций, которые содержат как условие, так и следствие. Например, для ассоциации A→B можно записать S(A→B)=P(A∩B)= (количество транзакций, содержащих A и B)/(общее число транзакций) Правило имеет поддержку s, если s% транзакций из всего набора содержат одновременно наборы элементов A и B или, другими словами, содержат оба товара.

Достоверность правила показывает, какова вероятность того, что из события A следует событие B. Достоверность ассоциативного правила A→B представляет собой меру точности правила и определяется как отношение количества транзакций, содержащих и условие и следствие, к количеству транзакций, содержащих только условие: C(A→B)=P(A|B)=P(A∩B)/P(A) (количество транзакций, содержащих A и B)/(количество транзакций, содержащих только A). Правило "Из A следует B" справедливо с достоверностью c, если c% транзакций из всего множества, содержащих набор элементов A, также содержат набор элементов B.

Границы поддержки и достоверности ассоциативного правила

При помощи использования алгоритмов поиска ассоциативных правил аналитик может получить все возможные правила вида "Из A следует B", с различными значениями поддержки и достоверности. Однако в большинстве случаев, количество правил необходимо ограничивать заранее установленными минимальными и максимальными значениями поддержки и достоверности. Если значение поддержки правила слишком велико, то в результате работы алгоритма будут найдены правила очевидные и хорошо известные. Слишком низкое значение поддержки приведет к нахождению очень большого количества правил, которые, возможно, будут в большей части необоснованными, но не известными и не очевидными для аналитика. Таким образом, необходимо определить такой интервал, который с одной стороны обеспечит нахождение неочевидных правил, а с другой - их

обоснованность. Если уровень достоверности слишком мал, то ценность правила вызывает серьезные сомнения. Например, правило с достоверностью в 3% только условно можно назвать правилом. Аналитики могут отдавать предпочтение правилам, которые имеют только высокую поддержку или только высокую достоверность, или оба этих показателя. Правила, для которых значения поддержки или достоверности превышают определенный, заданный пользователем порог, называются сильными правилами.

Методики поиска ассоциативных правил обнаруживают все ассоциации, которые удовлетворяют ограничениям на поддержку и достоверность, наложенным пользователем. Это приводит к необходимости рассматривать десятки и сотни тысяч ассоциаций, что делает невозможным обработку такого количества данных вручную. Число правил желательно уменьшить таким образом, чтобы проанализировать только наиболее значимые из них. Значимость часто вычисляется как разность между поддержкой правила и в целом и произведением поддержки только условия и поддержки только следствия. Если условие и следствие независимы, то поддержка правила примерно соответствует произведению поддержек условия и следствия, то есть $SAB \approx SASB$. Это значит, что хотя условие и следствие часто встречаются вместе, не менее часто они встречаются и по отдельности.

Субъективные меры значимости ассоциативных правил

Лифт вычисляется следующим образом: $L(A \rightarrow B)=C(A \rightarrow B)/S(B)$. Лифт – это отношение частоты появления условия в транзакциях, которые также содержат и следствие, к частоте появления следствия в целом. Значения лифта, большие, чем 1, показывают, что условие чаще появляется в транзакциях, содержащих следствие, чем в остальных. Можно сказать, что лифт является обобщенной мерой связи двух предметных наборов: при значениях лифта больше 1 связь положительная, при 1 она отсутствует, при значениях меньше 1 – отрицательная.

Левередж вычисляется следующим образом: T(A→B)=S(A→B)-S(A)S(B). Левередж – это разность между наблюдаемой частотой, с которой условие и следствие появляются совместно (т.е. с поддержкой ассоциации), и произведением частот появления (поддержек) условия и следствия по отдельности. Из ассоциаций с одинаковым лифтом ассоциация с большим левереджем представляет больший интерес, так как это говорит о том, что данное правило встречаются чаще. Улучшение вычисляется следующим образом: I(A→B)=S(A→B)/(S(A)S(B)). Улучшение показывает, полезнее ли правило случайного угадывания. Если I(A→B)>1, это значит, что вероятнее предсказать наличие набора B с помощью правила, чем угадать случайно.

3.3.2 Методы поиска ассоциативных правил

В процессе поиска ассоциативных правил может производится обнаружение всех ассоциаций, поддержка и достоверность для которых превышают заданный минимум. Простейший алгоритм поиска ассоциативных правил рассматривает все возможные комбинации условий и следствий, оценивает для них поддержку и достоверность, а затем исключает все ассоциации, которые не удовлетворяют заданным ограничениям. Число возможных ассоциаций с увеличением числа предметов растет экспоненциально. Поэтому в процессе генерации ассоциативных правил широко используются методики, позволяющие уменьшить количество ассоциаций, которое требуется проанализировать.

Впервые задача поиска ассоциативных правил была предложена для нахождения типичных шаблонов покупок, совершаемых в супермаркетах, поэтому иногда ее еще называют анализом рыночной корзины (market basket analysis) [20].

Алгоритм AIS

Первый алгоритм поиска ассоциативных правил, называвшийся AIS, был разработан сотрудниками исследовательского центра IBM Almaden в 1993 году. С этой работы начался интерес к ассоциативным правилам; на середину 90-х годов прошлого века пришелся пик исследовательских работ в этой области, и с тех пор каждый год появляется несколько новых алгоритмов.

В алгоритме AIS кандидаты множества наборов генерируются и подсчитываются «на лету», во время сканирования базы данных.

Алгоритм SETM

Создание этого алгоритма было мотивировано желанием использовать язык SQL для вычисления часто встречающихся наборов товаров. Как и алгоритм AIS, SETM также формирует кандидатов «на лету», основываясь на преобразованиях базы данных. Чтобы использовать стандартную операцию объединения языка SQL для формирования кандидата, SETM отделяет формирование кандидата от их подсчета. Неудобство алгоритмов AIS и SETM – излишнее генерирование и подсчет слишком многих кандидатов, которые в результате не оказываются часто встречающимися. Для улучшения их работы был предложен алгоритм Apriori.

Алгоритм Apriori

Работа данного алгоритма состоит из нескольких этапов, каждый из этапов состоит из следующих шагов:

– формирование кандидатов;

– подсчет кандидатов.

Формирование кандидатов – этап, на котором алгоритм, сканируя базу данных, создает множество *i*-элементных кандидатов (*i* – номер этапа). На этом этапе поддержка кандидатов не рассчитывается.

Подсчет кандидатов – этап, на котором вычисляется поддержка каждого *i*-элементного кандидата. Здесь же осуществляется отсечение кандидатов, поддержка которых меньше минимума, установленного пользователем (min_sup). Оставшиеся *i*-элементные наборы называем часто встречающимися [21].

Однако алгоритм Apriori уменьшает количество кандидатов, отсекая – априори – тех, которые заведомо не могут стать часто встречающимися, на основе информации об отсеченных кандидатах на предыдущих этапах работы алгоритма.

Алгоритм Apriori рассчитывает также поддержку наборов, которые не могут быть отсечены априори. Это так называемая негативная область, к ней принадлежат наборы-кандидаты, которые встречаются редко, их самих нельзя отнести к часто встречающимся, но все подмножества данных наборов являются часто встречающимися.

Последовательные шаблоны

Для расширения возможностей анализа транзакционных данных с учетом временного аспекта, последовательности появления предметов и ориентированности на конкретного клиента существует задача Data Mining под названием *последовательные шаблоны* (sequential pattern, time-serial sequential pattern) [22].

Теория последовательных шаблонов во многом основана на теории ассоциативных правил и, по сути, является ее расширением. В частности, базовыми понятиями в ней также являются транзакция, предметный набор, частота набора, поддержка и т.д. Кроме этого, для поиска

последовательных шаблонов широко используется адаптированный алгоритм Apriori и его модификации. Но при рассмотрении последовательных шаблонов необходимо учитывать ряд особенностей. Главная из них заключается в том, что если в ассоциативных правилах рассматривается только факт совместного появления товаров в одной транзакции, то в последовательных шаблонах рассматривается последовательность появления товаров. Последовательный шаблон – это всегда последовательность появления предметов и их групп.

Интуитивно понятно, что типичной последовательностью (шаблоном) может быть только такая последовательность, которая встречается в базе данных достаточно часто. Поэтому при поиске последовательных шаблонов возникает та же проблема, что и при поиске ассоциативных правил. Большое число рассматриваемых предметов порождает огромное количество возможных последовательностей, что приводит к серьезным вычислительным затратам при использовании полного перебора. Но и здесь применим принцип антимонотонности – последовательности, содержащие редкие события, не могут быть частыми, что позволяет существенно снизить пространство поиска.

Применение последовательных шаблонов выходит за рамки анализа рыночной корзины. Они позволяют выявлять типичные последовательности событий в самых разнообразных предметных областях.

Алгоритм AprioriAll

Является модификацией алгоритма Apriori. На каждом проходе используются частые последовательности, полученные на предыдущем проходе, для генерации последовательностей-кандидатов, а затем вычисляется их поддержка в процессе нового прохода. В дальнейшем она

используется для определения частых последовательностей. Частые предметные наборы, найденные на первом проходе, являются, по сути, частыми 1-последовательностями. Иногда этот процесс называют инициализацией. На каждом проходе базы данных алгоритм AprioriAll обрабатывает последовательности только определенной длины *k*. При этом анализ последовательностей всех длин может оказаться затратной по времени процедурой.

Алгоритм AprioriAll формирует частые последовательности-кандидаты всех возможных длин. Однако если из числа последовательностей определенной длины формируется мало частых последовательностей, то эту длину можно пропустить. Алгоритм использует как параметр длину последовательностей, анализируемых на предыдущем проходе, и возвращает длину последовательностей, которые будут анализироваться на следующем. Иными словами, длина последовательностей, искомых на следующем проходе, определяется длиной последовательностей, найденных на предыдущем [22].

Вопросы к главе 3

1. Дайте определение понятию стохастическая зависимость.
2. Какие задачи решает множественная регрессия?
3. Что такое уравнение регрессии?
4. Опишите идею метода наименьших квадратов.
5. Перечислите известные Вам показатели качества уравнения регрессии.
6. Как используют критерий Фишера для оценки качества моделей регрессии?
7. Какой вид имеют ассоциативные правила?
8. Объясните суть понятие поддержка правила.

9. Дайте определение понятию достоверность правила.
10. Основные шаги алгоритма Apriori.

ГЛАВА 4 МЕТОДЫ МОДЕЛИРОВАНИЯ НА ОСНОВЕ ТЕОРИИ НЕЧЕТКИХ МНОЖЕСТВ И МЯГКИХ ВЫЧИСЛЕНИЙ

Математическая теория нечетких множеств (fuzzy sets) и нечеткая логика (fuzzy logic) являются обобщениями классической теории множеств и классической формальной логики. Данные понятия были впервые предложены американским ученым Лотфи Заде (Lotfi Zadeh) в 1965 г. [23].

Концепция нечеткого множества зародилась у Заде “как неудовлетворенность математическими методами классической теории систем, которая вынуждала добиваться искусственной точности, неуместной во многих системах реального мира, особенно в так называемых гуманистических системах, включающих людей” [24].

Началом практического применения теории нечетких множеств можно считать 1975 г., когда Мамдани и Ассилиан (Mamdani and Assilian) построили первый нечеткий контролер для управления простым паровым двигателем. В 1982 Холмблад и Остергад (Holmblad and Osregaad) разработали первый промышленный нечеткий контроллер, который был внедрен в управление процессом обжига цемента на заводе в Дании. Успех первого промышленного контролера, основанного на нечетких лингвистических правилах “Если – то” привел к всплеску интереса к теории нечетких множеств среди математиков и инженеров. Несколько позже Бартоломеем Коско (Bart Kosko) была доказана теорема о нечеткой аппроксимации (Fuzzy Approximation Theorem), согласно которой любая математическая система может быть аппроксимирована системой, основанной на нечеткой логике. Другими словами, с помощью естественно-языковых высказываний-правил “Если – то”, с последующей их формализацией средствами теории нечетких множеств, можно сколько угодно точно отразить произвольную взаимосвязь “входы-выход” без использования сложного аппара-

та дифференциального и интегрального исчислений, традиционно применяемого в управлении и идентификации [25].

Использование нечетких множеств дает ряд преимуществ для моделирования, т.к. позволяет:

– включать в анализ качественные переменные;

– оперировать нечеткими входными данными;

– оперировать лингвистическими критериями;

– быстро моделировать сложные динамические системы и сравнивать их с заданной степенью точности;

Методы на основе теории нечетких множеств используются для решения большого круга задач:

– анализ новых рынков;

– биржевые игры;

– оценка политических рейтингов;

– выбор оптимальной ценовой стратегии и т.п.;

– новые архитектуры компьютеров для нечетких вычислений;

– элементная база нечетких компьютеров и контроллеров;

– инструментальные средства разработки;

– инженерные методы расчета и разработки нечетких систем управления, и т.п.

4.1 Основные понятия теории нечетких множеств

Характеристикой нечеткого множества выступает *функция принадлежности (Membership Function).* Обозначим через $MF_c(x)$ – степень принадлежности к нечеткому множеству C, представляющей собой обобщение понятия характеристической функции обычного множества. Тогда *нечетким множеством* C называется множество упорядоченных пар вида

$C=\{MF_c(x)/x\}$, $MF_c(x)$ [0,1]. Значение $MF_c(x)=0$ означает отсутствие принадлежности к множеству, 1 – полную принадлежность.

Пример [26]. Формализуем неточное определение "горячий чай". В качестве x (область рассуждений) будет выступать шкала температуры в градусах Цельсия. Очевидно, что она будет изменяется от 0 до 100 градусов. Нечеткое множество для понятия "горячий чай" может выглядеть следующим образом:

C = {0/0; 0/10; 0/20; 0,15/30; 0,30/40; 0,60/50; 0,80/60; 0,90/70; 1/80; 1/90; 1/100}.

Так, чай с температурой 60 ºС принадлежит к множеству "Горячий" со степенью принадлежности 0,80. Для одного человека чай при температуре 60 ºС может оказаться горячим, для другого – не слишком горячим.

Для нечетких множеств, как и для обычных, определены основные логические операции. Самыми основными, необходимыми для расчетов, являются пересечение и объединение.

Пересечение двух нечетких множеств (нечеткое "И"): A B: $MF_{AB}(x)=\min(MF_A(x), MF_B(x))$.

Объединение двух нечетких множеств (нечеткое "ИЛИ"): A B: $MF_{AB}(x)=\max(MF_A(x), MF_B(x))$.

В теории нечетких множеств разработан общий подход к выполнению операторов пересечения, объединения и дополнения, реализованный в так называемых треугольных нормах и конормах. Приведенные выше реализации операций пересечения и объединения – наиболее распространенные случаи *t*-нормы и *t*-конормы [26].

Для описания нечетких множеств вводятся понятия нечеткой и лингвистической переменных.

Нечеткая переменная описывается набором (*N*,*X*,*A*), где *N* – это название переменной, *X* – универсальное множество (область рассуждений), *A* – нечеткое множество на *X*.

Лингвистическая переменная

Лингвистической называется переменная, принимающая значения из множества слов или словосочетаний некоторого естественного или искусственного языка. Множество допустимых значений лингвистической переменной называется терм-множеством. Задание значения переменной словами, без использования чисел, для человека более естественно. Ежедневно мы принимаем решения на основе лингвистической информации типа: "очень высокая температура"; "длительная поездка"; "быстрый ответ"; "красивый букет"; "гармоничный вкус" и т.п. Психологи установили, что в человеческом мозге почти вся числовая информация вербально перекодируется и хранится в виде лингвистических термов [25].

Значениями лингвистической переменной могут быть нечеткие переменные, т.е. лингвистическая переменная находится на более высоком уровне, чем нечеткая переменная.

Каждая *лингвистическая переменная* состоит из:

– названия;

– базового терм-множества *T*. Элементы базового терм-множества представляют собой нечеткие переменные;

– универсального множества *X*;

– синтаксического правила *G*, по которому генерируются новые термы с применением слов естественного или формального языка;

– семантического правила *P*, которое каждому значению лингвистической переменной ставит в соответствие нечеткое подмножество множества *X*.

Пример [26]. Рассмотрим такое нечеткое понятие как "Цена акции". Это и есть название лингвистической переменной. Сформируем для нее базовое терм-множество, которое будет состоять из трех нечетких переменных: "Низкая", "Умеренная", "Высокая" и зададим область рассуждений в виде X = [100;200] (единиц). Последнее, что осталось сделать – построить функции принадлежности для каждого лингвистического терма из базового терм-множества T.

Совокупность функций принадлежности для каждого терма из базового терм-множества T обычно изображаются вместе на одном графике. Количество термов в лингвистической переменной редко превышает 7. На рисунке 4.1 приведен пример описанной выше лингвистической переменной "Цена акции".

Рис. 4.1. Описание лингвистической переменной "Цена акции"

Нечеткая истинность

В классической логике истинность может принимать только два значения: истинно и ложно. В нечеткой логике истинность "размытая". Для задания нечеткой истинности Заде предложил такие функции принадлежности термов "истинно" и "ложно":

$$\mu_{\text{"истинно"}}(u) = \begin{cases} 0, & 0 \le u \le a \\ 2 \cdot \left(\frac{u-a}{1-a}\right)^2, & a < u \le \frac{a+1}{2} \\ 1 - 2 \cdot \left(\frac{u-1}{1-a}\right)^2, & \frac{a+1}{2} < u \le 1 \end{cases};$$

$$\mu_{\text{"ложно"}}(u) = \mu_{\text{"истинно"}}(1-u), \quad u \in [0,1],$$

где $a \in [0,1]$ – параметр, определяющий носители нечетких множеств "истинно" и "ложно". Для нечеткого множества "истинно" носителем будет интервал $(a,1]$, а для нечеткого множества "ложно" – $[0,a)$.

Функции принадлежности нечетких термов "истинно" и "ложно" изображены на рисунке 4.2 [25]. Они построены при значении параметра a = 0.4.

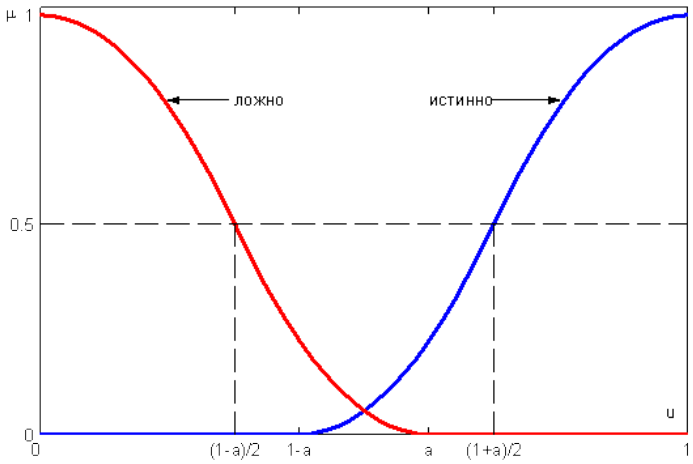


Рис. 4.2. Лингвистическая переменная "истинность" по Заде [25]

4.2 Нечеткий логический вывод

Нечетким логическим выводом называется получение заключения в виде нечеткого множества, соответствующего текущим значениях входов, с использованием нечеткой базы знаний и нечетких операций.

Основу нечеткого логического вывода составляет композиционное правило Заде [27]: если известно нечеткое отношение $\tilde{R}$ между входной (x)

и выходной (*y*) переменными, то при нечетком значении входной переменной $x = \tilde{A}$, нечеткое значения выходной переменной определяется так:

$$y = \tilde{A} \circ \tilde{R},$$

где $\circ$ – максминая композиция.

Пример [26]. Дано нечеткое правило "Если $x = \tilde{A}$, то $y = \tilde{B}$" с нечеткими множествами: $\tilde{A} = 0/1 + 0.1/2 + 0.5/3 + 0.8/4 + 1/5$ и $\tilde{B} = 1/5 + 0.8/10 + 0.4/15 + 0.2/20$. Определить значение выходной переменной y, если $x = \tilde{C} = 0.3/1 + 0.5/2 + 1/3 + 0.7/4 + 0.4/5$.

Вначале рассчитаем нечеткое отношение, соответствующее правилу "Если $x = \tilde{A}$, то $y = \tilde{B}$", применяя в качестве *t*-нормы операцию нахождения минимума:

$$\tilde{R} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0.1 & 0.1 & 0.1 & 0.1 \\ 0.5 & 0.5 & 0.4 & 0.2 \\ 0.8 & 0.8 & 0.4 & 0.2 \\ 1 & 0.8 & 0.4 & 0.2 \end{bmatrix}.$$

Теперь, по формуле $y = \tilde{C} \circ \tilde{R}$ рассчитаем нечеткое значение выходной переменной:

$$y = 0.7/5 + 0.7/10 + 0.4/15 + 0.2/20.$$

Основой для проведения операции нечеткого логического вывода является база правил, содержащая нечеткие высказывания в форме "Если – то" и функции принадлежности для соответствующих лингвистических термов. При этом должны соблюдаться следующие условия [26]:

– существует хотя бы одно правило для каждого лингвистического терма выходной переменной;

– для любого терма входной переменной имеется хотя бы одно правило, в котором этот терм используется в качестве предпосылки (левая часть правила).

В противном случае имеет место неполная база нечетких правил.

Пусть в базе правил имеется m правил вида:

$$R_1: \text{ЕСЛИ } x_1 \text{ это } A_{11} \ldots \text{ И } \ldots x_n \text{ это } A_{1n}, \text{ ТО } y \text{ это } B_1 \ldots$$

$$R_i: \text{ЕСЛИ } x_1 \text{ это } A_{i1} \ldots \text{ И } \ldots x_n \text{ это } A_{in}, \text{ ТО } y \text{ это } B_i \ldots$$

$$R_m: \text{ЕСЛИ } x_1 \text{ это } A_{i1} \ldots \text{ И } \ldots x_n \text{ это } A_{mn}, \text{ ТО } y \text{ это } B_m,$$

где x_k, $k = 1..n$ – входные переменные; y – выходная переменная; A_{ik} – заданные нечеткие множества с функциями принадлежности.

Результатом нечеткого вывода является четкое значение переменной y^* на основе заданных четких значений x_k, $k = 1..n$.

В общем случае механизм логического вывода включает следующие этапы: введение нечеткости (фазификация), нечеткий вывод, композиция и приведение к четкости, или дефазификация (см. рис. 4.3).

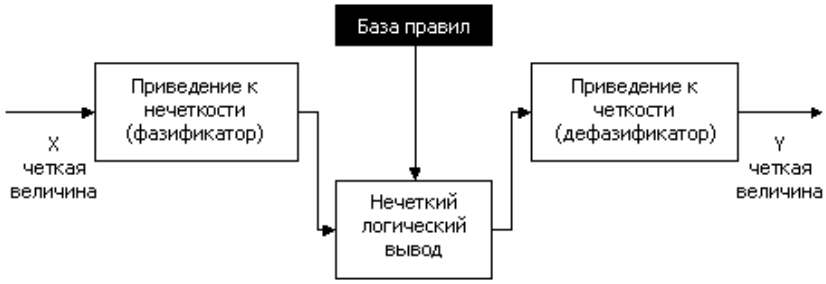


Рис. 4.3. Система нечеткого логического вывода [26]

Алгоритмы нечеткого вывода различаются главным образом видом используемых правил, логических операций и разновидностью метода дефазификации. Разработаны модели нечеткого вывода Мамдани, Сугено, Ларсена, Цукамото.

Рассмотрим подробнее нечеткий вывод на примере механизма Мамдани (Mamdani). Это наиболее распространенный способ логического вывода в нечетких системах. В нем используется минимаксная композиция нечетких множеств. Данный механизм включает в себя следующую последовательность действий.

Геометрический смысл такого значения – центр тяжести для кривой *MF*(*y*). Рисунок 4.4 графически показывает процесс нечеткого вывода по Мамдани для двух входных переменных и двух нечетких правил R1 и R2.

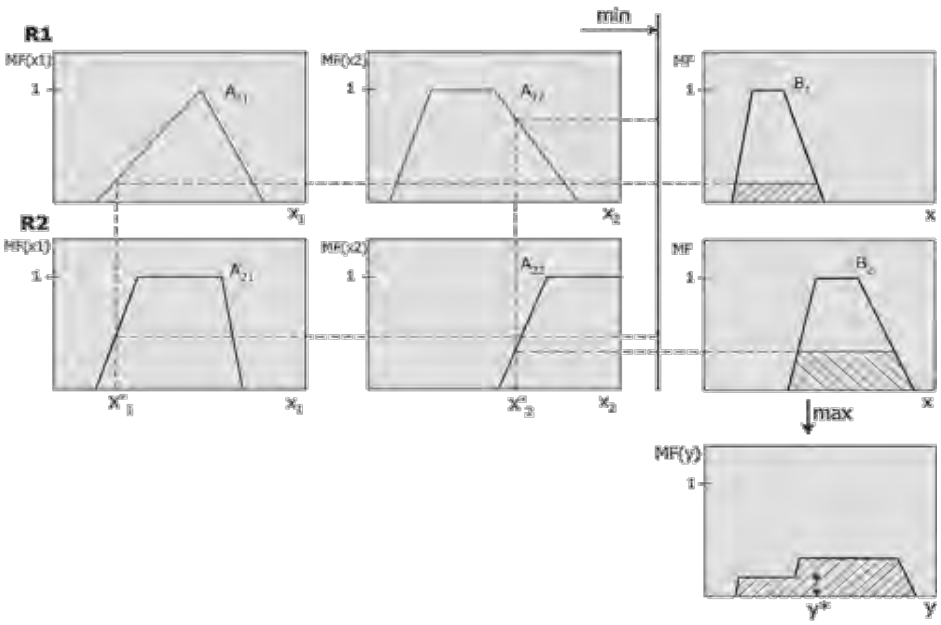


Рис. 4.4. Схема нечеткого вывода по Мамдани [26]

1. Процедура фазификации: определяются степени истинности, т.е. значения функций принадлежности для левых частей каждого правила (предпосылок). Для базы правил с m правилами обозначим степени истинности как $A_{ik}(x_k)$, i=1..m, k=1..n.

2. Нечеткий вывод. Сначала определяются уровни "отсечения" для левой части каждого из правил:

$$alfa_i = \min_i (A_{ik}(x_k))$$

Далее находятся "усеченные" функции принадлежности:

$$B_i^*(y) = \min_i (alfa_i, B_i(y))$$

3. Композиция, или объединение полученных усеченных функций, для чего используется максимальная композиция нечетких множеств:

$$MF(y) = \max_i (B_i^*(y))$$

где $MF(y)$ – функция принадлежности итогового нечеткого множества.

4. Дефазификация, или приведение к четкости. Существует несколько методов дефазификации. Например, метод среднего центра, или центроидный метод:

$$MF(y) = \max_i \left(B_i^*(y)\right).$$

4.3 Мягкие вычисления

В результате объединения нескольких технологий искусственного интеллекта появился специальный термин – "мягкие вычисления" (soft computing), который ввел Л. Заде в 1994 году [28].

Основной принцип мягких вычислений – это учет неточности, неопределенности, частичной истины и аппроксимации для достижения большего соответствия с реальностью, устойчивости решения, снижения затрат на его получение.

В настоящее время мягкие вычисления объединяют такие области как: нечеткая логика, искусственные нейронные сети, вероятностные рассуждения и эволюционные алгоритмы. Они дополняют друг друга и используются в различных комбинациях для создания гибридных интеллектуальных систем.

Ниже приводятся некоторые примеры методов, которые можно отнести к «мягким вычислениям» [26].

Нечеткие нейронные сети

Нечеткие нейронные сети (fuzzy-neural networks) осуществляют выводы на основе аппарата нечеткой логики, однако параметры функций принадлежности настраиваются с использованием алгоритмов обучения НС. Поэтому для подбора параметров таких сетей применим метод обратного распространения ошибки, изначально предложенный для обучения много-

слойного персептрона. Для этого модуль нечеткого управления представляется в форме многослойной сети. Нечеткая нейронная сеть как правило состоит из четырех слоев: слоя фазификации входных переменных, слоя агрегирования значений активации условия, слоя агрегирования нечетких правил и выходного слоя.

Наибольшее распространение в настоящее время получили архитектуры нечеткой НС вида ANFIS и TSK. Доказано, что такие сети являются универсальными аппроксиматорами.

Быстрые алгоритмы обучения и интерпретируемость накопленных знаний – эти факторы сделали сегодня нечеткие нейронные сети одним из самых перспективных и эффективных инструментов мягких вычислений.

Адаптивные нечеткие системы

Классические нечеткие системы обладают тем недостатком, что для формулирования правил и функций принадлежности необходимо привлекать экспертов той или иной предметной области, что не всегда удается обеспечить. Адаптивные нечеткие системы (adaptive fuzzy systems) решают эту проблему. В таких системах подбор параметров нечеткой системы производится в процессе обучения на экспериментальных данных. Алгоритмы обучения адаптивных нечетких систем относительно трудоемки и сложны по сравнению с алгоритмами обучения нейронных сетей, и, как правило, состоят из двух стадий:

1. Генерация лингвистических правил.
2. Корректировка функций принадлежности.

Первая задача относится к задаче переборного типа, вторая – к оптимизации в непрерывных пространствах. При этом возникает определенное противоречие: для генерации нечетких правил необходимы функции принадлежности, а для проведения нечеткого вывода – правила. Кроме того,

при автоматической генерации нечетких правил необходимо обеспечить их полноту и непротиворечивость.

Значительная часть методов обучения нечетких систем использует генетические алгоритмы. В англоязычной литературе этому соответствует специальный термин – Genetic Fuzzy Systems.

Значительный вклад в развитие теории и практики нечетких систем с эволюционной адаптацией внесла группа испанских исследователей во главе с Ф. Херрера (F. Herrera).

Нечеткие запросы

Нечеткие запросы к базам данных (fuzzy queries) – перспективное направление в современных системах обработки информации. Данный инструмент дает возможность формулировать запросы на естественном языке, например: "Вывести список недорогих предложений о съеме жилья близко к центру города", что невозможно при использовании стандартного механизма запросов. Для этой цели разработана нечеткая реляционная алгебра и специальные расширения языков SQL для нечетких запросов.

Нечеткие ассоциативные правила

Нечеткие ассоциативные правила (fuzzy associative rules) – инструмент для извлечения из баз данных закономерностей, которые формулируются в виде лингвистических высказываний. Здесь введены специальные понятия нечеткой транзакции, поддержки и достоверности нечеткого ассоциативного правила.

Нечеткие когнитивные карты

Нечеткие когнитивные карты (fuzzy cognitive maps) были предложены Б. Коско в 1986 г. и используются для моделирования причинных взаимо-

связей, выявленных между концептами некоторой области. В отличие от простых когнитивных карт, нечеткие когнитивные карты представляют собой нечеткий ориентированный граф, узлы которого являются нечеткими множествами. Направленные ребра графа не только отражают причинно-следственные связи между концептами, но и определяют степень влияния (вес) связываемых концептов. Активное использование нечетких когнитивных карт в качестве средства моделирования систем обусловлено возможностью наглядного представления анализируемой системы и легкостью интерпретации причинно-следственных связей между концептами. Основные проблемы связаны с процессом построения когнитивной карты, который не поддается формализации. Кроме того, необходимо доказать, что построенная когнитивная карта адекватна реальной моделируемой системе. Для решения данных проблем разработаны алгоритмы автоматического построения когнитивных карт на основе выборки данных.

Нечеткая кластеризация

Нечеткие методы кластеризации, в отличие от четких методов (например, нейронные сети Кохонена), позволяют одному и тому же объекту принадлежать одновременно нескольким кластерам, но с различной степенью. Нечеткая кластеризация во многих ситуациях более "естественна", чем четкая, например, для объектов, расположенных на границе кластеров. Наиболее распространены: алгоритм нечеткой самоорганизации c-means и его обобщение в виде алгоритма Густафсона-Кесселя.

Вопросы к главе 4

1. Дайте определение понятию «нечеткое множество».
2. Какие операции применимы к нечетким множествам?
3. Дайте определение понятию «лингвистическая переменная».

4. Приведите пример лингвистической переменной.

5. Приведите пример нечеткого правила.

6. Что понимается под фаззификацией?

7. Приведите пример фаззификации входной ЛП «скорость ветра».

8. Приведите пример агрегирования для нечетких высказываний: «температура воздуха средняя» и «скорость ветра небольшая», задав предварительно функции принадлежности.

9. Что понимается под дефаззификацией?

10. Какие методы можно отнести к технологии «мягких вычислений»? В чем их особенность?

ГЛАВА 5 МЕТОДЫ КОНЦЕПТУАЛЬНОГО МОДЕЛИРОВАНИЯ

5.1 Основные понятия концептуального моделирования

Концептуальное моделирование – это вид моделирования, при котором с помощью некоторых специальных знаков, символов, операций над ними или с помощью естественного или искусственного языков истолковывается основная мысль (концепция) относительно исследуемого объекта.

Концептуальная модель – модель предметной области, состоящей из перечня взаимосвязанных понятий, используемых для описания этой области, вместе со свойствами и характеристиками, классификацией этих понятий, по типам, ситуациям, признакам в данной области и законов протекания процессов в ней [29].

Основная задача концептуальной модели – облегчение восприятия информации обычным пользователем. Чтобы добиться данного результата, необходимо в первую очередь сделать эту модель наиболее простой. А во-вторых, постараться максимально ориентировать ее на выполнение определенных задач [30].

Создание концептуальной модели преследует следующие цели:

1) создать простую, последовательную и удобную в использовании и изучении структуру. С этой целью области задач разделяются на понятия, которые можно использовать для работы с разными объектами;

2) сохранить устойчивость терминологии. Это достигается тем, что концептуальная модель данных, состоящая на начальном этапе из словаря терминов, используется для распознавания каждого действия и объекта, расписанного в программе.

Выделяют три вида концептуальных моделей [30]:

– *логико-семантические*. Описание объекта в терминах соответствующих предметных областей знаний. Анализ таких моделей осуществляется средствами логики с привлечением специальных знаний;

– *структурно-функциональные*. Объект рассматривается как целостная система, которую расчленяют на отдельные подсистемы или элементы. Части системы связывают структурными отношениями, описывающими подчиненность, логическую и временную последовательность решения задач;

– *причинно-следственные*. Служит для объяснения и прогнозирования поведения объекта. Такие модели ориентированы на решение следующих задач: 1) выявление главных взаимосвязей между подсистемами; 2) выявление определенного влияния различных факторов на состояние объекта; 3) описание динамики интересующих разработчика параметров.

Разработка концептуальной модели

Построение концептуальной модели включает следующие этапы [31]:

1) определение типа системы;
2) описание внешних воздействий;
3) декомпозиция системы.

На первом этапе осуществляется сбор фактических данных (на основе работы с литературой и технической документацией, проведения натурных экспериментов, сбора экспертной информации и т. д.), а также выдвижение гипотез относительно значений параметров и переменных, для которых отсутствует возможность получения фактических данных.

На втором этапе происходит создание модели внешних воздействий. Модель внешних воздействий (ВВ) должна обладать следующими основными свойствами:

– совместимостью с моделью системы;

– представительностью;

– управляемостью;

– системной независимостью.

Свойство совместимости предполагает, что, во-первых, степень детализации описания ВВ соответствует детализации описания системы; во-вторых, модель ВВ должна быть сформулирована в тех же категориях предметной области, что и модель системы (например, если в модели системы исследуется использование ресурсов, то должны быть выражена в запросах на ресурсы) [31].

Представительность модели ВВ определяется ее способностью адекватно представить ВВ в соответствии с целями исследования. Другими словами, модель ВВ должны отвечать целям исследования системы. Например, если оценивается пропускная способность, то должны выбираться ВВ, «насыщающие» систему. Под управляемостью понимается возможность изменения параметров модели ВВ в некотором диапазоне, определяемом целями исследования.

Системная независимость – это возможность переноса модели ВВ с одной системы на другую с сохранением ее представительности. Данное свойство наиболее важно при решении задач сравнения различных систем или различных модификаций одной системы. Если модель ВВ зависит от конфигурации исследуемой системы или других ее параметров, то использование такой модели для решения задачи выбора невозможно.

На последнем этапе производится декомпозиция системы исходя из выбранного уровня детализации модели, который, в свою очередь, определяется тремя факторами:

– целями моделирования;

– объемом априорной информации о системе;

– требованиями к точности и достоверности результатов моделирования.

Способы представления концептуальных моделей:

– различные варианты диаграмм сущность-связь (Entity Relationship Diagram – ER-диаграммы);

– диаграммы IDEF0;

– диаграммы в UML;

– концептуальные карты (графы);

– когнитивные карты и пр.

5.2 Концептуальные карты

Концептуальной картой или концептуальной схемой называют схему, которая изображает отношения между понятиями (концептами). Это графический инструмент, который используется дизайнерами, инженерами, техническими писателями и другими, чтобы организовать и структурировать знания. Концептуальные карты способны развить логическое мышление и навыки обучения по выявлению связей, а также помогают увидеть, как отдельные идеи образуют большую целую идею.

Впервые концептуальные карты были предложены Новаком в начале 70-х гг. при изучении детского мышления и формирования первых научных понятий. Это исследование использовало идеи Дэвида Асубеля о формировании понятийного мышления. Концептуальные карты оказались эффективным инструментом отображения понятийной системы человека [32].

Концептуальные карты используются для стимулирования генерации идей, увеличения креативности, а также для мозгового штурма. Формализованные концептуальные карты используются в разработке программного обеспечения (например, диаграммы UML). Кроме этого, концептуальные карты широко используются в сфере образования и бизнеса.

Еще одним преимуществом использования концептуальных карт в качестве средства структурирования знаний является системный подход к изучению предметной области [33]. При этом достигаются:

– системность – концептуальная карта представляет целостный взгляд на предметную область;

– единообразие – материал, представленный в единой форме, гораздо лучше воспринимается и воспроизводится;

– научность – построение концептуальной карты позволяет восстановить недостающие логические связи во всей их полноте.

Пример концептуальной карты, описывающей, что такое концептуальная карта, представлен на рисунке 5.1.

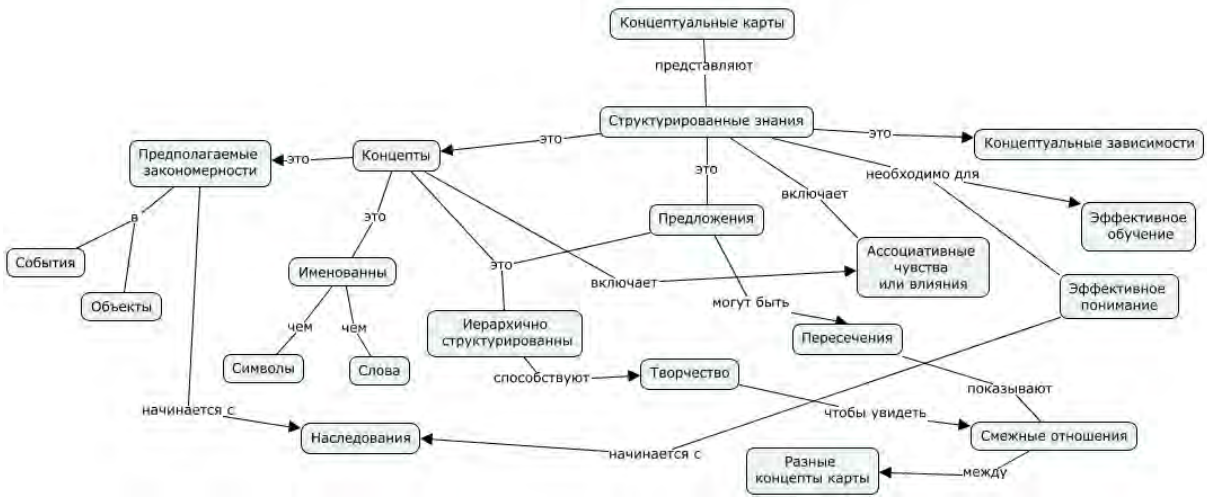


Рис. 5.1. Концептуальная модель понятия «Концептуальная карта» [34]

Виды концептуальных карт

Кластерные карты (рис. 5.2.). Эти карты являются полезным средством, при помощи которого можно генерировать и организовывать идеи в ходе мозгового штурма и прослеживать связи между ними. Работа над ними может быть использована как начальная точка в большом проекте или как задание, выполняемое перед началом исследовательской работы [34].

Рис. 5.2. Кластерная карта «Что такое дыхание весны» [34]

Карты причин (рис. 5.3) [34]. Эти карты помогают создавать графические образы причинно-следственных отношений. Анализ причин и следствий крайне важно для понимания сложных систем, таких как исторические события, романы, изменения в среде обитания животных.

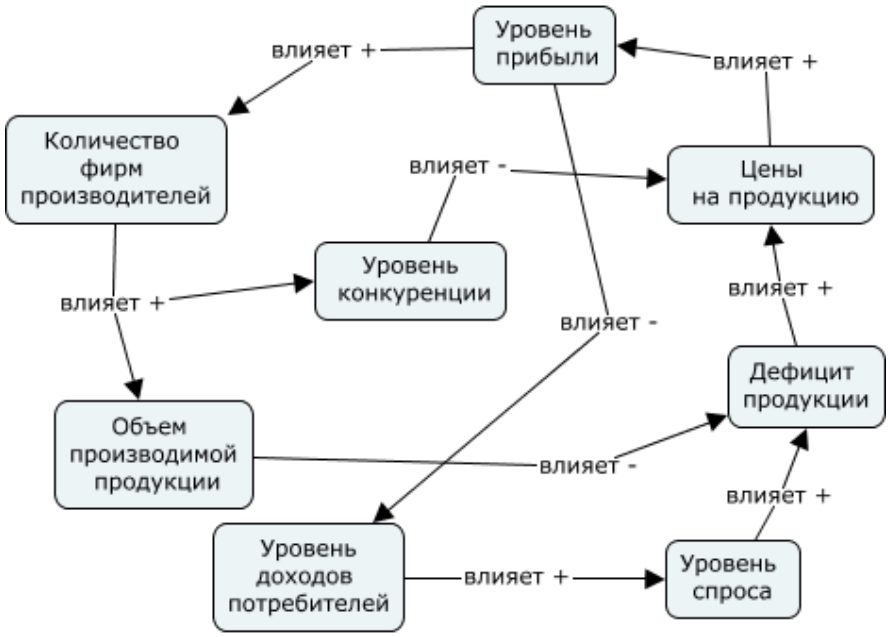


Рис. 5.3. Карта причин «Экономическая ситуация (Анализ продаж)»

Карта иерархий (рис. 5.4) [34].

Концептуальная карта отношений («*карта-паук*») (рис. 5.5). Часто используется, когда главная тема содержит очень много ответвлений и ко-

гда необходимо показать сопровождающие условия и понять первичность и вторичность свойств объекта [34].

Рис. 5.4. Карта иерархий "Организация служащих магазина"

Рис. 5.5. Карта отношений "Здоровье"

Карта блок-схемы (рис. 5.6) [34]. Позволяет посмотреть процесс и увидеть множество вариантов решения.

Рис. 5.6. Карта блок-схема "Включить люстру"

Этапы построения концептуальных карт

В простейшем случае построение концептуальной карты сводится к:

– определению контекста путем задания конкретного фокусирующего вопроса, определяющего главную тему и границы концептуальной карты;

– выделению концептов – базовых понятий данной предметной области (обычно не более 15–20 понятий);

– построению связей между концептами – определению соотношений и взаимодействий базовых понятий;

– упорядочению графа – уточнению, удалению лишних связей, снятию противоречий [33].

Применение концептуальных карт [34]:

– краткое изложение проблемы;

– выявление знаний;

– выявление концепций, взаимодействий иерархий из анализа источников;

– генерирование новых знаний;

– трансформация скрытых знаний в явные структурированные;

– сохранение данных в ассоциативных связях;

– моделирование совместных групповых знаний;

– обмен мнениями по прогнозам, тенденциям в рабочих группах;

– способ запоминания при изучении тех или иных явлений;

– глубокое освоение материала;

– коллективное развитие идей;

– мозговой штурм (брейнсторминг);

– анализ структуры комплексных идей;

– выстраивание цепочек аргументации, выявление нарушения аргументационной логики (ошибки, разрывы, пропуски и т.п.);

– изучение механизмов познания;

– обучение ораторскому искусству.

5.3 Когнитивное моделирование

5.3.1 Основные понятия когнитивного анализа

Методология когнитивного моделирования, предназначенная для анализа и принятия решений в плохо определенных ситуациях, была предложена Аксельродом [35]. В основе методологии когнитивного моделирования лежит когнитивная (познавательно-целевая) структуризация знаний об объекте и внешней для него среде, причем объект и внешняя среда разграничиваются «нечётко». Цель такой структуризации состоит в выявлении наиболее существенных (базисных) факторов, характеризующих «пограничный» слой взаимодействия объекта и внешней среды, и установлении качественных (причинно-следственных) связей.

Когнитивное моделирование – это исследование функционирования и развития слабоструктурированных систем и ситуаций посредством по-

строения модели сложной системы (ситуации) на основе когнитивной модели. Когнитивная модель отражает субъективные представления (индивидуальные или коллективные) исследуемой проблемы, связанной с функционированием и развитием систем. Основными элементами когнитивной модели являются базисные факторы и причинно-следственные связи между ними. *Базисные факторы* – это факторы, которые определяют и ограничивают наблюдаемые явления и процессы в системе и окружающей ее среде и интерпретированы субъектом управления как существенные, ключевые параметры, признаки этих явлений и процессов [35].

Качественные модели сложных и очень сложных систем достаточно эффективно строятся на основе математического аппарата знаковых и взвешенных графов, которые позволяют формализовать взаимодействие основных положительных и отрицательных обратных связей, существующих между процессами, определяющими функционирование и развитие сложной социально-политической, экономической или экологической системы. При построении таких моделей может быть использована неполная, нечеткая и даже противоречивая информация.

Цель когнитивной структуризации состоит в формировании и уточнении гипотезы о функционировании исследуемого объекта. Чтобы понять и проанализировать поведение сложной системы с помощью когнитивного подхода, строится структурная схема причинно-следственных связей (*когнитивная карта*).

Когнитивная карта – это субъективное представление эксперта о процессах в сложной динамической ситуации, формально представляемое в виде ориентированного знакового графа [36].

Впервые понятие «когнитивная карта» было введено в 1948 г. психологом Э. Толменом как образ пространственного окружения [37]. Когнитивная карта как образ внутренних представлений субъекта служит «инстру-

ментом для формирования и уточнения гипотезы о функционировании исследуемого объекта, рассматриваемого как сложная система» [38].

5.3.2 Типы когнитивных карт

Слабо формализованные когнитивные карты (СФКК) [39,40] используются, как правило, для формирования общего представления о ситуации и анализа (сравнения) точек зрения субъектов относительно некоторой ситуации, а иногда для моделирования понимания механизмов выработки решений. *Формальные когнитивные карты* (ФКК) используются для анализа и моделирования слабоструктурированных ситуаций, источником знаний о которых служат представления субъектов [38].

Формально, обязательной основой всех таких моделей является ориентированных граф, вершины которого соответствуют элементам (короткие фразы для СФКК и понятия, факторы или переменные для ФКК), а дуги интерпретируются как прямые причинные влияния между элементами. Обычно базис включает некоторые параметры такие, как знак влияния (как для ФКК, так и для СФКК) или сила влияния. На основе СФКК развиваются методы построения, анализа и сравнения карт на базе теории графов.

По особенности представляемых ситуаций ФКК могут быть динамическими и статическими. ФКК соответствует теоретическая (общая) модель, которая включает формальное описание карты и явную или неявную функцию агрегирования влияний на фактор. Различная интерпретация вершин, дуг и весов, также как различные функции агрегирования влияний на фактор, приводит к различным типам теоретических моделей ФКК (или типам ФКК) и формальным средствам их анализа.

Функции агрегирования влияний на фактор и параметры карт могут содержать время в явном виде, тогда будем говорить о картах с сильной

динамикой. В другом случае – будем говорить о когнитивных картах со слабой динамикой.

К когнитивным картам со слабой динамикой можно отнести нечеткие когнитивные карты Коско [41], логические когнитивные карты с реляционной алгеброй и др., вероятностные когнитивные карты Велмена и их модификации [42].

Например, когнитивным картам логического типа соответствует функция определения полного эффекта влияния переменной-причины на переменную-следствие и они воспроизводят *интуитивные* механизмы вывода, например, при моделировании многоагентных систем в задачах коллективного взаимодействия и прогнозирования поведения [43].

Когнитивные карты с сильной динамикой различаются по типу функций агрегирования влияний на фактор.

Рассмотрим *когнитивные карты* с сильной динамикой и базовой линейной импульсной моделью, предложенной на знаковых (взвешенных) графах (соответствующим картам) для решения задачи прогнозирования поведения сложной системы [44]. Идея этого подхода основана на анализе импульсных процессов на знаковых графах. Импульсный процесс может подвергаться воздействию внешних импульсов в любой момент времени и тогда функция вычисления значения факторов в момент времени t+1 имеет вид [44]:

$$v_i(t+1) = v_i(t) + p_i^0(t+1) + \sum_{j+1}^{n} sgn(u_j, u_i) p_j(t) \qquad (5.1)$$

где $p_j^0(t)$ внешний импульс или изменение в вершине u_j в момент t.

$$sgn(u_j, u_i) = \begin{cases} 1, & \text{если дуга } u_j, u_i \text{ положительная} \\ -1, & \text{если дуга } u_j, u_i \text{ отрицательная} \\ 0, & \text{если дуга } u_j, u_i \text{ отсутствует} \end{cases}$$

Для когнитивных карт с весами соответствующая формула будет иметь вид:

$$x_i(t+1) = x_i(t) + \sum_{j \in I_i} a_{ij} \Delta x_j(t) + g_j(t) \qquad (5.2)$$

где $x_i(t+1)$ и $x_i(t)$ значения i-го фактора в момент времени $t+1$ и t, соответственно, $\Delta x_j(t) = x_j(t) - x_j(t-1)$ – приращение (импульс) фактора x_j, a_{ij} – вес взаимовлияния между факторами x_j и x_i, I_i – индексы прямо влияющих факторов на фактор x_i; $g_j(t)$ – внешнее воздействие (например, управление).

5.3.3 Анализ влияний в когнитивных картах

При анализе ситуаций, опирающемся на описанные выше модели когнитивных карт, решаются два типа задач: *статические и динамические. Статический анализ* – это анализ текущей ситуации, заключающийся в выделении и сопоставлении путей влияния одних факторов на другие через третьи (каузальных цепочек). Динамический анализ – это генерация и анализ возможных сценариев развития ситуации во времени. Математическим аппаратом анализа является теория знаковых графов и нечетких графов.

Задачи статического анализа, рассматриваемые в терминах знаковых графов, это исследование влияний одних факторов на другие, исследование устойчивости ситуации в целом и поиск структурных изменений для получения устойчивых структур. Фактор v_i влияет на фактор v_j, если существует ориентированный путь от вершины v_i в вершину v_j. Суммарное влияние v_i на v_j положительно, если все пути от v_i к v_j положительны; отрицательно, если все пути отрицательны; неопределенно, если среди этих путей есть как положительные, так и отрицательные.

Одна из основных задач, решаемых в терминах знаковых графов – *это задача об устойчивости*. В этой задаче ребра графа интерпретируются как некоторые (необязательно каузальные) отношения. Если отношения сим-

метричны, то ситуация представляется неориентированным знаковым графом, вершины которого соответствуют субъектам отношений.

Неориентированный знаковый граф *сбалансирован*, если все его циклы положительны. В этом случае все вершины можно разбить на два класса так, что ребра, соединяющие вершины одного класса, положительны, а ребра, соединяющие вершины разных классов, отрицательны. В сбалансированной ситуации все субъекты отношения разбиты на две коалиции, находящиеся в оппозиции. Такая ситуация устойчива в том смысле, что ввиду однородности отношений членов одной коалиции как друг к другу, так и к членам другой коалиции, нет предпосылок для изменения ситуации. Пример - двухпартийная парламентская система. Динамика развития ситуации при этом не рассматривается; прогноз состоит в том, что если ситуация устойчива, она будет сохраняться и в дальнейшем; если же она неустойчива, то характер связей скорее всего будет меняться.

Если отношения между факторами несимметричны, то когнитивная карта является ориентированным знаковым графом. Положительный цикл – это контур положительной обратной связи; если факторам приданы некоторые веса (значения), то увеличение веса фактора в цикле ведет к его дальнейшему увеличению и, в конечном счете, неограниченному росту. Отрицательный цикл противодействует отклонениям от начального состояния, однако возможна неустойчивость в виде значительных колебаний, возникающих при прохождении возбуждения по циклу. Различают случаи линейного, экспоненциального роста значений факторов, а также случай знакопеременного изменения и роста значений факторов (резонанса).

Анализ устойчивости графа предполагает поиск структурных изменений графа для получения устойчивой сбалансированной структуры. Существует ряд методов, направленных на поиск структурных изменений графа для получения устойчивых структур.

5.3.4 Этапы когнитивного моделирования

Когнитивное моделирование представляет собой циклический процесс и содержит несколько взаимосвязанных этапов:

– когнитивная структуризация;

– структурный анализ когнитивной модели;

– сценарное моделирование развития ситуации;

– оценка и интерпретация результатов моделирования;

– мониторинг ситуации.

На начальном этапе проводится ***когнитивная структуризация*** информации о ситуации и процессах, оказывающих влияние на ее развитие. Этап когнитивной структуризации включает в себя сбор, анализ и синтез (структуризацию) информации, т. е. построение когнитивной карты, описывающей механизм и условия развития ситуации. Когнитивная структуризация проводится с целью формирования множества базисных факторов и определения причинно-следственных отношений между ними.

На каждом этапе формирования модели приходится принимать решения, от совокупности которых, в конечном счете, зависит адекватность построенной модели. В число таких задач входят:

– выбор самой модели;

– формирование набора факторов и связей между ними (включая знак связи);

– выбор шкал и весов связей;

– выбор методов вычисления влияний.

Множество базисных факторов, причинно-следственные отношения между ними и параметры факторов и отношений определяются по результатам анализа текстов, содержащихся в информационно-аналитической базе, и анкетирования или интервьюирования экспертов и лиц, принимающих решения (ЛПР).

Из множества базисных факторов ситуации выбираются подмножества целевых и управляющих факторов, а также начальные тенденции базисных факторов. В качестве управляющих выбираются факторы, относящиеся к объекту управления или к внешней среде, на которые субъект управления имеет возможность воздействовать; в качестве целевых – факторы, в наибольшей степени характеризующие состояние объекта управления и его цели.

Структурный анализ когнитивной модели

Для более эффективного управления ситуацией необходимо знать ее структурные свойства (рис. 5.7), т. е. особенности причинно-следственных отношений между базисными факторами.

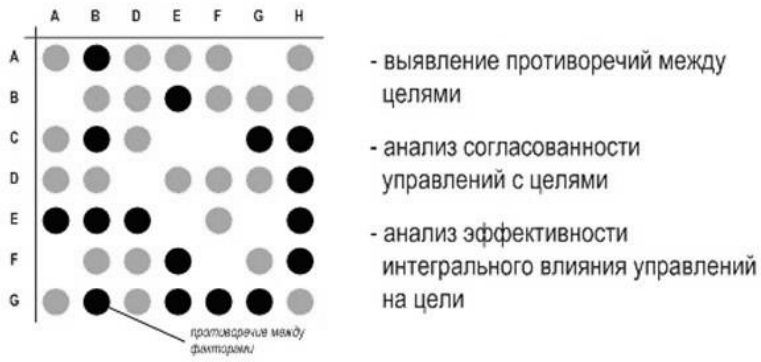


Рис. 5.7. Структурный анализ когнитивной модели

Суть задания непротиворечивого вектора целей состоит в том, чтобы желательное изменение одних целевых факторов не приводило к нежелательным изменениям других.

Управление ситуацией заключается в таком изменении управляющих факторов, которое приводило бы к желательным изменениям целевых факторов, т. е. в направлении оценки динамики. В связи с этим необходимо исследовать управляющие факторы на согласованность с целями и на эффективность их воздействия на целевые факторы. Согласованность управ-

ляющих факторов с вектором целей состоит в том, что никакое их изменение не вызовет изменения ни одной из целей в нежелательном направлении. Эффективность управляющего фактора определяется силой и характером его влияния на целевые факторы.

Сценарное моделирование развития ситуации

Моделирование может проводиться в режимах саморазвития и управляемого развития (рис. 5.8). Значение «тенденции» фактора в каждый момент определяется как сумма значения «тенденции» фактора в предыдущий момент и всех влияний, пришедших от «соседних» факторов. При определении результирующего значения «тенденции» фактора учитываются как собственно тенденции влияющих факторов, так и сила их влияния.

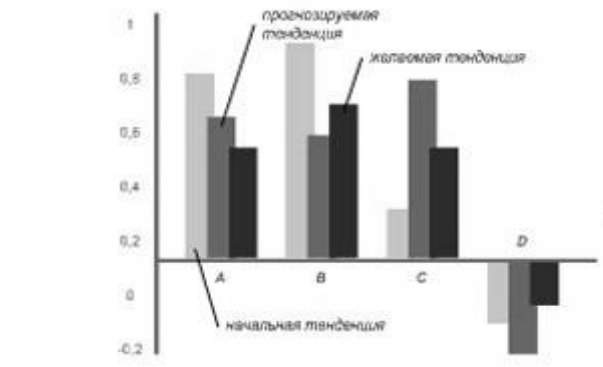


Рис. 5.8. Сценарное моделирование

Саморазвитие предполагает сохранение существующих тенденций факторов и, по сути, представляет собой экстраполяцию текущего положения с учетом взаимных влияний базисных факторов. Управляемое развитие ситуации подразумевает целенаправленное воздействие на один или несколько факторов, т. е. в качестве управления выступает изменение текущей тенденции фактора передаваемое на другие факторы по цепочке влияний.

Оценка и интерпретация результатов моделирования.

Для оценки эффективности принятых решений используется система показателей, характеризующих:

– степень достижения цели – коэффициент целедостижения;

– степень благоприятности ситуации для ЛПР – коэффициент благоприятности ситуации;

– объем и ценность ресурсов, необходимых для реализации управленческого решения, – ресурсоемкость управленческого решения;

– коэффициент эффективности решения, характеризующий отношение степени достижения целей к объему и ценности ресурсов, необходимых для реализации соответствующего решения.

Когнитивный мониторинг ситуации

На заключительном этапе в случае изменения текущей ситуации производится корректировка когнитивной модели, и повторяются процессы структурно-целевого анализа и моделирования развития ситуации.

Пример когнитивной модели

Рассмотрим пример построения когнитивной модели для анализа проблемы потребления электроэнергии в регионе [44] (рис. 5.9).

В соответствии с [44] исследуемую проблему достаточно полно можно описать семью факторами F, J, P, Q, R, C, U. Дугами на рисунке 5.9 отме-

чены существенные причинно-следственные отношения, влиянием остальных можно пренебречь.

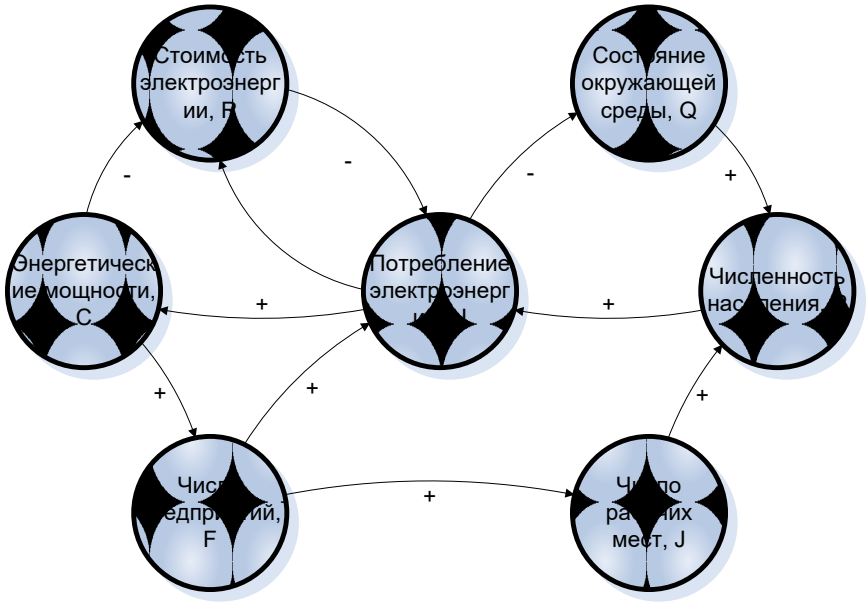


Рис. 5.9. Когнитивная карта «Потребление электроэнергии в регионе»

Дуга (Q, P) имеет знак « + », так как улучшение окружающей среды ведет к увеличению числа жителей, а ухудшение состояния окружающей среды вызывает отток населения. Дуга (U, Q) имеет знак « – », так как увеличение потребления энергии ухудшает состояние окружающей среды, а уменьшение потребления энергии благотворно сказывается на ее состоянии. Дуга (P, U) имеет знак « + » в виду того, что рост числа жителей вызывает увеличение потребления энергии и, наоборот, уменьшение населения приводит к падению потребления энергии.

Задав начальные значения и рассчитав значения факторов получим следующий сценарий развития ситуации (табл. 5.1).

Для улучшения ситуации (состояние окружающей среды), повысим стоимость электроэнергии (добавив управляющие воздействие 0.5) (табл. 5.2).

Табл. 5.1. Сценарий саморазвития

	Начальные значения	**Саморазвитие**
Число предприятий	0,25	0,30
Число рабочих мест	0,25	0,40
Численность населения	0,25	0,36
Состояние окружающей среды	0,10	-0,03
Стоимость электроэнергии	0,05	0,03
Энергетические мощности	0,10	0,15
Потребление электроэнергии	0,05	0,26

Табл. 5.2. Сценарий управляемого развития

	Начальные значения	**Управляемое развитие**
Число предприятий	0,25	0,29
Число рабочих мест	0,25	0,39
Численность населения	0,25	0,37
Состояние окружающей среды	0,10	0,01
Стоимость электроэнергии	0,20	0,22
Энергетические мощности	0,10	0,13
Потребление электроэнергии	0,05	0,17

Видно, что при увеличении стоимости электроэнергии, нам удалось улучшить состояние окружающей среды, незначительно увеличить численность населения, при этом снизив потребление электроэнергии в регионе, но потеряв некоторый процент энергетических мощностей.

Вопросы к главе 5

1. Что такое концептуальное моделирование?
2. Задачи и цели концептуальных моделей.

3. Виды концептуальных моделей.

4. Методологические подходы в области концептуального моделирования.

5. Перечислите типы концептуальных карт.

6. Для каких задач используются концептуальные карты?

7. Сформулируйте задачу когнитивного моделирования.

8. Дайте определение понятию «когнитивная модель».

9. Перечислите этапы когнитивного моделирования.

10. Что такое структуризация предметной области?

ГЛАВА 6 СРЕДЫ ИМИТАЦИОННОГО МОДЕЛИРОВАНИЯ

Любой алгоритм моделирования может быть записан на обычном процедурно ориентированном языке программирования. Преимуществами подобной организации работы являются:

- отсутствие необходимости в специальном программном обеспечении и его изучении;
- «прозрачность» логики модели не только для разработчиков, но и для широкого круга программистов;
- доступность встроенных функций, библиотечных подпрограмм, средств статистического анализа, редактирования вывода и других возможностей, обычно предоставляемых системами программирования широкого назначения;
- отсутствие ограничений па состав и логику модели, исключая налагаемые конфигурацией моделирующей вычислительной установки и режимом ее использования;
- возможность управления экспериментом, необходимая, в частности, для реализации методов уменьшения дисперсии результатов.

Тем не менее, составление имитационных моделей с использованием стандартных алгоритмических языков весьма трудоемко по следующим причинам:

- постоянно возрастает количество элементов и отслеживаемых процессов в моделируемых системах;
- имитационные модели сложны в отладке из-за богатой логики, взаимной зависимости ветвей алгоритма, большой трудоемкости прогонов и низкой точности результатов, способной в ряде случаев замаскировать ошибку;
- незначительные изменения модели требуют внесения в программу многочисленных поправок и проведения отладки практически заново.

Выходом здесь может служить использование некоторых стандартных схем в качестве моделей класса систем. Возможность создания типовых моделей базируется на двух обстоятельствах:

1) Даже в сложных моделях с тысячами элементов количество видов действий остается весьма ограниченным. Такие действия естественно описывать как подпрограммы — при необходимости с дополнительными параметрами.

2) Имитационные модели достаточно широких классов реальных процессов (работа комбината бытового обслуживания, вычислительной системы коллективного пользования, проведение ремонтных и регламентных работ на технике, распространение эпидемии и борьба с ней, система снабжения запчастями авиабазы в период боевых действий и т.п.) имеют сходную логику, опреде¬ляемую общей математической моделью.

Языки моделирования позволяют писать имитационные программы в такой форме, которая напоминает описание моделируемой системы, и притом так, что малым изменениям системы соответствуют малые изменения в программе, что обеспечивает быстроту и правильность составления программ моделирования. Чем уже проблемная ориентация, тем меньше работы по программированию. В пределе можно добиться полной параметризации системы, и тогда задание конкретной модели будет состоять лишь в указании числовых значений параметров. Обзор 350 языков моделирования приводится в книге Киндлера Е. «Языки моделирования».

Со временем появились новые инструменты моделирования, которые представляют весь арсенал новейших информационных технологий, включая развитые графические оболочки для конструирования моделей и интерпретации выходных результатов, мультимедийные средства и видео, поддерживающие анимацию в реальном масштабе времени, объектно-ориентированное программирование, Internet – решения и др.

В работе Лычкиной Н.Н. «Современные технологии имитационного моделирования и их применение в информационных бизнес-системах и системах поддержки принятия решений» представлены технологические возможности современных систем моделирования, которые характеризуются:

- универсальностью и гибкостью базовой и альтернативной к базовой концепций структуризации и формализации моделируемых динамических процессов, заложенных в систему моделирования. Сегодня популярны среди систем моделирования дискретного типа процессно-ориентированные концепции структуризации, основанные на сетевых парадигмах, автоматном подходе и некоторые другие; среди систем моделирования непрерывного типа – модели и методы системной динамики;
- наличием средств проблемной ориентации, когда система моделирования содержит наборы понятий, абстрактных элементов, языковые конструкции из предметной области соответствующего исследования;
- применением объектно-ориентированных специализированных языков программирования, поддерживающих авторское моделирование и процедуры управления процессом моделирования;
- наличием удобного и легко интерпретируемого графического интерфейса, когда блок-схемы дискретных моделей и системные потоковые диаграммы непрерывных реализуются на идеографическом уровне, параметры моделей определяются через подменю;
- использованием развитой двух- и трех-мерной анимации в реальном времени;
- возможностью для реализации нескольких уровней представления модели, средствами для создания стратифицированных описаний. Современные системы моделирования применяют структурно-функциональный

подход, многоуровневые иерархические, вложенные структуры и другие способы представления моделей на разных уровнях описания;

- наличием линеек и инструментов для проведения и анализа результатов сценарных, вариантных расчетов на имитационной модели;
- математической и информационной поддержкой процедур анализа входных данных, анализа чувствительности и широкого класса вычислительных процедур, связанных с планированием, организацией и проведением направленного вычислительного эксперимента на имитационной модели.
- Экспериментальные исследования на имитационной модели информативны, поэтому необходима реализация подхода Simulation Data Base, основанного на доступе к базам данных моделирования. Технологически это решается при помощи собственных специализированных аналитических блоков системы моделирования или за счет интеграции с другими программными средами;
- исполнительный модуль может функционировать вне среды для разработки модели;
- применением многопользовательского режима работы, интерактивного распределенного моделирования, разработками в области взаимодействия имитационного моделирования со Всемирной паутиной и др.

Перечень программного обеспечения и инструментальных средств имитационного моделирования можно посмотреть на сайте http://dic.academic.ru/dic.nsf , а также на сайте Национального общества имитационного моделирования: www.simulation.su. Ниже приведено краткое описание трех систем.

6.1 Simulink

Программа Simulink является приложением к пакету MATLAB. Simulink - интерактивная среда для моделирования и анализа широкого

класса динамических систем. Simulink предоставляет пользователю графический интерфейс для конструирования моделей из стандартных блоков при помощи технологии "darg-and-drop".

В Simulink входит большая библиотека блоков, позволяющая легко создавать модели, перемещая компоненты из библиотеки в рабочую область и соединяя их. Группируя блоки в подсистемы, можно создавать иерархические модели. Число блоков и связей в модели не ограничено.

Блок-диаграмы Simulink обеспечивают интерактивную среду для нелинейного моделирования. Моделирование выполняется с помощью меню или из командной строки. Результаты моделирования отображаются в процессе работы. Параметры модели можно менять в процессе моделирования.

Simulink является достаточно самостоятельным инструментом MATLAB и при работе с ним совсем не требуется знать сам MATLAB и остальные его приложения. С другой стороны доступ к функциям MATLAB и другим его инструментам остается открытым и их можно использовать в Simulink. Часть входящих в состав пакетов имеет инструменты, встраиваемые в Simulink (например, LTI-Viewer приложения Control System Toolbox – пакета для разработки систем управления). Имеются также дополнительные библиотеки блоков для разных областей применения (например, Power System Blockset – моделирование электротехнических устройств, Digital Signal Processing Blockset – набор блоков для разработки цифровых устройств и т.д).

При работе с Simulink пользователь имеет возможность модернизировать библиотечные блоки, создавать свои собственные, а также составлять новые библиотеки блоков.

При моделировании пользователь может выбирать метод решения дифференциальных уравнений, а также способ изменения модельного вре-

мени (с фиксированным или переменным шагом). В ходе моделирования имеется возможность следить за процессами, происходящими в системе. Для этого используются специальные устройства наблюдения, входящие в состав библиотеки Simulink. Результаты моделирования могут быть представлены в виде графиков или таблиц. Преимущество Simulink заключается также в том, что он позволяет пополнять библиотеки блоков с помощью подпрограмм написанных как на языке MATLAB, так и на языках C++, Fortran и Ada. С помощью Real-Time Workshop Вы можете генерировать С код моделей Simulink

6.2 Model Vision Studium (новое название Rand Model Designer)

Виртуальный испытательный стенд Model Vision Studium (MVS) - это графическая среда моделирования и проектирования сложных динамических систем, использующая объектно-ориентированную технологию. Практически важные сложные динамические системы чаще всего изучаются с помощью гибридных моделей, одновременно отражающих непрерывные и дискретные свойства реального поведения объекта. MVS - это современный инструмент для разработки и изучения гибридных моделей сложных физических и технических объектов.

Образ испытательного стенда и привычная для инженера технология проектирования составили основу технологии MVS. Графические образы отдельных компонент моделируемых устройств размещаются и соединяются на виртуальном стенде в единое устройство. К собранному устройству подсоединяются виртуальные измерительные приборы и генераторы внешних воздействий. Моделируемое поведение объекта воспроизводится в реальном времени. В любой момент можно приостановить испытания и вмешаться в ход эксперимента, чтобы изменить параметры или внешние воздействия. Широкий набор виртуальных устройств и возможность легко

и надежно создавать новые устройства, позволяет применять MVS практически во всех областях научной и инженерной деятельности.

Система разрабатывался при поддержки сотрудников кафедры "Распределенных вычислений и компьютерных сетей" факультета Технической Кибернетики Санкт-Петербургского Государственного Технического Университета. С 2014 года продукт получил название Rand Model Designer и поддерживается компанией MVSTUDIUM Group (http://www.mvstudium.com/index.htm).

Система позволяет описать модель на специальном графическом языке, а затем автоматически построить программу для воспроизведения ее поведения, использующую для этого современные численные методы. Пакет предназначен для исследования гибридных систем. Гибридными называют системы, обладающие одновременно "непрерывными" и "дискретными" свойствами.

В основе языка моделирования, реализованного в MVS, лежит специальная форма наглядного представления гибридного поведения - Карта Поведения. Использование карты поведения позволяет создать простой и удобный инструмент для моделирования гибридных систем, ориентированный на широкий круг прикладных пользователей (инженеров, преподавателей, аспирантов, студентов и, как показывает опыт, даже школьников).

Карта поведения

Карта поведения (bihavior chart или B-chart) - это ориентированный граф, в котором узлам приписываются некоторые локальные поведения, а дугам, называемым переходами, - условия перехода от одного поведения к другому и выполняемые при этом действия. Узел, в котором система находится в каждый конкретный момент времени, называется текущим. Один из узлов должен быть предварительно помечен как начальный, он автоматически становится текущим при создании карты состояний. Соответст-

вующее ему начальное поведение создается при создании экземпляра устройства. Смена текущего узла происходит в результате срабатывания переходов. Когда узел становится текущим, создается экземпляр приписанного ему локального поведения. Созданный экземпляр уничтожается, как только узел перестает быть текущим.

Локальное поведение может быть описано как

а) непрерывное поведением,

б) карта поведения (в этом случае узел называется гиперузлом) или

в) считаться пустым поведением NULL.

Графический образ карты поведения позволяет в наглядной форме представлять множества допустимых локальных поведений устройства, их области определения в фазовом пространстве и времена переходов от одного локального поведения к другому. При описании локальных поведений, предусмотрена возможность вводить локальные переменные (аналогичным локальным переменным программных единиц). В принятом в MVS подходе, когда локальные поведения создаются и уничтожаются одновременно с изменением поведения системы, текущий фазовый вектор может менять размерность.

В общем случае переход T из начального узла Nb в конечный узел Ne характеризуется: охраняющим предикатом G, запускающим событием E, и действиями A. Возможны три типа запускающего события:

1) некоторое логическое условие стало истинным (change event);

2) поступил внешний сигнал (signal event);

3) истекло определенное время после того как начальный узел Nb стал текущим (time event).

Семантика перехода следующая. Если узел Nb является текущим и предикат G истинен или отсутствует, переход T становится открытым, в противном случае переход закрыт. Если событие E не указано, то откры-

тый переход немедленно срабатывает. Если указано событие E, то открытый переход сработает только при его появлении и истинности предиката G (до появления события E переход может закрыться, если предикат G перестает быть истинным или узел Nb текущим). Срабатывание представляет собой следующую последовательность мгновенных действий:

1) узел Nb перестает быть текущим;

2) выполняется последовательность действий A;

3) узел Ne становится текущим.

Непрерывное поведение в общем случае задается совокупностью обыкновенных дифференциальных и алгебраических уравнений , а также формул вида

<выражение, не зависящее от s>.

Уравнения и формулы могут задаваться как в скалярной, так и в матричной форме.

В действиях переходов и правых частях уравнений и формул возможно использование алгоритмических функций и процедур, задаваемых либо с помощью встроенного алгоритмического языка - подмножества языка Ada, либо во внешних программных модулях.

Карта поведения представляет собой простую и наглядную форму описания процесса смены поведений. В частном случае устройства с чисто дискретным поведением (например, контроллер в примере 1), всем узлам карты поведения следует приписать пустые локальные поведения и тогда она превращается в обычную карту состояний (statechart) с единственным отличием. Это отличие связано с невозможностью перехода в локальную карту поведения по команде "покажи предисторию" ("history indicator"). В нашем случае поведение - это параметризованный класс, конкретный экземпляр которого создается при входе в узел карты поведения и уничтожается при выходе из него. Элемент с чисто непрерывным поведением (на-

пример, объект управления в примере 1) трактуется как гибридный с картой поведения, состоящей из единственного узла, которому приписано непрерывное поведение. Таким образом, дискретные аспекты поведения отражаются с помощью хорошо знакомого языка карт состояния, а непрерывные аспекты - с помощью привычного языка систем уравнений и формул.

Блоки и связи

Основным “строительным элементом” описания в MVS является устройство (device). Устройство - это некоторый активный объект, функционирующий параллельно и независимо от других объектов в непрерывном времени . Устройство является ориентированным блоком, т.е. все взаимодействия устройства с окружающим миром осуществляются только через его входы и выходы, составляющие интерфейс устройства. Все остальные свойства устройства инкапсулированы внутри него. В общем случае в описании устройства содержатся следующие элементы: входы, выходы, параметры конструктора, переменные состояния, поведение, локальная структура, анимация. Входы, выходы и переменные состояния являются фазовыми переменными и все вместе составляют фазовый вектор устройства. Идентификаторы блоков и переменных могут быть русскими.

Типы данных включают в себя скалярные, регулярные и комбинированные типы, а также типы, определяемые пользователем. Скалярные типы это - вещественные, целые, булевский, перечислимые, символьный и строковый типы. Регулярными типами являются одномерные и двумерные массивы с произвольными элементами, матрицы и векторы с вещественными элементами, а также списки. Комбинированные типы представляют собой записи с полями различных типов. Для передачи информации о дискретных событиях используется специальный тип "сигнал".

Описание устройства всегда строится как описание класса устройств. Для конкретных экземпляров устройства могут быть указаны специальные значения параметров конструктора. Устройства могут соединяться между собой однонаправленными функциональными связями и входить в состав других устройств, образуя иерархическую структуру. При описании локальной структуры возможно использование сложных многокомпонентных связей, агрегирование и расщепление связей. Возможно использование регулярных структур устройств (массивов и списков)

Конкретная моделируемая система, с которой будет проводиться вычислительный эксперимент, собирается в окне редактора виртуального стенда из экземпляров классов, определенных в данном проекте, и/или библиотечных классов.

Объектно-ориентированное моделирование

В MVS описание устройства всегда строится как описание класса устройств. Статический экземпляр локального устройства создается автоматически при создании экземпляра составного устройства. Статический экземпляр главного устройства проекта создается при запуске вычислительного опыта и уничтожается по его окончании. Динамические экземпляры устройств должны быть явно созданы или уничтожены с помощью специальных операторов в действиях переходов.

Вводимый класс устройств может наследовать свойства другого класса. Все устройства являются потомками предопределенного класса CDevice, которому приписаны все предопределенные соглашения о взаимодействии с исполняющей системой MVS. При наследовании, вы можете вводить новые параметры, фазовые переменные, константы, алгоритмические процедуры и функции, локальные поведения, локальные устройства и связи, узлы и переходы в карте поведения, анимационные окна и анимационные компоненты, а также переопределять и перегружать алгоритмиче-

ские процедуры и функции, поведения, узлы и переходы в карте поведения, анимационные окна и анимационные компоненты.

Классы устройств обладают свойством полиморфизма: вместо экземпляра класса-предка может использоваться экземпляр класса-потомка.

К проекту могут быть присоединены ранее созданные библиотеки классов и при создании своей модели вы можете использовать уже готовые классы устройств. Любой проект может быть превращен в библиотеку классов и любой класс из вашего проекта может быть добавлен в существующую библиотеку. Пакет поставляется со стандартной библиотекой классов SysLib, содержащей набор наиболее типовых линейных блоков, нелинейных блоков, и источников сигналов.

6.3. AnyLogic

Система разрабатывается AnyLogic Company (http://www.anylogic.ru/).

AnyLogic - единственный инструмент имитационного моделирования (ИМ), который поддерживает все подходы к созданию имитационных моделей: процессно-ориентированный (дискретно-событийный), системно динамический и агентный, а также любую их комбинацию.

AnyLogic применяется в диапазоне от микро-моделей «физического» уровня, где важны конкретные размеры, расстояния, скорости, времена, до макро-моделей «стратегического» уровня, на котором рассматривается глобальная динамика обратных связей, тенденции на длительных временных отрезках и оцениваются стратегические решения.

AnyLogic обеспечивает поддержку всех этапов имитационного моделирования: для различных типов динамических моделей – дискретных, непрерывных и гибридных. Создание модели, ее выполнение, оптимизация параметров, анализ полученных результатов, верификация модели – все эти этапы удобно выполнять в среде AnyLogic.

AnyLogic поддерживает все элементы системной динамики (накопители, потоки, обратные связи, задержки, вспомогательные переменные, табличные функции, массивы и уравнения над ними и т.д.), но, в отличие от традиционных инструментов, обеспечивает существенно лучшую структуризацию моделей за счёт понятия объекта, интерфейса и иерархии. Кроме того, в AnyLogic возможно определить сколь угодно сложную дискретно-событийную логику (например, при помощи диаграмм состояний или диаграмм процессов) и связать её с системно-динамической частью — только увязав структуру и поведение, возможно эффективно моделировать взаимодействие компании и её окружения.

Одним из наиболее важных преимуществ AnyLogic является возможность быстрого построения многоагентных моделей, которую не даёт ни один из существующих инструментов. Активные объекты AnyLogic могут создаваться и уничтожаться динамически, перемещаться, общаться друг с другом, иметь поведение, знания, цели, стратегию — то есть обладают всеми свойствами агентов. При помощи агентов моделируют рынки (агент — потенциальный покупатель), конкуренцию и цепочки поставок (агент — компания), население (агент — семья, житель города или избиратель) и много другое. Только агентные модели позволяют получить представление об общем поведении системы, исходя из предположений о поведении её элементов при отсутствии знания о глобальных законах — то есть в наиболее общем случае.

Комбинируя системную динамику на уровне стратегии с агентными моделями рынка и дискретными моделями производства и логистики, можно добиться наиболее адекватного представления глобальной цепочки поставок, обеспечив надёжный базис для принятия решений и, таким образом, получения конкурентного преимущества.

AnyLogic имеет исключительно развитый базовый язык дискретного и смешанного дискретно/непрерывного моделирования, на основе которого построены решения для конкретных областей: библиотека Enterprise Library, а также Material Flow Library (потоки материалов) и Healthcare Library (работа медицинских учреждений), включенные в состав продукта. Enterprise Library содержит традиционные объекты: очереди, задержки, конвейеры, ресурсы, и т.п., так что модель и анимация быстро строятся в стиле «перетащить и оставить» (drag-and-drop) и очень гибко параметризуется.

Реализация стандартных объектов открыта для пользователя, их функциональность может быть как угодно расширена, вплоть до создания собственных библиотек. Используя иерархию и регулярные структуры объектов, можно создавать масштабируемые модели. Специально разработанная техника анимации позволяет быстро связать модель с техническими чертежами.

Для решения многих задач необходимо учитывать не только привычные всем параметры, такие, например, как время, скорость или расстояние, но и физические размеры, геометрию и поведение объектов (агентов) и окружающей их среды, что требует от инструмента имитационного моделирования значительной гибкости и производительности, которые предоставляет AnyLogic. Благодаря естественной поддержке агентного моделирования, AnyLogic позволяет задавать различные модели поведения индивидуумов и их взаимодействия друг с другом и окружающей средой, вплоть до использования элементов искусственного интеллекта, что позволяет более адекватно моделировать систему.

Визуальная среда моделирования AnyLogic поддерживает проектирование, разработку, документирование модели, выполнение компьютерных экспериментов с моделью, включая различные виды анализа – от анализа

чувствительности, анализа риска до оптимизации параметров модели относительно некоторого критерия, регистрацию и запоминание результатов моделирования в базе данных для возможности их последующего использования и анализа.

Инструмент обладает большим спектром разнообразных возможностей проведения как отдельных прямых экспериментов типа “If-then (Если-то)”, так и серий таких экспериментов для решения разнообразных обратных задач.

В каждом компьютерном эксперименте пользователь может установить свои параметры (факторы) всех объектов модели, от которых зависит ее поведение, установить специфические условия останова выполнения модели и проведения эксперимента, менять установки при выборе численных методов и требований к точности решения и т.п.

Типы экспериментов, поддерживаемых в системе моделирования AnyLogic:

Простой эксперимент – это эксперимент “что-если” (когда модель выполняется заданное время при фиксированных начальных значениях параметров). Её типичное использование - это просмотр анимации модели или сбор статистики. Простой эксперимент может быть запущен как в реальном так и виртуальном режимах времени, результаты эксперимента могут быть сохранены, а результаты нескольких экспериментов можно представлять в виде диаграммы.

Оптимизационный эксперимент в AnyLogic позволяет найти наилучшее решение, т.е. такие значения параметров модели, при которых обращается в минимум или максимум некоторая определенная пользователем целевая функция. Значение целевой функции подсчитываются в AnyLogic каждый раз по окончании очередного выполнения модели, и алгоритм оптимизации автоматически выбирает новые значения параметров для оче-

редного запуска модели. Оптимизация в AnyLogic реализована с использованием пакета OptQuest фирмы OptTek. Для оптимизации пользователь должен в соответствующих окнах задать функционал, который следует минимизировать либо максимизировать, задать параметры и ограничения их диапазона, в которых должна выполняться оптимизация, а также указать ограничения, определяющие класс допустимых решений. Задав все это, пользователь может запустить оптимизацию, и пакет OptQuest после некоторого числа проб автоматически выберет наилучший метод оптимизации для данной модели среди тех методов, которые находятся в его библиотеке, и будет следовать выбранному оптимизационному алгоритму. Интерфейс AnyLogic позволяет пользователю также использовать свой алгоритм оптимизации.

Анализ чувствительности ориентирован на анализ поведения модели в зависимости от вариации входных параметров или нахождение параметров, к изменению которых модель наиболее чувствительна.

Нестандартный эксперимент. Если пользователю не подходит ни один из стандартных экспериментов, он может создать свой собственный эксперимент, используя Java. При описании эксперимента можно вызывать любые другие эксперименты.

AnyLogic позволяет построить компьютерную анимацию поведения модели, определив динамические графические элементы, представляющие процесс, и связав с каждым из этих элементов переменные модели. Изменение переменных будет отображаться в изменении координат положения и размеров соответствующих графических элементов, их ориентации, цвета, видимости и других характеристиках. Индикаторы позволяют отразить изменение переменных не только графиком.

Модели AnyLogic можно помещать на web-сайты в виде апплетов. Это уникальное свойство позволяет удалённым пользователям запускать инте-

рактивные модели в web-браузере без необходимости устанавливать какое-либо программное обеспечение.

Модели, построенные в AnyLogic, имеют открытую архитектуру и могут работать с любым офисным или корпоративным ПО, а также с пользовательскими модулями, написанными на различных языках. Модель может динамически читать и сохранять данные в электронных таблицах, базах данных, системах планирования корпоративных ресурсов (ERP) и управления взаимоотношениями с клиентами (CRM), а также быть встроена в производственный или контур управления.

AnyLogic поддерживает как стохастические так и детерминированные модели. Существуют различные способы описания стохастического поведения: использование различных законов распределения и/или статистики для описания времени между событиями; реализация недетерминированных алгоритмов, например, заявка теряется с определенной вероятностью. В случае, если несколько событий должны произойти в одно и то же время, AnyLogic позволяет выбрать событие, которое должно произойти, случайно, используя равномерное распределение.

AnyLogic поддерживает Stat::Fit, специализированное программное обеспечение для обработки статистики, которое позволяет подбирать распределения по имеющейся выборке.

Непрерывное моделирование

В AnyLogic объекты могут перемещаться в непрерывном 2D и 3D пространстве, остывать и нагреваться, воздействовать на другие объекты и т.д., что дает возможность удобного моделирование непрерывных процессов в AnyLogic, например физических систем, химических реакции, электрических сетей, состояния жидкостей или газов и т.д.

AnyLogic изначально спроектирован и разработан для моделирования всех трех типов процессов: дискретных, непрерывных и гибридных.

AnyLogic поддерживает нескольких типов уравнений:

— Дифференциальные: d(x)/dt = f(x,y,t)

— Алгебраические: g(x,y,t) = 0, find(x)

— Формулы: z = h(x.v.t)

В «имитационный исполнитель» AnyLogic встроен Решатель уравнений, таким образом,уравнения решаются численно.Численные методы включают: Эйлера. Рунге-Кутта, Радау5. Пауэлла, Ньютона, методы для решения жестких систем и т.д.

Функции, реализуемые в AnyLogic

— Математические функции. В AnyLogic встроены часто используемые математические функции.

— Алгоритмические функции позволяют использовать Java для задания сложных алгоритмов (if-then-else, loops, DB access, etc)

— Табличные функции предоставляют простой способ использования табличных данных в модели. AnyLogic поддерживает полиномиальную аппроксимацию и три типа интерполяции: ступенчатую, линейную и сплайн.

Библиотеки AnyLogic – это множество классов активных объектов, анимаций, классов сообщений и Java модулей, разработанных для упрощения создания моделей в какой-нибудь конкретной области, которые обеспечивают простоту повторного использования объектов в разных моделях. Существует возможность создания собственных библиотек.

AnyLogic включает библиотеки: Enterprise library, Material flow library, Dynamic Systems library, Business Graphics Library, Agent Based Library и Pedestrian Library

6.4. Программный комплекс «Моделирование в технических устройствах» (ПК «МВТУ»)

Программный комплекс «МВТУ» предназначен для исследования динамики и проектирования самых разнообразных систем и устройств. По своим возможностям он является альтернативой аналогичным зарубежным программным продуктам Simulink, VisSim и др. Удобный редактор структурных схем, обширная библиотека типовых блоков и встроенный язык программирования позволяют реализовывать модели практически любой степени сложности, обеспечивая при этом наглядность их представления. ПК «МВТУ» успешно применяется для проектирования систем автоматического управления, следящих приводов и роботов-манипуляторов, ядерных и тепловых энергетических установок, а также для решения нестационарных краевых задач (теплопроводность, гидродинамика и др.). Широко используется в учебном процессе, позволяя моделировать различные явления в физике, электротехнике, в динамике машин и механизмов, в астрономии и т.д. Может функционировать в многокомпьютерных моделирующих комплексах, в том числе и в режиме удаленного доступа к технологическим и информационным ресурсам.

ПК «МВТУ» реализует следующие режимы работы:

- МОДЕЛИРОВАНИЕ, обеспечивающий:
- моделирование процессов в непрерывных, дискретных и гибридных динамических системах, в том числе и при наличии обмена данными с внешними программами и устройствами;
- редактирование параметров модели в режиме «on-line»;
- расчет в реальном времени или в режиме масштабирования модельного времени;
- рестарт, архивацию и воспроизведение результатов моделирования;
- статистическую обработку сигналов, основанную на быстром преобразовании Фурье.
- ОПТИМИЗАЦИЯ, позволяющий решать задачи:

- минимизации (максимизации) заданных показателей качества;
- нахождения оптимальных параметров проектируемой системы в многокритериальной постановке при наличии ограничений на показатели качества и оптимизируемые параметры.
- АНАЛИЗ, обеспечивающий:
- расчет и построение частотных характеристик и годографов;
- расчет передаточных функций, их полюсов и нулей;
- реализацию метода D-разбиения на плоскости одного комплексного параметра.
- СИНТЕЗ, позволяющий конструировать регуляторы:
- по заданным желаемым частотным характеристикам;
- по заданному расположению доминирующих полюсов.
- КОНТРОЛЬ И УПРАВЛЕНИЕ, позволяющий создавать виртуальные аналоги:
- пультов управления с измерительными приборами и управляющими устройствами;
- мнемосхем с мультимедийными и анимационными эффектами.

К достоинствам ПК «МВТУ» относятся:

- открытость за счет встроенного языка и реализации нескольких механизмов обмена данными с внешними программами;
- простота построения сложных моделей благодаря использованию вложенных структур, векторизации сигналов и алгоритмов типовых блоков, удобным средствам задания параметров и уравнений;
- эффективные численные методы;
- большое число обучающих и демонстрационных примеров с подробными комментариями.

Для отечественных пользователей удобство работы с ПК «МВТУ» обусловлено также русскоязычным интерфейсом и наличием обширной

документации на русском языке. Учебная и демонстрационная версии ПК «МВТУ» вместе с полной документацией и набором демонстрационных примеров распространяются свободно. В учебной версии есть ограничения на сложность модели: порядок дифференциальных уравнений не выше 30, а число блоков не более 100. В демонстрационной версии таких ограничений нет, но модель нельзя сохранить.

ГЛАВА 7 ОСНОВЫ АГЕНТНОГО МОДЕЛИРОВАНИЯ В СИСТЕМЕ ANYLOGIC

7.1 Краткие сведения о системе AnyLogic

AnyLogic – отечественный профессиональный инструмент нового поколения, который предназначен для разработки и исследования имитационных моделей. Разработчик продукта – компания «Экс Джей Текнолоджис» (XJ Technologies), г. Санкт-Петербург; электронный адрес: www.xjtek.ru.

AnyLogic основан на объектно-ориентированной концепции. Другой базовой концепцией является представление модели как набора взаимодействующих, параллельно функционирующих активностей. Активный объект в AnyLogic – это объект со своим собственным функционированием, взаимодействующий с окружением. Он может включать в себя любое количество экземпляров других активных объектов. Графическая среда моделирования поддерживает проектирование, разработку, документирование модели, выполнение компьютерных экспериментов, оптимизацию параметров относительно некоторого критерия.

При разработке модели можно использовать элементы визуальной графики: диаграммы состояний (стейтчарты), сигналы, события (таймеры), порты и т.д.; синхронное и асинхронное планирование событий; библиотеки активных объектов.

Для реализации специальных вычислений и описания логики поведения объектов AnyLogic позволяет использовать язык Java.

Основными строительными блоками модели AnyLogic являются активные объекты, которые позволяют моделировать любые объекты реального мира. Класс определяет шаблон, в соответствии с которым

строятся отдельные экземпляры класса. Эти экземпляры могут быть определены как объекты других активных объектов. Активный объект является экземпляром класса активного объекта. Активные объекты могут содержать вложенные объекты, причем уровень вложенности не ограничен.

Активные объекты имеют четко определенные интерфейсы взаимодействия. Это облегчает создание систем со сложной структурой, а также делает активные объекты повторно используемыми. Создав класс активного объекта, вы можете создать любое количество объектов – экземпляров данного класса.

Основными средствами описания поведения объектов являются переменные, события и диаграммы состояний. Переменные отражают изменяющиеся характеристики объекта. События могут наступать с заданным интервалом времени и выполнять заданное действие. Диаграммы состояний (или стейтчарты) позволяют визуально представить поведение объекта во времени под воздействием событий или условий, они состоят из графического изображения состояний и переходов между ними (т.е. по сути это конечный автомат). Любая сложная логика поведения объектов модели может быть выражена с помощью комбинации стейтчартов, дифференциальных и алгебраических уравнений, переменных, таймеров и программного кода на Java [45]. Алгебраические и дифференциальные уравнения записываются аналитически. Интерпретация любого числа параллельно протекающих процессов в модели AnyLogic скрыта от пользователя. Никаких усилий разработчика модели для организации квазипараллелизма интерпретации не требуется; отслеживание всех событий выполняется системой автоматически.

Агенты в AnyLogic

Агент – это некоторая сущность, которая обладает активностью, автономным поведением, может принимать решения в соответствии с

некоторым набором правил, может взаимодействовать с окружением и другими агентами, а также может изменяться (эволюционировать). Многоагентные (или просто агентные) модели используются для исследования децентрализованных систем, динамика функционирования которых определяется не глобальными правилами и законами, а, наоборот, эти глобальные правила и законы являются результатом индивидуальной деятельности членов группы. Цель агентных моделей – получить представление об общем поведении системы исходя из знаний о поведении ее отдельных активных объектов и взаимодействии этих объектов в системе [46].

В среде AnyLogic агент создается с помощью базового элемента AnyLogic – активного объекта. В модели можно создавать классы активных объектов и далее использовать в модели любое число экземпляров этих классов. Активный объект имеет параметры, которые можно изменять извне, переменные, которые можно считать памятью агента, а также поведение.

Стейтчарты и таймеры могут выражать поведение: состояния агента и изменение состояний под воздействием событий и условий. Кроме того, агент может иметь интерфейс для взаимодействия с окружением, который реализуется с помощью интерфейсных объектов: портов и интерфейсных переменных.

7.2 Работа в системе AnyLogic

Постановка задачи

Создадим модель для моделирования процесса приобретения нового продукта под влиянием рекламной кампании, проводимой с целью выведения нового продукта на сложившийся рынок [47].

В этой модели интенсивность рекламы и вероятность того, что продукт будет приобретен под ее влиянием, полагаются постоянными.

Поэтому мы зададим эффективность рекламы константой. Эффективность рекламы определяется количеством людей купивших продукт под воздействием рекламы.

Шаг 1. Создание простейшей агентной модели

1. Щелкните мышью по кнопке панели инструментов **Создать**  . Задайте имя новой модели и сохраните проект.

2. Щелкните мышью по кнопке **Далее**. Откроется вторая страница **Мастера создания модели**. Здесь предлагается выбрать шаблон модели, на базе которого будет разрабатываться модель. Выберите **Агентная модель** в расположенном списке.

3. Далее необходимо задать имя класса агента и количество агентов, которое будет изначально создано. Задайте в качестве имени класса Person и введите в поле **Начальное количество агентов** 500. Автоматически будет создано 500 агентов (то есть, экземпляров класса Person, каждый из которых будет представлять отдельного агента).

4. После нажатия кнопки **Далее** откроется следующая страница для задания свойств пространства, в котором будут действовать агенты, и выбора фигуры анимации агента.

5. Установите флажок **Добавить пространство** и выберите ниже тип этого пространства: **Непрерывное**. Здесь же можно задать размерность пространства: в поле **Ширина** введем 600, а в поле **Высота** – 350.

6. При открытии следующей страницы **Мастера** необходимо определить будет ли задана сеть взаимосвязей агентов. Установите флажок **Использовать сеть** и оставьте выбранной опцию **Случайное**.

7. После нажатия кнопки **Далее** откроется последняя страница **Мастера создания модели**. Установите на ней флажок **Добавить простое поведение** для создания диаграммы состояний.

Пользовательский интерфейс AnyLogic

В левой части рабочей области находится панель **Проекты** (рис. 6.1). Панель **Проекты** обеспечивает легкую навигацию по элементам моделей, открытым в текущий момент времени. Поскольку модель организована иерархически, то она отображается в виде дерева: сама модель образует верхний уровень дерева; эксперименты, классы активных объектов и Java классы образуют следующий уровень; элементы, входящие в состав активных объектов, вложены в соответствующую подветвь дерева класса активного объекта и т.д.

В правой части рабочей области отображается панель **Палитра**, а внизу – панель **Свойства**. Панель **Палитра** содержит разделенные по категориям элементы, которые могут быть добавлены на графическую диаграмму класса активного объекта или эксперимента. Панель **Свойства** используется для просмотра и изменения свойств выбранного в данный момент элемента (или элементов) модели.

В центре рабочей области AnyLogic находится графический редактор в котором отображается диаграмма класса Main.

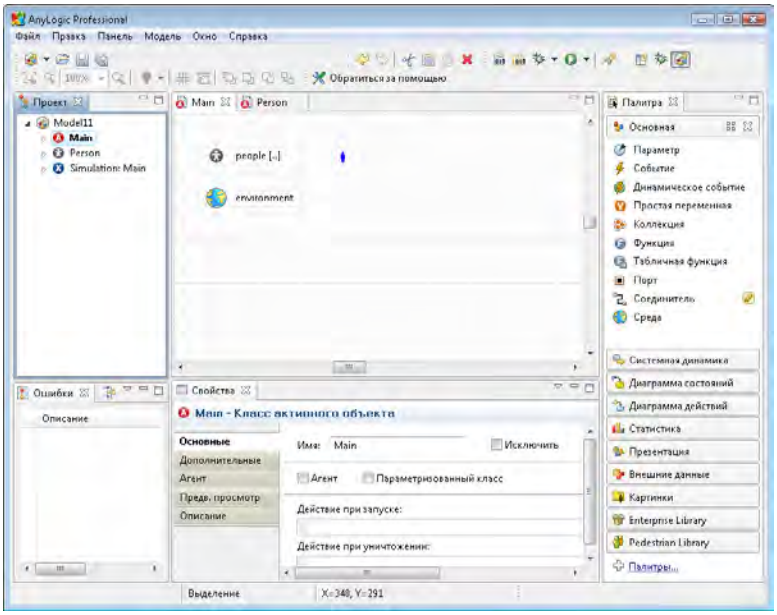


Рис. 6.1. Окно проекта

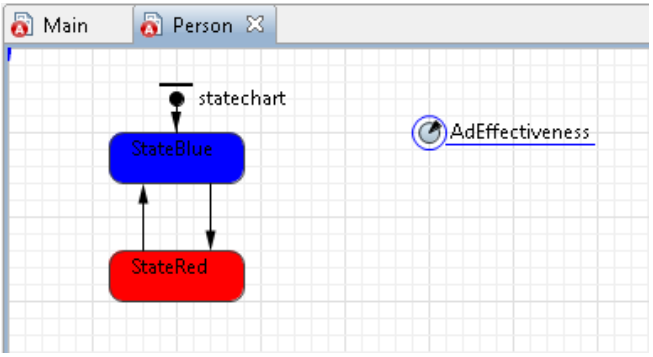
Созданная модель будет содержать классы активных объектов Main и Person. Активный объект Person будет моделировать агентов (людей). Этот класс активного объекта был автоматически объявлен агентом (тем самым он получил доступ к специальной функциональности агента).

Шаг 2. Моделирование продаж под влиянием рекламы

Характеристики модели задаются с помощью параметров. Мы зададим параметры в классе Person, потому что наши агенты задаются экземплярами именно этого класса. Задав значение параметра в классе, мы задаем его для всех агентов одновременно. Но при необходимости можно задать характеристики индивидуально для каждого агента

Откройте диаграмму класса Person, сделав двойной щелчок мышью по элементу Person в панели **Проекты**.

1. Перетащите элемент **Параметр** из палитры **Основная** на диаграмму класса:



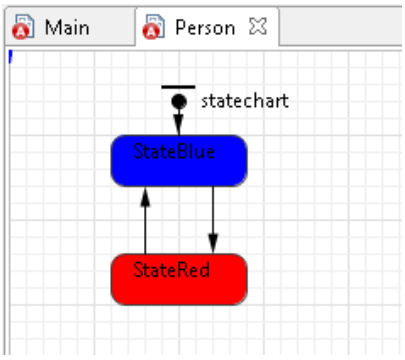
2. При помещении элемента на диаграмму класса, этот элемент будет считаться выбранным, и появится возможность изменять свойства элемента в расположенной в нижней части рабочей области панели **Свойства**. В дальнейшем для изменения свойств элемента нужно будет вначале щелчком мыши выделить его в графическом редакторе или в дереве элементов модели, отображаемом в панели **Проекты**.

3. Перейдите на страницу **Основные** панели **Свойства**, чтобы изменить свойства созданного параметра.

4. Измените имя параметра. Введите AdEffectiveness в поле **Имя**.

5. В поле **Значение по умолчанию** введите 0.01.

6. Откройте диаграмму класса Person, сделав двойной щелчок мышью по элементу Person в панели **Проекты**. На диаграмме класса Вы увидите следующую диаграмму состояний:

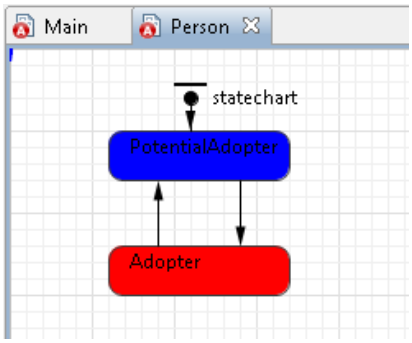


7. Измените имя верхнего состояния на PotentialAdopter (поле **Имя** на странице свойств перехода) Это начальное состояние. Если диаграмма состояний будет находиться в этом состоянии, то это будет означать, что этот человек еще не купил продукт.

8. Следующее состояние назовем Adopter. Если это состояние диаграммы будет активным, это будет означать, что этот человек уже купил продукт.

9. Процесс приобретения продукта человеком моделирует переход, ведущий из верхнего состояния в нижнее (рис.6.2). Нам нужно изменить его свойства, чтобы он срабатывал в нужный нам момент времени.

10. Время, через которое человек купит продукт, экспоненциально зависит от эффективности рекламы продукта. Поскольку время, необходимое человеку, чтобы принять решение о покупке продукта экспоненциально зависит от подверженности этого человека влиянию рекламы, то выберите из выпадающего списка **Происходит** с заданной интенсивностью и введите в поле свойства **Интенсивность** этого перехода имя созданного нами только что параметра AdEffectiveness.



11. Введите AdEffectiveness в расположенном ниже поле **Интенсивность**. Чтобы открыть **Мастер**, щелкните мышью в том месте поля (в нашем случае – поля **Интенсивность**, куда Вы хотите поместить имя, а затем нажмите Ctrl+пробел (при работе на Mac OS: Alt+пробел). Появится окно **Мастера подстановки кода**, перечисляющего переменные модели и функции, доступные в текущем контексте.

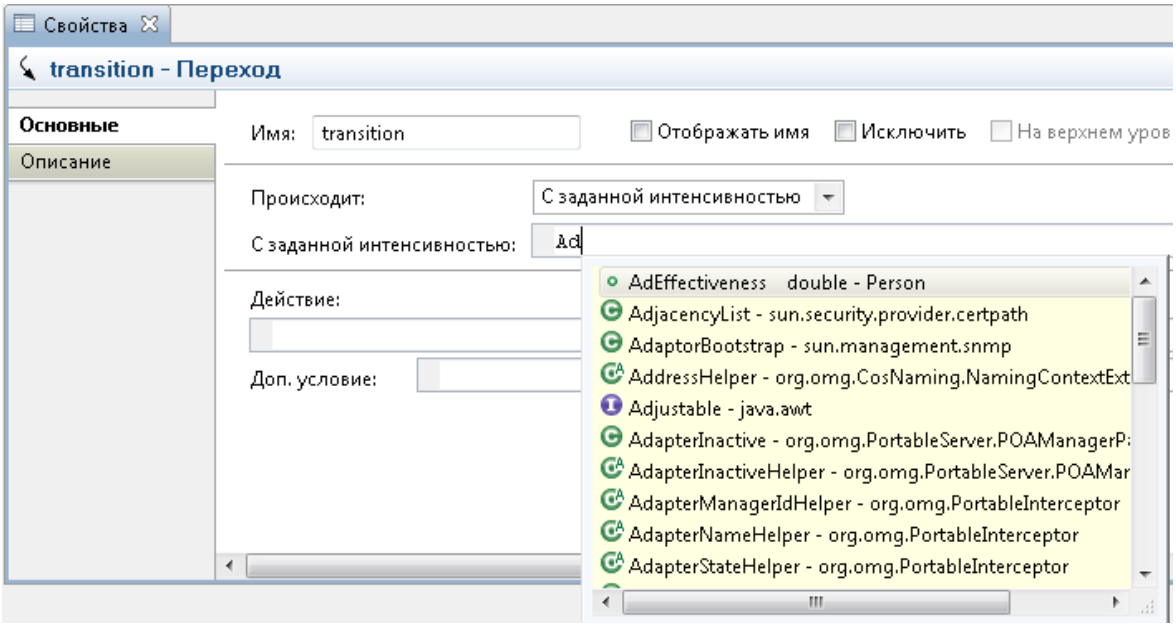
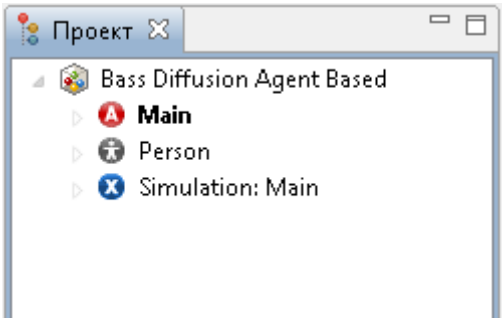


Рис. 6.2. Задание свойств перехода

12. Удалите переход, ведущий из нижнего состояния в верхнее, поскольку мы пока создаем простейшую модель, в которой человек, однажды приобретший продукт, навсегда остается его потребителем, и соответственно перехода из состояния Adopter в состояние PotentialAdopter пока что быть не должно.

Шаг 3. Настройка запуска модели

В панели **Проекты** эксперименты отображаются в нижней части дерева модели. Один эксперимент, названный Simulation, создается по умолчанию. Это простой эксперимент, позволяющий запускать модель с заданными значениями параметров, поддерживающий режимы виртуального и реального времени, анимацию и отладку модели.



Существуют также и другие типы экспериментов (оптимизационный эксперимент, эксперимент для оценки рисков, эксперимент для варьирования параметров), которые используются в тех случаях, когда параметры модели играют существенную роль, и требуется проанализировать, как они влияют на поведение модели, или когда нужно найти оптимальные значения параметров модели.

Под единицей модельного времени примем один год, а процесс распространения продукта в этой модели длится примерно 8 лет.

1. В панели **Проекты**, выделите эксперимент Simulation:Main.
2. На странице **Модельное время** панели **Свойства**, выберите **В заданное время** из выпадающего списка **Остановить** и введите в поле цифру 8.

Шаг 4. Запуск модели

1. Нажмите на кнопку **Построить модель**.
2. Щелкните мышью по кнопке панели инструментов **Запустить** и выберите из открывшегося списка эксперимент, который Вы хотите запустить.

3. Появится окно презентации модели. AnyLogic автоматически помещает на презентацию каждого простого эксперимента заголовок и кнопку, позволяющую запустить модель и перейти на презентацию, нарисованную для главного класса активного объекта этого эксперимента (Main).

4. На презентации можно увидеть моделируемых агентов. Каждый агент отображается своей фигуркой, которая меняет свой цвет в зависимости от того, приобрел ли данный агент рассматриваемый нами продукт или нет. Линиями на презентации будут соединены те агенты, между которыми существуют связи (в данный момент эти связи генерируются случайным образом).

Шаг 5. Подсчет потребителей продукта

Добавим возможность отслеживания того, сколько людей уже купило продукт, а сколько – еще нет.

1. Откройте диаграмму класса Main.
2. Выделите на диаграмме вложенный объект people.
3. Перейдите на страницу **Статистика** панели **Свойства**.
4. Щелкните мышью по кнопке **Добавить функцию сбора статистики**.
5. Введите potentialAadopters (имя функции) в поле **Имя**.
6. Оставьте выбранный по умолчанию **Тип** функции: **Кол-во**.
7. Задайте **Условие**:

item.statechart.isStateActive(item.PotentialAdopter)

Эта функция будет вести подсчет количества агентов, для которых выполняется заданное условие, т.е. тех агентов, которые находятся в текущий момент времени в состоянии PotentialAdopter (являются потенциальными потребителями продукта). Здесь item – это агент (элемент реплицированного объекта people).

8. Аналогично создайте еще одну функцию сбора статистики об агентах, которые уже приобрели продукт.

9. Добавьте временной график, отображающий динамику изменения численностей потребителей и потенциальных потребителей продукта. Для этого откройте диаграмму класса Main, сделав двойной щелчок мышью по элементу Main в панели **Проекты**.

10. Перетащите элемент **Временной график** из палитры **Статистика** на диаграмму класса.

11. Перейдите на страницу **Основные** панели **Свойства** и задайте элементы данных графика:

- введите people.potentialAdopters() в поле **Выражение**.
- введите Potential adopters в поле **Заголовок**. Эта строка будет отображаться в легенде диаграммы для данного элемента данных.

12. Аналогично добавьте еще один элемент данных – количество потребителей продукта, возвращаемое другой нашей статистической функцией: people.adopters(). Задайте Adopters в качестве заголовка этого элемента данных и измените свойства внешнего вида, как и в предыдущем случае **Временной диапазон** (рис.6.3)

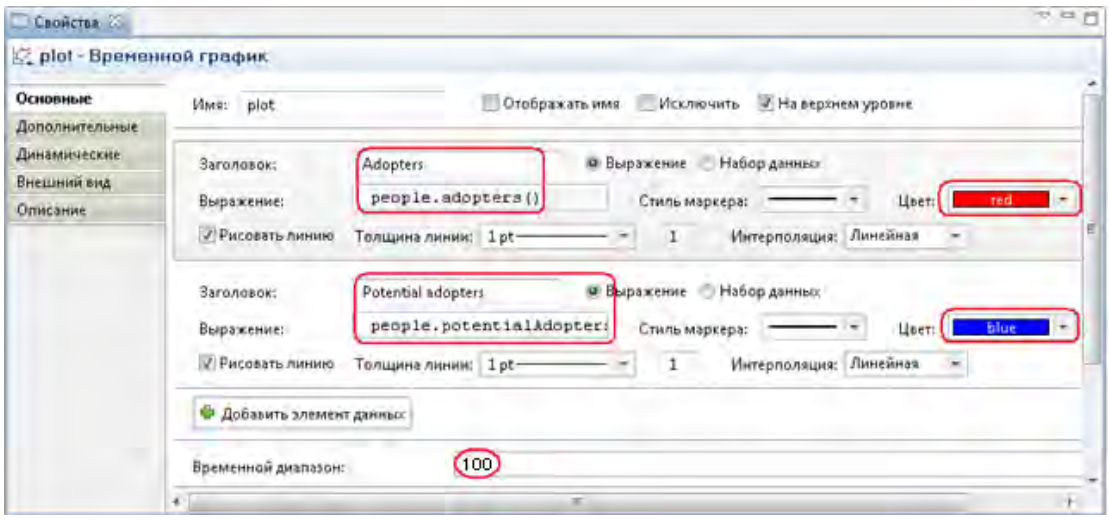


Рис. 6.3 Задание свойств графика

13. Запустите модель. С помощью диаграммы (рис.6.4) можно наблюдать за динамикой моделируемого процесса.

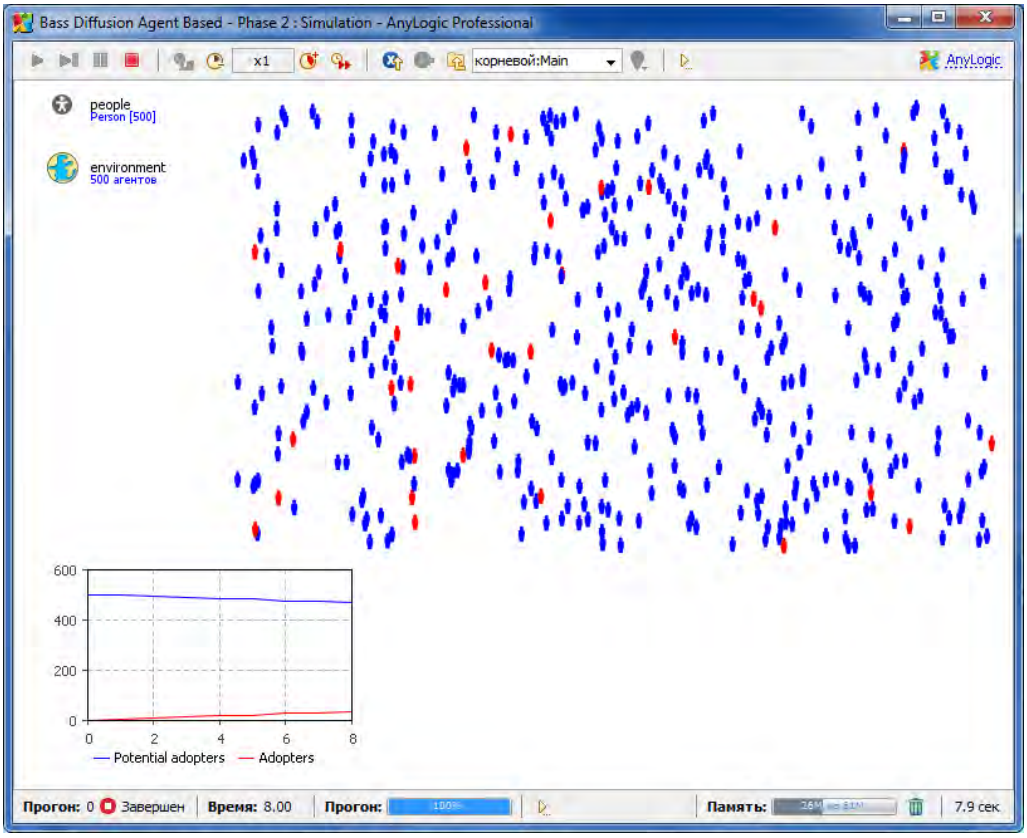


Рис. 6.4. Отображение моделируемого процесса

Контрольные вопросы:

1. Что такое агент?
2. Назовите основные средства описания поведения объектов. Каковы их особенности?
3. Опишите основные возможности пакета AnyLogic 6.
4. Что такое внешний интерфейс модели и какие возможности для их создания предлагает AnyLogic?
5. Какие средства визуализации работы модели предлагает AnyLogic?

6. Какой общий алгоритм реализован в AnyLogic для создания агентных моделей?

7. Какие есть возможности для организации пространства для "жизнедеятельности" агентов в модели и для чего это используется?

8. Чем отличаются непрерывное и дискретное модельные пространства?

9. Что такое «Активный объект» в AnyLogic и как они создаются?

10. Что такое объект «Environment» (окружающая среда агентов), как и зачем он включается в модель?

Как в AnyLogic можно задавать и управлять движением агентов?

ГЛАВА 8 ЗАДАНИЯ НА ЛАБОРАТОРНЫЕ РАБОТЫ

8.1 Лабораторная работа №1. Изучение элементов системы имитационного моделирования ANYLOGIC

Цель работы — выработка навыков разработки и исследования имитационных моделей процессов функционирования информационных систем в системе AnyLogic.

Содержание работы:

1. Изучить теоретические основы.

2. Получить задание у преподавателя.

3. Разработать модель в среде AnyLogic в соответствии с индивидуальным заданием.

4. Подготовить протокол и отчитать лабораторную работу.

Контрольные вопросы к лабораторной работе №1

1. Перечислите этапы разработки математических моделей.

2. Сформулируйте особенности построения моделей имитационного моделирования в среде AnyLogic.

3. Какие элементы системной динамики поддерживает среда AnyLogic?

4. Какие типы экспериментов поддерживаются в системе моделирования AnyLogic?

5. Что встроено в «имитационный исполнитель» AnyLogic?

8.2 Лабораторная работа №2. Моделирование систем массового Обслуживания (СМО)

Цель работы — выработка навыков моделирование СМО аналитическими методами теории массового обслуживания.

Содержание работы:

1. Изучить: подходы к моделированию случайных величин; модель многоканальной СМО; модель многоканальной СМО с отказами; модель

одноканальной СМО с ограниченной и неограниченной очередью; модель СМО с ограниченным временем ожидания.

2. Получить задание у преподавателя.

3. Построить модель в среде имитационного моделирования.

3. Подготовить протокол и отчитать лабораторную работу.

Контрольные вопросы к лабораторной работе №2

1. Из каких основных элементов состоит система массового обслуживания? Какие процессы в ней происходят?

2. Чем отличаются СМО от систем массового сервиса?

3. Дайте классификацию СМО и их обозначение.

4. Дайте характеристику процессов в системах обслуживания.

5. Изобразите графически зависимость количества времени обслуживания от времени.

6. Что такое период занятости и свободы системы обслуживания?

7. Изобразите пример зависимости числа клиентов в системе от времени. Как она связана с количеством времени, необходимым для освобождения системы обслуживания?

8. Какие основные показатели используются для характеристики СМО?

9. Чем интенсивность поступления клиентов в систему отличается от интенсивности поступления клиентов в очередь?

10. Сформулируйте основные принципы оптимизации систем обслуживания.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Бусленко, Н. П. Моделирование сложных систем / Н. П. Бусленко. – М. : Наука, 1978.

2. Советов, Б. Я. Моделирование систем / Б. Я. Советов, С. А. Яковлев. – М. : Высшая школа, 2007.

3. Колесов, Ю. Б. Объектно-ориентированное моделирование сложных динамических систем / Ю. Б. Колесов. – СПб. : Изд-во СПбГПУ, 2004. – 240 с.

4. Колесов, Ю. Б. Моделирование систем. Динамические и гибридные системы / Ю. Б. Колесов, Ю. Б. Сениченков. – СПб. : БХВ-Петербург, 2006. – 224 с.

5. Шеннон, Р. Имитационное моделирование систем – искусство и наука / Р. Шеннон. – М. : Мир, 1978. – 420 с.

6. Лычкина, Н. Н. Имитационное моделирование экономических процессов. Учебное пособие для слушателей программы eMBI / Н. Н. Лычкина. – Академия АйТи, 2005. – 164 с.

7. Сениченков, Ю. Б. Школа моделирования 2003. Урок 2. Моделирование движения объекта в среде MVS / Ю. Б. Сениченков, Н. В. Макарова, Ю. Ф. Титова // Компьютерные инструменты в образовании. – 2003. – №2. – С. 75-83.

8. Борщёв, А. В. От системной динамики и традиционного ИМ – к практическим агентным моделям: причины, технология, инструменты [Электронный ресурс] / А. В. Борщёв. – Режим доступа : http://www.gpss.ru/paper/borshevarc.pdf.

9. Forrester, J. Urban Dynamics / J. Forrester. – Portland : MIT Press, Productivity Press, 1969. – 287 p.

10. Sterman, J. Business Dynamics – Systems Thinking and Modeling for a Complex World / J. Sterman. – McGraw-Hill Higher Education, 2000.

11. Wooldridge, M. Agent Theories, Architectures, and Languages: A Survey / M. Wooldridge, N. R. Jennings // Intelligent Agents: Proceedings of the ECAI-94 Workshop on Agent Theories, Architecture and Languages. Amsterdam, The Netherlands, August 8-9, 1994, (Eds. M.J.Wooldridge and N.R.Jennings). – Springer Verlag, 1994. – P. 3-39.

12. Городецкий, В. И. Многоагентные системы (обзор) / В. И. Городецкий, М. С. Грушинский, А. В. Хабалов // Новости искусственного интеллекта. – 1998. – №2. – С. 64-116.

13. Wooldridge, M. Towards a Theory of Cooperative Problem Solving / M. Wooldridge,

N. Jennings. – 1994. – 152 p.

14. Wooldridge, M. Agent Theories, Architectures, and Languages: A Survey / M. Wooldridge, N. R. Jennings // Intelligent Agents: Proceedings of the ECAI-94 Workshop on Agent Theories, Architecture and Languages. Amsterdam, The Netherlands, August 8-9, 1994, (Eds. M.J.Wooldridge and N.R.Jennings). – Springer Verlag, 1994. – P. 3-39.

15. Zlotkin, J. S. Rosenschtein, Mechanisms for Automated Negotiation in State Oriented Domain / J. S. Zlotkin // Journal of Artificial Intelligence Research. – 1996. – No. 5. – P. 34-48.

16. Варшавский, В. И. Оркестр играет без дирижера / В. И. Варшавский, Д. А. Поспелов. – М : Наука, 1984.

17. Дрейпер, Н. Прикладной регрессионный анализ : в 2-х т. / Н. Дрейпер, Г. Смит. – Москва : Финансы и статистика, 1987. – Т. 2 – 185 с.

18. Петрович, М. Л. Регрессионный анализ и его математическое обеспечение на ЕС ЭВМ: Практическое руководство / М. Л. Петрович. – Москва : Финансы и статистика, 1982. – 199 с.

19. Ивахненко, А. Г. Индуктивный метод самоорганизации моделей сложных систем / А. Г. Ивахненко. – К.: Наук. думка, 1982. – 360 с.

20. Шахиди, А. Введение в анализ ассоциативных правил [Электронный ресурс] / А. Шахиди. – 2015. – Режим доступа : http://www.basegroup.ru/library/analysis/association_rules/intro/.

21. Чубукова, И. А. Data Mining: учебное пособие / И. А. Чубукова – М.: Интернет-университет информационных технологий: БИНОМ: Лаборатория знаний, 2006. – 382 с.

22. Орешков, В. Поиск последовательных шаблонов [Электронный ресурс] / В. Орешков. – 2012. – Режим доступа : http://www.basegroup.ru/library/analysis/association_rules/sequential_patterns_1/

23. Zadeh, L. Fuzzy sets / L. Zadeh // Information and Control. – 1965. – №8. – P. 338-353.

24. Заде, Л. Понятие лингвистической переменной и ее применение к принятию приближенных решений / Л. Заде. – М.: Мир, 1976. – 167 с.

25. Штовба, С. Д. Введение в теорию нечетких множеств и нечеткую логику [Электронный ресурс] / С. Д. Штовба. – 2014. – Режим доступа : http://matlab.exponenta.ru/fuzzylogic/book1/

26. BaseGroup Labs. Нечеткая логика – математические основы [Электронный ресурс]. – 2017. – Режим доступа : https://basegroup.ru/community/articles/fuzzylogic-math#comments

27. Zadeh, L. Fuzzy Logic, Neural Networks, and Soft Computing / L. Zadeh // Communications of the ACM, March 1994. – P. 77-84.

28. Леоненков, А. Нечеткое моделирование в среде MATLAB и fuzzyTech / А. Леоненков. – СПб. : БХВ-Петербург, 2003. – 736 с.

29. Толковый словарь по искусственному интеллекту [Электронный ресурс]. – 2017. – Режим доступа : http://www.raai.org/library/tolk/aivoc.html#L306

30. Что такое концептуальная модель [Электронный ресурс]. – 2017. – Режим доступа : http://fb.ru/article/144365/chto-takoe-kontseptualnaya-model

31. Классификационные признаки и классификация моделей [Электронный ресурс]. – 2017. – Режим доступа : http://sernam.ru/book_mm.php?id=8

32. Concept map [Электронный ресурс] / Википедия – электронная энциклопедия. – 2017. – Режим доступа : http://en.wikipedia.org/wiki/Concept_map

33. Муромцев, Д. И. Концептуальное моделирование знаний в системе Concept Map. / Д. И. Муромцев. – СПб. : СПб ГУ ИТМО, 2009. – 83 с.

34. Концептуальные карты [Электронный ресурс] / Intel. – 2017. – Режим доступа : http://www.intel.ru/content/dam/www/program/education/emea/ru/ru/documents/project-design1/instructional-strategies/graphic-organizers/concept-maps.pdf

35. Axelrod, R. Structure of Decision: the Cognitive Maps of Political Elites / R. Axelrod. – N.Y. : Priston Univ. Press, 1976. – 246 p.

36. Авдеева, З. К. Когнитивное моделирование для решения задач управления слабоструктурированными системами (ситуациями) / З. К. Авдеева, С. В. Коврига, Д. И. Макаренко // Управление большими системами: сборник трудов. – 2007. – № 16. – С. 26-39.

37. Tolman, E. C. Cognitive maps in rats and men (англ.) / E. C. Tolman // Psychological Review. – 1948. – Vol. 55, no. 4. – P. 189-208.

38. Авдеева, З. К. Формирование стратегии развития социально-экономических объектов на основе когнитивных карт / З. К. Авдеева, С. В. Коврига // Saarbrucken: LAP LAMBERT Academic Publishing GmbH & Co. KG, 2011. – 184 с.

39. Eden, C. The Practice of Making Strategy: Step by Step Guide / C. Eden, F. Ackerman, I. Brown. – L. : Stage. 2005. – 460 p.

40. Huff, A. S. (1990). Mapping strategic thought / A. S. Huff // Mapping strategic thought. – Chichester, UK : Wiley, 1990. – P. 11-49.

41. Kosko, B. Fuzzy cognitive maps / B. Kosko // International Journal of Man-Machine Studies, 1986. – Vol. 1. – P. 65–75.

42. Pena, A. Cognitive maps: an overview and their application for student modeling / A. Pena, H. Sossa, A. Gutiérrez // J. Comp. y Sist.. – 2007. – Vol.10, no. 3. – P. 45-52.

43. Chaib-draa, B. Causal maps: theory, implementation, and practical applications in multiagent environments / B. Chaib-draa // J. IEEE Trans. on Knowledge and Data Engineering. – 2002. – Vol.14, no. 6. – P. 26-39.

44. Робертс, Ф. С. Дискретные математические модели с приложением к социальным, биологическим и экологическим задачам / Ф. С. Робертс. – М. : Наука, 1986. – 280 с.

45. Карпов, Ю. Г. Имитационное моделирование систем. Введение в моделирование с AnyLogic 5 / Ю. Г. Карпов. – СПб. : БХВ-Петербург, 2009. – 400 с.

46. Маликов, Р. Ф. Практикум по имитационному моделированию сложных систем в среде AnyLogic : учеб. пособие / Р. Ф. Маликов. – Уфа: Изд-во БГПУ, 2013. – 296 с.

47. Киселева, М. В. Имитационное моделирование систем в среде AnyLogic : учеб.-метод. пособие / М. В. Киселёва. – Екатеринбург : УГТУ-УПИ, 2009. – 88 с.

Учебное издание

Наталья Петровна **Садовникова**
Данила Сергеевич **Парыгин**
Татьяна Владимировна **Ерещенко**
Николай Михайлович **Рашевский**

ИМИТАЦИОННОЕ МОДЕЛИРОВАНИЕ

Учебное пособие

Редактор *Л. Н. Рыжих*

Темплан 2021 г. (учебники и учебные пособия). Поз. № 19.
Подписано в печать 00.00.2021. Формат 60x84 1/16. Бумага газетная.
Гарнитура Times. Печать офсетная. Усл. печ. л. 4,0. Уч.-изд. л. 5,05.
Тираж 100 экз. Заказ

Волгоградский государственный технический университет.
400005, г. Волгоград, просп. В. И. Ленина, 28, корп. 1.

Отпечатано в типографии ИУНЛ ВолгГТУ.
400005, г. Волгоград, просп. В. И. Ленина, 28, корп. 7.