МИНИСТЕРСТВО СЕЛЬСКОГО ХОЗЯЙСТВА
И ПРОДОВОЛЬСТВИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

Учреждение образования
«БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ
АГРАРНЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

Н. Г. Серебрякова, О. Л. Сапун, А. П. Мириленко

ОСНОВЫ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ

*Рекомендовано Учебно-методическим объединением
по аграрному техническому образованию
в качестве пособия для студентов
учреждений высшего образования по специальностям
7-06-0812-01 «Техническое обеспечение производства
сельскохозяйственной продукции»,
7-06-0812-02 «Техническое обеспечение
хранения и переработки сельскохозяйственной продукции»,
7-06-0812-03 «Технический сервис в агропромышленном комплексе»,
7-06-0812-04 «Энергетическое обеспечение сельского хозяйства»,
7-06-0812-05 «Проектирование и производство
сельскохозяйственной техники»,
7-06-1021-01 «Охрана труда и эргономика»*

Минск
БГАТУ
2025

УДК 311(075)
ББК 60.6я7
С32

Рецензенты:
кафедра «Техническая эксплуатация автомобилей»
Белорусского национального технического университета
(кандидат технических наук, доцент,
заведующий кафедрой *А. С. Гурский*);
главный инженер УП «Митоком» *А. Л. Николаев*;
кандидат технических наук, доцент, доцент кафедры
«Программное обеспечение информационных систем и технологий»
Белорусского национального технического университета *Н. Н. Гурский*

Серебрякова, Н. Г.

С32 Основы информационных технологий : пособие / Н. Г. Серебрякова, О. Л. Сапун, А. П. Мириленко. – Минск : БГАТУ, 2025. – 268 с.
ISBN 978-985-25-0316-7.

Изложены основные понятия, принципы и инструменты современных информационных технологий. Рассмотрены актуальные данные о технологиях: история, архитектура и перспективы развития вычислительной техники, классификация и свойства информации, операционные системы, языки и технологии программирования. Приводятся сведения по основным программным средствам: текстовым и графическим редакторам, электронным таблицам и др.

Пособие ориентировано на магистрантов специальностей 7-06-0812-01 «Техническое обеспечение производства сельскохозяйственной продукции», 7-06-0812-02 «Техническое обеспечение хранения и переработки сельскохозяйственной продукции», 7-06-0812-03 «Технический сервис в агропромышленном комплексе», 7-06-0812-04 «Энергетическое обеспечение сельского хозяйства», 7-06-0812-05 «Проектирование и производство сельскохозяйственной техники», 7-06-1021-01 «Охрана труда и эргономика».

УДК 311(075)
ББК 60.6я7

ISBN 978-985-25-0316-7

СОДЕРЖАНИЕ

1. СОВРЕМЕННЫЕ ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ

1.1. История, современное состояние и перспективы развития ИТ-технологий

Вычислительная техника играет ключевую роль в современном мире, оказывая влияние на все аспекты нашей жизни. От первых механических калькуляторов до современных суперкомпьютеров и искусственного интеллекта, эволюция вычислительной техники была стремительной и многогранной. В этом учебном пособии мы рассмотрим историю развития вычислительной техники, ее современное состояние и перспективы на будущее.

1.1.1. История вычислительной техники

Ранние вычислительные устройства

История вычислительной техники начинается с древних времен. Первые вычислительные устройства, такие как абак и антикитерский механизм, были созданы для выполнения простых арифметических операций. Абак, изобретенный в Вавилоне около 2700 г. до н. э., является одним из самых древних вычислительных устройств. Антикитерский механизм, найденный в Греции и датируемый II веком до н. э., считается первым известным аналоговым компьютером, используемым для предсказания астрономических событий и явлений.

В XVII веке Блез Паскаль создал механический калькулятор, который мог выполнять сложение и вычитание. Это устройство, известное как «Паскалина», было разработано для помощи его отцу, который был сборщиком налогов. В XVIII веке Готфрид Вильгельм Лейбниц усовершенствовал эту идею, создав калькулятор, способный также выполнять умножение и деление. Лейбниц также разработал двоичную систему счисления, которая стала основой для современных цифровых компьютеров.

Первые компьютеры

Начало XIX века ознаменовалось появлением аналитической машины Чарльза Бэббиджа, которая считается предшественником современных компьютеров. Бэббидж разработал концепцию программируемого компьютера, который мог выполнять различные вычисления на основе введенных данных и программ. Хотя

аналитическая машина так и не была полностью построена при жизни Бэббиджа, его идеи оказали значительное влияние на последующие разработки.

Однако первые настоящие компьютеры появились только в середине XX века. В 1943 году был создан ENIAC (Electronic Numerical Integrator and Computer), первый полностью электронный цифровой компьютер. ENIAC занимал огромную площадь и потреблял большое количество энергии, но он стал важным шагом в развитии вычислительной техники. ENIAC использовался для выполнения баллистических расчетов во время Второй мировой войны.

Появление транзисторов и микропроцессоров

В 1947 году была изобретена транзисторная технология, что позволило создавать более компактные и энергоэффективные компьютеры. Транзисторы заменили вакуумные лампы, которые использовались в ранних компьютерах, таких как ENIAC. Это привело к значительному уменьшению размеров и увеличению надежности вычислительных устройств.

В 1971 году компания Intel выпустила первый микропроцессор, что стало еще одним важным шагом в развитии вычислительной техники. Микропроцессоры позволили создавать персональные компьютеры, которые стали доступны для широкой аудитории. Первый микропроцессор, Intel 4004, содержал 2300 транзисторов и мог выполнять до 60 000 операций в секунду.

Большой вклад в развитие вычислительной техники в СССР внесли советские ученые С. А. Лебедев и И. С. Брук. Под их руководством были созданы ЭВМ первого поколения МЭСМ – 1951 г. (малая электронная счетная машина), БЭСМ – 1952–1953 гг. (большая электронная счетная машина). К машинам первого поколения можно отнести МЭСМ, БЭСМ, М-1, М-2, М-3, «Стрела», «Минск-1», «Урал-1», «Урал-2», «Урал-3»,М-20, «Сетунь», БЭСМ-2, «Раздан». ЭВМ «Сетунь» была построена в 1953 году Н. П. Брусенцовым.

1.1.2. Современное состояние вычислительной техники

Персональные компьютеры и ноутбуки

Современные персональные компьютеры и ноутбуки стали неотъемлемой частью нашей повседневной жизни. Они используются для работы, учебы, развлечений и общения. Современные ПК оснаще-

ны мощными процессорами, большим объемом оперативной памяти и высокоскоростными накопителями, что позволяет выполнять сложные вычислительные задачи. Например, современные игровые компьютеры могут обрабатывать сложную графику и выполнять миллионы операций в секунду, обеспечивая плавный игровой процесс.

Серверы и дата-центры

Серверы и дата-центры играют ключевую роль в обеспечении работы интернета и облачных сервисов. Они хранят и обрабатывают огромные объемы данных, обеспечивая доступ к информации и услугам в любой точке мира. Современные дата-центры используют передовые технологии для повышения энергоэффективности и надежности. Например, компания Google использует специальные системы охлаждения и возобновляемые источники энергии для своих дата-центров, что позволяет снизить энергопотребление и уменьшить выбросы углекислого газа.

Суперкомпьютеры

Суперкомпьютеры используются для выполнения сложных научных и инженерных задач, таких как моделирование климата, разработка новых материалов и исследование космоса. Современные суперкомпьютеры способны выполнять триллионы операций в секунду, что делает их незаменимыми инструментами для научных исследований. Например, суперкомпьютер IBM Summit, разработанный для Национальной лаборатории Ок-Ридж, используется для моделирования молекулярных процессов и разработки новых лекарств.

Суперкомпьютеры «Яндекса»: «Червоненкис», «Галушкин» и «Ляпунов» занимают в мировом рейтинге 42, 69 и 79 места. Они оснащены процессорами AMD EPYC 7xxx и GPU NVIDIA A100, они обеспечивают производительность 21,5, 16 и 12,8 петафлопса соответственно.

Мобильные устройства

Мобильные устройства, такие как смартфоны и планшеты, стали важной частью нашей повседневной жизни. Они оснащены мощными процессорами, большими экранами и различными сенсорами, что позволяет выполнять широкий спектр задач. Современные смартфоны могут выполнять сложные вычисления, обрабатывать видео и изображения, а также поддерживать различные приложения и сервисы. Например, iPhone от компании Apple использует передовые технологии машинного обучения для улучшения качества фотографий и видео.

Облачные вычисления

Облачные вычисления стали важной частью современной вычислительной техники. Они позволяют пользователям и организациям арендовать вычислительные ресурсы по мере необходимости, что снижает затраты на оборудование и обслуживание. Облачные сервисы, такие как Amazon Web Services (AWS), Microsoft Azure и Google Cloud, предоставляют широкий спектр услуг, включая хранение данных, вычисления и машинное обучение. Например, компания Netflix использует облачные сервисы для хранения и обработки видеоконтента, что позволяет обеспечить высокое качество обслуживания для миллионов пользователей по всему миру.

1.1.3. Перспективы развития вычислительной техники

Квантовые компьютеры

Квантовые компьютеры представляют собой следующий большой шаг в развитии вычислительной техники. Они используют принципы квантовой механики для выполнения вычислений, что позволяет решать задачи, недоступные для классических компьютеров. Квантовые компьютеры могут значительно ускорить разработку новых лекарств, оптимизацию логистических цепочек и моделирование сложных физических систем. Например, компания IBM разрабатывает квантовые процессоры, которые могут выполнять миллионы квантовых операций в секунду.

Искусственный интеллект и машинное обучение

Искусственный интеллект (ИИ) и машинное обучение (МО) становятся все более важными в различных областях – от медицины до финансов. Современные системы ИИ способны анализировать большие объемы данных, распознавать образы и речь, а также принимать решения на основе обучения. Развитие ИИ и МО открывает новые возможности для автоматизации процессов и создания интеллектуальных систем. Например, компания DeepMind, принадлежащая Google, разработала систему AlphaGo, которая смогла победить чемпиона мира по игре го, что считалось невозможным для компьютеров.

Интернет вещей (IoT)

Интернет вещей (IoT) представляет собой сеть физических устройств, подключенных к интернету, которые могут собирать и обмениваться данными. IoT используется в различных областях, таких как умные дома, промышленность и здравоохранение. Развитие IoT позволяет создавать более эффективные и удобные

системы управления, а также открывает новые возможности для сбора и анализа данных. Например, умные дома могут использовать IoT-устройства для управления освещением, отоплением и безопасностью, что позволяет снизить энергопотребление и повысить комфорт.

Виртуальная и дополненная реальность

Виртуальная и дополненная реальность (VR и AR) становятся все более популярными в различных областях, таких как игры, образование и медицина. VR и AR позволяют создавать иммерсивные и интерактивные опыты, которые могут использоваться для обучения, тренировок и развлечений. Например, компания Oculus, принадлежащая Facebook, разрабатывает VR-шлемы, которые могут использоваться для виртуальных туров, симуляций и игр.

Блокчейн и криптовалюты

Блокчейн и криптовалюты представляют собой новые технологии, которые могут изменить различные аспекты нашей жизни. Блокчейн позволяет создавать децентрализованные системы, которые обеспечивают высокую степень безопасности и прозрачности. Криптовалюты, такие как Bitcoin и Ethereum, используют блокчейн для обеспечения безопасных и анонимных транзакций. Например, компания Ethereum разрабатывает платформу для содания смарт-контрактов, которые могут использоваться для автоматизации различных процессов, таких как финансовые транзакции и управление цепочками поставок.

1.2. Элементная база, архитектура, сетевая компоновка, производительность

1.2.1. Элементная база вычислительной техники

Элементная база вычислительной техники включает в себя основные компоненты, которые обеспечивают функционирование вычислительных систем. К таким компонентам относятся процессоры, память, накопители данных, ввод-вывод устройства и системы охлаждения.

Процессоры

Процессор (центральный процессор, ЦП) является основным вычислительным устройством в компьютере. Современные процессоры состоят из миллиардов транзисторов и могут выполнять

миллиарды операций в секунду. Основные характеристики процессоров включают тактовую частоту, количество ядер и архитектуру. Например, процессоры Intel и AMD используют различные архитектуры, такие как x86 и ARM, которые определяют их производительность и энергоэффективность.

Память

Память делится на оперативную память (RAM) и постоянную память (ROM). Оперативная память используется для временного хранения данных и программ, которые выполняются процессором. Постоянная память содержит системные данные и программы, которые необходимы для загрузки и работы операционной системы. Современные системы памяти используют технологии DDR (Double Data Rate) для повышения скорости передачи данных.

Накопители данных

Накопители данных используются для долговременного хранения информации. Существуют различные типы накопителей, такие как жесткие диски (HDD), твердотельные накопители (SSD) и оптические диски. SSD накопители стали популярными благодаря своей высокой скорости чтения и записи данных, а также надежности.

Устройства ввода и вывода

Устройства ввода и вывода (I/O) обеспечивают взаимодействие пользователя с вычислительной системой. К таким устройствам относятся клавиатура, мышь, монитор, принтер и другие периферийные устройства. Современные системы используют интерфейсы USB, HDMI и другие для подключения I/O устройств.

Системы охлаждения

Системы охлаждения необходимы для поддержания оптимальной температуры работы компонентов вычислительной системы. Существуют различные методы охлаждения, такие как воздушное охлаждение, жидкостное охлаждение и пассивное охлаждение. Эффективное охлаждение обеспечивает стабильную работу системы и предотвращает перегрев компонентов.

Архитектура вычислительных систем

Архитектура вычислительных систем определяет организацию и взаимодействие их компонентов. Существуют различные архитектурные подходы, такие как архитектура фон Неймана (рис. 1.1), гарвардская архитектура и архитектура с параллельной обработкой данных.

Архитектура фон Неймана

Рис. 1.1. Архитектура фон Неймана

Архитектура фон Неймана является основой для большинства современных компьютеров. Она включает в себя процессор, память, ввод-вывод устройства и системные шины, которые обеспечивают передачу данных между компонентами. Основное преимущество архитектуры фон Неймана заключается в ее универсальности и гибкости.

Гарвардская архитектура

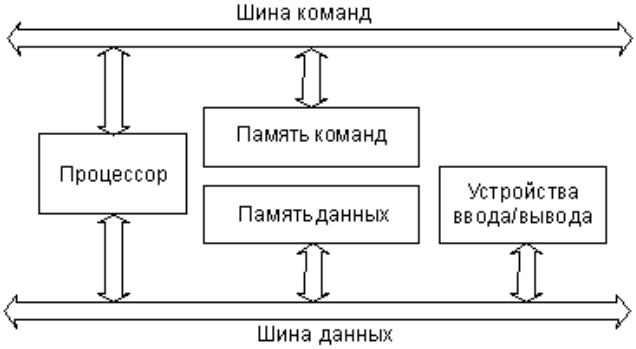


Рис. 1.2. Гарвардская архитектура

Гарвардская архитектура (рис. 1.2) отличается от архитектуры фон Неймана тем, что использует раздельные шины для памяти данных и памяти команд. Это позволяет увеличить скорость выполнения программ, так как данные и команды могут передаваться одновременно. Гарвардская архитектура используется в некоторых встроенных системах и микроконтроллерах.

Архитектура с параллельной обработкой данных

Архитектура с параллельной обработкой данных использует несколько процессоров или ядер для выполнения вычислений одновременно. Это позволяет значительно увеличить производительность системы при выполнении сложных задач, таких как обработка больших данных и моделирование. Примером такой архитектуры являются суперкомпьютеры и графические процессоры (GPU).

1.2.2. Сетевая компоновка

Сетевая компоновка определяет организацию и взаимодействие вычислительных систем в сети. Существуют различные топологии сетей, такие как звезда, кольцо, шина и сетка (рис. 1.3).

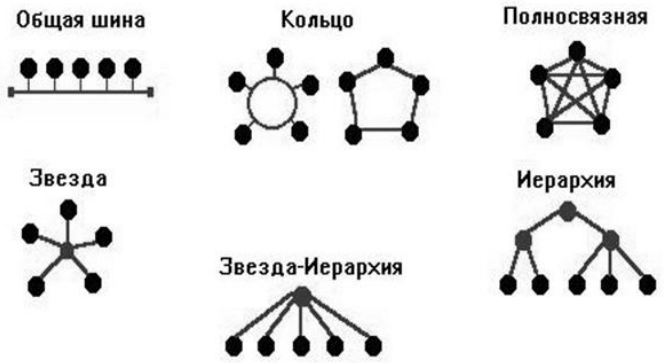


Рис. 1.3. Сетевая компоновка

Топология звезда

Топология звезда является наиболее распространенной топологией сети. В такой сети все устройства подключены к центральному коммутатору или маршрутизатору, который управляет передачей

данных между устройствами. Преимущество топологии звезда заключается в ее простоте и надежности, так как отказ одного устройства не влияет на работу других устройств в сети.

Топология кольцо

Топология кольцо используется в сетях, где все устройства подключены последовательно, образуя замкнутый контур. Данные передаются от одного устройства к другому по кольцу. Преимущество топологии кольцо заключается в ее простоте и эффективности при передаче данных на большие расстояния. Однако отказ одного устройства может привести к нарушению работы всей сети.

Топология шина

Топология шина используется в сетях, где все устройства подключены к общей шине передачи данных. Данные передаются от одного устройства к другому через шину. Преимущество топологии шина заключается в ее простоте и низкой стоимости. Однако при большом количестве устройств в сети может возникнуть проблема коллизий данных.

Топология сетка

Топология сетка используется в сетях, где каждое устройство подключено к нескольким другим устройствам, образуя сетку. Данные могут передаваться по нескольким маршрутам, что повышает надежность и производительность сети. Топология сетка используется в высокопроизводительных вычислительных системах и суперкомпьютерах.

1.2.3. Производительность вычислительных систем

Производительность вычислительных систем определяется их способностью выполнять вычисления и обрабатывать данные. Основные показатели производительности включают скорость выполнения операций, пропускную способность, время отклика и энергоэффективность.

Скорость выполнения операций

Скорость выполнения операций определяется тактовой частотой процессора и количеством операций, которые он может выполнить за единицу времени. Современные процессоры могут выполнять миллиарды операций в секунду, что позволяет им обрабатывать сложные задачи в реальном времени.

Пропускная способность

Пропускная способность определяет объем данных, который может быть передан между компонентами системы за единицу времени. Пропускная способность зависит от шины передачи данных, интерфейсов и сетевых протоколов. Высокая пропускная способность необходима для эффективной работы высокопроизводительных вычислительных систем.

Время отклика

Время отклика определяет задержку между отправкой запроса и получением ответа. Время отклика зависит от производительности процессора, скорости передачи данных и эффективности алгоритмов обработки данных. Низкое время отклика необходимо для интерактивных приложений и систем реального времени.

Энергоэффективность

Энергоэффективность определяет соотношение производительности системы и ее энергопотребления. Современные вычислительные системы используют различные технологии для повышения энергоэффективности, такие как динамическое управление частотой и напряжением, использование энергоэффективных компонентов и оптимизация алгоритмов.

Элементная база, архитектура, сетевая компоновка и производительность являются ключевыми аспектами вычислительной техники. Понимание этих аспектов позволяет разрабатывать эффективные и надежные вычислительные системы, которые могут решать сложные задачи и обеспечивать высокую производительность. В будущем можно ожидать дальнейшего развития этих технологий, что откроет новые возможности для научных исследований и практических применений.

1.3. Понятие информации. Источники информации. Инструментарий, классификация и виды информационных технологий

1.3.1. Понятие информации

Емкое понятие информационных технологий формируется двумя не менее емкими понятиями технология и информация. Термин технология (от др.- греч. λόγος – способ производства) – совокупность методов, процессов и материалов, используемых в какой-либо

отрасли деятельности, а также научное описание способов технического производства с применением современных методов и средств. Понятие информация (от лат. informatio – «разъяснение, осведомленность») – сведения о чем-либо, независимо от формы их представления.

В настоящее время не существует единого определения информации как научного термина. С точки зрения различных областей знания данное понятие описывается своим специфическим набором признаков. Понятие «информация» является базовым в курсе информатики, где невозможно дать его определение через другие, более «простые» понятия. В научном подходе выявлении наиболее характерных определений понятия информации привело к двум трактовкам понятия информации.

Во-первых, оно было расширено и включило обмен сведениями не только между человеком и человеком, но и между человеком и автоматом, автоматом и автоматом; обмен сигналами в животном и растительном мире. Передачу признаков от клетки к клетке и от организма к организму тоже стали рассматривать как передачу информации.

Во-вторых, была предложена количественная мера информации в работах К. Шеннона и А. Н. Колмогорова, что привело к созданию теории информации.

Согласно концепции К. Шеннона, информация – это снятая неопределенность, т. е. сообщения, которые должны снять в той или иной степени существующую у потребителя до их получения неопределенность, расширить его понимание объекта полезными сведениями. В противном случае эти сообщения не представляет никакой ценности.

Американский инженер Р. Хартли в 1928 году предложил формулу для определения количества информации *I,* содержащейся в сообщении, выбранном из множества *N* равновероятных сообщений:

$$I = log_2 N. \qquad (1.1)$$

Клод Шеннон развил идею Хартли для оценки количества информации при неодинаковой вероятности сообщений в наборе. Формула Шеннона имеет вид:

$$I = \sum_{i=1}^{n} \left(p_i \log_2 p_i \right), \qquad (1.2)$$

где p_i – вероятность появления i-го события.

Понятие информации обязательно объединяет двух участников – ее источник и приемник. Для существования информации необходим также канал приема-передачи информации. При взаимодействии этих трех компонентов появляется и проявляется информация.

1.3.2. Прием и передача информации

По способу передачи и приема различают информацию, передаваемую:

– визуально;

– звуками – аудиальную;

– ощущениями – тактильную;

– запахами и вкусами – органолептическую;

– информацию, передаваемую и воспринимаемую средствами вычислительной техники, – машинную.

Информацию, создаваемую и используемую человеком, по общественному назначению делят на три вида: личную, массовую, специальную.

В зависимости от области знаний различают научную, техническую, производственную, правовую, патентную и иную информацию. Каждый вид информации имеет свои особые смысловые нагрузки и ценности, свои требования к ее точности и достоверности, преимущественные технологии обработки, формы представления и носители.

1.3.3. Формы представления информации

– символьная (основанная на использовании символов: букв, цифр, знаков);

– текстовая (использует тексты, т. е. символы, расположенные в определенном порядке);

– графическая (различные виды изображений);

– звуковая.

Символьная форма – является наиболее простой, но практически она применяется только для передачи несложных сигналов о различных событиях. Примером может служить разный свет светофора, сообщающий о возможности, начала или прекращения движения пешеходам или водителям автотранспорта.

Текстовая форма также использует различные символы: буквы, цифры, математические, но информация заложена не только в этих символах, но и в их сочетании, в порядке следования.

Графическая форма представления информации является наиболее сложной. Сюда относятся фотографии, схемы, рисунки, чертежи, имеющие большое значение в деятельности человека.

Звуковая – устная или в виде записи и передачи лексем языка аудиальным путем.

1.3.4. Основные характеристики информации

1. Релевантность: информация должна быть актуальной и полезной для конкретной цели или задачи.

2. Точность: информация должна быть корректной и свободной от ошибок.

3. Своевременность: информация должна быть доступна в нужное время для принятия решений.

4. Полнота: информация должна быть достаточно полной для понимания и использования.

5. Доступность: информация должна быть легко доступной для пользователей.

6. Достоверность информации: определяется ее свойством отражать реально существующие объекты с необходимой точностью. Измеряется достоверность информации доверительной вероятностью необходимой точности, т. е. вероятностью того, что отображаемое информацией значение параметра отличается от истинного значения этого параметра в пределах необходимой точности.

В одном терминологическом ряду с понятием «информация» в информационных технологиях применяются понятия «данные» и «знания».

Данные с точки зрения информационных технологий – это документированная информация, циркулирующая в процессе ее обработки на электронно-вычислительных машинах. Данные хранятся в базах данных.

База данных – совокупность структурированной и взаимосвязанной информации, организованной по определенным правилам на материальных носителях. Базы данных могут объединяться в банки данных.

Банк данных – организационно-техническая система, включающая одну или несколько баз данных и систему управления ими.

Знания – это информация, на основании которой путем логических рассуждений могут быть получены определенные выводы.

Для обобщения накопленных знаний и обеспечения возможности использования их в практической деятельности, они так же, как и данные, организуются в базы знаний.

База знаний – совокупность формализованных знаний об определенной предметной области, представленных в виде фактов и правил.

Для представления информации используется алфавитный способ, основой которого является использование фиксированного конечного набора символов любой природы, называемого алфавитом. Символы из набора алфавита называются буквами, а любая конечная последовательность букв этого алфавита – словом. Примеры слов: AAD, 10110110. Символы алфавита при вводе в ЭВМ должны быть преобразованы в код.

В цифровых ЭВМ преимущественное распространение получило двоичное кодирование, при котором символы вводимой в ЭВМ информации представляются средствами двоичного алфавита, состоящего из двух символов 0 и 1, которые моделируются двумя фиксированными состояниями среды: есть напряжение, нет напряжения; включено устройство или выключено и т. д. Каждой букве ставится в соответствие последовательность нескольких двоичных знаков или двоичное слово. Такие последовательности называются кодовыми комбинациями.

Процесс получения кодовых комбинаций для представления букв одного алфавита средствами другого алфавита называется кодированием. Процесс обратного преобразования информации относительно ранее выполненного кодирования называется декодированием. Например, представление буквы А русского алфавита с помощью двоичных символов 11100001 есть процесс кодирования, обратный процесс получения буквы А из двоичного кода 11100001 есть декодирование. Полный набор кодовых комбинаций, соответствующий представлению всех букв одного алфавита средствами другого

алфавита, называется кодом. Число символов, составляющих кодовую комбинацию, называется длиной кода или разрядностью кода. Максимальное число кодовых комбинаций N при заданной разрядности n определяется выражением: $N = 2^n$.

В вычислительной технике обычно используется ASCII код (американский стандарт кодов для обмена информацией), позволяющий закодировать 256 символов. В настоящее время в приложениях операционной системы Windows применяется UNICOD, который позволяет закодировать 65 536 символов.

Для измерения объема передаваемой или хранимой информации используются следующие единицы измерения:

– бит – один двоичный символ. Он позволяет представить в двоичной форме одно из двух различимых состояний объекта – «0» или «1»;

– байт – восемь двоичных символов;

– слово – два байта для 16-разрядных ЭВМ или 4 байта для 32 разрядных ЭВМ, 8 байт для 64-разрядных ЭВМ;

– килобайт – 1024 бита (2^{10});

– мегабайт – 1 048 576 байт (2^{20}),

– гигабайт – 2^{30} байт, терабайт – 2^{40} байт, петабайт – 2^{50} байт.

1.3.5. Основные свойства информационных технологий

Информационная технология (ИТ) - комплекс взаимосвязанных научных, технологических, экономических, инженерных дисциплин, занимающихся изучением, созданием и применением методов, средств, правил, используемых для получения новой информации (сведений, знаний), сбора, обработки, анализа, хранения, интерпретации, выделения и применения данных и информации с целью удовлетворения информационных потребностей народного хозяйства и общества в требуемом объеме и заданного качества с применением современных программных и компьютерных средств. В настоящее время под информационными технологиями чаще всего подразумевают компьютерные информационные технологии.

Выделяют следующие основные характеристики информационной технологии:

– предметом процесса обработки являются данные;

– целью процесса является получение, хранение, обработка, передача информации;

– средствами осуществления процесса являются программные, аппаратные и программно-аппаратные вычислительные комплексы;

– процессы обработки данных разделяются на операции в соответствии с данной предметной областью;

– выбор управляющих воздействий на процессы осуществляется лицами, принимающими решения;

– критерием оптимизации процесса являются своевременность доставки информации пользователю, ее надежность, доступность, полнота.

Основные составляющие информационных технологий

1. Техническое обеспечение – это аппаратные средства и средства коммуникации, обеспечивающие работу ИТ. Как правило, включают персональные компьютеры, периферийные устройства, линии связи, сетевое оборудование и др.

2. Программное обеспечение (ПО) непосредственно реализует функции получения, обработки, хранения, отображения, поиска и анализа данных, обеспечивает взаимодействие пользователя с ЭВМ посредством пользовательского интерфейса. ПО находится в прямой зависимости от технического обеспечения.

3. Информационное обеспечение представляет собой совокупность проектных решений по видам, объемам, способам размещения и формам организации информации.

4. Методическое обеспечение – это комплекс нормативно методических и инструктивных материалов по подготовке и оформлению документов по эксплуатации технических средств и компьютерной сети, организации работы специалистов-пользователей и технического персонала.

5. Математическое обеспечение – это совокупность алгоритмов, математических методов, моделей обработки информации, используемых при решении функциональных задач и в процессе автоматизации проектных работ. Математическое обеспечение включает средства моделирования процессов проектирования и управления, методы и средства решения типовых задач экономики и управления, методы оптимизации запасов материальных ресурсов и принятия оптимальных управленческих решений.

6. Правовое обеспечение – это совокупность правовых норм, регламентирующих право отношения при создании, внедрении и использовании ИТ.

7. Лингвистическое обеспечение включает совокупность научно-технических терминов и других языковых средств, используемых в ИТ, а также правил формализации естественного языка, включающих методы сжатия и раскрытия текстовой информации с целью повышения эффективности автоматизированной обработки информации и облегчающих общение человека со средствами ИТ.

1.3.6. Классификация информационных технологий по функциям

1. *Технологии сбора и ввода данных* включают устройства и методы для сбора и ввода информации в вычислительные системы. Примеры включают сканеры, сенсоры, клавиатуры и голосовые интерфейсы.

2. *Автоматизация функций управления*. Цель такой ИТ удовлетворение информационных потребностей всех без исключения сотрудников организаций, имеющих дело с принятием решений, на основе различных видов отчетов.

3. *Электронный офис* обеспечивает организацию и поддержку коммуникационных процессов как внутри организации, так и с внешней средой на базе компьютерных сетей и других современных средств передачи и обработки информации. Технология автоматизации офиса строится на базе таких продуктов, как текстовые и табличные редакторы, электронная почта, электронный календарь, телеконференции, системы электронного документооборота и т. д.

4. *Системы поддержки принятия решений* обеспечивают переработку больших объемов информации и принятия решения. Особенность данной ИТ в том, что человек активно участвует в данном процессе на начальной и завершающей стадиях (вводит данные в компьютер и принимает окончательное решение на основе полученной информации), а компьютер под управлением человека создает новую информацию.

5. *Экспертная поддержка* дает возможность получать консультации экспертов по любым проблемам, о которых этими системами накоплены знания, и принимать обоснованные решения.

6. *Технологии хранения данных* включают системы и устройства для хранения информации. Примеры включают жесткие диски, твердотельные накопители (SSD), облачные хранилища и базы данных.

7. *Технологии обработки данных* включают методы и инструменты для обработки и анализа информации. Примеры включают процессоры, серверы, программное обеспечение для анализа данных и системы искусственного интеллекта.

8. *Технологии передачи данных* включают методы и инструменты для передачи информации между устройствами и системами. Примеры включают сетевые протоколы, кабели связи, беспроводные технологии (Wi-Fi, Bluetooth) и интернет.

9. *Технологии отображения и вывода данных* включают устройства и методы для отображения и вывода информации. Примеры включают мониторы, принтеры, проекторы и аудиосистемы.

1.3.7. Классификация информационных технологий по областям применения

1. *Бизнес-технологии*: ИТ, используемые в бизнесе для автоматизации процессов, управления данными и принятия решений. Примеры включают системы управления предприятием (ERP), системы управления взаимоотношениями с клиентами (CRM) и бизнес-аналитику.

2. *Научные технологии*: ИТ, используемые в научных исследованиях для сбора, обработки и анализа данных. Примеры включают суперкомпьютеры, программное обеспечение для моделирования и системы управления экспериментами.

3. *Образовательные технологии*: ИТ, используемые в образовании для обучения и развития студентов. Примеры включают электронные учебные материалы, системы управления обучением (LMS) и виртуальные классы.

4. *Медицинские технологии*: ИТ, используемые в здравоохранении для диагностики, лечения и управления медицинскими данными. Примеры включают системы электронных медицинских записей (EMR), телемедицину и медицинские устройства с подключением к интернету.

5. *Развлекательные технологии*: ИТ, используемые для создания и потребления развлекательного контента. Примеры включают видеоигры, стриминговые сервисы, социальные сети и виртуальную реальность (VR).

1.4. Операционные системы. Функциональные характеристики

Операционные системы (ОС) являются фундаментальным компонентом вычислительных систем, обеспечивающим управление аппаратными ресурсами и предоставляющим пользователям и приложениям удобный интерфейс для взаимодействия с компьютером. ОС выполняет множество функций, включая управление процессами, памятью, файловыми системами, вводом-выводом и обеспечение безопасности. В этом разделе мы рассмотрим основные концепции операционных систем, их классификацию, архитектуру и примеры современных ОС.

1.4.1. Основные функции операционных систем

Управление процессами

Управление процессами является одной из ключевых функций операционной системы. Процесс – это выполняемая программа, которая включает в себя код, данные и ресурсы. ОС отвечает за создание, завершение, приостановку и возобновление процессов, а также за их синхронизацию и взаимодействие.

Планирование процессов

Планирование процессов (схедулинг) – это механизм, который определяет порядок выполнения процессов на центральном процессоре (ЦП). Существует несколько алгоритмов планирования, таких как:

1. FCFS (First Come, First Served) – процессы выполняются в порядке их поступления.
2. SJF (Shortest Job First) – процессы выполняются в порядке возрастания их длительности.
3. Приоритетное планирование – процессы выполняются в порядке их приоритета.
4. Round Robin – процессы выполняются по кругу, каждому процессу выделяется определенное время (квант времени).

Синхронизация и взаимодействие процессов

Синхронизация процессов необходима для координации их выполнения и предотвращения конфликтов при доступе к общим ресурсам. ОС использует различные механизмы синхронизации, такие как мьютексы, семафоры и мониторы, для обеспечения корректного взаимодействия процессов.

Управление памятью

Управление памятью включает в себя распределение и освобождение памяти для процессов, а также обеспечение их изоляции и защиты. ОС использует различные механизмы управления памятью, такие как разделение памяти, виртуальная память и кэширование.

Разделение памяти

Разделение памяти позволяет выделять отдельные области памяти для каждого процесса, что предотвращает конфликты и обеспечивает изоляцию процессов. ОС использует таблицы разделения памяти для управления этими областями.

Виртуальная память

Виртуальная память позволяет процессам использовать больше памяти, чем физически доступно в системе. ОС использует страничную или сегментную организацию памяти для реализации виртуальной памяти. Страничная организация памяти делит память на страницы фиксированного размера, которые могут быть загружены в физическую память по мере необходимости.

Кэширование

Кэширование используется для ускорения доступа к часто используемым данным. ОС использует кэш-память для хранения копий данных, которые могут быть быстро доступны процессам.

Управление файловыми системами

Управление файловыми системами включает в себя создание, удаление, чтение и запись файлов, а также управление их атрибутами и правами доступа. ОС предоставляет интерфейс для взаимодействия пользователей и приложений с файловыми системами.

Файловые системы

Файловая система – это метод организации и хранения файлов на устройствах хранения данных. Существует несколько типов файловых систем, таких как:

1. *FAT (File Allocation Table).* Простая и широко используемая файловая система, поддерживающая небольшие объемы данных.
2. *NTFS (New Technology File System).* Современная файловая система, поддерживающая большие объемы данных, журналирование и шифрование.
3. *ext4 (Fourth Extended Filesystem).* Файловая система, используемая в Linux, поддерживающая журналирование и большие объемы данных.

Управление правами доступа

Управление правами доступа обеспечивает защиту файлов от несанкционированного доступа. ОС использует механизмы контроля доступа, такие как списки контроля доступа (ACL) и атрибуты файлов, для определения прав доступа пользователей и процессов.

Управление вводом-выводом

Управление вводом-выводом включает в себя взаимодействие с устройствами ввода-вывода, такими как клавиатура, мышь, монитор, принтер и сетевые интерфейсы. ОС предоставляет драйверы устройств и интерфейсы для управления вводом-выводом.

Драйверы устройств

Драйверы устройств – это программные модули, которые обеспечивают взаимодействие ОС с аппаратными устройствами. Драйверы устройств преобразуют команды ОС в команды, понятные устройствам, и наоборот.

Буферизация и кэширование

Буферизация и кэширование используются для ускорения ввода-вывода. ОС использует буферы для временного хранения данных, которые передаются между устройствами и процессами. Кэширование позволяет хранить часто используемые данные в быстрой памяти для ускорения доступа.

Обеспечение безопасности

Обеспечение безопасности включает в себя защиту системы от несанкционированного доступа, вредоносного ПО и других угроз. ОС использует различные механизмы безопасности, такие как аутентификация, авторизация, шифрование и контроль целостности.

Аутентификация и авторизация

Аутентификация – это процесс проверки подлинности пользователей и процессов. ОС использует методы аутентификации, такие как пароли, биометрические данные и сертификаты, для проверки подлинности пользователей. Авторизация – это процесс предоставления прав доступа пользователям и процессам. ОС использует механизмы контроля доступа для определения прав доступа пользователей и процессов.

Шифрование

Шифрование используется для защиты данных от несанкционированного доступа. ОС использует алгоритмы шифрования для защиты данных при их хранении и передаче. Шифрование обеспечивает конфиденциальность данных и предотвращает их утечку.

Контроль целостности

Контроль целостности обеспечивает защиту данных от несанкционированных изменений. ОС использует механизмы контроля целостности, такие как контрольные суммы и цифровые подписи, для проверки целостности данных.

1.4.2. Классификация операционных систем

Операционные системы можно классифицировать по различным признакам, таким как количество пользователей, количество задач, архитектура и область применения.

По количеству пользователей

1. *Однопользовательские ОС*: ОС, предназначенные для работы одного пользователя. Примеры включают Windows 95, Windows 98 и Mac OS.

2. *Многопользовательские ОС*: ОС, поддерживающие работу нескольких пользователей одновременно. Примеры включают Unix, Linux и Windows NT.

По количеству задач

1. *Однозадачные ОС*: ОС, выполняющие только одну задачу в любой момент времени. Примеры включают MS-DOS и ранние версии Windows.

2. *Многозадачные ОС*: ОС, поддерживающие выполнение нескольких задач одновременно. Примеры включают Windows, Linux и macOS.

По архитектуре

1. *Монолитные ОС*: ОС, в которых все компоненты работают в одном адресном пространстве. Примеры включают ранние версии Unix и Windows.

2. *Микроядерные ОС*: ОС, в которых основные функции выполняются в пользовательском пространстве, а ядро выполняет только минимальные функции. Примеры включают Minix и QNX.

3. *Гибридные ОС*: ОС, сочетающие элементы монолитных и микроядерных архитектур. Примеры включают Windows NT и macOS.

По области применения

1. *Настольные ОС*: ОС, предназначенные для использования на настольных компьютерах и ноутбуках. Примеры включают Windows, macOS и Linux.

2. *Серверные ОС*: ОС, предназначенные для использования на серверах. Примеры включают Windows Server, Linux и FreeBSD.

3. *Встроенные ОС*: ОС, предназначенные для использования в встроенных системах. Примеры включают VxWorks, FreeRTOS, QNX.

4. *Мобильные ОС*: ОС, предназначенные для использования на мобильных устройствах. Примеры включают Android, iOS и Windows Phone.

1.4.3. Архитектура операционных систем

Архитектура операционной системы определяет организацию и взаимодействие ее компонентов. Существует несколько архитектурных подходов, таких как монолитная архитектура, микроядерная архитектура и гибридная архитектура.

Монолитная архитектура

Монолитная архитектура характеризуется тем, что все компоненты ОС работают в одном адресном пространстве. Это позволяет обеспечить высокую производительность и эффективность, но усложняет разработку и отладку. Примеры монолитных ОС включают ранние версии Unix и Windows.

Преимущества монолитной архитектуры

1. *Высокая производительность*. Все компоненты работают в одном адресном пространстве, что уменьшает накладные расходы на взаимодействие.

2. *Эффективность*. ОС может оптимизировать использование ресурсов и уменьшить время выполнения задач.

Недостатки монолитной архитектуры

1. *Сложность разработки*. Разработка и отладка монолитных ОС требуют значительных усилий и времени.

2. *Низкая модульность*. Изменение одного компонента может потребовать изменения других компонентов, что усложняет обновление и поддержку.

Микроядерная архитектура

Микроядерная архитектура характеризуется тем, что основные функции ОС выполняются в пользовательском пространстве, а ядро выполняет только минимальные функции, такие как управление памятью и планирование процессов. Это позволяет обеспечить высокую модульность и безопасность, но может снизить производительность. Примеры микроядерных ОС включают Minix и QNX.

Преимущества микроядерной архитектуры

1. *Высокая модульность*. Компоненты ОС могут быть разработаны и отлажены независимо, что упрощает обновление и поддержку.

2. *Безопасность*. Ограничение функций ядра уменьшает вероятность ошибок и уязвимостей.

Недостатки микроядерной архитектуры

1. *Снижение производительности*. Взаимодействие между компонентами в пользовательском пространстве может увеличить накладные расходы и снизить производительность.

2. *Сложность взаимодействия*. Необходимость взаимодействия между компонентами в пользовательском пространстве может усложнить разработку и отладку.

Гибридная архитектура

Гибридная архитектура сочетает элементы монолитной и микроядерной архитектур. В гибридных ОС ядро выполняет основные функции, но некоторые компоненты могут работать в пользовательском пространстве. Это позволяет обеспечить баланс между производительностью и модульностью. Примеры гибридных ОС включают Windows NT и macOS.

Преимущества гибридной архитектуры

1. *Баланс между производительностью и модульностью*. Гибридные ОС могут оптимизировать использование ресурсов и уменьшить время выполнения задач, а также обеспечить модульность и безопасность.

2. *Гибкость*. Гибридные ОС могут адаптироваться к различным требованиям и условиям, что делает их универсальными.

Недостатки гибридной архитектуры

1. *Сложность разработки*. Разработка и отладка гибридных ОС требуют значительных усилий и времени.

2. *Компромиссы*. Необходимость балансировать между производительностью и модульностью может привести к компромиссам в дизайне и реализации.

1.4.4. Примеры современных операционных систем

Windows

Windows – это семейство операционных систем, разработанных компанией Microsoft. Windows поддерживает различные архитектуры, такие как x86, x86-64 и ARM, и используется на настольных компьютерах, ноутбуках, серверах и мобильных устройствах.

Основные компоненты Windows

1. *Ядро*. Ядро Windows выполняет основные функции ОС, такие как управление памятью, планирование процессов и управление вводом-выводом.

2. *Подсистема пользовательского интерфейса*. Подсистема пользовательского интерфейса предоставляет графический интерфейс для взаимодействия пользователей с ОС.

3. *Подсистема Win32*. Подсистема Win32 предоставляет API для разработки приложений, совместимых с Windows.

4. *Подсистема .NET*. Подсистема .NET предоставляет платформу для разработки приложений на основе .NET Framework.

Преимущества Windows

1. *Широкая совместимость*. Windows поддерживает широкий спектр аппаратных и программных решений, что делает его универсальным.

2. *Удобство использования*. Windows предоставляет удобный и интуитивно понятный графический интерфейс, что облегчает его использование.

3. *Поддержка и обновления*. Microsoft регулярно выпускает обновления и патчи безопасности для Windows, что обеспечивает его актуальность и безопасность.

Недостатки Windows

1. *Стоимость*. Windows является коммерческой ОС, что может ограничивать его использование в некоторых случаях.

2. *Безопасность*.Windows является популярной целью для вредоносного ПО, что требует постоянного внимания к вопросам безопасности.

Linux

Linux – это семейство операционных систем на основе ядра Linux. Linux поддерживает различные архитектуры, такие как x86, x86-64, ARM и другие, и используется на настольных компьютерах, серверах, встроенных системах и мобильных устройствах.

Основные компоненты Linux

1. *Ядро*. Ядро Linux выполняет основные функции ОС, такие как управление памятью, планирование процессов и управление вводом-выводом.

2. *Системные библиотеки*. Системные библиотеки предоставляют API для разработки приложений, совместимых с Linux.

3. *Системные утилиты*. Системные утилиты предоставляют инструменты для управления ОС и выполнения различных задач.

4. *Графический интерфейс*. Графический интерфейс предоставляет удобный интерфейс для взаимодействия пользователей с ОС.

Преимущества Linux

1. *Открытый исходный код*. Linux является ОС с открытым исходным кодом, что позволяет пользователям и разработчикам свободно модифицировать и распространять его.

2. *Гибкость и настраиваемость*. Linux предоставляет широкие возможности для настройки и кастомизации, что делает его универсальным.

3. *Безопасность*. Linux является более безопасной ОС по сравнению с Windows, что делает его предпочтительным для серверов и встроенных систем.

Недостатки Linux

1. *Сложность использования*. Linux может быть сложным для начинающих пользователей, что требует определенных знаний и навыков.

2. *Совместимость*. Некоторые программы и драйверы могут быть несовместимы с Linux, что ограничивает его использование в некоторых случаях.

macOS

macOS – это операционная система, разработанная компанией Apple для использования на компьютерах Mac. macOS поддерживает архитектуру x86-64 и ARM, и используется на настольных компьютерах, ноутбуках и рабочих станциях.

Основные компоненты macOS

1. *Ядро*. Ядро macOS выполняет основные функции ОС, такие как управление памятью, планирование процессов и управление вводом-выводом.

2. *Подсистема Cocoa*. Подсистема Cocoa предоставляет API для разработки приложений, совместимых с macOS.

3. *Подсистема Carbon*. Подсистема Carbon предоставляет API для разработки приложений, совместимых с классическими версиями Mac OS.

4. *Графический интерфейс*. Графический интерфейс предоставляет удобный и интуитивно понятный интерфейс для взаимодействия пользователей с ОС.

Преимущества macOS

1. *Интеграция с аппаратным обеспечением*. macOS оптимизирован для работы с аппаратным обеспечением Apple, что обеспечивает высокую производительность и надежность.

2. *Удобство использования*. macOS предоставляет удобный и интуитивно понятный графический интерфейс, что облегчает его использование.

3. *Безопасность*. macOS является более безопасной ОС по сравнению с Windows, что делает его предпочтительным для пользователей, заботящихся о безопасности.

Недостатки macOS

1. *Стоимость*. macOS является коммерческой ОС, что может ограничивать его использование в некоторых случаях.

2. *Совместимость*. Некоторые программы и драйверы могут быть несовместимы с macOS, что ограничивает его использование в некоторых случаях.

Android

Android – это операционная система, разработанная компанией Google для использования на мобильных устройствах. Android поддерживает архитектуру ARM и x86 и используется на смартфонах, планшетах, телевизорах и других устройствах.

Основные компоненты Android

1. *Ядро*. Ядро Android выполняет основные функции ОС, такие как управление памятью, планирование процессов и управление вводом-выводом.

2. *Системные библиотеки*. Системные библиотеки предоставляют API для разработки приложений, совместимых с Android.

3. *Среда выполнения Android*. Среда выполнения Android предоставляет виртуальную машину для выполнения приложений, написанных на языке Java.

4. *Приложения*. Приложения предоставляют функциональность для выполнения различных задач, таких как связь, навигация, развлечения и продуктивность.

Преимущества Android

1. *Открытый исходный код*. Android является ОС с открытым исходным кодом, что позволяет пользователям и разработчикам свободно модифицировать и распространять его.

2. *Гибкость и настраиваемость*. Android предоставляет широкие возможности для настройки и кастомизации, что делает его универсальным.

3. *Широкая поддержка устройств*. Android поддерживает широкий спектр устройств, что делает его популярным среди производителей и пользователей.

Недостатки Android

1. *Фрагментация*. Android используется на различных устройствах с различными версиями ОС, что может привести к проблемам совместимости и безопасности.

2. *Безопасность*. Android является популярной целью для вредоносного ПО, что требует постоянного внимания к вопросам безопасности.

iOS

iOS – это операционная система, разработанная компанией Apple для использования на мобильных устройствах. iOS поддерживает архитектуру ARM и используется на смартфонах, планшетах и других устройствах.

Основные компоненты iOS

1. *Ядро*. Ядро iOS выполняет основные функции ОС, такие как управление памятью, планирование процессов и управление вводом-выводом.

2. *Системные библиотеки*. Системные библиотеки предоставляют API для разработки приложений, совместимых с iOS.

3. *Среда выполнения iOS*. Среда выполнения iOS предоставляет виртуальную машину для выполнения приложений, написанных на языке Swift.

4. *Приложения*. Приложения предоставляют функциональность для выполнения различных задач, таких как связь, навигация, развлечения и продуктивность.

Преимущества iOS

1. *Интеграция с аппаратным обеспечением*. iOS оптимизирован для работы с аппаратным обеспечением Apple, что обеспечивает высокую производительность и надежность.

2. *Удобство использования*. iOS предоставляет удобный и интуитивно понятный графический интерфейс, что облегчает его использование.

3. *Безопасность*. iOS является более безопасной ОС по сравнению с Android, что делает его предпочтительным для пользователей, заботящихся о безопасности.

Недостатки iOS

1. *Закрытость*. iOS является закрытой ОС, что ограничивает возможности для настройки и кастомизации.

2. *Совместимость*. Некоторые приложения и устройства могут быть несовместимы с iOS, что ограничивает его использование в некоторых случаях.

Операционные системы играют ключевую роль в функционировании вычислительных систем, обеспечивая управление аппаратными ресурсами и предоставляя пользователям и приложениям удобный интерфейс для взаимодействия с компьютером. В этом разделе мы рассмотрели основные концепции операционных систем, их классификацию, архитектуру и примеры современных ОС. Понимание этих аспектов позволяет эффективно использовать операционные системы для решения различных задач и достижения целей. В будущем можно ожидать дальнейшего развития операционных систем, что откроет новые возможности для научных исследований, бизнеса и повседневной жизни.

1.5. Технологии программирования. Компилируемые, интерпретируемые и встраиваемые языки

Языки и технологии программирования являются фундаментальными инструментами в области информационных технологий и вычислительной техники. Они позволяют разработчикам создавать программное обеспечение, которое управляет аппаратными ресурсами, обрабатывает данные и взаимодействует с пользователями. В этом разделе мы рассмотрим основные концепции языков программирования, их классификацию, архитектуру и примеры современных технологий программирования.

1.5.1. Основные концепции языков программирования

Что такое язык программирования?

Язык программирования – это формальный язык, предназначенный для написания программ, которые управляют поведением компьютера. Языки программирования состоят из набора синтаксических и семантических правил, которые определяют, как должны быть написаны программы и как они будут выполняться.

Основные компоненты языка программирования

1. *Синтаксис*. Набор правил, определяющих структуру программы. Синтаксис включает в себя правила написания операторов, выражений и конструкций языка.

2. *Семантика*. Значение и смысл конструкций языка. Семантика определяет, как программа будет выполняться, и какие результаты она будет производить.

3. *Парадигма*. Основной стиль или подход к программированию, который используется в языке. Существует несколько парадигм программирования, таких как императивное, декларативное, объектно-ориентированное и функциональное программирование.

Парадигмы программирования

1. *Императивное программирование*. Парадигма, в которой программа состоит из последовательности команд, изменяющих состояние системы. Примеры языков: C, Pascal, Fortran.

2. *Декларативное программирование*. Парадигма, в которой программа описывает, что должно быть сделано, а не как это должно быть сделано. Примеры языков: SQL, Prolog.

3. *Объектно-ориентированное программирование (ООП)*. Парадигма, в которой программа состоит из объектов, которые взаимодействуют друг с другом. Объекты представляют собой экземпляры классов, которые включают в себя данные и методы для работы с этими данными. Примеры языков: Java, C++, Python.

4. *Функциональное программирование*. Парадигма, в которой программа состоит из функций, которые не изменяют состояние системы и не имеют побочных эффектов. Примеры языков: Haskell, Lisp, Erlang.

1.5.2. Классификация языков программирования

Языки программирования можно классифицировать по различным признакам, таким как уровень абстракции, парадигма программирования, область применения и типизация.

По уровню абстракции

1. *Машинные языки*. Языки, которые напрямую понимает процессор компьютера. Машинные языки состоят из двоичных кодов и являются низкоуровневыми языками. Пример: машинный код для архитектуры x86.

2. *Ассемблер*. Языки, которые используют мнемонические коды для представления машинных инструкций. Ассемблер является низкоуровневым языком, но более удобным для человека, чем машинный код. Пример: Assembly language для архитектуры x86.

3. *Высокоуровневые языки*. Языки, которые абстрагируют детали аппаратного обеспечения и предоставляют более удобный и понятный синтаксис для программиста. Высокоуровневые языки обычно независимы от конкретной архитектуры компьютера. Примеры: Java, Python, C++.

По парадигме программирования

1. *Императивные языки*. Языки, которые используют императивную парадигму программирования. Примеры: C, Pascal, Fortran.

2. *Декларативные языки*. Языки, которые используют декларативную парадигму программирования. Примеры: SQL, Prolog.

3. *Объектно-ориентированные языки*. Языки, которые поддерживают объектно-ориентированную парадигму программирования. Примеры: Java, C++, Python.

4. *Функциональные языки*. Языки, которые используют функциональную парадигму программирования. Примеры: Haskell, Lisp, Erlang.

По области применения

1. *Системные языки*. Языки, предназначенные для разработки операционных систем, драйверов устройств и встроенных систем. Примеры: C, Assembly language.

2. *Прикладные языки*. Языки, предназначенные для разработки прикладных программ, таких как офисные приложения, веб-приложения и мобильные приложения. Примеры: Java, Python, JavaScript.

3. *Специализированные языки*. Языки, предназначенные для решения специфических задач в определенных областях. Примеры: SQL для работы с базами данных, MATLAB для научных вычислений, Verilog для разработки цифровых схем.

По типизации

1. *Статически типизированные языки*. Языки, в которых типы данных определяются на этапе компиляции и не могут быть изменены во время выполнения программы. Примеры: Java, C++, Haskell.

2. *Динамически типизированные языки*. Языки, в которых типы данных определяются во время выполнения программы и могут быть изменены. Примеры: Python, JavaScript, Ruby.

1.5.3. Архитектура и компиляция языков программирования

Компиляция и интерпретация

Языки программирования могут быть компилируемыми или интерпретируемыми. Компиляция – это процесс перевода исходного кода программы в машинный код, который может быть выполнен процессором. Интерпретация – это процесс выполнения исходного кода программы напрямую, без предварительной компиляции.

Компилируемые языки

Компилируемые языки требуют предварительной компиляции исходного кода в машинный код. Компилятор анализирует исходный код, проверяет его на наличие синтаксических и семантических ошибок, оптимизирует его и генерирует машинный код. Примеры компилируемых языков: C, C++, Java.

Этапы компиляции

1. *Лексический анализ*. Разбиение исходного кода на лексемы (токены), такие как ключевые слова, операторы и идентификаторы.

2. *Синтаксический анализ*. Проверка последовательности лексем на соответствие грамматике языка и построение синтаксического дерева.

3. *Семантический анализ*. Проверка семантической корректности программы, такой как типизация данных и соответствие объявлений.

4. *Оптимизация*. Улучшение производительности программы за счет оптимизации кода.

5. *Генерация кода*. Преобразование оптимизированного кода в машинный код.

Интерпретируемые языки

Интерпретируемые языки выполняются напрямую интерпретатором, который анализирует и выполняет исходный код по мере его чтения. Интерпретатор выполняет те же этапы, что и компилятор, но делает это во время выполнения программы. Примеры интерпретируемых языков: Python, JavaScript, Ruby.

Преимущества и недостатки интерпретации

Преимущества:

1. *Гибкость*. Интерпретируемые языки позволяют изменять и тестировать код на лету, что ускоряет процесс разработки.

2. *Портативность*. Интерпретируемые языки могут быть легко перенесены на различные платформы, так как они не требуют компиляции.

Недостатки:

1. *Производительность*. Интерпретируемые языки обычно медленнее компилируемых языков, так как они выполняются напрямую интерпретатором.

2. *Ограниченная оптимизация*. Интерпретаторы имеют ограниченные возможности для оптимизации кода по сравнению с компиляторами.

1.5.4. Виртуальные машины

Виртуальная машина (VM) – это программное обеспечение, которое эмулирует работу реального компьютера. Виртуальные машины используются для выполнения байт-кода, который является промежуточным представлением исходного кода программы. Байт-код является независимым от архитектуры компьютера и может быть выполнен на любой платформе, поддерживающей соответствующую виртуальную машину.

Примеры виртуальных машин

1. *Java Virtual Machine (JVM)*. Виртуальная машина для выполнения байт-кода Java. JVM позволяет запускать Java-программы на любой платформе, поддерживающей JVM.

2. *Common Language Runtime (CLR)*. Виртуальная машина для выполнения байт-кода .NET. CLR позволяет запускать программы, написанные на языках .NET, таких как C# и VB.NET, на любой платформе, поддерживающей CLR.

Сборка мусора

Сборка мусора (Garbage Collection) – это механизм автоматического управления памятью, который освобождает память, занимаемую объектами, которые больше не используются программой. Сборка мусора позволяет избежать утечек памяти и упрощает разработку программ.

Алгоритмы сборки мусора

1. *Подсчет ссылок*. Алгоритм, который подсчитывает количество ссылок на каждый объект и освобождает память, занимаемую объектами, на которые нет ссылок.

2. *Метка и очистка*. Алгоритм, который помечает все достижимые объекты, начиная с корневых объектов, и освобождает память, занимаемую непомеченными объектами.

3. *Генерационная сборка мусора*. Алгоритм, который делит объекты на поколения в зависимости от их времени жизни и собирает мусор для каждого поколения отдельно.

1.5.5. Современные технологии программирования

Современные технологии программирования включают в себя различные инструменты, фреймворки и методологии, которые позволяют разработчикам создавать сложные и масштабируемые приложения. В этом разделе мы рассмотрим основные современные технологии программирования и их применение.

Веб-технологии

Веб-технологии используются для разработки веб-приложений, которые работают в браузерах. Веб-приложения могут быть как клиентскими, так и серверными, и включают в себя различные компоненты, такие как HTML, CSS, JavaScript и серверные языки программирования.

Клиентские технологии

1. *HTML (HyperText Markup Language).* Язык разметки, используемый для создания структуры веб-страниц. HTML определяет элементы веб-страницы, такие как заголовки, параграфы, ссылки и изображения.

2. *CSS (Cascading Style Sheets).* Язык стилей, используемый для оформления веб-страниц. CSS позволяет задавать стили для элементов HTML, такие как цвета, шрифты, отступы и размеры.

3. *JavaScript*. Язык программирования, используемый для создания интерактивных элементов веб-страниц. JavaScript позволяет обрабатывать события, изменять содержимое страницы и взаимодействовать с сервером.

Серверные технологии

1. *Node.js*. Платформа для выполнения JavaScript на сервере. Node.js позволяет создавать масштабируемые и высокопроизводительные серверные приложения.

2. *Django*. Фреймворк для разработки веб-приложений на языке Python. Django предоставляет инструменты для создания масштабируемых и безопасных веб-приложений.

3. *Ruby on Rails*. Фреймворк для разработки веб-приложений на языке Ruby. Ruby on Rails предоставляет инструменты для создания веб-приложений по принципу «конвенции превыше конфигурации».

Мобильные технологии

Мобильные технологии используются для разработки приложений для мобильных устройств, таких как смартфоны и планшеты. Мобильные приложения могут быть как нативными, так

и кроссплатформенными, и включают в себя различные компоненты, такие как пользовательский интерфейс, логика приложения и взаимодействие с аппаратными ресурсами.

Нативные технологии

1. *Android.* Операционная система для мобильных устройств, разработанная компанией Google. Android использует язык программирования Java и Kotlin для разработки приложений.

2. *iOS*. Операционная система для мобильных устройств, разработанная компанией Apple. iOS использует язык программирования Swift и Objective-C для разработки приложений.

Кроссплатформенные технологии

1. *React Native*. Фреймворк для разработки кроссплатформенных мобильных приложений на языке JavaScript. React Native позволяет создавать приложения для Android и iOS с использованием одного кода.

2. *Flutter*. Фреймворк для разработки кроссплатформенных мобильных приложений на языке Dart. Flutter позволяет создавать приложения для Android и iOS с использованием одного кода.

Облачные технологии

Облачные технологии используются для разработки приложений, которые работают в облачных средах. Облачные приложения могут быть как серверными, так и клиентскими, и включают в себя различные компоненты, такие как виртуальные машины, контейнеры и сервисы.

Облачные платформы

1. *Amazon Web Services (AWS)*. Облачная платформа от компании Amazon. AWS предоставляет различные сервисы, такие как виртуальные машины, базы данных, хранилища данных и машинное обучение.

2. *Microsoft Azure*. Облачная платформа от компании Microsoft. Azure предоставляет различные сервисы, такие как виртуальные машины, базы данных, хранилища данных и искусственный интеллект.

3. *Google Cloud Platform (GCP)*. Облачная платформа от компании Google. GCP предоставляет различные сервисы, такие как виртуальные машины, базы данных, хранилища данных и машинное обучение.

Контейнеризация

Контейнеризация – это технология, которая позволяет упаковывать приложения и их зависимости в контейнеры, которые могут быть выполнены на любой платформе. Контейнеры обеспечивают изоляцию и портативность приложений.

1. *Docker*. Платформа для контейнеризации приложений. Docker позволяет создавать, развертывать и управлять контейнерами.

2. *Kubernetes*. Платформа для оркестрации контейнеров. Kubernetes позволяет управлять кластерами контейнеров, обеспечивая их масштабируемость и надежность.

1.5.6. Технологии искусственного интеллекта

Технологии искусственного интеллекта (ИИ) используются для разработки приложений, которые могут анализировать данные, распознавать образы, принимать решения и выполнять другие сложные задачи. ИИ включает в себя различные компоненты, такие как машинное обучение, нейронные сети и обработка естественного языка.

Машинное обучение

Машинное обучение – это раздел ИИ, который использует алгоритмы для обучения моделей на основе данных. Машинное обучение позволяет создавать модели, которые могут предсказывать результаты, классифицировать объекты и принимать решения.

1. *Scikit-learn*. Библиотека для машинного обучения на языке Python. Scikit-learn предоставляет инструменты для классификации, регрессии, кластеризации и других задач машинного обучения.

2. *TensorFlow*. Платформа для машинного обучения, разработанная компанией Google. TensorFlow предоставляет инструменты для создания и обучения нейронных сетей.

Нейронные сети

Нейронные сети – это модели машинного обучения, которые имитируют работу человеческого мозга. Нейронные сети состоят из слоев нейронов, которые обрабатывают входные данные и производят выходные результаты.

1. *Convolutional Neural Networks (CNN)*. Тип нейронных сетей, используемый для обработки изображений. CNN используют сверточные слои для извлечения признаков из изображений.

2. *Recurrent Neural Networks (RNN).* Тип нейронных сетей, используемый для обработки последовательных данных, таких как текст и временные ряды. RNN используют циклические связи для обработки последовательностей.

Обработка естественного языка

Обработка естественного языка (NLP) – это раздел ИИ, который использует алгоритмы для анализа и понимания человеческого языка. NLP позволяет создавать приложения, которые могут распознавать речь, переводить текст, анализировать настроения и выполнять другие задачи, связанные с языком.

1. *NLTK (Natural Language Toolkit).* Библиотека для обработки естественного языка на языке Python. NLTK предоставляет инструменты для токенизации, лемматизации, частеречной разметки и других задач NLP.

2. *spaCy.* Библиотека для обработки естественного языка на языке Python. spaCy предоставляет инструменты для токенизации, лемматизации, частеречной разметки, распознавания именованных сущностей и других задач NLP.

1.5.7. Технологии блокчейна

Технологии блокчейна используются для разработки децентрализованных приложений, которые обеспечивают безопасность, прозрачность и неизменяемость данных. Блокчейн включает в себя различные компоненты, такие как криптография, консенсусные алгоритмы и смарт-контракты.

1.5.8. Криптография

Криптография – это наука о защите информации с использованием математических алгоритмов. Криптография обеспечивает конфиденциальность, целостность и аутентичность данных.

1. *Symmetric Encryption.* Тип криптографии, который использует один и тот же ключ для шифрования и дешифрования данных. Примеры алгоритмов: AES, DES.

2. *Asymmetric Encryption.* Тип криптографии, который использует разные ключи для шифрования и дешифрования данных. Примеры алгоритмов: RSA, ECC.

1.5.9. Консенсусные алгоритмы

Консенсусные алгоритмы – это механизмы, которые обеспечивают согласованность данных в децентрализованных системах. Консенсусные алгоритмы позволяют участникам сети достигать согласия о состоянии данных.

1. *Proof of Work (PoW).* Консенсусный алгоритм, который использует вычислительные ресурсы для достижения согласия. PoW используется в биткоине и других криптовалютах.

2. *Proof of Stake (PoS).* Консенсусный алгоритм, который использует доли участников для достижения согласия. PoS используется в эфириуме и других криптовалютах.

1.5.10. Смарт-контракты

Смарт-контракты – это программы, которые автоматически выполняют условия контракта при выполнении определенных условий. Смарт-контракты обеспечивают автоматизацию и безопасность выполнения контрактов.

1. *Solidity*. Язык программирования для разработки смарт-контрактов на платформе Ethereum. Solidity позволяет создавать смарт-контракты для различных приложений, таких как финансы, логистика и управление цепочками поставок.

2. *Hyperledger Fabric*. Платформа для разработки смарт-контрактов для бизнес-приложений. Hyperledger Fabric предоставляет инструменты для создания смарт-контрактов для различных отраслей, таких как финансы, здравоохранение и логистика.

Языки и технологии программирования являются фундаментальными инструментами в области информационных технологий и вычислительной техники. Они позволяют разработчикам создавать программное обеспечение, которое управляет аппаратными ресурсами, обрабатывает данные и взаимодействует с пользователями. В этом разделе мы рассмотрели основные концепции языков программирования, их классификацию, архитектуру и примеры современных технологий программирования. Понимание этих аспектов позволяет эффективно использовать языки и технологии программирования для решения различных задач и достижения целей. В будущем можно ожидать дальнейшего развития языков и технологий программирования, что откроет новые возможности для научных исследований, бизнеса и повседневной жизни.

1.6. Процедурное, объектно-ориентированное и логическое программирование

Выбор парадигмы программирования является фундаментальным при создании программного обеспечения. Он определяет структуру кода, способ мышления разработчика и, в конечном итоге, эффективность решения задачи. В рамках кандидатского минимума по ИТ необходимо четко понимать различия, преимущества и области применения ключевых парадигм: процедурной, объектно-ориентированной и логической. Данный раздел посвящен их сравнительному анализу.

1.6.1. Процедурное программирование (ПП)

Основная концепция: программа представляется в виде последовательности инструкций (процедур или подпрограмм), которые выполняются шаг за шагом для изменения состояния данных. Центральными понятиями являются данные и процедуры (функции), которые над этими данными оперируют.

Ключевые принципы

Императивность. Программа, как решить задачу, детально определяя последовательность шагов.

Модульность. Код организуется в отдельные блоки – процедуры (functions) и функции (subroutines). Это позволяет избегать дублирования, структурировать программу и повторно использовать код.

Локальные и глобальные данные. Данные, как правило, отделены от кода. Процедуры работают с данными, которые передаются им через параметры или являются глобальными.

Достоинства

1. Простота и прямолинейность для небольших задач.
2. Эффективное использование ресурсов (памяти и процессорного времени), что критично для систем низкого уровня.
3. Легкость понимания потока выполнения для линейных алгоритмов.

Недостатки

1. Сложность управления большими проектами: при росте программы количество глобальных данных и взаимосвязей между процедурами становится трудно контролируемым.
2. Низкая степень повторного использования кода.

3. Проблема безопасности: глобальные данные могут быть непреднамеренно изменены любой процедурой.

Языки-представители: C, Pascal, Fortran, BASIC.

Области применения: системное программирование (ОС, драйверы), научные вычисления, микроконтроллеры, скрипты для автоматизации.

1.6.2. Объектно-ориентированное программирование (ООП)

Основная концепция. Программа представляется в виде совокупности объектов, которые взаимодействуют друг с другом путем отправки сообщений (вызова методов). Каждый объект является экземпляром класса, который выступает в роли шаблона, описывающего структуру (данные) и поведение (методы) объекта.

Ключевые принципы (четыре «столпа»):

Инкапсуляция. Объединение данных (полей, свойств) и методов, которые с ними работают, в единую структуру – класс. Сокрытие внутренней реализации объекта от внешнего мира (реализуется через модификаторы доступа: public, private, protected).

Наследование. Возможность создания нового класса (потомка) на основе существующего (родителя). Потомок наследует все свойства и методы родителя и может добавлять свои. Это обеспечивает повторное использование кода и иерархическую классификацию.

Полиморфизм. Возможность объектов с одинаковой спецификацией (интерфейсом) иметь различную реализацию. Один и тот же метод может работать по-разному в зависимости от того, объект какого класса его вызывает.

Абстракция. Моделирование только тех характеристик объекта, которые существенны для текущей задачи, игнорируя нерелевантные детали.

Достоинства

Упрощение разработки сложных систем за счет моделирования предметной области через объекты.

1. Повышение повторного использования и сопровождаемости кода.

2. Улучшение защищенности данных благодаря инкапсуляции.

Недостатки

1. Более высокий порог вхождения по сравнению с процедурным программированием.

2. Большой расход памяти и незначительное снижение производительности из-за сложности системы.

3. Риск создания излишне сложных иерархий наследования («запах кода» God Object).

Языки-представители: Java, C++, C#, Python, Ruby.

Области применения: корпоративные приложения, веб-разработка (бэкенд), desktop-приложения, игровая индустрия.

1.6.3. Логическое программирование (ЛП)

Основная концепция: в отличие от императивных парадигм, логическое программирование является декларативным. Программист не описывает последовательность действий, а декларирует факты об объектах и их отношениях, а также правила логического вывода. Затем система автоматически, с помощью механизма логического вывода, находит решение на основе этих фактов и правил.

Ключевые принципы

1. *Программа как набор утверждений.* Программа состоит из базы знаний (набора фактов и правил) и запроса к ней.

2. *Поиск с возвратом (Backtracking).* Механизм вывода (например, алгоритм унификации и поиск в глубину в Prolog) пытается найти все возможные решения, удовлетворяющие запросу, перебирая варианты и «откатываясь» назад в случае неудачи.

3. *Отношения, а не функции.* Основная конструкция – не функция, возвращающая значение, а предикат, который определяет отношение между аргументами и может быть истинным или ложным.

Достоинства

1. Идеально для задач, связанных с символьными вычислениями, искусственным интеллектом и доказательством теорем.

2. Высокий уровень абстракции: программист описывает, что нужно найти, а не как это сделать.

3. Краткость кода для задач своего класса.

Недостатки

1. Неэффективность для вычислительных задач, требующих сложных алгоритмов.

2. Сложность контроля над порядком выполнения и потреблением ресурсов.

3. Узкая специализация и непривычный для большинства разработчиков стиль мышления.

Языки-представители: Prolog, Mercury, Datalog.

Области применения: экспертные системы, базы знаний, лингвистическая обработка (парсинг), формальная верификация, математическая логика.

Не существует «лучшей» парадигмы программирования. Выбор зависит от решаемой задачи, требований к производительности, опыта команды и контекста проекта. Современная разработка часто носит мультипарадигменный характер. Например, язык Python поддерживает процедурный, объектно-ориентированный и в ограниченной степени функциональный подходы, позволяя разработчику использовать наиболее подходящий инструмент для каждой конкретной подзадачи.

2. ОСНОВНЫЕ ПРОГРАММНЫЕ СРЕДСТВА ОБРАБОТКИ ИНФОРМАЦИИ

2.1. Программное обеспечение. Средства хранения, обработки и визуализации данных

Программное обеспечение (ПО) – это совокупность компьютерных программ получения, поиска, передачи, хранения, обработки данных и необходимых для их эксплуатации документов. По назначению программное обеспечение разделяется на:

– системное ПО;

– прикладное ПО;

– инструментальное обеспечение разработки программ.

Системное программное обеспечение представляет собой совокупность взаимосвязанных программ, которые обеспечивают функционирование средств вычислительной техники как таковых без выполнения операций по реализации функций офисных технологий. Системное ПО являются продолжением аппаратного обеспечения компьютера. Оно подразделяется на базовое и сервисное.

Базовое ПО включает: операционные системы, командно-файловые процессоры (операционные оболочки), системные утилиты.

Операционная система – совокупность программных средств, обеспечивающая управление аппаратной частью компьютера и прикладными программами, а также их взаимодействие между собой и пользователем.

Командно-файловые процессоры или операционные оболочки – это специальные программы, предназначенные для облегчения общения пользователя с командами операционной системы (например, Norton Commander, Total Commander и др.).

Системные утилиты (от латин. utilitas – польза) – программы, служащие для вспомогательных операций обработки данных или обслуживания компьютера (диагностика, тестирование аппаратных или программных средств, оптимизация использования дискового пространства, восстановления разрушенной на магнитном диске информации и т. д.).

Сервисное ПО – программы и программные комплексы для выполнения вспомогательных сервисных услуг, например, антивирусные программы.

Прикладное программное обеспечение представляет собой совокупность программных комплексов, обеспечивающих решение конкретных задач при реализации тех или иных функций офисных технологий. Прикладное программное обеспечение работает только при наличии системного программного обеспечения. Прикладное ПО включает пакеты прикладных программ (ППП), которые называют также приложениями.

Пакеты прикладных программ (ППП) – это комплекс взаимосвязанных программ для решения задач определенного класса конкретной предметной области. ППП могут быть общего или специального назначения.

К пакетам прикладных программ общего назначения относятся:

– текстовые процессоры;
– табличные процессоры;
– системы управления базами данных;
– графические редакторы;
– системы разработки презентаций;
– системы обработки финансово-экономической информации;
– системы управления проектами;
– экспертные системы и системы поддержки принятия решения;
– пакеты прикладных программ специального назначения.

Пакеты прикладных программ специального назначения ориентированы на решение задач в определенной предметной области. К ним относятся: пакеты компьютерной математики для научно-технических расчетов (например, Matematica, Mathcad, Matlab), пакеты моделирования и компьютерного инженерного анализа (например, Ansys.Inc, Nastran и др.); пакеты статистической обработки данных; системы поддержки принятия решений (Assistant Choice, Multi expert и др.), корпоративные информационные системы (ERWin, BPWin и др.), обучающие программы, бухгалтерские и экономические пакеты и др. Прикладные программы пользователей служат для решения практических задач в различных предметных областях. Они автоматизируют практически все виды человеческой деятельности.

Инструментальное обеспечение разработки программ выполняется комплексом программного обеспечения, с помощью которого могут разрабатываться и адаптироваться к конкретным условиям применения те или иные функциональные программы, например для офисных технологий. Для разработки программ существуют

и применяются мощные системы программирования – это совокупность программ для разработки, отладки и внедрения новых программных продуктов. Системы программирования обычно содержат трансляторы; среду разработки программ; библиотеки справочных программ; отладчики; редакторы связей и др.

2.1.2. Текстовые редакторы, их возможности и назначение

Текстовый редактор Microsoft Word

В настоящее время широкое распространение получил текстовый редактор Microsoft Word 2016 более поздние версии, представляющие собой интегрированную программную среду для создания и редактирования документов произвольной структуры. Он обеспечивает ввод, редактирование и форматирование текста, вставку диаграмм, таблиц и рисунков, обмен данными с другими приложениями Windows, работу с гипертекстовыми документами, просмотр веб-страниц и размещение документов на веб-страницах, подготовку писем и их рассылку по электронной почте. Текстовый процессор содержит большой набор шаблонов, облегчающих создание стандартных документов, позволяет создавать записи в блоге.

Аналогичная версия текстового процессора представлена в офисном пакете Open Office – одном из ведущих систем для обработки текстов, электронных таблиц, презентаций, графиков, баз данных и многого другого. Этот мультиязычный и мультиплатформенный офисный пакет с открытым исходным кодом доступен на многих языках и работает на всех персональных компьютерах. Совместим с основными офисными пакетами. Его можно загрузить и использовать совершенно свободно для любых целей бесплатно.

Для верстки и дизайна бизнес-публикаций часто применяется программа PageMaker. Тесная интеграция с программами Adobe Photoshop, Illustrator, Acrobat вплоть до уровня «drag and drop» и совместимость с текстовым редактором Word 2010 обеспечивает удовлетворение большинства требований по верстке и оформлению.

Запуск программы осуществляется командой Пуск – Программы – Microsoft Office 2016 – Microsoft Office Word 2016 или щелчком мыши по пиктограмме W на рабочем столе или в строке состояния. После запуска программы на выполнение на экране отображается рабочее окно Microsoft Word 2016 (рис. 2.1). Выход из программы осуществляется командой Файл – Выход главного меню или комбинацией клавиш Alt + F5.

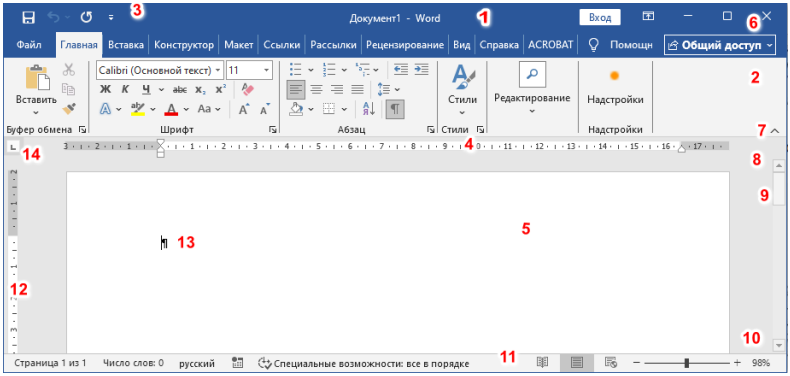


Рис. 2.1. Окно программы MS Word 2016:
1 – строка заголовка; *2* – лента инструментов; *3* – панель быстрого доступа; *4* – горизонтальная линейка прокрутки; *5* – окно документа; *6* – кнопки свертывания, развертывания и закрытия окна; *7* – кнопка деления окна документа; *8* – кнопка управления линейками; *9* – вертикальная линейка прокрутки; *10* – кнопка перехода к объекту; *11* – строка состояния; *12* – вертикальная линейка; *13* – курсор (точка вставки); *14* – установка табуляторов

В верхней части расположена строка заголовка 1, в которой выводится имя редактируемого документа и название программы. В левой части строки заголовка размещается кнопка системного меню W, а в правой – кнопки свертывания, развертывания/восстановления и закрытия окна 6. Ниже строки заголовка расположены меню Файл и Лента 2, панель быстрого доступа 3, рабочее окно и строка состояния 11.

В рабочем окне расположены: окно документа 5; горизонтальная и вертикальная линейки прокрутки 9. Горизонтальная линейка прокрутки появляется в том случае, когда ширина окна программы становится уже окна документа. Выше вертикальной линейки прокрутки расположены кнопка управления делением окна документа разделительной линией на две части по горизонтали 7 и кнопка управления линейками 8, позволяющая показать или убрать линейки, не обращаясь к командам ленты Вид. На вертикальной линейке прокрутки расположена кнопка Выбор объекта перехода 10.

В окне документа расположен курсор ввода (точка вставки), который отмечает место вставки текста или других объектов в документ 13; горизонтальная и вертикальная линейки прокрутки 4, 12. Над вертикальной линейкой прокрутки расположена кнопка маркера табуляторов 14.

Меню Файл

Кнопка меню Файл (кнопка Office) расположена в верхнем левом углу экрана. Это меню содержит команды работы с файлами: создания, открытия, закрытия, сохранения, печати и др. Для выхода из меню щелкните по ярлычку ленты Главная или ярлычку другой ленты.

Лента инструментов

В программах офисного пакета Microsoft Office 2016 самым главным нововведением последних лет было использование ленточного интерфейса, заменившего меню и панели инструментов предыдущих версий. Лента с расположенными на ней инструментами размещается вверху окна. Лента состоит из трех элементов: вкладки, группы и команды.

Вкладки

На вкладках собраны команды по их функциональному назначению. Доступ к вкладкам осуществляется с помощью ярлычков. Имеется возможность добавлять на ленту новые кнопки, а также добавлять новую вкладку, новую группу на существующую вкладку с любым набором команд.

Группы. Каждая вкладка содержит несколько групп команд, объединенных по функциональному назначению. Например, на вкладке Главная имеется пять групп: Буфер обмена, Шрифт, Абзац, Стили, Редактирование.

Команды – это кнопка, поле для ввода информации или меню. Все кнопки снабжены всплывающими подсказками.

Ленту можно изменять и дополнять. Для настройки ленты необходимо вызвать контекстное меню ленты щелчком правой кнопкой мыши по свободному участку ленты и выбрать пункт меню Настройка ленты или выберите команду Параметры – Настройка ленты в меню Файл.

Панель быстрого доступа

Панель быстрого доступа позволяет разместить на ней те кнопки, которые должны быть всегда под рукой и которых нет на вкладке Главная, например, кнопки Сохранить, Откат, создать новый документ. Справа от установленных кнопок расположена кнопка раскрывающегося списка Настройка панели быстрого доступа. Эта кнопка открывает всплывающее меню, в котором содержится набор команд. Команду можно поместить на панель, щелкнув по ней мышью. Для удаления команды с Панели быстрого доступа выделите команду в правом списке и щелкните по кнопке Удалить.

Строка состояния

Строка состояния расположена в нижней части окна программы. В левой части строки состояния выводится информация о редактируемом документе: номер текущей страницы, общее число страниц в документе, число слов в документе, используемый язык ввода. В правой части строки состояния расположены кнопки управления режимами просмотра документа, а также управление масштабом его представления. Аналогичные команды размещены в группах Режимы просмотра документа и Масштаб вкладки Вид.

Линейки прокрутки

Линейки прокрутки служат для просмотра документа. На вертикальной линейке прокрутки имеется несколько кнопок. Кнопки с одиночной стрелкой служат для перемещения текста на одну строку в соответствующем направлении, двойные стрелки перемещают текст на страницу. Быстрое перемещение по тексту удобно с помощью ползунка: зацепите ползунок мышью и перемещайте в требуемом направлении. При этом слева от ползунка появляется всплывающее окно сообщения, в котором отображаются номер страницы и наименование раздела. На вертикальной линейке прокрутки имеется кнопка – Выбор объекта перехода. При щелчке мышью по этой кнопке открывается меню, в котором можно выбрать объект для быстрого перехода: страница, разделы, примечания, сноски, концевые сноски, поля, таблицы, рисунки, заголовки, исправления, а также команды: Найти, Переход и Отмена.

Создание и открытие документов

Начало работы с документом начинается с открытия меню Файл. Создание нового документа можно выполнить так: ввести в окне программы Microsoft Word команду Файл, Создать. Открывается окно диалога, в котором следует выбрать Новый документ и щелкните по кнопке Создать. Все документы создаются на основе шаблонов. Документ, который создается командой Новый документ, Создать, основан на шаблоне Нормальный по умолчанию. Microsoft Word 2016 предлагает пользователю большое число шаблонов разного назначения. Имеется возможность создать свой пользовательский шаблон, который будет сохранен в папке Мои шаблоны.

Сохранение документа

Рекомендуется регулярно сохранять работу на диске: командой Файл, Сохранить или Файл, Сохранить как. По умолчанию предлагается тип файла Документ Word, расширение имени

файла .docx. Можно установить режим автосохранения: выберите в меню Файл команду Параметры – Сохранение и установите флажок Автосохранение каждые 5 минут.

Печать документов

Для вывода документа на печать необходимо воспользоваться командой Печать меню Файл или соответствующей кнопкой панели быстрого доступа.

Ввод текста

Текст документа вводится после точки ввода, которая отображается мигающей вертикальной чертой (курсор). При вводе длинного текста автоматически осуществляется переход на другую строку. При нажатии клавиши Enter курсор ввода переходит на другую строку в положение абзацного отступа. Для вставки пустой строки нажмите клавишу Enter. Чтобы разделить строку на две строки, установите курсор в точку раздела и нажмите клавишу Enter.

Выделение текста

Все операции в редакторе: копирование, перемещение, удаление – выполняются над выделенным текстом. Для выделения текста можно использовать несколько способов:

– отдельное слово выделяется двойным щелчком мыши;

– абзац выделяется тройным щелчком мыши;

– другой способ выделения абзаца – установите указатель мыши на поле выделения и протяните мышью по этому полю до последней строки выделяемого текста;

– для выделения произвольного фрагмента текста установите указатель мыши в начало выделяемого текста и протяните мышью до конца выделяемого текста.

Перемещение или копирование текста

Для копирования текста можно пользоваться командами Копирование и Вставка группы Буфер обмена вкладки Главная или использовать прием перетаскивания выделенного текста мышью при нажатой клавише Ctrl. Эти операции удобно выполнять также с помощью контекстного меню.

Перемещение отличается от копирования тем, что исходный текст удаляется. В этом случае для помещения текста (объекта) в буфер целесообразно использовать команду Вырезать группы Буфер обмена. При перемещении объекта мышью клавиша Ctrl не нажимается. Команда Вставить группы Буфер обмена имеет

подменю, позволяющее выбрать режим вставки объекта. Полезно также запомнить комбинации клавиш: [Ctrl + C] – забрать выделенный текст в буфер обмена и [Ctrl + V] – вставить текст из буфера обмена.

Поля

Документ, подготовленный пользователем, содержит ряд элементов: поля, заголовки, форматированный текст, абзацы, колонтитулы, таблицы, рисунки, графики, формулы и т. п. Поля ограничивают текст документа. Левое поле используется обычно для подшивки и принимается равным 20–25 мм, правое поле – не менее 10 мм. Верхнее и нижнее поля используются для размещения колонтитулов. Размеры полей можно настраивать. Настройка параметров страницы осуществляется с помощью команд группы Параметры страницы на вкладке Разметка страницы или линейками.

Для удобства размещения таблиц, рисунков и других объектов, вставляемых в документ, полезно установить на страницы границы текста. Граница текста отображается пунктирной линией только в электронном документе. Для ее установки введите команду Файл – Параметры – Дополнительно и в группе «Показывать содержимое документа» установите флажок «Показывать границы текста».

Заголовки

Заголовки оформляются с помощью стилей заголовков из списка Стили, расположенного в группе Стили на вкладке Главная ленты. Здесь же имеются два списка для изменения стилей документа. Выделенные стили можно настраивать с помощью контекстного меню. Стили, используемые в заголовках, не должны использоваться в тексте документа для выделения других фрагментов текста. При оформлении заголовков с помощью стилей открывается возможность автоматизировать процедуру составления оглавления документа.

Оформление шрифтов

Чтобы сделать текст более читабельным, удобным для чтения и восприятия пользователем, применяют различное оформление шрифта. К параметрам шрифта, подлежащим настройке, относятся: тип шрифта, размер, начертание (полужирный, курсив, подчеркнутый), цвет, межсимвольный интервал и ряд дополнительных эффектов. Настройку параметров шрифтов осуществляют с помощью команд группы Шрифт вкладки Главная ленты или окна диалога

Шрифт, которое вызывается кнопкой, расположенной в правом нижнем углу группы Шрифт. Окно диалога имеет две вкладки: Шрифт и Дополнительно. На вкладке Шрифт настраиваются практически все параметры шрифтов. Для печатного текста наиболее подходящими являются шрифты Times New Roman – шрифт с засечками и Arial – прямоугольный шрифт.

Оформление абзацев

Абзац – это фрагмент текста, который содержит, как правило, законченную мысль. Для текстового процессора абзац – это текст, заключенный между двумя нажатиями клавиши Enter. При нажатии клавиши Enter автоматически вставляется символ возврата каретки. Точка вставки (курсор) переводится на новую строку в положение отступа первой строки. Параметры абзацев настраиваются с помощью команд группы Абзац вкладки Главная ленты или окна диалога Абзац, вызываемого кнопкой, расположенной в правом нижнем углу группы Абзац. Выравнивание может принимать четыре значения: по левому краю, по центру, по правому краю и по ширине. Отступы слева и справа определяют расстояние текста от границы текста, но не от края страницы. Первая строка может иметь три значения: без выступа, выступ вправо – красная строка или выступ влево – висячая строка. Часть параметров абзацев можно регулировать с помощью горизонтальной линейки. На горизонтальной линейке имеется три маркера: два в левой части и один в правой части линейки. Эти маркеры позволяют устанавливать отступы и выступ первой строки.

Табуляторы

Выше вертикальной линейки прокрутки находится кнопка для установки табуляторов. Табуляторы позволяют управлять размещением текста. По умолчанию текст выравнивается по левому краю, этому состоянию соответствует маркер табуляции ∟, маркер ⊥ означает выравнивание по центру, а маркер ┘ – выравнивание по правому краю, четвертый тип маркера – выравнивание по десятичному разделителю и пятый тип маркера – с чертой. Маркер применяется к выделенному абзацу или абзацам. Установить позиции табуляции можно также с помощью команды Табуляция окна диалога Абзац.

Границы и заливка

Границы и заливка применяются для выделения текста, оформления страниц и абзацев. Для этой цели можно воспользоваться кнопкой Границы в группе Абзац вкладки Главная ленты или

командой Границы страниц в группе Фон страницы вкладки Разметка страницы ленты. Выделите текст, который требуется заключить в рамку, и введите команду Границы страниц из группы Фон страницы – открывается окно диалога Границы и Заливка. Окно диалога имеет три вкладки. Вкладка Граница позволяет установить границы на выделенный фрагмент текста, а вкладка Страница позволяет установить границы страницы для всего документа или раздела, первой страницы или всех страниц, кроме первой.

Списки

Списки могут быть нумерованные, маркированные и многоуровневые (иерархические). Для оформления списков необходимо воспользоваться кнопками в группе Абзац вкладки Главная. Набранный текст следует предварительно выделить, затем выбрать в раскрывающемся списке вида списка нужный образец маркера (номера) и щелкнуть по нему мышью. При необходимости можно изменить значок или стиль оформления номера.

Группа Редактирование

В группе Редактирование вкладки Главная расположены кнопки Найти, Заменить и Выделить для выполнения редактирования. Первые две кнопки предназначены для поиска и замены текста в документе. А кнопка Выделить содержит подменю: Выделить все, Выбор объектов, Выделить текст, имеющий такой же формат, Область выделения.

Оформление документа

Документ – материальный носитель с зафиксированной на нем в любой форме информацией должен быть определенным образом оформлен. В него могут вставляться различные объекты: номера страниц, сноски, рисунки, формулы, таблицы, диаграммы, оглавление и т. п. Рассмотрим некоторые из возможностей текстового процессора по оформлению документов. Вставка различных объектов в документ осуществляется с помощью вкладок ленты Вставка, Ссылки, Рецензирование.

Колонтитулы

Колонтитул – это короткий текст, расположенный в верхнем или нижнем поле страницы. Это может быть краткое сообщение, например, фамилия автора документа, наименование раздела, номер страницы, дата и время разработки документа, рисунки и т. п. По умолчанию все колонтитулы одинаковые, однако можно

изменить эту настройку. В группе Текст вкладки Вставка собран ряд команд для вставки текста и объектов: Буквица, Надпись, Строка подписи, Экспресс-блоки, WordArt, а также команды вставки даты и времени и объектов.

Вставка номера страниц

Вставка номеров страниц осуществляется командой Номер страницы группы Колонтитулы вкладки Вставка. При установке номера страницы необходимо также указать, с какой страницы следует начать нумерацию страниц: с четной страницы, с нечетной страницы или выбрать другой вариант из списка Начать раздел в окне диалога Параметры страницы на вкладке Источник бумаги.

Надпись

Данный объект позволяет вставлять предварительно отформатированный текст. Это по сути дела текстовый документ, заключенный в рамку, к которому можно применять все элементы форматирования текста, а также вставлять другие объекты: рисунки, картинки, формулы и т. п.

Буквица

Позволяет создать большую заглавную букву в начале. Для создания буквицы выделите заглавную букву в первом слове предложения и введите команду Вставка, Текст, Буквица. В окне диалога выберите способ размещения буквицы: в тексте или на поле.

Дата и время

Дата и время вставляются командой Дата и Время из группы Текст. После ввода команды открывается одноименное окно диалога, которое позволяет вставлять в документ поле даты и времени.

Строка подписи

Строка подписи – позволяет вставить цифровую подпись, заверяющую подлинность документа. Для создания цифровой подписи законодательством Республики Беларусь установлены специальные процедуры.

WordArt

Данная команда позволяет вставлять броские, красочные заголовки, выбираемые из коллекции шрифтов. При выборе команды на экране появляется шаблон, в который необходимо ввести свой текст. К данному тексту можно добавлять некоторые элементы форматирования, например, курсив.

Команда Объект

Данная команда позволяет вставлять текст из файлов, а также загружать в документ объекты, подготовленные другими приложениями: Word, Excel, Power Point, Mathcad, редактор формул Microsoft Equation 3.0 и др.

Вставка спецсимволов

Вставка символов, отсутствующих на клавиатуре, осуществляется с помощью команды Символы из группы Символы.

Вставка формул

Для записи в текст документа формул применяется специальная программа Microsoft Equation 3.0, которая вызывается командой Объект, Вставка объекта из группы Текст вкладки Вставка. При загрузке программы на экране появляется панель инструментов, которая содержит набор элементов для вставки в формулы. При выборе любого значка открывается всплывающая панель с набором значков, соответствующих выбранной теме. Каждый значок имеет знакоместа, куда вписываются символы. Выбор знакоместа осуществляется мышью.

Вставка рисунков

Вставка рисунков осуществляется с помощью команд из группы Иллюстрации вкладки Вставка. Технология вставки рисунков следующая: щелкните мышью по значку Картинка – откроется окно диалога Картинка. Откройте список Искать объекты и установите флажки у той группы объектов, которые хотите найти, щелкните по кнопке Начать. Открывается библиотека рисунков. Выберите нужный рисунок и щелкните по нему мышью – рисунок вставляется в документ в точке вставки. Команда Рисунок позволяет вставить рисунок из файла.

Сетка

Для удобства рисования можно установить на лист сетку командой Сетка из группы. Показать вкладки Вид. Границы сетки действуют как магнит, притягивая рисуемые объекты к узлам сетки.

Вставка объектов вкладки Ссылки

Вкладка Ссылки содержит шесть групп команд: Оглавление, Сноски, Ссылки и списки литературы, Названия, Предметный указатель, Таблица ссылок. Указанные группы команд позволяют подготовить к изданию брошюру, дипломный проект, диссертацию или учебное пособие в соответствии с требованиями, предъявляемыми к печатной продукции.

Оглавление

Оглавление – это список заголовков разделов, подразделов с указанием номеров страниц. В редакторе Word вставка оглавления предельно упрощена. Для вставки оглавления выполните следующее:

– создайте заголовки и подзаголовки к тексту документа и оформите их с помощью стилей заголовков группы Стили вкладки Главная ленты. Заголовок первого уровня должен быть самым крупным. Заголовки следующих уровней имеют меньшую высоту шрифта. Редактор позволяет создавать до 9 уровней заголовков;

– укажите место вставки списка;

– введите команду Оглавление на вкладке Ссылки. Откроется список стандартных вариантов оформления оглавления. Если эти варианты не устраивают, то щелкните по кнопке Оглавление в конце списка. В этом случае открывается окно диалога для выбора другого формата и числа уровней оглавления.

Сноски

Сноска – это краткий комментарий к тексту, пояснение какого-либо термина, описание событий, связанных с какой-нибудь датой, и тому подобное. Различают обычную и концевую сноски. Обычная сноска помещается в конце текущей страницы, концевая сноска – в конце документа. Для вставки сноски установите курсор после текста, к которому будет относиться сноска, введите команду Вставить сноску из группы Сноска.

Ссылки и списки литературы

Ссылки облегчают создание списка литературных источников, использованных при разработке документа. В группе Ссылки и списки литературы на вкладке Ссылки размещены кнопки Управление источниками, Стиль, Список литературы и Вставить ссылку. Список Стиль позволяет выбрать стиль оформления списка литературы. Список Вставить ссылку добавляет в текст документа ссылку на источник фрагмента текста, вставленного в документ, в соответствии с выбранным стилем. Ссылка выбирается из списка.

Названия. Команда Вставить название из группы Названия на вкладке Ссылки позволяет автоматизировать нумерацию рисунков, таблиц, формул и облегчает составление списка иллюстраций документа. При вводе команды открывается окно диалога Название. После номера рисунка следует ввести название этого рисунка. Номера рисунков программа формирует автоматически.

Перекрестная ссылка. Для создания перекрестной ссылки выполните следующее: в месте перекрестной ссылки напишите нужный текст; выделите слово Сноски (на этом месте может быть любой текст, так как он будет заменен на выбранный) и введите команду Вставка, Перекрестная ссылка – открывается окно диалога Перекрестная ссылка; выберите в списке «Тип ссылки» требуемый тип, например, Заголовок; в списке «Вставить ссылку на:» выберите объект, на который будете ссылаться (текст заголовка, номер страницы и др.); в окне «Для какого заголовка:» выделите заголовок Сноски; нажмите кнопку Вставить. Если предполагается создавать гиперссылку, то установите флажок «Вставить как гиперссылку» в окне диалога. Для завершения работы по созданию перекрестной ссылки щелкните по кнопке Вставить.

Закладка. Закладка предназначена для присвоения имени определенной позиции в документе. На закладки можно ссылаться при создании гиперссылок и ряда других полей Word. Для вставки закладки выделите объект (текст, формулу, заголовок, таблицу и др.) и введите команду Вставка, Ссылки, Закладка. Укажите имя закладки. Чтобы просмотреть закладки, введите команду Файл, Параметры, Дополнительно, и в группе Показывать содержимое документа установить/снять флажок. Показывать закладки. Закладки выделяются квадратными скобками. Для удаления закладки введите команду Вставка, Ссылки, Закладка, выделите нужное имя закладки и щелкните по кнопке Удалить в окне диалога.

Создание таблиц

Команды для работы с таблицами в текстовом процессоре Word 2016 размещаются в группе команд Таблица вкладки Вставка и на вкладке Работа с таблицами ленты. Вкладка Работа с таблицами открывается при создании или выделении таблицы. Она имеет две вкладки: Конструктор и Макет.

Создание таблицы осуществляется командой Таблица на вкладке Вставка. После ввода команды открывается меню команды Таблица. Оно позволяет создать новую таблицу несколькими способами:

– выделить мышью нужное число строк и столбцов в макете таблицы;

– вставить таблицу командой Вставить таблицу. После ввода команды откроется окно диалога, в котором необходимо указать требуемое число строк и столбцов;

– нарисовать таблицу;

– вставить таблицу Excel;

– вставить таблицу из набора шаблонов таблиц – Экспресс-таблицу.

Таблица состоит из строк и столбцов. По умолчанию все столбцы нумеруются латинскими буквами от *A* до *Z*, а строки цифрами. Поэтому при написании формул к ячейке следует обращаться по ее адресу, например, *S*1. Каждая ячейка таблицы представляет собой самостоятельный документ, в который можно помещать числа, текст, рисунки, даты, таблицы. Редактируется текст в ячейке по тем же правилам, что и в основном документе.

Шаблоны

Шаблон – это набор параметров форматирования текста, абзацев, списков, элементов автотекста, макросов. Редактор Word имеет стандартный шаблон Normal, который загружается при открытии нового документа. Кроме того, он имеет большое количество других шаблонов для создания записок, факсов, писем, публикаций и т. п. Чтобы найти требуемый шаблон, введите команду Файл, Создать, затем выберите подходящий по содержанию шаблон и щелкните по кнопке Создать. Шаблон может содержать поля ввода с текстом, который нужно заменить. В документ можно вставлять поля различного вида с помощью команды Вставка – Экспресс-блоки – Поле.

Word 2016 позволяет создавать и собственные шаблоны – пользовательские. Для создания пользовательского шаблона введите команду Файл, Создать и выберите в окне диалога папку Мои шаблоны: откроется окно диалога Создать. Активизируйте значок Новый документ, установите в группе Создать переключатель Шаблон и щелкните по кнопке ОК. При сохранении документа шаблон сохраняется с расширением .dotx (.dot) в папке Мои шаблоны.

В шаблон документа введите постоянный текст и вставьте необходимые поля с помощью команды Вставка, Экспресс-блоки (Автотекст, Поле и Организатор стандартных блоков). В шаблон можно вставлять такие объекты, как текст (A), поле ввода (ab), флажки, списки. Элементы управления для вставки в шаблоны размещены в группе Элементы управления вкладки Разработчик ленты. В режиме ограниченной функциональности эти кнопки недоступны.

Создание шаблона пользователя

В группе Элементы управления вкладки Разработчик имеются следующие элементы для создания полей:

– форматированный текст (Аа) – ввод форматированного текста;

– обычный текст (Аа) – ввод обычного текста;

– рисунок – вставка элемента управления содержимым «рисунок»;

– коллекция стандартных блоков – служит для вставки стандартных блоков;

– поле со списком – содержит текстовое поле с раскрывающимся списком, позволяет вносить данные во время работы;

– раскрывающийся список – содержит список данных;

– выбор даты – служит для вставки даты;

– флажок – может иметь два состояния: установлен или снят.

Установка элементов управления осуществляется в режиме конструктора (режим установлен по умолчанию). Настройка параметров элементов управления также осуществляется в режиме конструктора, вызов свойств объектов осуществляется кнопкой Свойства в группе Элементы управления или через контекстное меню.

Слияние документов. Подготовка документов к рассылке

Часто приходится рассылать один и тот же документ по разным адресам, например, приглашение на юбилей учебного заведения, симпозиум и др. Программа Word позволяет автоматизировать эту работу. Для подготовки к рассылке документа, оформления конвертов и наклеек имеется Мастер слияния документов. Для выполнения операции слияния необходимо подготовить шаблон документа с полями слияния и источник данных для заполнения этих полей. Шаблон документа может быть подготовлен заранее и сохранен на диске как документ Word или создан в процессе выполнения работы. Источник данных представляет собой обычную таблицу. Таблица может быть подготовлена средствами Word или другими приложениями, например, с помощью электронной таблицы, системы управления базой данных и др. Каждая строка таблицы является записью, относящейся к одному объекту, а каждая ячейка – элементом данных. Выполнение операций по подготовке документов к рассылке осуществляется с помощью команд Начать слияние и Выбрать получателей вкладки Рассылка.

Макрокоманды

Макрокоманды, или макросы, представляют собой набор команд, с помощью которых можно автоматизировать выполнение повторяющейся задачи. Макросы позволяют легко и быстро выполнять часто повторяющиеся действия и тем самым повышать производительность труда пользователя. Прикладные приложения Макрокоманды хранятся в шаблонах. Выбором шаблона для сохранения ограничивается круг документов, в которых можно использовать ту или иную макрокоманду. Место хранения макроса указывается при его создании.

Макрокоманда может быть написана с использованием встроенного языка программирования Visual Basic или записана с помощью средств записи программы Word. Макрокоманда имеет имя. Имя макрокоманды может содержать до 36 символов, в имени не допускается использование пробелов. Макрокоманде можно назначить комбинацию клавиш или кнопку на панели инструментов. Запись макроса средствами записи программы Word. Проще всего создать макрос с помощью встроенных средств записи. Продумайте последовательность действий, необходимых для выполнения нужной операции, а затем повторите их в режиме записи макроса. Это необходимо делать всегда, так как в макрос будут записаны все команды, в том числе и ошибочные.

Запись макроса с помощью встроенного языка программирования Visual Basic

Для записи макроса с помощью встроенного языка программирования Visual Basic введите команду Visual Basic в группе Код вкладки – Разработчик открывается окно разработки проекта. Если нет вкладки Разработчик, то выполните следующие действия: введите команду Файл – Параметры – Настройка ленты и в списке Основные вкладки установите флажок «Разработчик». Введите команду Insert – Module – открывается окно разработки программы Module1. Введите команду Insert – Procedure – открывается окно добавления процедуры Add Procedure. Введите имя процедуры, например, Codirovka, установите переключатели Sub – ключевое слово заголовка процедуры и Public – общая (процедура будет доступна всем формам проекта) и щелкните по кнопке ОК – программа возвращается в окно проекта Module1. В этом окне появится шаблон процедуры, две строки команд, между которыми необходимо записать текст программы:

Public Sub Codirovka()Текст программы
End Sub

Между командами Public Sub и End Sub записывается или копируется туда текст необходимой программы макроса.

2.1.3. Графические редакторы

С развитием информационных технологий пользователи получили возможность создавать работы, требующие внедрения графических объектов. Для этих целей разработаны специальные программные средства – графические редакторы, позволяющие удовлетворить потребности работы с графикой, как опытных пользователей, так и непрофессионалов.

В настоящее время активно применяются программы Adobe PhotoShop (цветоделение и обработка изображений), Quark Press (верстка периодики), CorelDraw (векторный графический редактор), PowerPoint (разработка сценария и стиля презентаций, слайд-фильмы), Auto-CAD (черчение и конструирование), Adobe Illustrator (дизайнерство), Corel ArtShow (библиотека иллюстраций, созданных художниками всего мира), Microsoft Paint – многофункциональный, но в то же время простой в использовании растровый графический редактор компании Microsoft, входящий в состав всех операционных систем Windows, начиная с первых версий.

Существует специальная область информатики, изучающая методы и средства создания и обработки изображений с помощью программных и аппаратных вычислительных комплексов, – компьютерная графика. Она охватывает все виды и формы представления изображений, доступных для восприятия человеком либо на экране монитора, либо в виде копии из внешнего носителя.

В зависимости от способа формирования изображений компьютерную графику принято подразделять на растровую, векторную и фрактальную.

Растровая графика применяется при разработке электронных (мультимедийных) документов и полиграфических изданий. Она образует изображение множеством точек (пикселей), с каждой из которых можно работать отдельно (в том числе и закрашивать в определенный цвет). Большинство графических редакторов предназначены для работы с растровыми изображениями, ориентированы

не столько на создание изображений, сколько на их обработку. В интернете применяются главным образом растровые иллюстрации. Одним из недостатков растровой графики является снижение качества изображений при их увеличении. Если в оригинале предусмотрено определенное количество точек, то при большом масштабе увеличивается их размер, становятся заметны элементы растра, что искажает саму иллюстрацию.

Векторная графика образует изображение системой отдельных объектов, которыми могут быть различные геометрические фигуры, составленные из прямых, дуг, окружностей. Если в растровой графике базовым элементом изображения является точка, то в векторной графике – линия (вектор). Линия описывается математически как единый объект, и потому объем данных для отображения объекта средствами векторной графики существенно меньше, чем в растровой графике. Из простейших объектов создаются более сложные. Программные средства для работы с векторной графикой предназначены в первую очередь для создания иллюстраций и в меньшей степени для их обработки, но широко используются в рекламных средствах и издательствах, при создании чертежей и карт.

Фрактальная графика очень похожа на векторный способ отображения графической информации. Фрактальная графика, как и векторная, базируется на математических вычислениях. Однако базовым элементом фрактальной графики является сама математическая формула, т. е. изображение строится исключительно по уравнениям. Таким способом строят как простейшие регулярные изображения, так и сложные иллюстрации, имитирующие, например, природный ландшафт.

Отдельной областью считается трехмерная (3D) графика, реализующая приемы и методы построения объемных моделей объектов в пространстве. Как правило, в ней сочетаются векторный и растровые методы формирования изображений. Популярным пакетом обработки трехмерной графики является 3D Studio Max.

Основные форматы графических файлов: BMP – стандартный формат растровых изображений воспринимается всеми графическими редакторами; GIF – распространенный формат, получил популярность благодаря высокой степени сжатия; форматы JPEG и PCX.

Растровый графический редактор компании Microsoft Paint

Microsoft Paint – многофункциональный, но в то же время простой в использовании растровый, графический редактор компании Microsoft, входящий в состав всех операционных систем Windows, начиная с первых версий.

Первая версия Paint появилась в Windows 1.0. В Windows 3.0 был переименован в PaintBrush, но позже опять был переименован в Paint. В Windows 7 Paint впервые был полностью переработан, получил ленточный интерфейс, дополнительные кисти и фигуры, схожие с библиотекой Microsoft Office. Краткий обзор нововведений:

1. 9 разновидностей кисти (Brush).

2. Обновилась библиотека фигур: к стандартным эллипсу, прямоугольнику, вектору, кривой, многограннику и скругленному прямоугольнику добавилось еще 17 фигур, среди которых: треугольник равнобедренный, треугольник прямоугольный, ромб, пяти- и шестиугольник, стрелки вправо, влево, вверх и вниз; звезды: четырех-, пяти- и шестиугольная; прямоугольный, круглый и «думающие» пузыри для комиксов, сердце и молния.

3. Нарисовав фигуру, можно еще настроить ее параметры: повернуть, растянуть, изменить цвет и фактуру.

4. 7 разновидностей заливки/контура.

5. Также в меню «Вид» добавлены: новая линейка, режим предпросмотра печати.

6. Возможность получения материала для редактирования со сканера.

7. Теперь возможно использовать разные стили для каждого фрагмента текста внутри одной рамки.

Многократное увеличение или уменьшение инструмента. Выберите один из инструментов: «Кисть», «Ластик», «Линия» или «Распылитель» и нажмите клавиши Ctrl и NumPad +. Чем дольше держать нажатой эту комбинацию, тем больше будет увеличиваться инструмент. Соответственно, если зажать Ctrl и NumPad, то инструмент будет уменьшаться.

Пипетка. Инструмент «одноразового» действия – после применения автоматически возвращает тот инструмент, который был активен до ее включения. Левой кнопкой берет основной цвет, правой – фоновый. С нажатым Ctrl берет «третий» цвет.

Заливка. В связи с тем, что в Paint не используется полупрозрачность, заливка аккуратно и четко заполняет области, обведенные любыми кривыми линиями. Пользуясь заливкой и пипеткой, можно быстро стереть множество деталей одного цвета на фоне другого – достаточно залить этот фон цветом деталей, а затем вернуть ему его цвет.

Замена цвета. Инструмент «Ластик» работает, фактически рисуя вторым – «фоновым» – цветом там, где им проведут при нажатой левой кнопке мыши. Однако если им водить при нажатой правой кнопке, то он будет «стирать» фоновым цветом только то, что нарисовано первым – «основным» – цветом.

Выделение. Выделенный фрагмент оказывается «плавающим» (он может быть перенесен в любое место рабочей области без изменения самой картинки), а его место заполняется фоновым цветом. При этом если в момент начала перетаскивания нажата клавиша Ctrl, в начальной позиции остается «штамп» – туда впечатывается копия плавающего выделения (при первоначальном перемещении получается так, как будто унесена копия выделенного, а на исходном месте ничего не изменилось). Если нажата клавиша Shift, то подобный штамп делается и во всех промежуточных точках перемещения.

Прозрачное выделение. Прозрачным считается цвет, который в момент выделения назначен фоновым (назначается правой кнопкой мыши – как с палитры, так и с рабочей области инструментом взятия цвета (пипеткой)).

Пользовательская кисть. Сделайте сбоку от своего изображения свободное поле, отодвинув в сторону правый или нижний край полотна за имеющийся на нем маркер (добавленная область будет закрашена «фоновым» цветом). Нарисуйте там, на фоновом цвете, картинку, которую хотите использовать как кисть. Выделите эту часть изображения в режиме прозрачного фона и, зажав Ctrl, чтобы оставить ее копию для следующих применений, перенесите выделение в то место, где должен начаться штрих такой кистью. Теперь зажмите клавишу Shift и перемещайте выделенный фрагмент. Он будет оставлять след – так же, как оставляет след инструмент кисть (можно сказать, что шлейф от изображения примет вид карточной колоды).

Вставка. При вставке размер рабочей области увеличивается так, чтобы вмещать в себя вставляемое изображение. Пользуясь этим, можно измерять размеры изображений – достаточно перед вставкой уменьшить за нижний правый угол размеры рабочей области до минимальных, а после вставки посмотреть на «атрибуты» изображения.

Графический редактор Adobe Photoshop

Adobe Photoshop – многофункциональный графический редактор, разработанный фирмой Adobe Systems. В основном работает с растровыми изображениями, однако имеет некоторые векторные инструменты. Часто эту программу называют просто Photoshop. В настоящее время Photoshop доступен на платформах Windows, OS X в мобильных системах iOS и Android. Для версий 8.0 и CS6 возможен запуск под Linux с помощью альтернативы Windows API – Wine.

Несмотря на то, что изначально программа была разработана как редактор изображений для полиграфии, в данное время она широко используется и в веб-дизайне. Photoshop тесно связан с другими программами для обработки медиафайлов, анимации и другого творчества. Photoshop CS3 в версии Extended поддерживает также работу с трехмерными слоями.

Первая версия Photoshop появилась в 1987 году. Ее создал студент Мичиганского университета Томас Нолл (англ. Thomas Knoll) для платформы Macintosh. Дальнейшее развитие программа получила в коммерческой версии Photoshop CS3, которая была разработана в апреле 2007 года. Список нововведений включает в себя новый интерфейс, увеличенную скорость работы, новые фильтры и инструменты, а также приложение, позволяющее осуществлять предварительный просмотр работы в шаблонах популярных устройств, например, мобильных телефонов.

В программе Adobe Photoshop Extended можно открывать и работать с 3D-файлами, создаваемыми другими программами. Возможно использовать трехмерные файлы для внедрения в двумерное фото. Доступны некоторые операции для обработки 3D-модели, такие как работа с каркасами, выбор материалов из текстурных карт, настройка света.

Также можно создавать надписи на 3D-объекте, вращать модели, изменять их размер и положение в пространстве. Программа включает в себя также команды по преобразованию плоских фотографий

в трехмерные объекты определенной формы, такие как, например, банка, пирамида, цилиндр, сфера, конус и др.

Для имитации движения в Photoshop можно создавать кадры мультипликации, используя слои изображения. Можно создавать видеоизображения, основанные на одной из многих заданных пиксельных пропорций. После редактирования можно сохранить работу в виде файла GIF-анимации или PSD, который впоследствии можно проиграть в других видеопрограммах, таких как Adobe Premiere Pro или Adobe After Effects. Доступно открытие или импортирование видеофайлов и последовательности изображений для редактирования и ретуширования, создание видеоряда мультипликации и экспорт работ в файл формата QuickTime, GIF-анимацию или последовательность изображений. Видеокадры можно отдельно редактировать, трансформировать, клонировать, применять к ним маски, фильтры, разные способы наложения пикселов, на них можно рисовать, используя различные инструменты.

С помощью программы Photoshop Extended можно рассматривать MatLab-изображения, обрабатывать их в программе PhotoShop, комбинировать команды MatLab с технологиями обработки изображений PhotoShop. Как только устанавливается соединение с программой PhotoShop из программы MatLab и осуществляется ввод команд в командную строку MatLab, эти управляющие воздействия незамедлительно выполняются в PhotoShop. Файлы, подготовленные в программе MatLab, имеют расширение m, fig, rpt, mat, mdl. Коммуникация между PhotoShop и MatLab использует интерфейс PhotoShop, JavaScript и библиотечный интерфейс MatLab.

Графический редактор CorelDRAW

Программа CorelDRAW, разработанная фирмой Corel, была выпущена в 2002 г. Эта программа обладает удивительной универсальностью и мощностью, будучи в равной степени полезной и в промышленном дизайне, и в разработке рекламной продукции, и в подготовке публикаций, и в создании изображений для web-страниц. Это делает программу весьма эффективной в качестве первого программного средства для приступающих к изучению машинной графики в целом или векторной графики в частности. Пользовательский интерфейс CorelDRAW построен очень рационально, с высокой степенью унификации. Все изображения, с которыми работают программа, разделяются на два класса: точечные и векторные.

Объектно-ориентированный подход

CorelDRAW представляет собой интегрированный объектно-ориентированный пакет программ для работы с иллюстративной графикой. CorelDRAW представляет собой не отдельную программу, ориентированную на решение какой-либо одной четко поставленной задачи, а совокупность программ, ориентированных на решение множества различных задач, возникающих при работе пользователя в определенной прикладной области, а именно – в области иллюстративной графики. Интегрированность пакета следует понимать в том смысле, что входящие в него программы могут легко обмениваться данными или последовательно выполнять различные действия над одними и теми же данными.

Иллюстративная графика – это прикладная ветвь машинной графики, сравнительно недавно выделившаяся в отдельное направление наряду с графикой деловой, научной и инженерной. К области иллюстративной графики относятся в первую очередь рисунки, коллажи, рекламные объявления, заставки, постеры – все, что принято называть художественной продукцией. Объекты иллюстративной графики отличаются от объектов других прикладных областей своей первичностью – они не могут быть построены автоматически по некоторым исходным данным, без участия художника или дизайнера. В отличие от них такие графические изображения, как диаграммы (деловая графика), чертежи и схемы (инженерная графика), графики функций (научная графика), представляют собой лишь графический способ представления первичных исходных данных – как правило, таблицы (или аналитической модели, представленной в другой форме). В этом состоит их вторичность, производность.

Особенность объектной ориентации пакета состоит в том, что все операции, выполняющиеся в процессе создания и изменения изображений, пользователь проводит не с изображением в целом и не с его мельчайшими, атомарными частицами (пикселами точечного изображения), а с объектами – семантически нагруженными элементами изображения. Начиная со стандартных объектов (кругов, прямоугольников, текстов и т. д.), пользователь может строить составные объекты (например, значок в рассмотренном выше примере) и манипулировать ими как единым целым. Таким образом, изображение становится иерархической структурой, на самом верху которой находится иллюстрация в целом, а в самом низу – стандартные объекты.

Вторая особенность объектной ориентации пакета состоит в том, что каждому стандартному классу объектов ставится в соответствие уникальная совокупность управляющих параметров, или атрибутов класса.

Третья особенность объектной ориентации пакета состоит в том, что для каждого стандартного класса объектов определен перечень стандартных операций. Например, описанный выше прямоугольник можно развернуть, масштабировать, закруглить ему углы, преобразовать его в объект другого класса – замкнутую кривую.

2.1.4. Электронные таблицы на примере Microsoft Excel

Электронная таблица – интерактивная система обработки информации, упорядоченной в виде таблицы с поименованными строками и столбцами. Табличный процессор − категория прикладного программного обеспечения, предназначенного для работы с электронными таблицами. Инструментарий электронных таблиц включает мощные математические функции, позволяющие вести сложные инженерные, финансовые, статистические и прочие расчеты. Одними из самых популярных табличных процессоров сегодня являются Microsoft Excel, входящий в состав пакета Microsoft Office, и Open Office Calc, входящий в состав открытого для доступа пакета Open Office.

Для загрузки программы Microsoft Excel следует щелкнуть мышью по значку приложения Microsoft Excel на рабочем столе или панели задач операционной системы Microsoft Windows 7. Для выхода из программы введите команду Файл – Выход.

Рабочее окно Microsoft Excel представляет собой стандартное окно Microsoft Office со специфическими элементами. В верхней части окна расположена строка заголовка, в которой располагаются кнопка системного меню, название приложения и имя файла, загруженного в окно (в начале работы по умолчанию выводится имя файла Книга 1), кнопки свертывания и развертывания окна программы. Ниже расположены: кнопка меню Файл, лента, панель быстрого доступа, строка ввода данных, рабочее окно, строка состояния.

Лента имеет ту же структуру, что и лента Microsoft Word 2010. Она имеет восемь постоянно открытых вкладок, ярлычки которых видны на экране: Главная, Вставка, Разметка страницы, Формулы, Данные, Рецензирование, Вид и Надстройки. Каждая вкладка со-

стоит из групп команд, в которых расположены кнопки управления. У большинства групп в правом нижнем углу имеется кнопка, которая вызывает окно диалога настройки параметров. Для увеличения рабочей области ленту можно свернуть соответствующей командой контекстного меню или комбинацией клавиш Ctrl + F1. По умолчанию на экран не выводится одна важная вкладка ленты – Разработчик. Чтобы добавить ее на ленту, выполните следующее: введите команду Файл – Параметры – Настройка ленты, в окне Основные вкладки установите флажок «Разработчик» и щелкните по кнопке ОК.

Панель быстрого доступа содержит команды, используемые наиболее часто, состав этих команд может настраиваться пользователем. Положение Панели быстрого доступа может быть определено пользователем: над лентой или под лентой. Для изменения положения панели быстрого доступа воспользуйтесь контекстным меню.

Строка формул имеет три поля: поле адреса ячейки (Имя), поле управляющих клавиш и поле ввода данных (Строка формул). Поле Имя представляет собой раскрывающийся список, в нем указан адрес текущей ячейки или ее имя. Если щелкнуть мышью по этому полю, ввести адрес ячейки и нажать клавишу Enter, то курсор электронной таблицы перейдет в указанную ячейку. Этот прием перехода к нужной ячейке называется непосредственной адресацией.

Строка состояния расположена в нижней части рабочего окна. В левой части строки состояния выводится информация о текущем состоянии электронной таблицы. В правой части расположены кнопки управления режимами просмотра таблицы: обычный, разметка страницы и страничный, а также элементы управления масштабом представления информации. Строка состояния имеет контекстное меню, которое позволяет управлять представлением информации в строке состояния. Над строкой состояния размещены кнопки навигации, ярлычки листов, специальная кнопка Вставить лист и горизонтальная линейка прокрутки.

Рабочая книга, рабочий лист

Информация в электронной таблице сохраняется в виде рабочих книг. Имя книги выводится в строке заголовка. Рабочая книга состоит из листов различного типа. Максимально возможное число листов в рабочей книге – 256. Рабочий лист состоит из пронумерованных строк и столбцов. Столбцы рабочих листов озаглавлены

латинскими буквами от *A* до *Z* и их комбинациями по два и три символа, например, *AA*, *AB*, *IU*, …,*XFD*. Строки пронумерованы цифрами.

Рабочий лист содержит 16 385 столбца и 1 058 576 строк. На пересечении строк и столбцов образованы ячейки. В одной из ячеек расположен контур выделения – курсор электронной таблицы. Рабочий лист имеет номер, который указан на ярлыке. Если щелкнуть правой кнопкой мыши по ярлыку, то откроется контекстное меню с перечнем команд для управления рабочим листом. Рабочие листы можно добавлять, удалять, копировать, переименовывать, перемещать, группировать, разгруппировывать.

Ячейка и ее свойства

Ячейка является основным элементом таблицы. В качестве содержания ячейки выступают числовые и текстовые константы, а также выражения (формулы).

Ячейка – область, образованная пересечением строки и столбца. Она обозначается номером столбца и строки, на пересечении которых находится. Например, C1, AV9999.

Диапазон (группа) – непрерывная область ячеек, обозначенная номерами начальной и конечной ячеек, разделенными двоеточием или точкой, например, A1:C10, D8.H12. Ячейке или диапазону может быть присвоено уникальное имя. Ячейка характеризуется следующими параметрами: адрес, содержание, значение, формат, статус.

Адрес ячейки: адрес ячейки может быть абсолютным, относительным и смешанным.

Относительный адрес: A1, E7. Относительный адрес в операциях копирования автоматически настраивается.

Абсолютный адрес: A1, E7. Абсолютный адрес ячейки не меняется в операциях копирования, вставки или удаления ячеек, строк и столбцов.

Смешанный адрес: $A1, A$1. Если ячейке присвоен смешанный адрес, то при копировании будет меняться только тот параметр, перед которым не стоит знак $. Например: $D6 – при копировании ячейки будет меняться только номер строки; D$6 – при копировании будет меняться только адрес столбца.

Присвоение имени ячейке

Ячейке или диапазону ячеек может быть присвоено имя. В Microsoft Excel 2010 для работы с именами создана отдельная группа – Определенные имена на вкладке Формулы. Присвоение имени осуществляется командой Присвоить имя.

Для присвоения имени ячейке или диапазону ячеек необходимо:

1. Выделить ячейку (диапазон ячеек).

2. Ввести команду Формула, Присвоить имя.

3. Выбрать область действия имен из списка Область: книга или лист. Ввести в строке ввода Имя диалогового окна Создание имени имя ячейки и щелкнуть по кнопке ОК. Область, которой присваивается имя, отображается в строке Диапазон.

Содержание ячейки

Содержание ячейки – это то, что вводится в нее через строку ввода. Поэтому ячейка может либо быть пустой, либо содержать данные: текст, текстовую константу, формулу, дату, время.

Значение ячейки. Значением ячейки могут быть число, текстовая константа, дата, время, сообщения об ошибках. Значением пустой ячейки и ячейки, содержащей текст, является ноль.

Текстовая константа – это строка символов, используемая в выражениях как операнд, при вводе текстовой константы она заключается в скобки и в кавычки.

Формат ячейки

К формату ячейки относятся такие параметры ячейки, как ширина, режим отображения формул, формат отображения числовых величин, размещение содержимого ячейки, шрифт, цвет, границы, статус ячейки. Все основные параметры настройки свойств ячейки в Excel 2016 вынесены на вкладку Главная ленты.

Статус ячейки

Ячейка может иметь два статуса: защищена или не защищена. В защищенную ячейку нельзя внести информацию или изменить ее содержание. Установка режима защиты осуществляется командами вкладки Защита окна диалога Формат ячеек. Эта команда позволяет установить защиту на ячейку или скрыть формулу. По умолчанию для всех ячеек установлен режим защиты. Режим защиты ячейки вступает в силу только после защиты листа командами Защитить лист из группы Изменения вкладки Рецензирование. Для отмены защиты ячейки достаточно отменить защиту листа командой Снять защиту листа. Можно защитить также структуру книги.

Ввод данных

Данные вводятся в Строку ввода данных или непосредственно в ячейку. В первом случае выделите ячейку, в которую вводятся данные, и щелкните по Строке ввода данных. Введите нужную информацию. Для окончания ввода нажмите клавишу Enter. Во вто-

ром случае выделите ячейку и вводите данные прямо в ячейку. По окончании ввода данных нажмите клавишу Enter. Для очистки ячейки выделите ее и нажмите клавишу Delete или Пробел и Enter.

Курсор таблицы, или контур выделения, представляет собой рамку, окаймляющую всю ячейку. В правом нижнем углу рамки на пересечении сторон располагается маленький черный квадрат – маркер заполнения. Этот маркер используется для заполнения ячеек рядом данных с постоянным шагом, а также для копирования формул.

Ввод текста. Признаком текста при вводе данных является апостроф ('), например, 'Исходные данные. По умолчанию вводимые данные в этом случае воспринимаются как текст.

Ввод даты. Дата вводится в формате ДД.ММ.ГГ или ДД.ММ.ГГГГ: день, месяц, год (17.05.14). В качестве разделителя используется точка. Электронная таблица позволяет выводить дату на экран в различных форматах.

Ввод текстовых констант. Для ввода текстовых констант необходимо ввести символ = и текст в кавычках: =”Текст”. Для преобразования чисел или числовых значений выражений в текстовые константы следует воспользоваться функцией ТЕКСТ(). Тип данных в ячейке определяется при первом вводе.

Ввод формул. Признаком формулы является знак =. Если при вводе формулы допущена ошибка, то программа выдает сообщение об ошибке. При вводе формулы без знака «равно» программа воспринимает вводимые данные как текст. Адреса ячеек вводятся только латинскими символами. При вводе вещественных чисел, используется десятичная запятая, а не точка. Для ввода формул или ознакомления с функциями Excel можно использовать Мастера функций.

Примеры записи формул

=A2+2 – сложение;

=ЕСЛИ(A2<B2;C3;D2·E17) – условное выражение. Если значение в ячейке A2 меньше значения в ячейке B2, то результат будет равен значению ячейки C3, иначе – произведению значений ячеек D2 и E17.

При записи формул, для указания адреса ячеек, значения которых не должны изменяться при копировании формул, следует обязательно использовать абсолютный адрес.

Формула может содержать ссылки на ячейки таблицы, расположенные, в том числе, и на другом рабочем листе.

Методика работы с электронной таблицей

1. Выбор ячейки. Установите курсор на ячейку или щелкните по ней мышью.

2. Выбор группы ячеек. Установите указатель мыши на первую ячейку группы, нажмите левую клавишу и протащите указатель по всем ячейкам группы. В конце области выделения отпустите клавишу мыши.

3. Для выбора нескольких несвязанных областей необходимо нажать и удерживать клавишу Ctrl, а затем выделить требуемые области.

4. Для отмены выделения ячеек щелкните мышью по чистому полю в любом месте экрана.

5. Выбор строк и столбцов. Для выбора одной строки (столбца) щелкните мышью по номеру строки (столбца). Для выбора группы строк (столбцов) установите указатель мыши на номер первой строки (столбца), нажмите клавишу мыши и протащите ее указатель по всем строкам (столбцам) выделяемой группы. Отпустите клавишу мыши.

Копирование ячеек. При копировании ячеек можно использовать команды Копировать и Вставить из группы Буфер обмена вкладки Главная ленты, команды контекстного меню.

Копирование с использованием маркера автозаполнения. Если копирование осуществляется в соседние ячейки, то его удобно выполнить с использованием маркера автозаполнения: выделите копируемую ячейку; зацепите мышью за маркер автозаполнения (подведите курсор к черному квадратику в правом нижнем углу курсора таблицы так, чтобы указатель мыши превратился в черный крестик) и протащите указатель мыши по всем ячейкам назначения.

Копирование с помощью команд группы Буфер обмена или контекстного меню

Копирование с помощью мыши. Для копирования содержимого ячеек, содержащих текстовую информацию или формулы с абсолютными адресами данных, выделите с помощью мыши ячейку или блок ячеек, подлежащих копированию, нажмите и удерживайте клавишу Ctrl, перетащите выделенные ячейки на новое место, отпустите кнопку мыши, а затем отпустите кнопку Ctrl.

Для копирования формул, содержащих ссылки на ячейки с относительными или смешанными адресами, следует воспользоваться командой Вставить, Специальная вставка: выделите копируемые ячейки; укажите место вставки; введите команду Вставить, Специальная вставка из группы Буфер обмена и щелкните по кнопке Вставить связь. В этом случае в формуле сохраняются ссылки на ячейки, содержащие данные. При изменении данных в ячейках автоматически будут меняться значения в ячейках источника данных и в ячейках назначения.

Оформление таблицы. Для оформления таблицы: обрамления, заливки цветом – можно воспользоваться вкладками Граница и Заливка окна диалога – Формат ячеек или одноименными кнопками в группе Шрифт вкладки Главная. Можно воспользоваться также командами Форматировать как таблицу и Стили ячеек из группы Стили вкладки Главная.

Предварительный просмотр. Настройка параметров страниц

Для просмотра таблицы введите команду Разметка страницы в группе Режимы просмотра книги вкладки Вид. В этом режиме лист представляется в том виде, как будет выглядеть при печати, можно просмотреть начало и конец страниц, верхний и нижний колонтитулы, на экран выводятся горизонтальная и вертикальная линейки. Для изменения параметров страницы откройте вкладку Разметка страницы и обратитесь к группе Параметры страницы.

Сохранение, открытие и печать таблицы

Перед печатью целесообразно сохранить документ на диске, для этого необходимо выполнить следующее: введите команду Сохранить или Сохранить как из меню Файл, укажите в строке ввода «Имя Файла» имя файла, при необходимости измените тип файла .xlsx на .xls для совместимости с предыдущими версиями программы Excel. Выберите диск и папку, щелкните по кнопке ОК. Открытие сохраненного ранее документа осуществляется командой Файл, Открыть. Выберите соответствующий диск, папку.

Настройка параметров таблицы

Приступая к работе с электронной таблицей, полезно ознакомиться с некоторыми настройками. Настройка параметров электронной таблицы осуществляется командой Файл, Параметры. После ввода команды открывается окно диалога Параметры Excel. Команды собраны в отдельные группы: язык, интервал времени, через который рабочая книга будет сохраняться и др.

Категории функций электронной таблицы

Для удобства вычисления в табличных процессорах имеются встроенные функции: математические/тригонометрические, инженерные, текстовые, статистические, финансовые, даты и времени, логические, функции для работы с базами данных/списками, информационные и функции категории ссылки/массивы; функции проверки свойств и значений и др. Кроме того, Excel содержит большое число надстроечных функций, которые используются для создания компьютерных программ в Excel, а также имеется возможность создания пользовательских функций и программ на Visual Basic for Applications. В группе Формулы размещены списки категорий функций, что позволяет легко находить нужные функции. Список Автосумма содержит функции Сумма, Среднее, Максимум, Минимум, Число. Команда Число вызывает функцию Счет, которая подсчитывает число непустых ячеек в выделенной области.

Каждый список содержит команду Вставить функцию, аналогичная команда имеется в группе Библиотека функций. Эта команда вызывает окно диалога Мастер функций. Мастер функций содержит окно для поиска функции по ее краткому описанию, список категорий функций и окно выбора функции. При создании функций пользователя в списке категорий появляется группа Пользовательские.

Генерирование данных

Часто бывает необходимо сгенерировать последовательность чисел, дат. Для этой цели можно использовать механизм автозаполнения. Чтобы заполнить несколько ячеек прогрессией, необходимо записать в смежные ячейки данные, отличающиеся на величину шага, выделить эти ячейки и перетащить маркер автозаполнения выделенного диапазона ячеек. Можно также воспользоваться командой Заполнить из группы Редактирование вкладки Главная. Данная команда имеет подменю: влево, вправо, вверх, вниз, прогрессия, выровнять.

Графические возможности электронной таблицы

Виды иллюстраций деловой графики. Табличные процессоры предлагают различные виды иллюстраций деловой графики (диаграмм), причем их построение облегчено за счет использования «Мастера диаграмм» − встроенных автоматизированных пошаговых процедур, позволяющих выбрать тип диаграммы и для него выполнить все необходимые операции, в том числе оформления различными компонентами.

Гистограмма показывает изменение данных за определенный период времени и иллюстрирует соотношение отдельных их значений. Категории располагаются по горизонтали, а значения − по вертикали. Гистограмма с накоплением демонстрирует вклад отдельных элементов в общую сумму.

Линейчатая диаграмма отражает соотношение отдельных компонентов. Категории расположены по горизонтали, а значения по вертикали. Она ориентирована на сопоставление значений. Линейчатая диаграмма с накоплением демонстрирует вклад отдельных элементов в общую сумму.

График представляет варианты отображений изменений данных за равные промежутки времени.

Круговая диаграмма отражает как абсолютную величину каждого элемента ряда данных, так и его вклад в общую сумму. На круговой диаграмме может быть представлен только один ряд данных. Такую диаграмму рекомендуется использовать, когда необходимо подчеркнуть какой-либо значительный элемент.

Точечная диаграмма показывает взаимосвязь между числовыми значениями в нескольких рядах и представляет две группы чисел в виде одного ряда точек в координатах *x* и *y*. Она отображает нечетные интервалы данных и часто используется для предоставления данных научного характера. При подготовке данных следует расположить в одной строке или столбце все значения переменной *x*, а соответствующие значения y – в смежных строках или столбцах.

Поверхностная диаграмма используется для поиска наилучшего сочетания двух наборов данных. Так, на топографической карте области с одним значением выделяются одинаковым узором и шрифтом.

Диаграмма с областями подчеркивает величину изменения в течение определенного периода времени, показывая сумму введенных значений, а также вклад отдельных значений в общую сумму.

Кольцевая диаграмма, как и круговая диаграмма, показывает вклад каждого элемента в общую сумму, но в отличие от круговой диаграммы может содержать несколько рядов данных.

Биржевая диаграмма часто используется для демонстрации цен на акции. Этот тип диаграммы применяется для отображения научных данных, например, изменения температуры.

Построение графиков и диаграмм

Для построения графиков и диаграмм в электронной таблице используется группа Диаграммы вкладки Вставка, а также вкладки Конструктор, Макет и Формат вкладки Работа с диаграммами. Непосредственно в группе Диаграммы можно выбрать тип диаграммы и ее вид. Мастер диаграмм позволяет использовать 11 стандартных типов диаграмм. Для построения графиков функций необходимо использовать Точечную диаграмму Тип График используется только для построения линейных диаграмм, так как он не позволяет связать функцию с аргументом. Выбранный вид диаграммы отображается на экране.

После построения графика активизируется вкладка Работа с диаграммами. Вкладка Конструктор позволяет изменить тип диаграммы и сохранить его как шаблон, поменять местами строки и столбцы при построении графиков функций и диаграмм.

Вкладка Макет позволяет управлять оформлением диаграммы. Другие типы позволяют также эффективно компоновать диаграммы и выбрать стиль их оформления (цвет линий, стиль линий, тип диаграммы, шрифт и так далее). Многообразны и доступны возможности оформления диаграмм, например, вставка и оформление легенд, меток данных, оформление осей, возможность вставки линий сеток и т. д.

Работа с матрицами

Электронная таблица позволяет выполнять линейные преобразования матриц: умножение, деление матриц на число, прибавление или вычитание чисел, а также операции над матрицами: сложение, умножение матриц, транспонирование, вычисление обратной матрицы и определителей. Средствами Excel можно решать системы линейных алгебраических уравнений, задачи многомерной оптимизации и др. Для этой цели электронная таблица имеет ряд функций для работы с матрицами, например:

МОБР (массив) – вычисление обратной матрицы;

МОПРЕД (массив) – вычисление определителя матрицы;

МУМНОЖ (массив; массив) – умножение матриц;

ТРАНСП (массив) – транспонирование матриц и др.

Решение систем линейных алгебраических уравнений

С помощью встроенных функций МОБР, МУМНОЖ и МОПРЕД операции решения систем линейных алгебраических уравнений выполняются достаточно эффективно. Например,

можно воспользоваться формулой вычисления вектора неизвестных через обратную матрицу A−1 и вектор свободных членов B: $X = A - 1 \cdot B$.

Пример. Решить систему линейных алгебраических уравнений матричным методом:

$$65{,}18x + 36{,}31y + 23{,}76z = 86{,}56$$
$$-17{,}98x + 23{,}89y + 27{,}55z = -38{,}07$$
$$23{,}75x + 13{,}95y + 58{,}12z = 53{,}97$$

Решение:

1. Внесите в ячейки B6–D8 значения коэффициентов при неизвестных.

2. Внесите в ячейки F6–F8 значения свободных членов системы уравнений.

3. Выделите диапазон ячеек B12: D14 и введите формулу МОБР(B6:D8), для завершения операции ввода нажмите комбинацию клавиш Ctrl + Shift + Enter.

5. Выделите диапазон ячеек F12:F14 и введите формулу МУМНОЖ(B12: D14; F6:F8). Для завершения ввода формулы нажмите комбинацию клавиш Ctrl + Shift + Enter. В ячейках F12–F14 появятся значения корней системы уравнений: 1,652508; – 0,88768; 0,466381.

2.1.5. Система подготовки презентаций

Презентация – это упорядоченная последовательность слайдов и слайд-фильмов, раздаточные материалы, а также конспект и планы докладов, хранящиеся в одном файле.

Слайд – отдельная страница презентации, включающая разные объекты презентаций: заголовок, текст, графику, диаграммы, таблицы, рисунки, рисованные объекты, фотографии, формулы, видеоклипы, видеофильмы. Слайды можно распечатывать на бумаге или на прозрачной пленке.

Раздаточный материал – это распечатанные в компактном виде несколько слайдов на одной странице с целью закрепления восприятия слушателями темы доклада и возможности самостоятельно вернуться к теме доклада.

Конспект доклада – текст доклада, при печати которого на каждой странице будут выведены уменьшенное изображение слайда и текст, составляющий его содержание.

Как правило, для создания презентаций используются два программных продукта: Microsoft PowerPoint или Open Office Impress. Файлы Power Point имеют расширение .pptx.

Общие принципы создания презентаций

Существуют некоторые общие принципы, которыми рекомендуется руководствоваться при создании презентаций.

1. На восприятие одного слайда необходимо от 1 до 5 минут.

2. На слайде может быть от 20 до 60 слов.

3. Каждый слайд должен иметь свой заголовок, т. е. один слайд – одна мысль.

4. Информация должна быть структурирована – списки, таблицы и рисунки работают лучше, чем текстовые блоки.

5. Презентация должна быть выдержана в едином строгом стиле.

Классификация презентаций

Презентации можно классифицировать по некоторым признакам, определяющим требования к их оформлению и представлению: по способу представления, по способу управления представлением, по области применения.

По способу представления информации презентации делятся на линейные и со сценарием. В линейных презентациях материал расположен по порядку: начало, продолжение, завершение. Такой вид презентации используется в обучающих презентациях, рекламных роликах и др. Презентации со сценарием предполагают показ слайдов, снабженных анимированными объектами, видеоматериалом, звуковым сопровождением, а также спецэффектами.

По способу управления представлением презентации можно разделить на интерактивные и непрерывные. Интерактивные презентации выполняются под управлением пользователя. Непрерывные презентации позволяют предоставлять информацию в автоматическом режиме.

По области применения презентации можно разделить на следующие виды.

Маркетинговые – отражают направления деятельности компании, виды услуг.

Торговые – используются дилерами или торговыми агентами при заключении сделок, продаже товаров и услуг.

Обучающие презентации – используются при обучении персонала, в учебных заведениях при чтении лекций, выполнении лабораторных работ и практических заданий.

Корпоративные презентации – ориентированы на потенциальных инвесторов или освещают финансовую деятельность компании и т. д.

Работа в Microsoft Power Point

Рабочее окно программы Power Point представляет собой стандартное окно Windows и включает строку заголовка, ленту, панель быстрого доступа, рабочее окно, строку состояния.

В рабочем окне размещены три объекта: шаблон слайда, окно структуры документа и Заметки к слайду. Шаблон слайда занимает большую часть экрана, на нем, собственно, и ведется разработка слайда. Окно структуры документа предназначено для быстрого просмотра и перемещения по слайдам и имеет две вкладки: Слайды и Структура. В режиме Слайды информация выводится в виде последовательности мини-слайдов. В режиме Структура на экран выводится только текстовая информация, разделенная на кадры, рисунки не выводятся. В данном окне можно копировать, вставлять, удалять и перемещать слайды. Заметки к слайду могут содержать дополнительную текстовую информацию, поясняющую содержание текущего слайда.

Структура ленты и строки состояния рабочего окна программы Power Point аналогичны структуре ленты и строки состояния этих объектов в текстовом процессоре Word. В левой части строки состояния слева указаны номер текущего слайда и общее число слайдов в презентации, тема оформления слайдов, используемый язык редактирования документа. В правой части строки состояния расположены кнопки режимов просмотра, настройки масштаба и кнопка Вписать слайд в окно. Кнопка Вписать слайд в окно подгоняет размеры слайда к размеру рабочего окна, сохраняя соотношение сторон слайда.

Методика создания презентаций

Можно указать несколько способов создания презентации: начать с чистого листа и все операции выполнять в Power Point; использовать для презентации заранее подготовленный текст документа; использовать шаблоны Power Point; использовать существующую презентацию.

Начать с чистого листа. Этот способ можно использовать, когда мало текстовой информации и предполагается доклад с демонстрацией кадров, давая пояснения. При больших объемах текста этот путь не эффективен.

Использование готового текстового документа. Для этого надо руководствоваться следующим алгоритмом:

1. Выполнить некоторые подготовительные операции над исходным текстом: создать копию текстового документа и использовать ее для дальнейшей работы; выделить в документе фрагменты текста приемлемого объема, которые будут помещены в один слайд, и оформить их стилем заголовка № 2; расставить заголовки к каждому фрагменту текста и оформить их стилем заголовка № 1; определить текст, который будет помещен в качестве заметок к слайдам; сохранить полученный документ на диске и закрыть документ в текстовом процессоре.

2. Загрузить документ в программу Power Point: ввести команду Файл, Открыть, установить в окне диалога Открытие документа тип файлов Все файлы; загрузить подготовленный файл презентации. Программа автоматически формирует слайды, внося заголовки, оформленные стилем Заголовок 1, в заголовок слайда, а текст, оформленный стилем Заголовок 2, в слайд.

3. Открыть в текстовом процессоре Word подготовленный ранее файл презентации: вставить пустые слайды в местах размещения таблиц, формул, рисунков.

Добавление новых слайдов в Power Point осуществляется командой Создать слайд вкладки Главная ленты; скопировать и вставить в слайды необходимые рисунки; скопировать и вставить в область Заметки к слайду текст заметок.

Использование шаблона

Power Point предоставляет пользователю возможность создать презентацию на определенную тему, используя готовые шаблоны. Для этого введите команду Файл, Создать: в открывшемся окне диалога можно выбрать понравившуюся тему. Представленная здесь же презентация Power Point 2010 позволит ознакомиться с возможностями программы, загрузив шаблон и открыв его в режиме показа. Можно найти подходящие шаблоны на сайте Office.com.

Использование существующей презентации. В этом случае важно сохранить стиль оформления, а не содержание. Сохраните слайды, которые будут вам необходимы, и замените в них текст. Остальные слайды удалите. В принципе, достаточно оставить заголовок и один из слайдов, чтобы сохранить стиль оформления документа.

Управление внешним видом рабочей среды

Лента содержит несколько постоянно открытых вкладок: Главная, Вид, Вставка, Дизайн, Переходы, Анимация, Показ, Рецензирование. Каждая из вкладок содержит набор команд, сгруппированных по функциональному назначению. Управление внешним видом рабочего окна программы осуществляется с помощью вкладки Вид и группы команд в правой части строки состояния, о которых упоминали ранее. Вкладка Вид содержит несколько групп команд, позволяющих управлять режимами просмотра, настраивать параметры слайдов, выдач, заметок, внешний вид окна редактора, управлять масштабом изображения, открытыми окнами.

Режимы просмотра презентации и образцы документов

Power Point предоставляет пользователю четыре режима просмотра презентации: Обычный, Сортировщик слайдов, Чтение и Показ слайдов.

Обычный режим просмотра слайдов предназначен для разработки презентации.

Сортировщик слайдов позволяет быстро просматривать презентацию и осуществлять перемещение отдельных слайдов или групп слайдов в требуемое положение. Этот режим соответствует многостраничному режиму просмотра документов в текстовом процессоре Word.

Режим чтения – это полноэкранный режим просмотра документа. В этом режиме отображаются и спецэффекты, примененные к кадрам.

Показ слайдов – это представление презентации целевой аудитории. Управление показом в ручном режиме осуществляется с помощью клавиш управления курсором «вверх» и «вниз», PgUp, PgDn, а также команд контекстного меню. Команда Страница заметок из группы Режимы просмотра презентаций позволяет просмотреть страницу заметок вместе со слайдом.

Создание слайдов

Power Point 2010 имеет мощные средства для разработки слайдов: ввода и редактирования текста, вставки различных объектов. Все необходимые элементы для создания слайда сосредоточены на вкладке Главная. На ней размещены:

– группа Буфер обмена – содержит команды вырезания, копирования, вставки с использованием буфера обмена, форматирования по образцу – достаточно щелкнуть дважды мышью по кнопке Форматирование по образцу, чтобы применить этот формат к другим частям документа;

– группа Слайды позволяет создать новый слайд и предоставляет пользователю шаблоны размещения информации на слайде.

– группы Шрифт, Абзац, Рисование, Редактирование аналогичны соответствующим командам Word 2016.

Вкладка Рецензирование позволяет проверять орфографию, добавлять примечания к тексту заметок, пользоваться справочниками, переводчиком текста с других языков, выбирать язык редактирования справок, примечаний, интерфейса.

Шаблоны слайдов содержат местозаполнители. Пустые местозаполнители служат для ввода текста, местозаполнители с картинками служат для вставки объектов: таблиц, графиков и диаграмм, рисунков SmartArt, рисунков из файлов, картинок, клипов мультимедиа и даже фильмов.

Вставка объектов

Вкладка Вставка Power Point 2016 мало чем отличается от одноименной вкладки ленты Word 2016. Здесь наше внимание может привлечь группа Мультимедиа. Она предоставляет возможность вставки видеоматериалов из файла, видеосайта, организатора клипов. Список типов видеофайлов, которые можно использовать в Power Point, практически не ограничен. Для воспроизведения файла звука или видео требуется универсальный проигрыватель, который воспроизводит мультимедийные файлы и управляет такими устройствами воспроизведения, как приводы компакт- и видеодисков.

Группа Ссылки позволяет вставить гиперссылки на веб-страницу, рисунок, адрес электронной почты или программу.

Объект SmartArt служит для визуального представления информации в виде списков, последовательности переходов, иерархии, отражения связи между объектами. После выбора элемента SmartArt открывается вкладка ленты для работы с данным объектом, содержащая две вкладки Конструктор и Формат, позволяющие вести настройку параметров объектов.

Оформление слайдов и представление презентации

Выполняется оформление слайдов с помощью вкладок Дизайн, Переходы и Анимация.

Вкладка Дизайн предоставляет пользователю возможность выбрать тему из предлагаемого списка. Выбранная тема может быть применена ко всем слайдам или к выделенным слайдам. Обычно выбирается одинаковый тип для всех слайдов. Исключение можно сделать для титульного листа.

Вкладка Переходы позволяет создать эффекты, происходящие при смене кадра. После выбора варианта перехода можно изменить его параметры. Здесь же можно подобрать звук, сопровождающий смену кадра и временные параметры перехода.

Вкладка Анимация – это добавление к тексту или объекту специального видео- или звукового эффекта. Можно применить анимацию к тексту, рисункам, фигурам, таблицам, графическим элементам SmartArt и другим объектам в презентации. К одному и тому же объекту анимационные эффекты можно применять последовательно один за другим.

Подготовка и представление презентации

Настройка параметров представления презентации осуществляется с помощью команд вкладки Показ слайдов. Power Point позволяет использовать несколько режимов показа слайдов:

– непрерывный показ с начала;

– показ с текущего кадра;

– произвольный показ только отдельных кадров;

– широковещательный показ удаленным зрителям, использующим средства веб-браузера.

Создание страниц заметок

Страницы заметок можно создавать непосредственно во время разработки каждого слайда. По окончании создания презентации страницы заметок можно оформить с помощью команды Вид, Образец заметок. При печати слайдов в верхней части страницы выводится миниатюра слайда.

Раздаточные материалы

Раздаточные материалы – это распечатанные слайды. Подготовка раздаточного материала к печати осуществляется командой Вид, Образец выдач. После ввода команды появляется вкладка Образец выдач. На этой вкладке можно установить параметры страницы, ориентацию выдач (книжная или альбомная), ориентацию слайда (также книжная или альбомная), число слайдов на странице, указать необходимость вывода верхнего и нижнего колонтитулов, даты и времени, изменить тему и фон.

Сохранение и использование презентаций

Сохранение презентации осуществляется командами Сохранить, Сохранить как и Сохранить и отправить меню Файл. Первые две команды сохраняют файлы на своем компьютере. Команда сохранить и отправить позволяет выполнить при сохранении некоторые

преобразования и отправить файл по назначению. В частности, при сохранении презентацию можно преобразовать в видеофайл – команда Создать видео.

При отправке презентации по электронной почте рекомендуется использовать форматы .pdf и .xps. Достоинством этих форматов является то, что документы этих форматов одинаково выглядят на большинстве компьютеров, сохраняются шрифты, исходное форматирование и изображения, содержимое нельзя легко изменить.

Презентации в онлайн-сервисе Canva

В современном мире визуальная коммуникация играет ключевую роль в передаче информации. Презентации стали неотъемлемой частью образовательного процесса, бизнес-встреч, конференций и многих других сфер деятельности. Одним из популярных инструментов для создания презентаций является онлайн-сервис Canva. Canva предоставляет пользователям мощные и удобные инструменты для создания профессиональных презентаций без необходимости глубоких знаний в области дизайна.

Мы рассмотрим основные возможности и преимущества использования Canva для создания презентаций, а также подробно остановимся на процессе создания презентации в этом сервисе (рис. 2.2).

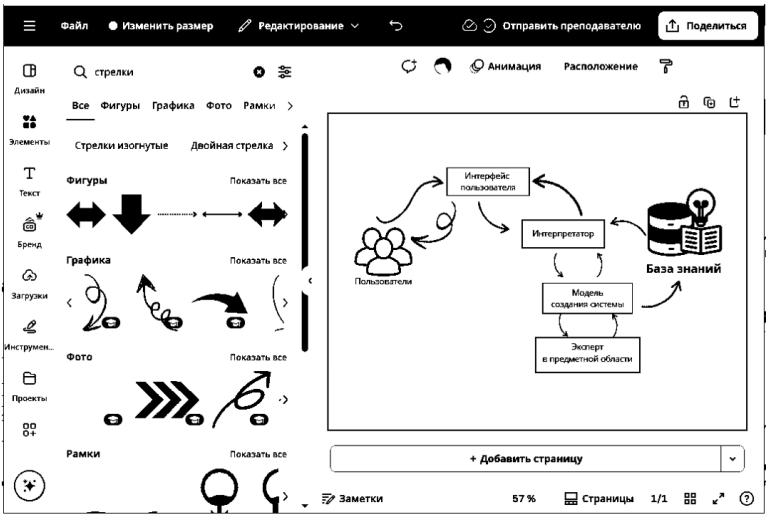


Рис. 2.2. Интерфейс Canva

Основные возможности Canva

Canva – это онлайн-сервис для создания графического дизайна, который предлагает широкий спектр инструментов и шаблонов для различных типов проектов, включая презентации. Основные возможности Canva включают:

1. **Шаблоны:** Canva предоставляет тысячи готовых шаблонов для презентаций, которые можно легко адаптировать под свои нужды. Шаблоны охватывают различные темы и стили, что позволяет быстро создать профессиональную презентацию.

2. **Редактор перетаскивания:** интуитивно понятный интерфейс Canva позволяет легко добавлять и настраивать элементы презентации с помощью функции перетаскивания. Это делает процесс создания презентации простым и удобным.

3. **Библиотека элементов:** Canva предлагает обширную библиотеку элементов дизайна, таких как иконки, изображения, иллюстрации, формы и линии. Пользователи могут легко добавлять эти элементы в свои презентации для улучшения визуального восприятия.

4. **Текстовые инструменты:** Canva предоставляет мощные инструменты для работы с текстом, включая различные шрифты, стили и эффекты. Пользователи могут легко настраивать текстовые элементы для создания привлекательных и информативных слайдов.

5. **Мультимедийные элементы:** Canva позволяет добавлять в презентации мультимедийные элементы, такие как видео, аудио и анимации. Это делает презентации более динамичными и увлекательными.

6. **Коллаборация:** Canva поддерживает совместную работу над проектами, что позволяет нескольким пользователям одновременно работать над одной презентацией. Это особенно удобно для командной работы и совместных проектов.

7. **Экспорт и публикация:** Canva предоставляет различные варианты экспорта и публикации презентаций, включая сохранение в форматах PDF, PPTX, а также публикацию в интернете и социальных сетях.

Процесс создания презентации в Canva

Создание презентации в Canva – это простой и удобный процесс, который включает несколько основных этапов. Рассмотрим каждый из этих этапов подробнее.

Регистрация и вход в систему

Первым шагом для создания презентации в Canva является регистрация и вход в систему. Пользователи могут зарегистрироваться на сайте Canva, используя электронную почту или аккаунты в социальных сетях, таких как Google или Facebook. После регистрации пользователи получают доступ к основным функциям и инструментам Canva.

Создание нового проекта

После входа в систему пользователи могут создать новый проект, выбрав опцию «Создать дизайн» на главной странице Canva. В открывшемся меню необходимо выбрать тип проекта «Презентация». Canva предложит различные шаблоны презентаций, из которых можно выбрать подходящий или начать с пустого слайда.

Выбор шаблона

Canva предлагает тысячи готовых шаблонов для презентаций, охватывающих различные темы и стили. Пользователи могут просматривать шаблоны и выбирать подходящий для своей презентации. Шаблоны можно фильтровать по темам, цветам и стилям, что упрощает процесс поиска. После выбора шаблона пользователи могут приступить к его настройке и адаптации под свои нужды.

Настройка слайдов

После выбора шаблона пользователи могут приступить к настройке слайдов. Canva предоставляет различные инструменты для настройки слайдов, включая изменение фона, добавление текста, изображений, иконок и других элементов дизайна. Пользователи могут легко перетаскивать элементы на слайды и настраивать их размер, положение и стиль.

Добавление текста

Текст является важным элементом любой презентации. Canva предоставляет мощные инструменты для работы с текстом, включая различные шрифты, стили и эффекты. Пользователи могут добавлять текстовые блоки на слайды и настраивать их размер, цвет, шрифт и выравнивание. Canva также предлагает готовые текстовые шаблоны, которые можно использовать для быстрого создания привлекательных текстовых элементов.

Добавление изображений и иконок

Изображения и иконки помогают улучшить визуальное восприятие презентации и сделать ее более привлекательной. Canva предлагает обширную библиотеку изображений и иконок, которые

можно легко добавлять на слайды. Пользователи могут искать изображения и иконки по ключевым словам и выбирать подходящие для своей презентации. После добавления изображений и иконок их можно настраивать, изменяя размер, положение и стиль.

Добавление мультимедийных элементов

Мультимедийные элементы, такие как видео, аудио- и анимации, делают презентации более динамичными и увлекательными. Canva позволяет добавлять видео- и аудиофайлы на слайды, а также использовать анимации для создания эффектов перехода между слайдами. Пользователи могут загружать свои собственные мультимедийные файлы или использовать готовые элементы из библиотеки Canva.

Настройка анимаций и переходов

Анимации и переходы помогают сделать презентацию более динамичной и увлекательной. Canva предоставляет различные варианты анимаций и переходов, которые можно применять к элементам слайдов и между слайдами. Пользователи могут выбирать подходящие анимации и переходы и настраивать их параметры, такие как скорость и направление.

Коллаборация и совместная работа

Canva поддерживает совместную работу над проектами, что позволяет нескольким пользователям одновременно работать над одной презентацией. Это особенно удобно для командной работы и совместных проектов. Пользователи могут приглашать других пользователей для совместной работы, предоставляя им доступ к проекту и настраивая права доступа.

Предварительный просмотр и тестирование

Перед завершением работы над презентацией важно провести предварительный просмотр и тестирование. Canva предоставляет возможность предварительного просмотра презентации, что позволяет проверить ее внешний вид и функциональность. Пользователи могут просматривать презентацию в режиме слайд-шоу и вносить необходимые корректировки.

Экспорт и публикация

После завершения работы над презентацией пользователи могут экспортировать ее в различные форматы, такие как PDF и PPTX, или опубликовать в интернете и социальных сетях. Canva предоставляет различные варианты экспорта и публикации, что делает процесс распространения информации удобным и простым.

Пример создания презентации в Canva

Рассмотрим пример создания презентации в Canva на тему «Введение в информационные технологии». В этой презентации будут представлены основные концепции и области применения информационных технологий.

Шаг 1. Регистрация и вход в систему

Перейдите на сайт Canva и зарегистрируйтесь, используя электронную почту или аккаунт в социальной сети.

Войдите в систему, используя свои учетные данные.

Шаг 2. Создание нового проекта

На главной странице Canva нажмите кнопку «Создать дизайн».

В открывшемся меню выберите тип проекта «Презентация».

Шаг 3. Выбор шаблона

Просмотрите доступные шаблоны презентаций и выберите подходящий для темы «Введение в информационные технологии».

Нажмите на выбранный шаблон, чтобы открыть его в редакторе Canva.

Шаг 4. Настройка слайдов

Настройте фон слайдов, выбрав подходящий цвет или изображение из библиотеки Canva.

Добавьте текстовые блоки для заголовков и подзаголовков слайдов.

Настройте шрифты, цвета и стили текста для создания привлекательных текстовых элементов.

Шаг 5. Добавление текста

Добавьте текстовые блоки на слайды и введите информацию об основных концепциях и областях применения информационных технологий.

Настройте размер, цвет и шрифт текста для улучшения визуального восприятия.

Шаг 6. Добавление изображений и иконок

Добавьте изображения и иконки, связанные с информационными технологиями, из библиотеки Canva.

Настройте размер, положение и стиль изображений и иконок для улучшения визуального восприятия слайдов.

Шаг 7. Добавление мультимедийных элементов

Добавьте видео или аудио файлы, связанные с информационными технологиями, на слайды.

Используйте анимации для создания эффектов перехода между слайдами.

Шаг 8. Настройка анимаций и переходов

Выберите подходящие анимации и переходы для элементов слайдов и между слайдами.

Настройте параметры анимаций и переходов, такие как скорость и направление.

Шаг 9. Коллаборация и совместная работа

Пригласите других пользователей для совместной работы над презентацией.

Настройте права доступа для совместной работы.

Шаг 10. Предварительный просмотр и тестирование

Просмотрите презентацию в режиме слайд-шоу.

Внесите необходимые корректировки для улучшения внешнего вида и функциональности.

Шаг 11. Экспорт и публикация

Экспортируйте презентацию в формате PDF или PPTX.

Опубликуйте презентацию в интернете или социальных сетях для распространения информации.

Canva – это мощный и удобный онлайн-сервис для создания презентаций, который предлагает широкий спектр инструментов и шаблонов для различных типов проектов. Основные возможности Canva, такие как шаблоны, редактор перетаскивания, библиотека элементов, текстовые инструменты, мультимедийные элементы, коллаборация и экспорт, делают процесс создания презентаций простым и удобным даже для новичков.

Использование Canva для создания презентаций имеет ряд преимуществ, включая простоту использования, широкий выбор шаблонов, обширную библиотеку элементов, мультимедийные возможности, поддержку совместной работы и удобные варианты экспорта и публикации. Эти преимущества делают Canva популярным инструментом среди пользователей различных уровней подготовки.

2.2. Сервисные инструментальные средства

Сервисная программная система – это программный продукт, изменяющий и дополняющий пользовательский и программный интерфейсы операционной системы. Сервисные системы используются для обеспечения эффективного взаимодействия пользователя,

компьютера и сетевых систем. Их характеризует то, что, с одной стороны, они не входят в состав операционных систем, а с другой – предназначены для реализации служебных функций по управлению компьютером. Иными словами, сервисные программы – это вспомогательные инструменты, расширяющие и дополняющие функциональность операционных систем. Среди множества сервисных систем особое место занимают файловые менеджеры, например, программа Проводник Windows (англ. Windows Explorer), встроенная в операционную систему Windows.

2.2.1. Файловый менеджер

Файловый менеджер (англ. file) – компьютерная программа, предоставляющая интерфейс пользователя для работы с файловой системой и файлами. Файловый менеджер позволяет выполнять наиболее частые операции над файлами – создание, открытие/проигрывание/просмотр, редактирование, перемещение, переименование, копирование, удаление, изменение атрибутов и свойств, поиск файлов и назначение прав. Помимо основных функций, многие файловые менеджеры включают ряд дополнительных возможностей, например, таких, как работа с сетью (через FTP, NFS и т. п.), резервное копирование, управление принтерами и др. Применяются различные типы файловых менеджеров.

2.2.2. Электронные словари и переводчики

Электронный словарь и электронный переводчик – это устройства, предназначенные для перевода отдельных слов (основная масса переводит только по одному слову) или фраз с иностранного языка на русский или в обратную сторону. Некоторые модели имеют возможность озвучить переведенное слово или фразу. Принцип работы и словаря, и переводчика практически один и тот же. Это устройства, которые позволяют получить для любого иностранного слова множество вариантов перевода, и ориентированы на различные предметные области изучаемого иностранного языка. Каждый переводчик или словарь имеет свой словарный запас, чем он больше, тем больше у вас шансов найти нужное слово без труда.

Электронный словарь – узконаправленное устройство с определенным спектром выполняемых функций и предлагаемых опций с вариантами переводов иностранных слов или фраз. Такой

словарь зачастую только позволяет быстро найти нужное слово с учетом морфологии и возможностью поиска словосочетаний (примеров употребления). Наиболее популярные виды словарей: Free Online Dictionary of Computing, FreeDict, Jar-gon file, WordNet. Часто используемые программы: AtomicDic, GoldenDict, StarDict, программы, сайты и др.: ABBYY Lingvo, DICT – сетевой протокол, Мультитран, Polyglossum, МультиЛекс (русско-английские, немецкие, французские, испанские, итальянские, португальские и многоязычные словари, включающие толковые словари и тематические словарные базы для перевода специализированной лексики).

Электронный переводчик в отличие от словаря, как правило, имеет разговорник, разбитый по темам и включающий в себя наиболее часто употребляемые фразы и выражения, которые можно также прослушать при наличии функции произношения. Во многих переводчиках есть дополнительные технические возможности, помогающие в изучении языка: транскрипция, обучающие программы в виде электронных учебников по грамматике, функция орфографического корректора, обучающие игры, экзамены TOEFL и др. Также имеются дополнительные функции, не относящиеся к функциям перевода слов: мировое время, перевод мер и весов, соответствие размеров одежды и обуви, будильник, калькулятор, метрические преобразования, записная книжка, которые будут очень кстати, если отправляться в путешествие или рабочую командировку по разным странам.

Программа ABBYY Lingvo

Российская компания ABBYY – ведущий мировой разработчик программного обеспечения и поставщик услуг в области распознавания документов, лингвистики и перевода. Запустите словарь ABBYY Lingvo, выберите нужное направление перевода (англо-русское или русско-английское) и введите незнакомое слово в строку ввода. Нажмите кнопку «Перевод» или клавишу Enter. В карточке словаря вы можете посмотреть переводы слова, примеры употребления, ознакомиться с транскрипцией и послушать произношение слова.

Система перевода PROMT Professional

Программа PROMT Professional – профессиональная система перевода текстов с английского, французского, немецкого, испанского, итальянского языков на русский язык и обратно, с расширенным

набором настроек для точного перевода документов по различным тематикам (деловых контрактов, технических описаний, финансовых и других документов).

Основные возможности переводчика PROMT Professional:

1. Пользовательские словари и онлайновые словарные базы дополняют основной словарный запас программы.

2. Перевод файлов .pdf,.doc, .xls, .ppt, .msg,.html, .xml, Open Office.org Writer.

3. Тексты pdf -файлов можно переводить непосредственно в редакторе PROMT с сохранением исходного форматирования, а также в интерфейсе программ Adobe Acrobat и Adobe Professional.

4. Использование базы переведенных текстов позволяет экономить время при работе с типовыми документами и поддерживать единую корпоративную терминологию.

5. Перевод без ошибок и опечаток.

6. Возможность переводить несколько файлов одновременно.

7. Удобный, простой и понятный интерфейс. Удобные, полные и понятные справочные материалы

8. Личный кабинет для доступа к пользовательским ресурсам и сервисам системы PROMT.

2.2.3. Программы архивации данных

Для уменьшения места, необходимого для содержания копий, и удобства эксплуатации создано достаточно много программ, позволяющих работать с архивными файлами. Архивный файл может содержать один и более файлов в сжатом виде. Можно как извлекать из него исходные файлы в их первоначальном виде, так и добавлять в этот файл новые. Среди популярных программ архивации, работающих в среде Windows, можно назвать такие, как WinZip, WinRar, WinArj, NetZip и др.

Архиватор WinZip

После установки этой программы создается не только меню WinZip, но и ярлыки на рабочем столе и в кнопке Пуск, в контекстное меню для папок и файлов добавляются пункты, связанные с архивацией. Программа может быть запущена последовательностью Пуск, Программы, WinZip.

Создание и изменение архива. Работа с архивом начинается с создания файла архива при помощи последовательности команд File, New Archive (как в любых программах Windows, любая

команда или операция может быть осуществлена разными способами, в частности, клавишами Ctrl+N или кнопкой New на панели инструментов. Файл архива будет создан после задания его имени. При установке флажка Add dialog будет открыто дополнительное диалоговое окно Add, в котором можно указать имена и параметры файлов, добавляемых в архив. В противном случае (если флажок Add dialog не установлен) будет создан пустой архивный файл.

При добавлении файлов в уже существующий архив необходимо открыть выбранный файл архива. Для этого используется команда открытия File, Open Archive. Для включения файла в архив используется команда добавления Actions, Add. В окне Add кроме указания файла (файлов) для добавления в архив задаются параметры включения в архив: например, активизация параметра Update (and add) files обновляются файлы, которые уже есть в архиве и добавляются новые. При добавлении файлов можно указать пароль для недопущения несанкционированного доступа к файлам архива. Команда добавления запускается нажатием кнопки Add.

Извлечение файлов из архива. При извлечении файла из архива извлекаемый файл из архива не удаляется. Для извлечения файлов из архива необходимо открыть файл архива и запустить команду извлечения. Команда извлечения файла из архива может быть вызвана последовательностью команд Actions, Extract. В появившемся окне необходимо в поле Extract to ввести имя папки, в которую будут помещаться извлекаемые из архива файлы, а в поле Files – имена извлекаемых файлов.

2.2.4. Программы распознавания текстов

Наиболее распространенное программное средство оптического распознавания текста ABBYY FineReader, которое поддерживает такие функции, как автоматическая сетевая установка, работа с сетевыми сканерами, удобные инструменты администрирования, распределенная обработка документов, поддержка многопроцессорных систем, распознавание изображений, полученных с помощью цифровой камеры. Если под рукой нет сканера, можно сделать снимок документа с помощью фотоаппарата. Программа адаптируется к этим изображениям, учитывая такие их особенности, как неравномерная освещенность, нечеткая фокусировка и искажение строк.

2.3. Системы управления базами данных. Структура данных, модели данных. Создание базы данных и таблиц

Системы управления базами данных (СУБД) являются ключевым компонентом современных информационных систем. Они обеспечивают эффективное хранение, управление и обработку данных, что делает их неотъемлемой частью различных приложений, от веб-сайтов до корпоративных систем. СУБД можно классифицировать по различным критериям, таким как модель данных, архитектура, способ доступа к данным и другие характеристики. В этом разделе мы рассмотрим основные классификации систем управления базами данных.

2.3.1. Системы управления базами данных. Классификация

Одним из основных критериев классификации СУБД является модель данных, которая определяет способ организации и манипулирования данными. Рассмотрим основные модели данных и соответствующие им СУБД.

Реляционные СУБД

Реляционные СУБД (РСУБД) основаны на реляционной модели данных, предложенной Эдгаром Коддом в 1970 году. Данные организованы в виде таблиц (отношений), состоящих из строк (записей) и столбцов (полей). Каждая таблица представляет собой отдельную сущность, связи между ними устанавливаются с помощью ключей.

Основные особенности реляционных СУБД

Табличная структура. Данные хранятся в таблицах, что обеспечивает простоту и понятность организации данных.

SQL. Для работы с данными используется язык структурированных запросов (SQL), который предоставляет мощные инструменты для выполнения различных операций.

Нормализация. Реляционные СУБД поддерживают процесс нормализации, который позволяет минимизировать избыточность данных и избежать аномалий при их изменении.

Примеры реляционных СУБД

*MySQL***.** Популярная открытая СУБД, широко используемая в веб-разработке.

PostgreSQL. Мощная открытая СУБД с поддержкой расширенных функций и возможностей.

Oracle Database. Коммерческая СУБД, известная своей производительностью и надежностью.

Microsoft SQL Server. Коммерческая СУБД от Microsoft, широко используемая в корпоративных системах.

Иерархические СУБД

Иерархические СУБД основаны на иерархической модели данных, в которой данные организованы в виде дерева. Каждая запись имеет одного родителя и может иметь несколько дочерних записей. Эта модель хорошо подходит для представления данных с четкой иерархической структурой (рис. 2.3).

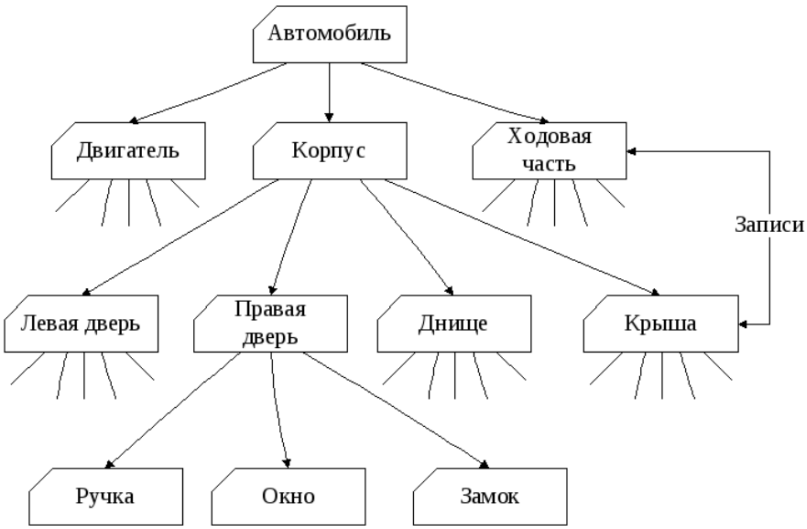


Рис. 2.3. Пример иерархической СУБД

Основные особенности иерархических СУБД

Древовидная структура. Данные организованы в виде дерева, что обеспечивает простоту навигации и поиска.

*Эффективность***.** Иерархические СУБД обеспечивают высокую производительность при работе с данными, имеющими четкую иерархическую структуру.

Примеры иерархических СУБД

IBM Information Management System (IMS). Одна из первых иерархических СУБД, разработанная IBM.

Windows Registry. Иерархическая база данных, используемая в операционных системах Windows для хранения конфигурационных данных.

Сетевые СУБД

Сетевые СУБД основаны на сетевой модели данных, которая является расширением иерархической модели. В сетевой модели данные организованы в виде графа, где каждая запись может иметь несколько родителей и несколько дочерних записей. Это позволяет представлять более сложные связи между данными (рис. 2.4).

Основные особенности сетевых СУБД

Графовая структура. Данные организованы в виде графа, что обеспечивает гибкость при представлении сложных связей.

Эффективность. Сетевые СУБД обеспечивают высокую производительность при работе с данными, имеющими сложные взаимосвязи.

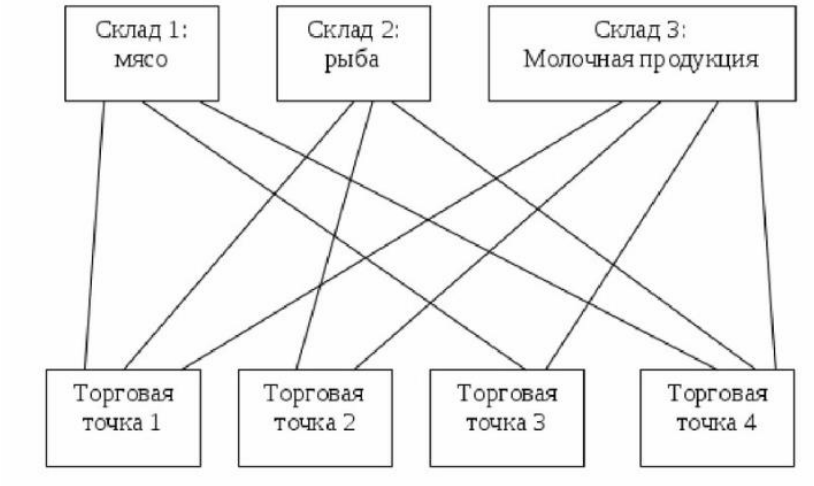


Рис. 2.4. Пример сетевой СУБД

Примеры сетевых СУБД:

Integrated Data Store (IDS). Одна из первых сетевых СУБД, разработанная компанией Honeywell.

CODASYL DBTG: Стандарт сетевой модели данных, разработанный конференцией по языкам систем данных (CODASYL).

Объектно-ориентированные СУБД

Объектно-ориентированные СУБД (ООСУБД) основаны на объектно-ориентированной модели данных, в которой данные организованы в виде объектов. Каждый объект имеет свойства (атрибуты) и методы (функции), что позволяет инкапсулировать данные и поведение в одном месте.

Основные особенности объектно-ориентированных СУБД:

Объектная структура. Данные организованы в виде объектов, что обеспечивает гибкость и модульность.

Инкапсуляция. Объектно-ориентированные СУБД поддерживают инкапсуляцию, что позволяет скрывать детали реализации и предоставлять интерфейсы для работы с данными.

Наследование. Объектно-ориентированные СУБД поддерживают наследование, что позволяет создавать иерархии объектов и повторно использовать код.

Примеры объектно-ориентированных СУБД:

*ObjectDB – о*бъектно-ориентированная СУБД для Java, поддерживающая стандарт JPA (Java Persistence API).

db4o – объектно-ориентированная СУБД для Java и .NET, известная своей простотой и производительностью.

NoSQL СУБД

NoSQL СУБД (Not Only SQL) представляют собой класс систем управления базами данных, которые не используют реляционную модель данных. Они предназначены для работы с большими объемами неструктурированных и полуструктурированных данных и обеспечивают высокую производительность и масштабируемость.

Основные особенности NoSQL СУБД

Гибкость. NoSQL СУБД обеспечивают гибкость при работе с неструктурированными и полуструктурированными данными.

Масштабируемость. NoSQL СУБД обеспечивают высокую масштабируемость, что позволяет обрабатывать большие объемы данных.

Производительность. NoSQL СУБД обеспечивают высокую производительность при работе с большими объемами данных.

Примеры NoSQL СУБД:

MongoDB. Документоориентированная СУБД, которая хранит данные в виде документов JSON.

Cassandra. Колонковая СУБД, разработанная для обработки больших объемов данных с высокой производительностью.

Redis. Ключ-значение СУБД, известная своей высокой производительностью и поддержкой различных структур данных.

Neo4j. Графовая СУБД, предназначенная для работы с графовыми данными и сложными взаимосвязями.

2.3.2. Создание базы данных и таблиц

Microsoft Access – это система управления базами данных (СУБД), разработанная компанией Microsoft. Она входит в состав пакета Microsoft Office, предназначена для создания и управления реляционными базами данных. Access предоставляет пользователям мощные инструменты для хранения, обработки и анализа данных, создания пользовательских интерфейсов и отчетов. Мы рассмотрим возможности и применение Microsoft Access.

Пример создания базы данных в Microsoft Access

Рассмотрим пример создания простой базы данных в Microsoft Access для управления информацией о студентах и курсах. В этой базе данных будут две таблицы: «Студенты» и «Курсы», а также формы и отчеты для работы с данными.

Создание таблиц

1. Откройте Microsoft Access и создайте новую базу данных.
2. Перейдите на вкладку «Создание» и выберите «Конструктор таблиц».
3. Создайте таблицу «Студенты» со следующими полями:
    a. ID: Числовой тип, первичный ключ.
    b. Имя: Текстовый тип.
    c. Фамилия: Текстовый тип.
    d. Дата рождения: Дата/время.
    e. Адрес: Текстовый тип.
    f. Телефон: Текстовый тип.
4. Сохраните таблицу под именем «Студенты».
5. Создайте таблицу «Курсы» со следующими полями:
    a. ID: Числовой тип, первичный ключ.
    b. Название: Текстовый тип.
    c. Описание: Текстовый тип.
    d. Дата начала: Дата/время.
    e. Дата окончания: Дата/время.
6. Сохраните таблицу под именем «Курсы».

Создание формы

1. Перейдите на вкладку «Создание» и выберите «Мастер форм».

2. Выберите таблицу «Студенты» и добавьте все поля в форму.

3. Выберите внешний вид формы, например, «В один столбец».

4. Задайте имя формы, например, «Форма студентов», и нажмите «Готово».

5. Повторите шаги 1–4 для создания формы для таблицы «Курсы».

Создание запроса

1. Перейдите на вкладку «Создание» и выберите «Конструктор запросов».

2. Добавьте таблицу «Студенты» в запрос.

3. Добавьте поля «Имя», «Фамилия» и «Дата рождения» в запрос.

4. Сохраните запрос под именем «Запрос студентов».

5. Повторите шаги 1–4 для создания запроса для таблицы «Курсы».

Создание отчета

1. Перейдите на вкладку «Создание» и выберите «Мастер отчетов».

2. Выберите таблицу «Студенты» и добавьте все поля в отчет.

3. Выберите способ группировки данных, например, «Без группировки».

4. Выберите способ сортировки данных, например, «По фамилии».

5. Задайте имя отчета, например, «Отчет студентов» и нажмите «Готово».

6. Повторите шаги 1–5 для создания отчета для таблицы «Курсы».

Microsoft Access является мощной и удобной системой управления базами данных, которая предоставляет широкие возможности для создания и управления реляционными базами данных. Она обладает интуитивно понятным графическим интерфейсом, поддерживает реляционную модель данных и язык SQL, а также предоставляет множество встроенных шаблонов и инструментов для автоматизации задач.

2.4. Структурированные и неструктурированные массивы данных

В современной информационной среде данные являются ключевым активом. Эффективность их обработки, хранения и анализа напрямую зависит от формы их организации. Все массивы данных принято классифицировать на два фундаментальных типа: структурированные и неструктурированные. Понимание их различий,

преимуществ и недостатков критически важно для проектирования архитектуры информационных систем, выбора инструментов анализа и формулирования задач в рамках исследований по ИТ.

2.4.1. Структурированные данные

Определение: структурированные данные – это информация, представленная в строго определенном формате и модели данных. Они высокоорганизованы и следуют заранее заданной схеме (модели), что делает их простыми для ввода, обработки, запросов и анализа с помощью алгоритмов.

Ключевые характеристики:

1. *Четкая схема (Schema)*. Структура данных определяется заранее. Например, таблица в реляционной базе данных имеет строго заданные столбцы (атрибуты) с определенными типами данных (integer, varchar, date и т. д.).

2. *Табличная форма*. Наиболее распространенная форма представления – строки и столбцы. Каждая строка представляет собой запись (объект), а каждый столбец – определенный признак (атрибут) этого объекта.

3. *Легкость обработки*. Благодаря строгой организации такие данные легко поддаются машинной обработке. Для работы с ними существуют мощные и стандартизированные инструменты, прежде всего язык SQL (Structured Query Language).

4. *Высокая машиночитаемость*. Данные могут быть однозначно интерпретированы программным обеспечением без необходимости сложного предварительного анализа.

Основные источники и примеры:

1. Реляционные базы данных (MySQL, PostgreSQL, Oracle).
2. Данные корпоративных информационных систем (ERP, CRM).
3. Электронные таблицы (Excel, Google Sheets).
4. Логи транзакций.
5. Данные с датчиков, передаваемые в строгом формате.

Преимущества:

1. *Простота анализа*. Идеальны для классического бизнес-анализа, построения отчетов и дашбордов.

2. *Целостность данных*. Механизмы транзакций (ACID) в реляционных СУБД гарантируют точность и consistency данных.

3. *Эффективность*. Выполнение сложных запросов с соединениями (JOIN) и агрегацией оптимизировано.

Недостатки:

1. *Негибкость*. Изменение схемы данных (добавление нового столбца) на работающей системе часто является сложной и дорогостоящей операцией.

2. *Ограниченность*. Непригодны для хранения информации, которая по своей природе не укладывается в табличную структуру.

2.4.2. Неструктурированные данные

Определение. Неструктурированные данные – это информация, которая не имеет предопределенной модели или не организована в заранее заданном порядке. Это до 80 %–90 % всех данных в цифровом мире.

Ключевые характеристики:

1. *Отсутствие четкой схемы*. Данные не следуют единому шаблону. Их структура, если она существует, является внутренней и неочевидной для машины.

2. *Разнообразие форматов*. Представлены в огромном множестве форм и видов.

3. *Сложность автоматической обработки*. Для извлечения ценной информации из таких данных требуются сложные методы обработки естественного языка (NLP), компьютерного зрения, машинного обучения и текстологического анализа.

Основные источники и примеры:

1. *Текстовые данные*. Документы (Word, PDF), электронные письма, сообщения в мессенджерах и соцсетях, новостные ленты, книги.

2. *Мультимедийный контент*. Изображения, фотографии, видео- и аудиозаписи.

3. *Веб-контент*. HTML-страницы, данные с сенсоров и IoT-устройств, часто передаваемые в форматах JSON или XML (которые считаются полуструктурированными).

4. *Прочее*. Файлы логов веб-серверов, презентации.

2.4.3. Полуструктурированные данные

Это подкатегория, занимающая промежуточное положение. Данные не имеют строгой схемы как в реляционных таблицах, но содержат внутренние метки или теги, которые позволяют отделять элементы и иерархии. Классические примеры – JSON (JavaScript Object Notation) и XML (eXtensible Markup Language). Они гибки, как неструктурированные данные, но при этом поддаются парсингу и анализу благодаря своей внутренней маркировке.

Преимущества:

1. *Богатство информации*. Содержат огромный пласт информации, часто именно в них скрыты глубинные инсайты и паттерны.

2. *Гибкость и масштабируемость*. Позволяют хранить данные практически любого типа без необходимости заранее определять их структуру.

3. *Естественность*. Большая часть информации, генерируемой человеком и машинами, по своей природе неструктурирована.

Недостатки:

1. *Сложность анализа*. Требуют применения сложных и ресурсоемких технологий (Big Data frameworks, AI).

2. *Проблемы с хранением и управлением*. Традиционные реляционные СУБД неэффективны для работы с такими данными. Для их хранения используются NoSQL СУБД (документные, ключ-значение), Data Lakes.

3. *Вопросы качества*. Данные могут быть шумными, противоречивыми и требовать сложной предобработки (data cleaning) (табл. 2.1).

Таблица 2.1

Сравнительная таблица

Критерий	Структурированные данные	Неструктурированные данные
Формат	Строгий, предопределенный	Произвольный, разнообразный
Схема	Жесткая, схема перед данными	Отсутствует или гибкая (схема после данных)
Источники	Базы данных, транзакционные системы	Соцсети, документы, email, медиафайлы
Объем	Обычно 10-20% от всех данных	80 %–90 % от всех данных
Масштабируемость	Вертикальное (увеличение мощности сервера)	Горизонтальное (добавление новых серверов)
Основные инструменты	SQL, реляционные СУБД	NoSQL, Data Lakes, Hadoop, Spark, AI
Цель анализа	Отчетность, бизнес-метрики, SQL-агрегация	Поиск инсайтов, паттернов, прогнозное моделирование

3. СЕТЕВЫЕ ТЕХНОЛОГИИ И ИНТЕРНЕТ

3.1. Основы веб-технологий

Веб-технологии основаны на глобальной компьютерной сети. Как и любые объекты исследований, компьютерные сети можно классифицировать по различным признакам.

По территориальной распространенности – это глобальные и локальные компьютерные сети.

Глобальные сети (WAN, Wide Area Networks) позволяют организовать взаимодействие между компьютерами на больших расстояниях. Эти сети работают на относительно низких скоростях и могут вносить значительные задержки в передачу информации. Протяженность глобальных сетей может составлять тысячи километров, и они интегрированы с сетями масштаба страны.

Локальные сети (LAN, Local Area Networks) обеспечивают наивысшую скорость обмена информацией между компьютерами, и типичная локальная сеть занимает пространство в одно или несколько зданий. Протяженность локальных компьютерных сетей составляет всего лишь несколько километров.

Сравнительно недавно появились городские сети или сети мегаполисов (MAN, Metropolitan Area Networks). Такие сети предназначены для обслуживания территории крупного города – мегаполиса.

В своем классическом построении глобальные и локальные компьютерные сети отличаются по следующим признакам.

Протяженность и качество линий связи. Локальные компьютерные сети по определению отличаются от глобальных сетей небольшими расстояниями между узлами сети. Это делает возможным использование в локальных сетях более качественных линий связи.

Сложность методов передачи данных. В условиях низкой надежности физических каналов и различных внутренних стандартах на передачу данных в разных странах в глобальных сетях требуются более сложные, чем в локальных сетях, методы передачи данных и соответствующее оборудование.

Разнообразие услуг. Высокие скорости обмена данными дали возможность реализовать в локальных сетях широкий набор услуг, таких как услуги файловой службы, услуги печати, услуги баз

данных, и др., в то время как глобальные сети в основном были предназначены для почтовых и файловых услуг с ограниченными возможностями.

Масштабируемость. Локальные сети обладают плохой масштабируемостью из-за жесткости базовых топологий, определяющих способ подключения станций и длину линии. При этом характеристики сети резко ухудшаются при достижении определенного предела по количеству узлов или протяженности линий связи. Глобальным сетям присуща хорошая масштабируемость, так как они изначально разрабатывались в расчете на работу с произвольными топологиями и сколь угодно большим количеством абонентов.

Ведомственная принадлежность. По принадлежности различают ведомственные и государственные сети. Ведомственные принадлежат одной организации (компании) и располагаются на ее территории.

Государственные сети – сети, используемые в государственных структурах.

По скорости передачи данных. По скорости передачи информации компьютерные сети делятся на низко-, средне- и высокоскоростные.

Низкоскоростные сети. Скорость передачи данных до 10 Мбит/с. Примеры: старые телефонные линии, некоторые беспроводные сети (например, ранние версии Wi-Fi стандарта 802.11).

Среднескоростные сети. Скорость передачи данных от 10 Мбит/с до 100 Мбит/с. Примеры: Fast Ethernet (100 Мбит/с), некоторые современные беспроводные сети (например, Wi-Fi стандарта 802.11n).

Высокоскоростные сети. Скорость передачи данных от 100 Мбит/с и выше. Примеры: Gigabit Ethernet (1 Гбит/с), 10 Gigabit Ethernet (10 Гбит/с), современные оптоволоконные сети, Wi-Fi стандарта 802.11ac и 802.11ax (Wi-Fi 6).

По типу среды передачи сети разделяются на:

– проводные – коаксиальные, на витой паре, оптоволоконные;

– беспроводные – с передачей информации по радиоканалам, в инфракрасном диапазоне.

По организации взаимодействия компьютеров

С точки зрения организации взаимодействия компьютеров, сети делят на одноранговые и иерархические сети (с выделенным сервером).

3.2. Семиуровневая модель структуры протоколов связи

Существуют два основных метода коммутации (соединения абонентов) в сетях – коммутация каналов и коммутация пакетов. Сети с коммутацией каналов исторически появились первыми в виде первых телеграфных и телефонных сетей. Коммутация каналов подразумевает образование составного физического канала из последовательно соединенных отдельных канальных участков для прямой передачи данных между узлами сети. В основе принципа работы большинства компьютерных сетей (в том числе сети Интернет) положен метод пакетной коммутации.

Коммутация пакетов – эта схема была специально разработана для компьютерных сетей, где различные компьютеры сети могут иметь различное быстродействие. Это основной метод коммутации в современных сетях. При коммутации пакетов все передаваемые сообщения разбиваются передающим компьютером на небольшие части (от 46 до 1500 байт), называемые пакетами. Каждый пакет снабжается заголовком, в котором указывается адресная информация, необходимая для доставки пакета к принимающему компьютеру, а также номер пакета, используемый для «сборки» сообщения на принимающем компьютере. Пакеты транспортируются в сети как независимые информационные блоки. Специальные устройства сети коммутаторы принимают пакеты от передающих компьютеров и на основании адресной информации передают их друг другу до конечного принимающего компьютера. За счет буферизации (задержки) пакета во внутренней памяти коммутатора (если требуемый участок сети занят передачей другой информации) выравнивается скорость передачи данных в сети в целом и повышается пропускная способность сети.

С другой стороны, по своей сущности компьютерные сети являются совокупностью компьютеров и сетевого оборудования, соединенных каналами связи. Поскольку компьютеры и сетевое оборудование могут быть разных производителей, то возникает проблема их совместимости. Без принятия всеми производителя общепринятых правил построения оборудования создание компьютерной сети было бы невозможно. Поэтому разработка и создание компьютерных сетей может происходить только в рамках утвержденных стандартов.

В основу стандартизации компьютерных сетей положен принцип декомпозиции, т. е. разделения сложных задач на отдельные более простые подзадачи. Каждая подзадача имеет четко определенные функции и строго установленные связи между подзадачами. При более внимательном рассмотрении работы компьютера в сети можно выделить две основные подзадачи:

– взаимодействие программного обеспечения пользователя с физическим каналом связи (посредством сетевой карты) в пределах одного компьютера;

– взаимодействие компьютера через канал связи с другим компьютером.

Современное программное обеспечение компьютера имеет многоуровневую модульную структуру, т. е. программный код, написанный программистом и видимый на экране монитора (модуль верхнего уровня), проходит несколько уровней обработки, прежде чем превратится в электрический сигнал (модуль нижнего уровня), передаваемый в канал связи.

При взаимодействии компьютеров через канал связи оба компьютера должны выполнять ряд соглашений. Например, они должны согласовать величину и форму электрических сигналов, длину сообщений, методы контроля достоверности и т. д. Соглашения должны быть такими, чтобы они были поняты каждым модулем на соответствующем уровне каждого компьютера.

Суть работы многоуровневого протокола можно пояснить как «письмо в конверте». Каждый уровень протокола надписывает на «конверте» свою информацию. Сетям нужно только понимать «надпись» на «конверте», чтобы передать его в место назначения.

На рис. 3.1 схематически показана модель взаимодействия двух компьютеров в сети. Для упрощения показаны четыре уровня модулей для каждого компьютера. Процедура взаимодействия каждого уровня этих компьютеров может быть описана в виде набора правил взаимодействия каждой пары модулей соответствующих уровней.

Формализованные правила, определяющие последовательность и формат сообщений, которыми обмениваются модули, лежащие на одном уровне, но в различных компьютерах, называются протоколами.

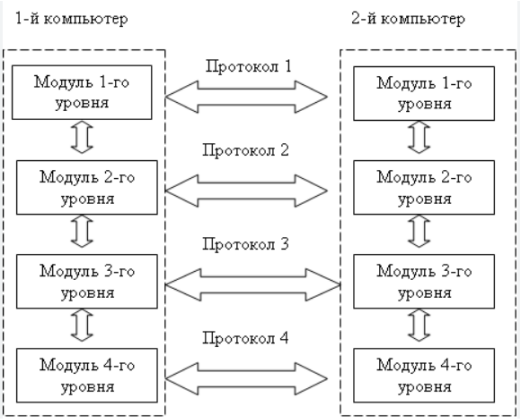


Рис. 3.1. Взаимодействие двух компьютеров в сети

Модули, реализующие протоколы соседнего уровня и находящиеся в одном компьютере, также взаимодействуют друг с другом в соответствии с четко определенными правилами и с помощью стандартизованных форматов сообщений. Эти правила называются интерфейсом и определяют набор сервисов, предоставляемых данным уровнем соседнему уровню.

Другими словами, в сетевых технологиях традиционно принято, что протоколы определяют правила взаимодействия модулей одного уровня, но в разных компьютерах, а интерфейсы – соседних уровней в одном компьютере. Модули, таким образом, должны обрабатывать: во-первых, свой собственный протокол, а, во-вторых, интерфейсы с соседними уровнями.

Иерархически организованный набор протоколов для взаимодействия компьютеров в сети называется стеком коммуникационных протоколов.

Коммуникационные протоколы могут быть реализованы как программно, так и аппаратно. Протоколы нижних уровней, как правило, реализуются комбинацией программно-аппаратных средств, а протоколы верхних уровней – чисто программными средствами.

Отметим, что протоколы каждого уровня обладают независимостью друг от друга, т. е. протокол любого уровня может быть изменен, не оказывая при этом никакого влияния на протокол другого уровня. Главное, чтобы интерфейсы между уровнями обеспечивали необходимые связи между ними.

Принцип взаимодействия компьютеров в сети можно объяснить на примере сотрудничества двух фирм. Два генеральных менеджера каждой из фирм осуществляют сделки между собой на основании заключенных договоров и соглашений. Указанные взаимодействия являются «протоколом уровня генеральных менеджеров». На каждой из фирм у менеджеров есть секретари, причем каждый менеджер имеет свой метод и стиль работы с секретарем. Один, например, предпочитает устные указания, а второй дает только письменные распоряжения. Таким образом, каждая фирма имеет свой собственный интерфейс «главный менеджер – секретарь», что не мешает, однако нормально работать генеральным менеджерам между собой. Секретари, в свою очередь, договорились обмениваться информацией с помощью факсов, реализуя протокол «секретарь – секретарь». В случае, если секретари перейдут на электронную почту, то генеральные менеджеры этого даже и не заметят главное, чтобы секретари выполняли их распоряжения, т. е. должен безукоризненно работать интерфейс «менеджер – секретарь». С другой стороны, менеджеры могут заключить совершенно новый договор, т. е. изменить «протокол уровня генеральных менеджеров». Передача не старого, а нового договора на уровне секретарей пройдет для этих секретарей абсолютно не замеченной.

В рассмотренном примере мы определили два уровня протоколов – уровень генеральных менеджеров и уровень секретарей. Каждый из указанных уровней имеет свой собственный протокол, который может быть изменен независимо от протокола другого уровня. Такую независимость обеспечивает правильное функционирование интерфейсов «менеджер – секретарь». Независимость протоколов каждого уровня друг от друга и взаимодействие самих уровней посредством интерфейсов является важнейшей предпосылкой для создания ряда стандартных протоколов для компьютерных сетей.

В начале 80-х была создана модель взаимодействия двух компьютеров в сети, названная OSI (Open system interconnection – взаимодействие открытых систем). Она была предложена Международной организацией по стандартизации (ISO – International Standards Organization). В модели OSI реализована более подробная семиуровневая архитектура, причем каждому из уровней соответствует определенная функция.

Рассмотрим принцип взаимодействия двух компьютеров в рамках модели OSI.

Взаимодействие компьютеров в сети начинается с того, что приложение (программа пользователя) одного компьютера обращается к прикладному уровню другого компьютера, например, к файловой системе. Приложение первого компьютера формирует с помощью операционной системы сообщение стандартного формата, состоящее из заголовка и поля данных.

Заголовок содержит служебную информацию, которую необходимо передать через сеть прикладному уровню другого компьютера, чтобы сообщить ему, какую работу необходимо выполнить. (Например, о размере файла и где он находится). Кроме этого, в заголовке имеется информация для следующего нижнего уровня, чтобы он «знал», что делать с этим сообщением. В поле данных находится информация, которую необходимо поместить в найденный файл (рис. 3.2).

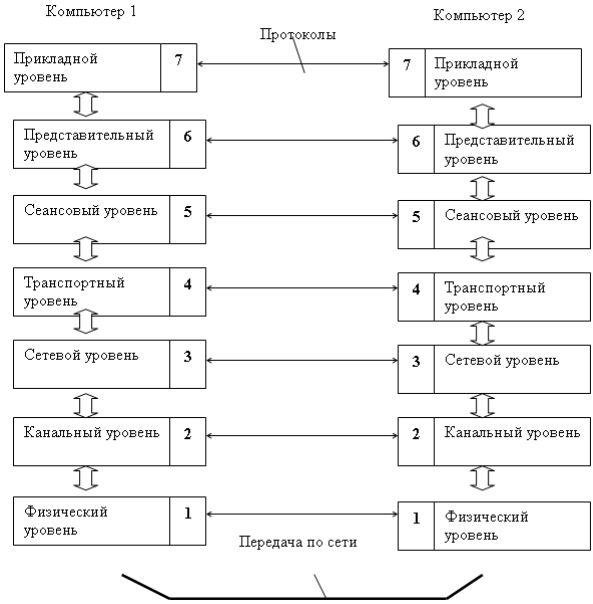


Рис. 3.2. Модель OSI

Сформировав сообщение, прикладной уровень направляет его представительному уровню. Прочитав заголовок, представительный уровень выполняет требуемые действия над сообщением и добавляет к сообщению собственную служебную информацию – заголовок представительного уровня, в котором содержатся указания для протоколов

представительного уровня второго компьютера. Полученное в результате сообщение передается вниз сеансовому уровню, который, в свою очередь, добавляет свой заголовок и т. д. При достижении сообщением нижнего, физического уровня, у него имеется множество заголовков, добавленных на каждом предыдущем уровне (сообщение вложено внутрь, как в матрешку). В таком виде оно и передается по сети.

Второй компьютер принимает его на физическом уровне и последовательно перемещает вверх с уровня на уровень. Каждый из них анализирует и обрабатывает заголовок своего уровня, выполняя соответствующие последнему функции, а затем удаляет этот заголовок и передает сообщение дальше вышележащему уровню. Отметим, что сообщение может оканчиваться также некоторой служебной информацией – «концевиком». (Например, дополнительными контрольными разрядами).

Как видно из рис. 3.3, информация, передающаяся в линию связи, содержит большое количество служебных заголовков, которые по величине могут превосходить даже собственно данные. В результате взаимодействия протоколов всех уровней и их единому стандарту на прикладном уровне второго компьютера получаются данные, переданные первым компьютером.

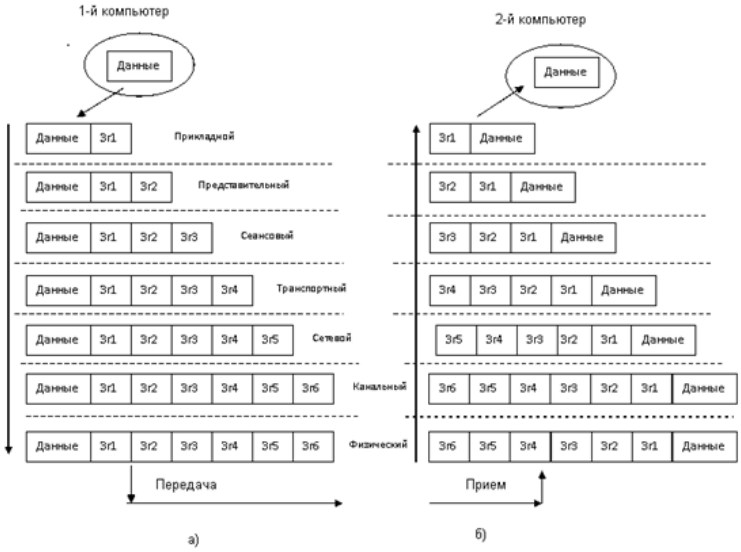


Рис. 3.3. Взаимодействие компьютеров в сети

В стандарте OSI для обозначения единиц данных, с которыми имеют дело протоколы различных уровней, используются специальные названия: кадр (frame), пакет (packet), дейтаграмма (datagram), сегмент (segment).

Рассмотри подробнее уровни модели OSI.

Физический уровень (Physical layer) имеет дело с передачей битов информации по физическим каналам связи. Такими каналами могут быть, например, коаксиальный кабель, витая пара, оптоволоконный кабель. На этом уровне стандартизируются характеристики электрических сигналов, уровни напряжения и тока, тип кодировки информации, скорость передачи сигналов, а также типы разъемов и назначение каждого контакта.

Канальный уровень (Data Link layer) обеспечивает надежную передачу данных через физический канал. Канальный уровень оперирует блоками данных, называемых кадрами. Основной задачей канального уровня является прием кадра из сети и отправка его в сеть. При выполнении этой задачи канальный уровень осуществляет физическую адресацию передаваемых сообщений, контролирует соблюдение правил использования физического канала, выявляет неисправности, управляет потоками информации. Примерами протоколов канального уровня для локальных сетей являются протоколы Ethernet, Token Ring, FDDI, для глобальных – PPP, SLIP, LAP-B, LAP-D.

Для реализации протоколов канального уровня используется специальное оборудование: коммутаторы. Раньше использовались концентраторы и мосты, которые в настоящее время сняты с производства.

Сетевой уровень (Network layer) служит для образования единой системы, объединяющей несколько сетей, причем эти сети могут быть различной топологии, использовать совершенно различные принципы сообщений между конечными узлами и обладать произвольной структурой. Сети соединяются между собой специальными устройствами, называемыми маршрутизаторами. Маршрутизатор – это устройство, которое собирает данные о топологии межсетевых соединений и на ее основании пересылает пакеты информации из одной сети в другую. Последовательность маршрутизаторов, через которые проходит пакет, называется маршрутом, а выбор маршрута называется маршрутизацией. Маршрутизация является главной

задачей сетевого уровня. На сетевом уровне действуют три протокола: сетевой протокол для определения правил передачи пакетов от конечных узлов к маршрутизаторам и между маршрутизаторами; протокол маршрутизации – для сбора информации о топологии сети; протокол разрешения адресов – для отображения адреса узла, используемого на сетевом уровне в локальный адрес сети (ARP-адрес).

Транспортный уровень (Transport layer) предназначен для оптимизации передачи данных от отправителя к получателю с той степенью надежности, которая требуется. Основная задача транспортного уровня – это обнаружение и исправление ошибок в сообщениях, пришедших с описанных выше уровней.

Начиная с транспортного уровня, все дальнейшие протоколы реализуются программным обеспечением компьютера, включаемого обычно в состав сетевой операционной системы.

Сеансовый уровень (Session layer) управляет диалогом между двумя компьютерами. На этом уровне устанавливаются правила начала и завершения взаимодействия. На сеансовом уровне определяется, какая из сторон является активной в данный момент, а какая принимает данные. Приложение должно различать разные потоки данных в пределах одного соединения. Например, приложение может одновременно запрашивать два файла с одного сервера, при этом, благодаря сеансовому уровню, оно будет различать эти два потоки.

Представительный уровень (Presentation layer) выполняет преобразование данных между устройствами с различными форматами данных, не меняя при этом содержания. Благодаря этому уровню информация, передаваемая прикладным уровнем одного компьютера, всегда понятна прикладному уровню другого компьютера. На этом уровне, как правило, происходит шифрование и дешифрование данных, благодаря которому обеспечивается секретность передаваемого сообщения.

Прикладной уровень (Application layer) является пользовательским интерфейсом для работы с сетью. Этот уровень непосредственно взаимодействует с пользовательскими прикладными программами, предоставляя им доступ в сеть. С помощью протоколов этого уровня пользователи сети получают доступ к разделяемым ресурсам, таким как файлы, принтеры, гипертекстовые web-страницы, электронная почта и т. д.

Необходимо отметить, что три нижних уровня модели OSI – физический, канальный и сетевой – зависят от сети, т. е. их протоколы тесно связаны с технической реализацией сети и с используемым коммутационным оборудованием. Три верхних уровня – сеансовый, представления и прикладной – ориентированы на программное обеспечение и мало зависят от особенностей построения сети (топологии, оборудования и т. д.). Транспортный уровень является промежуточным. Он скрывает детали функционирования нижних уровней от верхних уровней. Благодаря этому уровню можно разрабатывать приложения, не зависящие от технических средств транспортировки сообщений.

Модель OSI является так называемой открытой системой, т. е. она имеет опубликованные, общедоступные спецификации и стандарты, принятые в результате достижения согласия многих разработчиков и пользователей после всестороннего обсуждения. Эта модель доступна всем разработчикам и для ее использования не требуется получения специальных лицензий. Если две сети построены с соблюдением правил открытости, то у них есть возможность использования аппаратных и программных средств разных производителей, придерживающихся одного и того же стандарта, такие сети легко сопрягаются друг с другом, просты в освоении и обслуживании.

3.3. Компьютерные сети

Рассмотрим компьютерные сети на основе глобальной сети Интернет. Эта сеть развивалась в полном соответствии с требованиями, предъявляемыми к открытым системам. В разработке стандартов глобальной сети Интернет принимали и принимают участие тысячи специалистов и пользователей этой сети из различных университетов, научных организаций и фирм-производителей вычислительной аппаратуры и программного обеспечения из разных стран.

Само название стандартов, определяющих работу интернет-сети – Request For Comments – переводится как «запрос на комментарии», т. е. для введения новых стандартов в этой сети проводится открытый опрос мнений пользователей, и только затем вносятся изменения. В результате сеть Интернет сумела объединить в себе самое разнообразное оборудование и программное обеспечение огромного числа сетей, разбросанных по всему миру.

Стек проколов TCP/ IP (Transmission Control Protocol/ Internet Protocol) является базовым протоколом сети Интернет. Протокол TCP/ IP предложил сотрудник DARPA Роберт Канн в 1972 году для взаимодействия компьютера с открытой сетевой архитектурой.

Основными преимуществами протокола TCP/ IP являются:

Независимость от сетевой технологии отдельной сети. TCP/ IP не зависит от оборудования, так как он определяет только элемент передачи, который называется дейтаграммой, и описывает способ ее движения по сети.

Всеобщая связанность сетей. Протокол позволяет любой паре компьютеров взаимодействовать друг с другом. Каждому компьютеру назначается логический адрес, а каждая передаваемая дейтаграмма содержит адреса отправителей и получателей. Промежуточные маршрутизаторы используют адрес получателя для принятия решения о маршрутизации.

Подтверждение. Протокол TCP/IP обеспечивает подтверждение правильности прохождения информации при обмене между отправителем и получателем.

Стандартные прикладные протоколы. Протокол TCP/IP включает в свой состав поддержку основных приложений, таких как электронная почта, передача файлов, удаленный доступ и т. д.

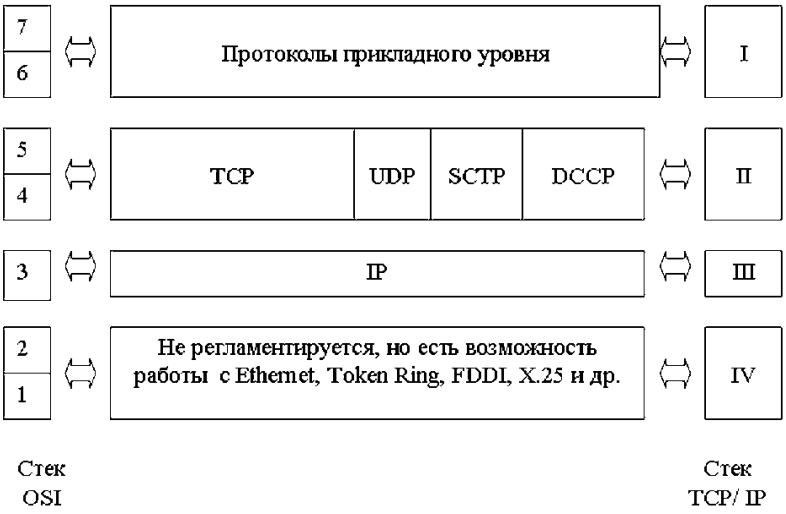


Рис. 3.4. Соответствие стека TCP/IP модели OSI

В стеке TCP/ IP определены 4 уровня взаимодействия, каждый из которых берет на себя определенную функцию по организации надежной работы глобальной сети:

Уровень I. Прикладной уровень.

Уровень II. Основной (транспортный) уровень.

Уровень III. Уровень межсетевого взаимодействия.

Уровень IV. Уровень сетевых интерфейсов.

Соответствие стека TCP/IP модели OSI показано на рис 3.4.

Как видно из рисунка 3.4, протокол TCP занимает транспортный и сеансовый уровень, а на сетевом уровне используется протокол IP.

IP-адресация компьютеров в сети построена на концепции сети, состоящей из хостов. Хост представляет собой объект сети, который может передавать и принимать IP-пакеты, например, компьютер или маршрутизатор.

Отметим, что в модели TCP/IP программные модули, соответствующие транспортному и сеансовому уровню, устанавливаются только на конечных компьютерах.

Программный модуль протокола TCP/ IP реализуется в операционной системе компьютера в виде отдельного системного модуля (драйвера). Интерфейс между прикладным уровнем и TCP представляет собой библиотеку вызовов, такую же, как, например, библиотека системных вызовов для работы с файлами. Пользователь может самостоятельно настраивать протокол TCP/ IP для каждого конкретного случая (количество пользователей сети, пропускная способность физических линий связи и т. д.).

Уровень межсетевого взаимодействия

Уровень межсетевого взаимодействия является стержнем всей архитектуры протокола, который реализует концепцию передачи пакетов в режиме без установления соединений, то есть дейтаграммным способом. Именно этот уровень обеспечивает возможность перемещения пакетов по сети, используя тот маршрут, который в данный момент является оптимальным. Этот уровень также называют уровнем Интернет, подчеркивая его основную функцию – передачу данных через составную сеть. Основным протоколом уровня межсетевого взаимодействия является протокол IP (Internet Protocol). IP-протокол проектировался для передачи пакетов в составных сетях, состоящих из большого количества

локальных сетей, поэтому он хорошо работает в сетях со сложной топологией. Так как IP- протокол является дейтаграммным протоколом, то он не гарантирует доставку пакетов до узла назначения.

Основной (транспортный) уровень

Так как на сетевом уровне не происходит установление соединения, то нет никаких гарантий, что межсетевым уровнем пакеты будут доставлены в место назначения неповрежденными.

Обеспечение надежной связи между двумя конечными компьютерами осуществляет основной уровень стека TCP/IP, называемый также транспортным.

На этом уровне работают основные протоколы:

TCP (Transmission Control Protocol) – основной протокол управления передачей;

UDP (User Datagramm Protocol) – протокол дейтаграмм пользователя;

SCTP (Stream Control Transmission Protocol) – протокол передачи с управлением потоком;

DCCP (Datagram Congestion Control Protocol) – протокол предоставляет механизмы для отслеживания перегрузок в сети, избегая необходимости создавать их на прикладном уровне.

Прикладной уровень

Прикладной уровень объединяет все службы пользователей сети. Прикладной уровень реализуется различными программными системами и постоянно расширяется. Наиболее известными прикладными службами в сети Интернет, например, являются электронная почта (E-mail), система новостей UseNet, всемирная паутина World Wide Web (WWW), передача файлов (FTP), удаленный терминал и терминальные серверы (TELNET) и др.

Уровень сетевых интерфейсов

В отличие от физического и канального уровня модели OSI в архитектуре стека TCP/ IP существует несколько другая интерпретация уровня сетевых интерфейсов. Протоколы этого уровня должны обеспечить интеграцию в составную сеть локальных сетей, использующих различные технологии. Поэтому разработчики той или другой технологии должны предусмотреть возможность инкапсуляции (включения) в свои кадры IP-пакетов. Уровень сетевых интерфейсов в протоколах TCP/IP не регламентируется, но он поддерживает все популярные стандарты физического и канального

уровней: Ethernet, Token Ring, FDDI, Gigabit Ethernet, Fast Ethernet и др. Разработаны спецификации для соединения с сетями X.25, frame relay, ATM.

Отметим, что в настоящее время каждый из разработчиков сетевых технологий канального и физического уровня стремится обеспечить их совместимость с протоколом TCP/ IP.

Адресация в сети Интернет

Для адресации компьютеров в сети Интернет используются IP-адреса.

IP-адреса отправителя и получателя принято называть адресом хоста.

IP-адрес любого из хостов состоит из адреса (номера) сети (сетевого префикса) и адреса хоста в этой сети.

В соответствии с принятым в момент разработки IP-протокола соглашением адрес представляется четырьмя десятичными числами, разделенными точками. Каждое из этих чисел не может превышать 255 и представляет один байт 4-байтного IP-адреса. Выделение всего лишь четырех байт для адресации всей сети Интернет связано с тем, что в то время массового распространения локальных сетей еще не предвиделось. О персональных компьютерах и рабочих станциях вообще не было речи.

В результате под IP-адрес было отведено 32 бита, из которых первые 8 бит обозначали сеть, а оставшиеся 24 бита – компьютер в сети. IP-адрес назначается администратором сети во время конфигурирования компьютеров и маршрутизаторов.

Главным координатором, занимающимся распределением и управлением IP-адресов, доменов и обратных доменов, является некоммерческая организация IANA (Internet Assigned Numbers Authority), агентство по выделению имен и уникальных параметров протоколов Internet). www.iana.org/, работающая по управлением организации Internet Corporation for Assigned Names and Numbers, или ICANN – международной некоммерческой организации при участии правительства США для регулирования вопросов, связанных с доменными именами, IP-адресами и прочими аспектами функционирования сети Интернет.

Отметим, что маршрутизатор может входить сразу в несколько сетей, поэтому каждый порт маршрутизатора имеет свой IP-адрес. Таким же образом и конечный компьютер так же может входить

в несколько сетей, а значит иметь несколько IP-адресов. Таким образом, IP- адрес характеризует не отдельный компьютер или маршрутизатор, а одно сетевое соединение. Как уже отмечалось выше, адрес состоит из двух частей – номера сети и номера узла в сети. Для того чтобы определить, какая часть адреса относится к номеру сети, а какая к номеру узла, в начале адреса несколько бит отводится для определения класса сети.

3.4. Поисковые системы, библиографические каталоги и сервисы при организации научного исследования

На современном этапе развития науки, характеризующемся экспоненциальным ростом объема публикаций, эффективная организация исследовательской деятельности немыслима без использования специализированного цифрового инструментария. Ключевую роль в этом процессе играют два типа ресурсов: академические поисковые системы и библиографические сервисы. Их грамотное применение позволяет исследователю не просто находить релевантные источники, но и системно управлять ими на всех этапах работы – от постановки проблемы до оформления рукописи и формирования списка цитирований.

3.4.1. Академические поисковые системы

Академические поисковые системы представляют собой специализированные инструменты, индексирующие научный контент (статьи, диссертации, книги, препринты, патенты). В отличие от универсальных систем (Google, Yandex), они ориентированы на глубокий анализ метаданных публикаций и установление связей между ними.

Ключевые игроки и их особенности:

1. *Google Академия (Google Scholar)*. Крупнейший и наиболее доступный бесплатный ресурс. Индексирует материалы из самых разнообразных источников: рецензируемые журналы, репозитории университетов, научные сайты. Преимущества: простота использования, широкий охват, функция формирования цитирований в стандартных стилях (APA, ГОСТ и др.). Недостатки: непрозрачный алгоритм ранжирования, возможное включение в выдачу материалов низкого качества, меньший контроль за метаданными по сравнению с платными базами.

2. *Scopus и Web of Science (WoS).* Это коммерческие библиографические и реферативные базы данных, считающиеся «золотым стандартом» в научном мире.

3. *Scopus* (издательство Elsevier) отличается широчайшим охватом журналов по техническим, естественнонаучным и медицинским дисциплинам, а также социальным наукам. Ключевые метрики: CiteScore, SJR (SCImago Journal Rank).

4. *Web of Science* (издательство Clarivate) исторически сильна в фундаментальных науках. Опирается на тщательно отбираемый индекс цитирования (Science Citation Index Expanded, Social Sciences Citation Index и др.). Ключевая метрика – импакт-фактор журнала (Journal Impact Factor).

Преимущества. Высокое качество проиндексированных журналов, мощные аналитические инструменты для анализа публикационной активности и цитирований, возможность построения библиометрических карт науки.

Недостатки. Платный доступ (как правило, через подписку университета).

PubMed – бесплатная поисковая система по биомедицинской и жизненной тематике, поддерживаемая Национальными институтами здоровья США (NIH). Содержит ссылки на полные тексты в PubMed Central и на сайтах издательств.

РИНЦ (Российский индекс научного цитирования) и eLibrary.ru – крупнейшая национальная база данных, агрегирующая информацию о публикациях российских ученых. Основа для оценки продуктивности научных организаций и researchers в России. Позволяет отслеживать индекс Хирша, импакт-факторы российских журналов.

3.4.2. Тактика эффективного поиска

1. Использование расширенного поиска (операторы AND, OR, NOT, поиск по конкретным полям: автор, заголовок, ключевые слова).

2. Анализ списка литературы релевантных статей (поиск «вглубь»).

3. Использование функции «Цитирующие статьи» (поиск «вперед») для отслеживания новейших исследований, развивающих идеи исходной публикации.

4. Сохранение поисковых запросов и настройка оповещений о новых публикациях по заданной теме.

3.4.3. Библиографические менеджеры: систематизация и автоматизация

Библиографические сервисы (менеджеры) – это программное обеспечение, предназначенное для создания, хранения, организации и использования персональных коллекций библиографических ссылок.

Функциональные возможности:

1. *Создание личной библиографической базы данных*. Импорт ссылок из поисковых систем и библиотечных каталогов одним кликом (через прямую интеграцию или по DOI/ISBN).

2. *Структурирование и организация*. Сортировка записей по проектам, темам, присвоение тегов, аннотирование, полнотекстовый поиск по прикрепленным PDF-файлам.

3. *Интеграция с текстовыми редакторами*. Установка плагинов для MS Word, LibreOffice или Google Docs позволяет автоматически вставлять цитаты в текст и форматировать список литературы в нужном стиле (более 10 000 предустановленных стилей, включать ГОСТы).

4. *Совместная работа*. Многие современные менеджеры (Zotero, Mendeley) поддерживают создание общих библиотек для научных групп.

Популярные сервисы:

Zotero. Бесплатный, с открытым исходным кодом. Отличается простотой использования, мощным браузерным расширением для захвата ссылок, гибкостью за счет подключаемых модулей.

Mendeley. Сочетает функции менеджера и социальной сети для researchers. Имеет встроенный PDF-ридер с функцией аннотирования, предлагает рекомендации на основе содержимого библиотеки.

EndNote. Один из старейших коммерческих продуктов. Обладает очень широкими возможностями по кастомизации стилей цитирования, интегрирован с Web of Science. Часто считается стандартом в крупных издательствах и университетах.

3.4.4. Стратегия организации исследования

Гармоничное сочетание поисковых систем и библиографических менеджеров формирует ядро методологии современного ученого.

Поэтапная модель работы:

1. *Разведка (Scoping)*. Использование Google Академия и тематических баз данных (Scopus/WoS) для получения общего представления о поле исследования, выявления ключевых работ, авторов и журналов. Анализ обзоров литературы (review articles).

2. *Глубинный поиск (Deep Search).* Формулирование точных поисковых запросов, использование расширенных возможностей платформ. Систематический импорт всех найденных релевантных материалов в библиографический менеджер.

3. *Синтез и организация (Synthesis)*. Внутри менеджера происходит структурирование источников: создание папок по главам или темам, добавление аннотаций, выделение ключевых цитат. Это превращает коллекцию ссылок в структурированную базу знаний по проекту.

4. *Написание (Writing)*. Интеграция менеджера с текстовым процессором для безошибочного оформления цитат и библиографии. Динамическое обновление списка литературы при его изменении.

5. *Мониторинг (Monitoring)*. Использование функций оповещения в поисковых системах для отслеживания новых публикаций по теме даже на этапе написания работы.

В условиях информационной перегрузки владение цифровым инструментарием становится не просто удобным навыком, а критически важным компонентом научной компетентности. Поисковые системы выступают в роли «радара», сканирующего академический ландшафт, а библиографические менеджеры – это «интеллектуальный штаб», где собранная информация систематизируется и превращается в основу для нового знания. Их комплексное применение значительно повышает продуктивность, эффективность и, в конечном счете, качество проводимого научного исследования.

3.5. Облачные технологии (Cloud Computing)

Современная цифровая трансформация опирается на два фундаментальных столпа: облачные технологии, которые предоставляют практически неограниченные вычислительные ресурсы по запросу,

и интернет вещей (IoT), который стирает грань между физическим и цифровым миром. Эти парадигмы не просто существуют параллельно; они взаимно усиливают друг друга, создавая экосистему, где данные от миллиардов устройств находят смысл и применение в мощных облачных центрах обработки данных.

Определение: Облачные технологии – это модель предоставления повсеместного и удобного сетевого доступа по требованию к общему пулу конфигурируемых вычислительных ресурсов (например, серверов, систем хранения данных, приложений, сервисов), которые могут быть оперативно предоставлены и выпущены с минимальными эксплуатационными затратами или взаимодействием с поставщиком услуг.

Ключевые характеристики (по версии NIST):

1. *Самообслуживание по требованию.* Пользователь может автоматически получать вычислительные мощности (например, время сервера, ресурсы хранения) без взаимодействия с человеком-оператором.

2. *Универсальный доступ по сети*. Услуги доступны через сеть (Интернет) по стандартным механизмам с различных клиентских платформ (смартфоны, ноутбуки, рабочие станции).

3. *Объединение ресурсов (Resource Pooling)*. Ресурсы провайдера объединяются в пул для обслуживания множества потребителей по модели мультитенантности (multi-tenant), с динамическим назначением и переназначением ресурсов в соответствии со спросом.

4. *Эластичность (Elasticity).* Ресурсы могут быть оперативно и гибко выделены и освобождены (часто автоматически) для быстрого масштабирования в соответствии с изменяющимся спросом.

5. *Учет потребления (Measured Service)*. Использование ресурсов автоматически контролируется, оптимизируется и фиксируется провайдером, что обеспечивает прозрачность для both поставщика и потребителя.

Модели обслуживания (Service Models):

1. *IaaS (Infrastructure as a Service).* Предоставление фундаментальных вычислительных инфраструктур (виртуальные серверы, сетевое оборудование, системы хранения). Пример: Amazon EC2, Google Compute Engine.

2. *PaaS (Platform as a Service)*. Предоставление среды для разработки и развертывания приложений без необходимости управления базовой инфраструктурой. Пример: Google App Engine, Microsoft Azure App Services.

3. *SaaS (Software as a Service).* Предоставление готового к использованию программного обеспечения, работающего в облачной инфраструктуре и доступного через тонкий клиент (например, веб-браузер). Пример: Google Workspace, Salesforce, Microsoft 365.

Модели развертывания (Deployment Models):

1. *Публичное облако*. Ресурсы принадлежат провайдеру и предоставляются всем клиентам.

2. *Частное облако*. Инфраструктура используется исключительно одной организацией.

3. *Гибридное облако*. Комбинация двух или более различных облачных инфраструктур (частных и публичных), остающихся уникальными объектами, но связанных между собой стандартизированными технологиями.

Преимущества:

1. *Экономическая эффективность*. Оплата по факту использования (pay-as-you-go), отсутствие капитальных затрат на собственные серверы.

2. *Масштабируемость и гибкость*. Возможность мгновенно увеличить или уменьшить мощности в зависимости от нагрузки.

3. *Надежность и отказоустойчивость*. Крупные провайдеры обеспечивают высокий уровень uptime и географическое резервирование данных.

4. *Скорость развертывания*. Новые приложения и сервисы можно запустить за считанные минуты.

3.5. Интернет вещей (Internet of Things, IoT)

Определение. Интернет вещей – это концепция вычислительной сети физических объектов (вещей), оснащенных встроенными технологиями для взаимодействия друг с другом и с внешней средой,

а также для удаленного сбора и обмена данными с минимальным участием человека.

Архитектура IoT-системы (упрощенно):

Периферийный уровень (устройства и датчики). «Вещи» – это датчики (измеряющие температуру, влажность, движение) и актуаторы (выполняющие действия – включить свет, открыть клапан). Они собирают данные из физического мира.

Сетевой уровень (шлюзы и связь). Устройства передают данные через сети связи (Wi-Fi, Bluetooth, Zigbee, LPWAN – LoRaWAN, NB-IoT) на шлюзы, которые агрегируют и предварительно обрабатывают данные.

Уровень платформы (Облако/периферийные вычисления). Данные поступают в облачную платформу для глубокой обработки, хранения и анализа.

Уровень приложений (визуализация и управление). Пользовательский интерфейс (веб-панель, мобильное приложение), который представляет обработанные данные в понятном виде и позволяет управлять устройствами.

Ключевые технологии IoT:

Микроконтроллеры и сенсоры. Дешевые и энергоэффективные вычислительные модули (Arduino, ESP32, Raspberry Pi).

Протоколы связи. Специализированные протоколы для малого энергопотребления и работы на больших расстояниях (MQTT, CoAP).

Платформы IoT (IoT Platforms). Специализированные облачные сервисы для управления устройствами, данными и правилами (Google Cloud IoT Core, AWS IoT, Azure IoT Hub).

Синергия облаков и IoT: симбиоз технологий

Облачные технологии и IoT образуют идеальный симбиоз:

IoT как генератор данных, облако – как мозг. Миллиарды IoT-устройств генерируют колоссальные объемы данных (Big Data). Облако предоставляет практически бесконечные ресурсы для их приема, хранения и сложного анализа с помощью алгоритмов машинного обучения и искусственного интеллекта.

Масштабируемость. Облако позволяет IoT-системе effortlessly масштабироваться. При подключении тысяч новых датчиков облачная инфраструктура автоматически предоставит дополнительные ресурсы для обработки возросшей нагрузки.

Глобальная доступность и управление. Облачные платформы позволяют управлять географически распределенной сетью устройств из единого центра, мониторить их состояние и удаленно обновлять прошивки.

Экономическая модель. Модель оплаты по мере использования в облаках идеально подходит для IoT-стартапов и проектов, позволяя начинать с малого и наращивать инфраструктуру вместе с ростом числа устройств.

Пример синергии: Умный город

IoT. Датчики на дорогах отслеживают трафик, камеры наблюдения фиксируют события, умные счетчики учитывают потребление ресурсов.

Облако. Все данные стекаются в облачный дата-центр. Алгоритмы ИИ анализируют потоки трафика в реальном времени и оптимизируют работу светофоров, выявляют правонарушения, прогнозируют потребление воды и электроэнергии.

Результат. Городская инфраструктура становится эффективнее, безопаснее и удобнее для жителей.

Вызовы и будущее

Безопасность. Каждое IoT-устройство – это потенциальная точка входа для атак. Защита периметра из миллионов устройств – ключевая задача.

Задержки (Latency). Для критически важных приложений (беспилотные автомобили, промышленная автоматизация) задержка при передаче данных в облако и обратно может быть неприемлема. Решение – Периферийные вычисления (Edge Computing), когда данные обрабатываются не в центральном облаке, а на самом устройстве или на локальном шлюзе, что значительно ускоряет реакцию.

Облачные технологии и Интернет вещей – это две стороны одной медали цифровизации. IoT оцифровывает физический мир, генерируя данные, а облака предоставляют интеллект для их осмысления и превращения в практические действия и бизнес-ценность.

4. ЗАЩИТА ИНФОРМАЦИИ

4.1. Концепция обеспечения информационной безопасности

Безопасность информационных технологий и систем является одной из важнейших составляющих проблемы обеспечения безопасности организации.

В общем случае информационную безопасность можно определить как защищенность информации, ресурсов и поддерживающей инфраструктуры от случайных или преднамеренных воздействий естественного или искусственного характера, которые могут нанести неприемлемый ущерб субъектам информационных отношений – производителям, владельцам и пользователям информации и поддерживающей инфраструктуре.

Требования по обеспечению безопасности в различных информационных системах могут существенно отличаться, однако они всегда направлены на достижение трех основных свойств:

1) целостности – данные и информация, на основе которой принимаются решения, должны быть достоверными, точными и защищенными от возможных непреднамеренных и злоумышленных искажений;

2) доступности (готовности) – данные, информация и соответствующие службы, автоматизированные сервисы, средства взаимодействия и связи должны быть доступны и готовы к работе всегда, когда в них возникает необходимость;

3) конфиденциальности – засекреченная информация должна быть доступна только тому, кому она предназначена.

Информационная безопасность не сводится исключительно к защите от несанкционированного доступа к информации – это принципиально более широкое понятие (рис. 4.1)

Основой программы обеспечения информационной безопасности является многоуровневая политика безопасности, отражающая подход организации к защите своих информационных активов и ресурсов. Под политикой безопасности понимается совокупность документированных методологий и управленческих решений, а также распределение ролей и ответственности, направленных на защиту информации, информационных систем и ассоциированных с ними ресурсов (рис. 4.2).

Рис. 4.1. Структура информационной безопасности

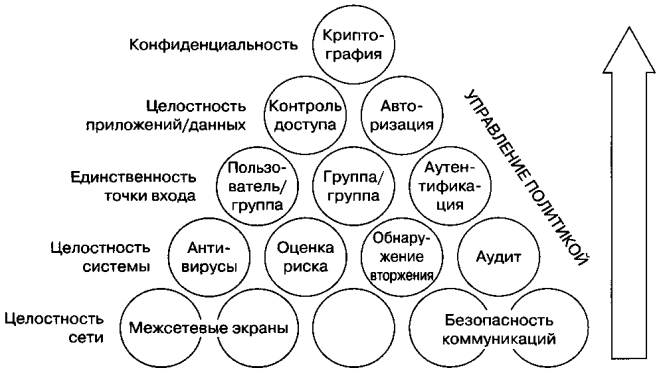


Рис. 4.2. Функциональная модель системы информационной безопасности

Исторически первым оценочным стандартом, получившим широкое распространение и оказавшим огромное влияние на базу стандартизации ИБ во многих странах, стал стандарт Министерства обороны США. Критерии оценки доверенных компьютерных систем – стандарт Министерства обороны США (англ. Department of Defense Trusted Computer System Evaluation Criteria, TCSEC, DoD 5200.28-STD, December 26, 1985), более известный под именем «Оранжевая книга» (англ. «Orange Book») из-за цвета обложки – стандарт Министерства обороны США, устанавливающий основные условия для оценки эффективности средств компьютерной безопасности,

содержащихся в компьютерной системе. Критерии используются для определения, классификации и выбора компьютерных систем, предназначенных для обработки, хранения и поиска важной или секретной информации. Данный стандарт получил международное признание и оказал исключительно сильное влияние на последующие разработки в области информационной безопасности(ИБ).

Аналогом «Оранжевой книги» является международный стандарт ISO/IEC 15408, опубликованный в 2005 году. Это более универсальный и совершенный стандарт, но вопреки распространенному заблуждению, он не заменял собой «Оранжевую книгу» в силу разной юрисдикции документов – «Оранжевая книга» используется исключительно Министерством обороны США, в то время как ISO/IEC 15408 ратифицировали множество стран. В последние годы Правительством Республики Беларусь предприняты конкретные шаги по формированию государственной политики в области информатизации, которые актуализируют проблемы разработки правовой основы, включая меры ответственности за правонарушения в инфосфере.

Каких-либо абсолютных систем (в том числе и безопасных) в нашей жизни не существует. Поэтому и было предложено оценивать лишь степень доверия, которое можно оказать той или иной системе. В стандарте заложен понятийный базис информационной безопасности. Безопасность и доверие оцениваются в данном стандарте с точки зрения управления доступом к информации, что и является средством обеспечения конфиденциальности и целостности.

Вслед за «Оранжевой книгой» появилась целая «Радужная серия». Наиболее значимой в ней явилась интерпретация «Оранжевой книги» для сетевых конфигураций (англ. National Computer Security Center. Trusted Network Interpretation, NCSC-TG-005, 1987), где в первой части интерпретируется «Оранжевая книга», а во второй части описываются сервисы безопасности, специфичные для сетевых конфигураций.

Политики безопасности должны быть подробными, четко определенными и обязательными для компьютерной системы. Есть две основных политики безопасности:

– мандатная политика безопасности – обязательные правила управления доступом напрямую, основанные на индивидуальном разрешении, разрешении на доступ к информации и уровне конфиденциальности запрашиваемой информации. Другие

косвенные факторы являются существенными и окружающими. Эта политика также должна точно соответствовать закону, главной политике и прочим важным руководствам, в которых устанавливаются правила;

– дискреционная политика безопасности – предоставляет непротиворечивый набор правил для управления и ограничения доступа, основанный на идентификации тех пользователей, которые намерены получить только необходимую им информацию.

Индивидуальная ответственность в независимости от политики должна быть обязательной. Есть три требования по условиям ответственности:

– аутентификация – процесс, используемый для распознавания индивидуального пользователя;

– авторизация – проверка разрешения индивидуальному пользователю на получение информации определенного рода;

– аудит – контролируемая информация должна избирательно храниться и защищаться в мере, достаточной для отслеживания действий аутентифицированного пользователя, затрагивающих безопасность.

Компьютерная система должна содержать аппаратные и/или программные механизмы, которые могут независимо определять обеспечивается ли достаточная уверенность в том, что система исполняет указанные выше требования. Вдобавок уверенность должна включать гарантию того, что безопасная часть системы работает только так, как запланировано. Для достижения этих целей необходимо два типа гарантий и соответствующих им элементов:

Механизмы гарантий:

– операционная гарантия – уверенность в том, что реализация спроектированной системы обеспечивает осуществление принятой стратегии защиты системы. Сюда относятся системная архитектура, целостность системы, анализ скрытых каналов, безопасное управление возможностями и безопасное восстановление;

– гарантия жизненного цикла – уверенность в том, что система разработана и поддерживается в соответствии с формализованными и жестко контролируемыми критериями функционирования. Сюда относятся тестирование безопасности, задание на проектирование и его проверка, управление настройками и соответствие параметров системы заявленным;

– гарантии непрерывной защиты – надежные механизмы, обеспечивающие непрерывную защиту основных средств от преступных и/или несанкционированных изменений.

Безопасная система – это система, которая обеспечивает управление доступом к информации, таким образом, что только авторизованные лица или процессы, действующие от их имени, получают право работы с информацией.

Под доверенной системой в стандарте понимается система, использующая аппаратные и программные средства для обеспечения одновременной обработки информации разной категории секретности группой пользователей без нарушения прав доступа.

Политика безопасности – набор законов, правил, процедур и норм поведения, определяющих, как организация обрабатывает, защищает и распространяет информацию. Причем, политика безопасности относится к активным методам защиты, поскольку учитывает анализ возможных угроз и выбор адекватных мер противодействия.

Уровень гарантированности подразумевает меру доверия, которая может быть оказана архитектуре и реализации информационной системы, и показывает, насколько корректны механизмы, отвечающие за реализацию политики безопасности (пассивный аспект защиты).

В группе «Подотчетность» должны быть следующие требования:

1) *Идентификация и аутентификация*. Все субъекты должны иметь уникальные идентификаторы. Контроль доступа должен осуществляться на основании результатов идентификации субъекта и объекта доступа, подтверждения подлинности их идентификаторов (аутентификации) и правил разграничения доступа. Данные, используемые для идентификации и аутентификации, должны быть защищены от несанкционированного доступа, модификации и уничтожения и должны быть ассоциированы со всеми активными компонентами компьютерной системы, функционирование которых критично с точки зрения безопасности.

2) *Регистрация и учет*. Для определения степени ответственности пользователей за действия в системе, все происходящие в ней события, имеющие значение с точки зрения безопасности, должны отслеживаться и регистрироваться в защищенном протоколе (то есть должен существовать объект компьютерной системы, потоки от которого и к которому доступны только субъекту администрирования). Система регистрации должна осуществлять анализ

общего потока событий и выделять из него только те события, которые оказывают влияние на безопасность для сокращения объема протокола и повышения эффективности его анализа. Протокол событий должен быть надежно защищен от несанкционированного доступа, модификации и уничтожения.

Доверенная вычислительная база – совокупность защитных механизмов информационной системы (как программные, так и аппаратные), реализующих политику безопасности.

Монитор обращений – контроль за выполнением субъектами (пользователями) определенных операций над объектами, путем проверки допустимости обращения (данного пользователя) к программам и данным разрешенному набору действий.

Обязательные качества для монитора обращений:

1. Изолированность (неотслеживаемость работы).
2. Полнота (невозможность обойти).
3. Верифицируемость (возможность анализа и тестирования).
4. Ядро безопасности – конкретная реализация монитора обращений, обладающая гарантированной неизменностью.
5. Периметр безопасности – граница доверенной вычислительной базы.

Механизмы реализации безопасности

Добровольное управление доступом – это метод ограничения доступа к объектам, основанный на учете личности субъекта или группы, в которую субъект входит. Добровольность управления состоит в том, что некоторое лицо (обычно владелец объекта) может по своему усмотрению давать другим субъектам или отбирать у них права доступа к объекту.

Большинство операционных систем и СУБД реализуют именно добровольное управление доступом. Главное его достоинство – гибкость, главные недостатки – рассредоточенность управления и сложность централизованного контроля, а также оторванность прав доступа от данных, что позволяет копировать секретную информацию в общедоступные файлы или секретные файлы в незащищенные каталоги.

Безопасность повторного использования объектов – важное на практике дополнение средств управления доступом, предохраняющее от случайного или преднамеренного извлечения секретной информации из «мусора». Безопасность повторного использования

должна гарантироваться для областей оперативной памяти (в частности, для буферов с образами экрана, расшифрованными паролями и т. п.), для дисковых блоков и магнитных носителей в целом. Важно обратить внимание на следующий момент. Поскольку информация о субъектах также представляет собой объект, необходимо позаботиться о безопасности «повторного использования субъектов». Когда пользователь покидает организацию, следует не только лишить его возможности входа в систему, но и запретить доступ ко всем объектам. В противном случае новый сотрудник может получить ранее использовавшийся идентификатор, а с ним и все права своего предшественника.

Современные интеллектуальные периферийные устройства усложняют обеспечение безопасности повторного использования объектов. Действительно, принтер может буферизовать несколько страниц документа, которые останутся в памяти даже после окончания печати. Необходимо предпринять специальные меры, чтобы «вытолкнуть» их оттуда.

Метки безопасности предусмотрены для субъектов (степень благонадежности) и объектов (степень конфиденциальности информации). Метки безопасности содержат данные об уровне секретности и категории, к которой относятся данные. Согласно «Оранжевой книге», метки безопасности состоят из двух частей – уровня секретности и списка категорий. Уровни секретности, поддерживаемые системой, образуют упорядоченное множество, которое может выглядеть, например, так: совершенно секретно; секретно; конфиденциально; несекретно.

Для разных систем набор уровней секретности может различаться. Категории образуют неупорядоченный набор. Их назначение – описать предметную область, к которой относятся данные. В военном окружении каждая категория может соответствовать, например, определенному виду вооружений. Механизм категорий позволяет разделить информацию по отсекам, что способствует лучшей защищенности. Субъект не может получить доступ к «чужим» категориям, даже если его уровень благонадежности «совершенно секретно». Специалист по танкам не узнает тактико-технические данные самолетов.

Главная проблема, которую необходимо решать в связи с метками, это обеспечение их целостности. Во-первых, не должно быть непомеченных субъектов и объектов, иначе в меточной безопасности

появятся легко используемые бреши. Во-вторых, при любых операциях с данными метки должны оставаться правильными. В особенности это относится к экспорту и импорту данных. Например, печатный документ должен открываться заголовком, содержащим текстовое и/или графическое представление метки безопасности. Аналогично при передаче файла по каналу связи должна передаваться и ассоциированная с ним метка, причем в таком виде, чтобы удаленная система могла ее разобрать, несмотря на возможные различия в уровнях секретности и наборе категорий.

Одним из средств обеспечения целостности меток безопасности является разделение устройств на многоуровневые и одноуровневые. На многоуровневых устройствах может храниться информация разного уровня секретности (точнее, лежащая в определенном диапазоне уровней). Одноуровневое устройство можно рассматривать как вырожденный случай многоуровневого, когда допустимый диапазон состоит из одного уровня. Зная уровень устройства, система может решить, допустимо ли записывать на него информацию с определенной меткой. Например, попытка напечатать совершенно секретную информацию на принтере общего пользования с уровнем «несекретно» потерпит неудачу.

Принудительное управление доступом основано на сопоставлении меток безопасности субъекта и объекта. Субъект может читать информацию из объекта, если уровень секретности субъекта не ниже, чем у объекта, а все категории, перечисленные в метке безопасности объекта, присутствуют в метке субъекта. В таком случае говорят, что метка субъекта доминирует над меткой объекта. Субъект может записывать информацию в объект, если метка безопасности объекта доминирует над меткой субъекта. В частности, «конфиденциальный» субъект может писать в секретные файлы, но не может – в несекретные (разумеется, должны также выполняться ограничения на набор категорий). На первый взгляд подобное ограничение может показаться странным, однако оно вполне разумно. Ни при каких операциях уровень секретности информации не должен понижаться, хотя обратный процесс вполне возможен.

Описанный способ управления доступом называется принудительным, поскольку он не зависит от воли субъектов, на месте которых могут оказаться даже системные администраторы. После того, как зафиксированы метки безопасности субъектов и объектов,

оказываются зафиксированными и права доступа. В терминах принудительного управления нельзя выразить предложение «разрешить доступ к объекту X еще и для пользователя Y». Конечно, можно изменить метку безопасности пользователя Y, но тогда он, скорее всего, получит доступ ко многим дополнительным объектам, а не только к X.

Принудительное управление доступом реализовано во многих вариантах операционных систем и СУБД, отличающихся повышенными мерами безопасности. В частности, такие варианты существуют для SunOS и СУБД Ingres. Независимо от практического использования принципы принудительного управления являются удобным методологическим базисом для начальной классификации информации и распределения прав доступа. Удобнее мыслить в терминах уровней секретности и категорий, чем заполнять неструктурированную матрицу доступа. Впрочем, в реальной жизни добровольное и принудительное управление доступом сочетается в рамках одной системы, что позволяет использовать сильные стороны обоих подходов.

Политика безопасности является важнейшим звеном в формировании информационной безопасности, поэтому принятие решения о ее разработке, внедрении и неукоснительном выполнении всегда принимается высшим руководством организации. Формирование политики информационной безопасности должно сводиться к следующим практическим шагам.

Шаг 1. Определение используемых руководящих документов и стандартов в области информационной безопасности, а также основных положений политики, включая:

– принципы администрирования системы информационной безопасности и управление доступом к вычислительным и телекоммуникационным средствам, программам и ИР, а также доступом в помещения, где они располагаются;

– принципы контроля состояния систем защиты информации, способы информирования об инцидентах в области информационной безопасности и выработку корректирующих мер, направленных на устранение угроз;

– принципы использования ИР персоналом компании и внешними пользователями;

– антивирусную защиту и защиту против действий хакеров;

– вопросы резервного копирования данных и информации;

– проведение профилактических, ремонтных и восстановительных работ;

– обучение и повышение квалификации персонала.

Шаг 2. Разработка методологии выявления и оценки угроз и рисков их осуществления, определение подходов к управлению рисками: является ли достаточным базовый уровень защищенности или требуется проводить полный вариант анализа рисков.

Шаг 3. Структуризация контрмер по уровням требований к безопасности.

Шаг 4. Порядок сертификации на соответствие стандартам в области информационной безопасности. Должна быть определена периодичность проведения совещаний на уровне руководства, включая периодический пересмотр положений политики информационной безопасности, а также порядок обучения всех категорий пользователей ИС по вопросам информационной безопасности.

4.2. Методы и средства защиты информации

Существуют следующие методы обеспечения безопасности информации в информационных системах:

– препятствие;

– управление доступом;

– механизмы шифрования;

– противодействие атакам вредоносных программ;

– регламентация;

– принуждение;

– побуждение.

Препятствие – метод физического преграждения пути злоумышленнику к защищаемой информации (аппаратуре, носителям информации и т. д.).

Управление доступом – методы защиты информации регулированием использования всех ресурсов ИС и ИТ. Эти методы должны противостоять всем возможным путям несанкционированного доступа к информации.

Управление доступом включает следующие функции зашиты:

– идентификацию пользователей, персонала и ресурсов системы (присвоение каждому объекту персонального идентификатора);

– опознание (установление подлинности) объекта или субъекта по предъявленному им идентификатору;

– проверку полномочий (проверка соответствия дня недели, времени суток, запрашиваемых ресурсов и процедур установленному регламенту);

– разрешение и создание условий работы в пределах установленного регламента;

– регистрацию (протоколирование) обращений к защищаемым ресурсам;

– реагирование (сигнализация, отключение, задержка работ, отказ в запросе и т. п.) при попытках несанкционированных действий.

Механизмы шифрования – криптографическое закрытие информации. Эти методы защиты все шире применяются как при обработке, так и при хранении информации на магнитных носителях. При передаче информации по каналам связи большой протяженности этот метод является единственно надежным.

Противодействие атакам вредоносных программ предполагает комплекс разнообразных мер организационного характера и использование антивирусных программ. Цели принимаемых мер – это уменьшение вероятности инфицирования АИС, выявление фактов заражения системы; уменьшение последствий информационных инфекций, локализация или уничтожение вирусов; восстановление информации в ИС.

Регламентация – создание таких условий автоматизированной обработки, хранения и передачи защищаемой информации, при которых нормы и стандарты по защите выполняются в наибольшей степени.

Принуждение – метод защиты, при котором пользователи и персонал ИС вынуждены соблюдать правила обработки, передачи и использования защищаемой информации под угрозой материальной, административной или уголовной ответственности.

Побуждение – метод защиты, побуждающий пользователей и персонал ИС не нарушать установленные порядки за счет соблюдения сложившихся моральных и этических норм.

Вся совокупность технических средств подразделяется на аппаратные и физические.

Аппаратные средства – устройства, встраиваемые непосредственно в вычислительную технику, или устройства, которые сопрягаются с ней по стандартному интерфейсу.

Физические средства включают различные инженерные устройства и сооружения, препятствующие физическому проникновению злоумышленников на объекты защиты и осуществляющие защиту персонала (личные средства безопасности), материальных средств и финансов, информации от противоправных действий. Примеры физических средств: замки на дверях, решетки на окнах, средства электронной охранной сигнализации и т. п.

Программные средства – это специальные программы и программные комплексы, предназначенные для защиты информации в ИС. Как отмечалось, многие из них слиты с ПО самой ИС.

Из средств ПО системы защиты выделим еще программные средства, реализующие механизмы шифрования (криптографии). Криптография – это наука об обеспечении секретности и/или аутентичности (подлинности) передаваемых сообщений.

Организационные средства осуществляют своим комплексом регламентацию производственной деятельности в ИС и взаимоотношений исполнителей на нормативно-правовой основе таким образом, что разглашение, утечка и несанкционированный доступ к конфиденциальной информации становятся невозможным или существенно затрудняются за счет проведения организационных мероприятий. Комплекс этих мер реализуется группой информационной безопасности, но должен находиться под контролем первого руководителя.

Законодательные средства защиты определяются законодательными актами страны, которыми регламентируются правила пользования, обработки и передачи информации ограниченного доступа и устанавливаются меры ответственности за нарушение этих правил.

Морально-этические средства защиты включают всевозможные нормы поведения (которые традиционно сложились ранее), складываются по мере распространения ИС и ИТ в стране и в мире или специально разрабатываются. Морально-этические нормы могут быть неписаные (например, честность) либо оформленные в некий свод (устав) правил или предписаний. Эти нормы, как правило, не являются законодательно утвержденными, но поскольку их несоблюдение приводит к падению престижа организации, они считаются обязательными для исполнения.

Кодирование и декодирование – процесс представления информации в определенной стандартной форме и обратный процесс восстановления информации по такому же ее представлению.

В математической литературе кодированием называется отображение произвольного множества A в множество конечных последовательностей в некотором алфавите B, а декодированием – обратное отображение. Примерами кодирования являются: представление натуральных чисел в r-ичной системе счисления, причем каждому числу $N = i, 2, ...$ ставится в соответствие слово $b1b2 ... bl$ в алфавите $Br = \{0, 1, ..., r – 1\}$ такое, что $b1$ не равно 0 и $b1rl – 1 + ... + bl – 1r + bl = N$; преобразование текстов на русском языке с помощью телеграфного кода в последовательности, составленные из посылок тока и пауз различной длительности; отображение, применяемое при написании цифр почтового индекса. Исследование различных свойств кодирования и декодирования и построение эффективных в определенном смысле кодирований, обладающих требуемыми свойствами, составляет проблематику теории кодирования. Обычно критерий эффективности кодирования так или иначе связан с минимизацией длин кодовых слов (образов элементов множества A), а требуемые свойства кодирования связаны с обеспечением заданного уровня помехоустойчивости, понимаемой в том или ином смысле. В частности, под помехоустойчивостью понимается возможность однозначного декодирования при отсутствии или допустимом уровне искажений в кодовых словах. Помимо помехоустойчивости, к кодированию может предъявляться ряд дополнительных требований. Проблематика теории кодирования в основном создавалась под влиянием разработанной К. Шенноном теории передачи информации. Источником новых задач теории кодирования служат создание и совершенствование автоматизированных систем сбора, хранения, передачи и обработки информации. Методы решения задач теории кодирования главным образом комбинаторные, теоретико-вероятностные и алгебраические. Произвольное кодирование f множества (алфавита) A словами в алфавите B можно распространить на множество A^* всех слов в A (сообщений) следующим образом:

$$f(a_1, a_2, \ldots, a_k) = f(a_1) f(a_2) \ldots f(a_k),$$

где $a_i \in A$, $i = 1, 2, \ldots, k$.

Одним из направлений теории кодирования является изучение общих свойств кодирования и построение алгоритмов распознавания этих свойств. В частности, для побуквенных и автоматных

кодирований найдены необходимые и достаточные условия для того, чтобы: 1) декодирование было однозначным; 2) существовал декодирующий автомат, т. е. автомат, реализующий декодирование с некоторой ограниченной задержкой; 3) существовал самонастраивающийся декодирующий автомат (позволяющий в течение ограниченного промежутка времени устранить влияние сбоя во входной последовательности или в работе самого автомата).

4.2.1. Защита от несанкционированного доступа к данным

Несанкционированный доступ (НСД) злоумышленника на компьютер опасен не только возможностью прочтения и/или модификации обрабатываемых электронных документов, но и возможностью внедрения злоумышленником управляемой программной закладки, которая позволит ему предпринимать следующие действия:

1. Читать и/или модифицировать электронные документы, которые в дальнейшем будут храниться или редактироваться на компьютере.
2. Осуществлять перехват различной ключевой информации, используемой для защиты электронных документов.
3. Использовать захваченный компьютер в качестве плацдарма для захвата других компьютеров локальной сети.
4. Уничтожить хранящуюся на компьютере информацию или вывести компьютер из строя путем запуска вредоносного программного обеспечения.

Защита компьютеров от НСД является одной из основных проблем защиты информации, поэтому в большинство операционных систем и популярных пакетов программ встроены различные подсистемы защиты от НСД. Например, выполнение аутентификации пользователей при входе в операционные системы семейства Windows. Однако не вызывает сомнений тот факт, что для серьезной защиты от НСД встроенных средств операционных систем недостаточно. К сожалению, реализация подсистем защиты большинства операционных систем достаточно часто вызывает нарекания из-за регулярно обнаруживаемых уязвимостей, позволяющих получить доступ к защищаемым объектам в обход правил разграничения доступа. Выпускаемые же производителями программного обеспечения пакеты обновлений и исправлений объективно несколько отстают от информации об обнаруживаемых

уязвимостях. Поэтому в дополнение к стандартным средствам защиты необходимо использование специальных средств ограничения или разграничения доступа. Данные средства можно разделить на две категории:

1. Средства ограничения физического доступа.
2. Средства защиты от несанкционированного доступа по сети.

Наиболее надежное решение проблемы ограничения физического доступа к компьютеру – использование аппаратных средств защиты информации от НСД, выполняющихся до загрузки операционной системы. Средства защиты данной категории называются «электронными замками».

4.2.2. Средства защиты от несанкционированного доступа по сети

Наиболее действенными методами защиты от несанкционированного доступа по компьютерным сетям являются виртуальные частные сети (VPN – Virtual Private Network) и межсетевое экранирование. Рассмотрим их подробно.

Виртуальные частные сети

Виртуальные частные сети обеспечивают автоматическую защиту целостности и конфиденциальности сообщений, передаваемых через различные сети общего пользования, прежде всего, через Интернет. Фактически, VPN – это совокупность сетей, на внешнем периметре которых установлены VPN-агенты (рис. 5.4). VPN-агент – это программа (или программно-аппаратный комплекс), собственно обеспечивающая защиту передаваемой информации путем выполнения описанных ниже операций.

Перед отправкой в сеть любого IP-пакета VPN-агент производит следующее:

Из заголовка IP-пакета выделяется информация о его адресате. Согласно этой информации на основе политики безопасности данного VPN-агента выбираются алгоритмы защиты (если VPN-агент поддерживает несколько алгоритмов) и криптографические ключи, с помощью которых будет защищен данный пакет. В том случае, если политикой безопасности VPN-агента не предусмотрена отправка IP-пакета данному адресату или IP-пакета с данными характеристиками, отправка IP-пакета блокируется.

С помощью выбранного алгоритма защиты целостности формируется и добавляется в IP-пакет электронная цифровая подпись (ЭЦП), имитоприставка или аналогичная контрольная сумма.

С помощью выбранного алгоритма шифрования производится зашифрование IP-пакета.

С помощью установленного алгоритма инкапсуляции пакетов зашифрованный IP-пакет помещается в готовый для передачи IP-пакет, заголовок которого вместо исходной информации об адресате и отправителе содержит соответственно информацию о VPN-агенте адресата и VPN-агенте отправителя, т. е. выполняется трансляция сетевых адресов.

Пакет отправляется VPN-агенту адресата. При необходимости производится его разбиение и поочередная отправка результирующих пакетов.

При приеме IP-пакета VPN-агент производит следующее:

Из заголовка IP-пакета выделяется информация о его отправителе. В том случае, если отправитель не входит в число разрешенных (согласно политике безопасности) или неизвестен (например, при приеме пакета с намеренно или случайно поврежденным заголовком), пакет не обрабатывается и отбрасывается.

Согласно политике безопасности выбираются алгоритмы защиты данного пакета и ключи, с помощью которых будет выполнено расшифрование пакета и проверка его целостности.

Выделяется информационная (инкапсулированная) часть пакета и производится ее расшифрование.

Производится контроль целостности пакета на основе выбранного алгоритма. В случае обнаружения нарушения целостности пакет отбрасывается.

Пакет отправляется адресату (по внутренней сети) согласно информации, находящейся в его оригинальном заголовке.

VPN-агент может находиться непосредственно на защищаемом компьютере (например, компьютеры «удаленных пользователей»). В этом случае с его помощью защищается информационный обмен только того компьютера, на котором он установлен, однако описанные выше принципы его действия остаются неизменными.

Основное правило построения VPN – связь между защищенной ЛВС и открытой сетью должна осуществляться только через VPN-агенты. Категорически не должно быть каких-либо способов

связи, минующих защитный барьер в виде VPN-агента, т. е. должен быть определен защищаемый периметр, связь с которым может осуществляться только через соответствующее средство защиты. Политика безопасности является набором правил, согласно которым устанавливаются защищенные каналы связи между абонентами VPN. Такие каналы обычно называют туннелями, аналогия с которыми просматривается в следующем:

Вся передаваемая в рамках одного туннеля информация защищена как от несанкционированного просмотра, так и от модификации.

Инкапсуляция IP-пакетов позволяет добиться сокрытия топологии внутренней ЛВС: из Интернет обмен информации между двумя защищенными ЛВС виден как обмен информацией только между их VPN-агентами, поскольку все внутренние IP-адреса в передаваемых через Интернет IP-пакетах в этом случае не фигурируют.

Правила создания туннелей формируются в зависимости от различных характеристик IP-пакетов, например, основной при построении большинства VPN протокол IPSec (Security Architecture for IP) устанавливает следующий набор входных данных, по которым выбираются параметры туннелирования и принимается решение при фильтрации конкретного IP-пакета:

1. IP-адрес источника. Это может быть не только одиночный IP-адрес, но и адрес подсети или диапазон адресов.
2. IP-адрес назначения. Также может быть диапазон адресов, указываемый явно, с помощью маски подсети или шаблона.
3. Идентификатор пользователя (отправителя или получателя).
4. Протокол транспортного уровня (TCP/UDP).
5. Номер порта, с которого или на который отправлен пакет.

Межсетевой экран представляет собой программное или программно-аппаратное средство, обеспечивающее защиту локальных сетей и отдельных компьютеров от несанкционированного доступа со стороны внешних сетей путем фильтрации двустороннего потока сообщений при обмене информацией. Фактически межсетевой экран является «урезанным» VPN-агентом, не выполняющим шифрование пакетов и контроль их целостности, но в ряде случаев имеющим ряд дополнительных функций, наиболее часто из которых встречаются следующие:

– антивирусное сканирование;
– контроль корректности пакетов;

– контроль корректности соединений (например, установления, использования и разрыва TCP-сессий);

– контент-контроль.

Межсетевые экраны, не обладающие описанными выше функциями и выполняющими только фильтрацию пакетов, называют пакетными фильтрами. По аналогии с VPN-агентами существуют и персональные межсетевые экраны, защищающие только компьютер, на котором они установлены. Межсетевые экраны также располагаются на периметре защищаемых сетей и фильтруют сетевой трафик согласно настроенной политике безопасности.

4.3. Классы безопасности компьютерных систем

В критериях впервые введены четыре уровня доверия – D, C, B и A, которые подразделяются на классы. Классов безопасности всего шесть – C1, C2, B1, B2, B3, A1 (перечислены в порядке ужесточения требований).

Уровень D. Данный уровень предназначен для систем, признанных неудовлетворительными.

Уровень C. Иначе – произвольное управление доступом.

Класс C1

Политика безопасности и уровень гарантированности для данного класса должны удовлетворять следующим важнейшим требованиям:

Доверенная вычислительная база должна управлять доступом именованных пользователей к именованным объектам.

Пользователи должны идентифицировать себя, причем аутентификационная информация должна быть защищена от несанкционированного доступа.

Доверенная вычислительная база должна поддерживать область для собственного выполнения, защищенную от внешних воздействий.

Должны быть в наличии аппаратные или программные средства, позволяющие периодически проверять корректность функционирования аппаратных и микропрограммных компонентов доверенной вычислительной базы.

Защитные механизмы должны быть протестированы (нет способов обойти или разрушить средства защиты доверенной вычислительной базы).

Должны быть описаны подход к безопасности и его применение при реализации доверенной вычислительной базы.

Класс С2

В дополнение к С1:

Права доступа должны гранулироваться с точностью до пользователя. Все объекты должны подвергаться контролю доступа.

При выделении хранимого объекта из пула ресурсов доверенной вычислительной базы необходимо ликвидировать все следы его использования.

Каждый пользователь системы должен уникальным образом идентифицироваться. Каждое регистрируемое действие должно ассоциироваться с конкретным пользователем.

Доверенная вычислительная база должна создавать, поддерживать и защищать журнал регистрационной информации, относящейся к доступу к объектам, контролируемым базой.

Тестирование должно подтвердить отсутствие очевидных недостатков в механизмах изоляции ресурсов и защиты регистрационной информации.

Уровень В также именуется принудительное управление доступом.

Класс В1 в дополнение к С2:

Доверенная вычислительная база должна управлять метками безопасности, ассоциируемыми с каждым субъектом и хранимым объектом.

Доверенная вычислительная база должна обеспечить реализацию принудительного управления доступом всех субъектов ко всем хранимым объектам.

Доверенная вычислительная база должна обеспечивать взаимную изоляцию процессов путем разделения их адресных пространств.

Группа специалистов, полностью понимающих реализацию доверенной вычислительной базы, должна подвергнуть описание архитектуры, исходные и объектные коды тщательному анализу и тестированию.

Должна существовать неформальная или формальная модель политики безопасности, поддерживаемой доверенной вычислительной базой.

Класс B2 в дополнение к B1:

Снабжаться метками должны все ресурсы системы (например, ПЗУ), прямо или косвенно доступные субъектам.

К доверенной вычислительной базе должен поддерживаться доверенный коммуникационный путь для пользователя, выполняющего операции начальной идентификации и аутентификации.

Должна быть предусмотрена возможность регистрации событий, связанных с организацией тайных каналов обмена с памятью.

Доверенная вычислительная база должна быть внутренне структурирована на хорошо определенные, относительно независимые модули.

Системный архитектор должен тщательно проанализировать возможности организации тайных каналов обмена с памятью и оценить максимальную пропускную способность каждого выявленного канала.

Должна быть продемонстрирована относительная устойчивость доверенной вычислительной базы к попыткам проникновения.

Модель политики безопасности должна быть формальной. Для доверенной вычислительной базы должны существовать описательные спецификации верхнего уровня, точно и полно определяющие ее интерфейс.

В процессе разработки и сопровождения доверенной вычислительной базы должна использоваться система конфигурационного управления, обеспечивающая контроль изменений в описательных спецификациях верхнего уровня, иных архитектурных данных, реализационной документации, исходных текстах, работающей версии объектного кода, тестовых данных и документации.

Тесты должны подтверждать действенность мер по уменьшению пропускной способности тайных каналов передачи информации.

Класс B3 в дополнение к B2:

Для произвольного управления доступом должны обязательно использоваться списки управления доступом с указанием разрешенных режимов.

Должна быть предусмотрена возможность регистрации появления или накопления событий, несущих угрозу политике безопасности системы. Администратор безопасности должен немедленно

извещаться о попытках нарушения политики безопасности, а система, в случае продолжения попыток, должна пресекать их наименее болезненным способом.

Доверенная вычислительная база должна быть спроектирована и структурирована таким образом, чтобы использовать полный и концептуально простой защитный механизм с точно определенной семантикой.

Процедура анализа должна быть выполнена для временных тайных каналов.

Должна быть специфицирована роль администратора безопасности. Получить права администратора безопасности можно только после выполнения явных, протоколируемых действий.

Должны существовать процедуры и/или механизмы, позволяющие произвести восстановление после сбоя или иного нарушения работы без ослабления защиты.

Должна быть продемонстрирована устойчивость доверенной вычислительной базы к попыткам проникновения.

Уровень А – верифицируемая безопасность.

Класс А1 в дополнение к В3:

Тестирование должно продемонстрировать, что реализация доверенной вычислительной базы соответствует формальным спецификациям верхнего уровня.

Помимо описательных, должны быть представлены формальные спецификации верхнего уровня. Необходимо использовать современные методы формальной спецификации и верификации систем.

Механизм конфигурационного управления должен распространяться на весь жизненный цикл и все компоненты системы, имеющие отношение к обеспечению безопасности.

Должно быть описано соответствие между формальными спецификациями верхнего уровня и исходными текстами.

Такова классификация, введенная в «Оранжевой книге». Коротко ее можно сформулировать так:

уровень С – произвольное управление доступом;

уровень В – принудительное управление доступом;

уровень А – верифицируемая безопасность.

Публикация «Оранжевой книги» стала эпохальным событием в области информационной безопасности. Появился общепризнанный понятийный базис, без которого даже обсуждение проблем ИБ было бы затруднительным.

Отметим, что огромный идейный потенциал «Оранжевой книги» пока во многом остается невостребованным. Прежде всего, это касается концепции технологической гарантированности, охватывающей весь жизненный цикл системы – от выработки спецификаций до фазы эксплуатации. При современной технологии программирования результирующая система не содержит информации, присутствующей в исходных спецификациях, теряется информация о семантике программ.

Для реализации функций безопасности могут использоваться следующие механизмы и их комбинации.

Криптография – наука о математических методах обеспечения конфиденциальности информации, подлинности авторства и невозможности отказа от него. Она изучает методы шифрования информации – обратимого преобразования исходного текста на основе секретного алгоритма и/или ключа в шифрованный текст.

В криптографии используются симметричные и асимметричные методы шифрования, системы электронной цифровой подписи (ЭЦП), хеш-функции, управление ключами, получение скрытой информации, квантовая криптография. Ключ представляет собой параметр шифра, определяющий выбор конкретного преобразования данного текста. В современных шифрах алгоритм шифрования известен, и криптографическая стойкость шифра определяется секретностью ключа.

В зависимости от структуры используемых ключей методы шифрования делятся на:

– симметричное шифрование – симметричные алгоритмы шифрования используют один и тот же ключ для шифрования и расшифровки сообщений;

– асимметричное шифрование основано на использовании пары криптографических ключей, один из которых является частным и известен только конечным пользователям, другой (публичный) может быть доступен всем;

– электронная цифровая подпись (ЭЦП) – реквизит электронного документа, предназначенный для защиты его от подделки, полученный в результате криптографического преобразования информации с использованием закрытого ключа электронной цифровой подписи и позволяющий идентифицировать владельца сертификата ключа подписи, установить отсутствие искажения информации

в электронном документе, обеспечить авторство подписавшегося. ЭЦП используется физическими и юридическими лицами в качестве аналога собственноручной подписи для придания электронному документу юридической силы, равной юридической силе документа на бумажном носителе, подписанного собственноручной подписью правомочного лица и скрепленного печатью. ЭЦП – это программно-криптографическое средство, которое обеспечивает: проверку целостности документов; конфиденциальность документов; установление лица, отправившего документ. Использование ЭЦП позволяет сократить время, затрачиваемое на оформление сделки и обмен документацией; усовершенствовать и удешевить процедуру подготовки, доставки, учета и хранения документов; гарантировать достоверность документации; повысить конфиденциальность информационного обмена и минимизировать риск финансовых потерь; организовать электронный документооборот. При использовании ЭЦП документ обрабатывается по сложному математическому алгоритму, который формирует одно большое число – хеш-код, которое связано с исходными данными таким образом, что при внесении в них изменений хеш-код окажется некорректным и сообщение нельзя будет расшифровать. Хеш-код подписывается с помощью частного ключа пользователя. Данные, закодированные с помощью частного ключа автора сообщения, могут быть расшифрованы только с помощью его же открытого ключа. Любой человек может проверить подлинность документа, расшифровав хеш-код с помощью открытого ключа автора и сравнив его с другим хеш-кодом, сгенерированным из полученных данных.

Подделать ЭЦП невозможно – это требует огромного количества вычислений, которые не могут быть реализованы при современном уровне математики и вычислительной техники за приемлемое время, т. е. пока информация, содержащаяся в подписанном документе, сохраняет актуальность. Дополнительная защита от подделки обеспечивается сертификацией Удостоверяющим центром открытого ключа подписи:

– механизмы управления доступом. Могут располагаться на любой из участвующих в общении сторон или в промежуточной точке;

– механизмы контроля целостности данных. Различаются два аспекта целостности: целостность отдельного сообщения или поля

информации и целостность потока сообщений или полей информации. Для проверки целостности потока сообщений (то есть для защиты от кражи, переупорядочивания, дублирования и вставки сообщений) используются порядковые номера, временные штампы, криптографическое связывание или иные аналогичные приемы:

– механизмы аутентификации. Аутентификация может достигаться за счет использования паролей, личных карточек или иных устройств аналогичного назначения, криптографических методов, устройств измерения и анализа биометрических характеристик;

– механизмы дополнения трафика;

– механизмы управления маршрутизацией. Маршруты могут выбираться статически или динамически. Оконечная система, зафиксировав неоднократные атаки на определенном маршруте, может отказаться от его использования. На выбор маршрута способна повлиять метка безопасности, ассоциированная с передаваемыми данными;

– механизмы нотаризации. Служат для заверения таких коммуникационных характеристик, как целостность, время, личности отправителя и получателей. Заверение обеспечивается надежной третьей стороной, обладающей достаточной информацией. Обычно нотаризация опирается на механизм электронной подписи.

4.4. Организационно-правовые аспекты защиты информации и авторское право

Мероприятия по защите информации охватывают множество аспектов законодательного, организационного и программно-технического характера. Для каждого из них формулируется ряд задач, выполнение которых необходимо для защиты информации. Перечислим самые общие из них.

В нормативно-законодательном аспекте необходимо:

– определить круг нормативных документов, применение которых требуется при проектировании и реализации системы информационной безопасности;

– установить требования по категорированию информации на основе нормативных документов;

– установить базовые требования к системе информационной безопасности и ее компонентам на основе нормативных документов.

В организационном аспекте требуется:

– установить соответствие защищаемой информации и информации по подсистемам и ресурсам ИС, в которых производится хранение, обработка и передача информации конечному пользователю (должно быть организовано ведение реестра ресурсов, содержащих информацию, значимую по критериям конфиденциальности, целостности и доступности);

– определить набор служб, обеспечивающих доступ к ИР системы (необходима выработка и согласование типовых профилей пользователей, ведение реестра таких профилей);

– сформировать политику безопасности, включающую в себя описание границ и способов контроля безопасного состояния системы, условий и правил доступа различных пользователей к ресурсам системы, мониторинг деятельности пользователей.

В процедурном аспекте следует:

– организовать физическую защиту помещений и компонентов ИС, включая сети и телекоммуникационные устройства;

– обеспечить решение задач информационной безопасности при управлении персоналом;

– сформировать, утвердить и реализовать план реагирования на нарушения режима безопасности;

– внести дополнения, связанные со спецификой ликвидации последствий несанкционированного доступа, в план восстановительных работ.

В программно-техническом аспекте необходимо:

– обеспечить архитектурную и инфраструктурную полноту решений, связанных с хранением, обработкой и передачей конфиденциальной информации;

– гарантировать реализационную непротиворечивость механизмов безопасности по отношению к функционированию ИС в целом;

– выработать и реализовать проектные и программно-аппаратные решения по механизмам безопасности. При формулировании требований к обеспечению информационной безопасности и построению соответствующей функциональной модели следует учитывать следующие важные моменты.

Во-первых, для каждого сервиса основные требования к информационной безопасности (доступность, целостность, конфиденциальность) трактуются по-своему. Целостность с точки зрения

СУБД и с точки зрения почтового сервера – вещи принципиально разные. Бессмысленно говорить о безопасности локальной или иной сети вообще, если сеть включает в себя разнородные компоненты. Необходимо анализировать защищенность конкретных сервисов и устройств, функционирующих в сети. Для разных сервисов и защиту строят по-разному. Во-вторых, основная угроза информационной безопасности организаций, как было отмечено ранее, в большей степени исходит не от внешних злоумышленников, а от собственных сотрудников.

В деле обеспечения информационной безопасности успех может принести только комплексный подход. Для защиты интересов субъектов информационных отношений необходимо сочетать меры следующих уровней:

– законодательного;

– административного (приказы и другие действия руководства организаций, связанных с защищаемыми информационными системами);

– процедурного (меры безопасности, ориентированные на людей);

– программно-технического.

Законодательный уровень является важнейшим для обеспечения информационной безопасности.

На законодательном уровне различаются две группы мер:

– меры, направленные на создание и поддержание в обществе негативного (в том числе с применением наказаний) отношения к нарушениям и нарушителям информационной безопасности (назовем их мерами ограничительной направленности);

– направляющие и координирующие меры, способствующие повышению образованности общества в области информационной безопасности, помогающие в разработке и распространении средств обеспечения информационной безопасности (меры созидательной направленности).

Самое важное на законодательном уровне – создать механизм, позволяющий согласовать процесс разработки законов с реалиями и прогрессом информационных технологий. Законы не могут опережать жизнь, но важно, чтобы отставание не было слишком большим, так как на практике, помимо прочих отрицательных моментов, это ведет к снижению информационной безопасности.

В современном мире глобальных сетей нормативно-правовая база должна быть согласована с международной практикой. Особое внимание следует обратить на то, что желательно привести национальные стандарты и сертификационные нормативы в соответствие с международным уровнем информационных технологий вообще и информационной безопасности в частности. Есть целый ряд оснований для того, чтобы это сделать. Одно из них – необходимость защищенного взаимодействия с зарубежными организациями. Второе – доминирование аппаратно-программных продуктов зарубежного производства.

Подводя итог, можно наметить следующие основные направления деятельности на законодательном уровне:

– разработка новых законов с учетом интересов всех категорий субъектов информационных отношений;

– обеспечение баланса созидательных и ограничительных (в первую очередь преследующих цель наказать виновных) законов;

– интеграция в мировое правовое пространство;

– учет современного состояния информационных технологий.

4.5. Требования к хранению и безопасности предметных данных

В цифровой экономике данные являются ключевым активом, а их сохранность и целостность – критически важным условием функционирования любой организации. Процессы хранения и обеспечения безопасности предметных (пользовательских, операционных, критических) данных неразрывно связаны и регулируются комплексом требований. Эти требования исходят из технических особенностей систем, бизнес-задач, а также из жестких предписаний законодательства. Понимание и реализация этих требований – обязательная компетенция IT-специалиста.

4.5.1. Требования к хранению данных

Требования к хранению определяют, как данные должны сохраняться, организовываться и управляться в течение всего их жизненного цикла.

Доступность (Availability)

Суть: обеспечение бесперебойного доступа к данным для авторизованных пользователей и систем в любое время.

Методы обеспечения:

Резервирование (Redundancy): размещение данных на нескольких дисках (RAID-массивы), в нескольких серверах (кластеризация) и даже в географически распределенных дата-центрах (репликация).

Отказоустойчивость (Fault Tolerance): способность системы продолжать работу при сбое одного или нескольких компонентов.

Плановое техническое обслуживание без остановки сервиса (hot-swap компонентов).

Целостность (Integrity)

Суть: гарантия точности, непротиворечивости и неизменности данных на протяжении всего жизненного цикла. Данные не должны быть искажены или случайно изменены.

Методы обеспечения:

Механизмы транзакций (ACID). В реляционных СУБД гарантируют, что операции либо выполняются целиком, либо не выполняются вовсе.

Контрольные суммы (Checksums) и хеш-функции. Для обнаружения повреждений данных при передаче или хранении.

Система версионности (Versioning). Позволяет отслеживать изменения и откатываться к предыдущим состояниям данных.

Конфиденциальность (Confidentiality)

Суть: обеспечение того, что доступ к данным имеют только авторизованные субъекты (пользователи, системы, процессы).

Методы обеспечения. (Более детально рассмотрены в разделе безопасности).

Шифрование данных на rest (при хранении) *и на transit* (при передаче).

Масштабируемость (Scalability)

*Суть: в*озможность инфраструктуры хранения увеличивать объемы и производительность по мере роста количества данных.

Подходы:

Вертикальное масштабирование (Scale-Up). Добавление ресурсов (памяти, дисков) к существующему серверу. Имеет физические ограничения.

Горизонтальное масштабирование (Scale-Out). Добавление новых серверов в кластер (например, как в Hadoop, NoSQL БД). Более гибкий и современный подход.

Управление жизненным циклом данных (Data Lifecycle Management – DLM)

Суть: политика управления данными от их создания до окончательного удаления.

Этапы:

1. Создание и прием.

2. Хранение и активное использование.

3. *Архивация*. Перемещение редко используемых, но нужных для соблюдения законодательства данных на более медленные и дешевые носители.

4. *Удаление (Disposal)*. Безвозвратное и безопасное уничтожение данных по истечении сроков хранения.

4.5.2. Требования к безопасности данных

Требования к безопасности определяют меры защиты данных от несанкционированного доступа, использования, раскрытия, нарушения, изменения или уничтожения.

Регламентирующие документы (Compliance)

1. Безопасность данных heavily driven by compliance. Требования диктуются:

2. Федеральный закон № 152-ФЗ «О персональных данных»: Обязателен для всех операторов ПДн в РФ. Требует локализации баз данных на территории РФ, получения согласий на обработку, реализации мер защиты (СЗПДн).

3. GDPR (General Data Protection Regulation): Регламент ЕС, имеющий экстерриториальное действие. Устанавливает жесткие правила по согласию, правам субъектов данных (право на забвение) и уведомлению об утечках.

4. Отраслевые стандарты: PCI DSS (для платежных данных), ГОСТ Р ИСО/МЭК 27001-2022 (информационная безопасность), ФСТЭК, ФСБ.

Технические меры защиты

Шифрование:

At rest. Шифрование данных на дисках (например, BitLocker, LUKS) и в базах данных.

In transit. Шифрование данных при передаче по сети (TLS/SSL протоколы).

Управление доступом:

Аутентификация. Проверка личности пользователя (логин/пароль, 2FA, сертификаты).

Авторизация. Определение прав доступа аутентифицированного пользователя (RBAC – Role-Based Access Control, ACL – Access Control Lists).

Мониторинг и аудит. Непрерывное протоколирование всех событий доступа и изменения данных для выявления аномалий и расследования инцидентов (SIEM-системы).

Защита от вредоносного ПО. Антивирусы, системы предотвращения вторжений (IPS).

Организационные меры защиты

1. Разработка и внедрение политик информационной безопасности.

2. Регулярное обучение и повышение осведомленности сотрудников.

3. Проведение проверок и аудитов безопасности.

4. План реагирования на инциденты (Incident Response Plan).

Требования к хранению и безопасности предметных данных формируют жёсткий каркас, внутри которого функционирует любая современная информационная система. Их игнорирование ведет к прямым финансовым потерям, репутационным рискам и административной ответственности. Глубокое понимание не только технических, но и юридических аспектов этих требований позволяет проектировать не просто эффективные, но и надежные и соответствующие законодательству IT-решения, что является неотъемлемой частью профессиональной подготовки к кандидатскому минимуму по ИТ.

5. МАТЕМАТИЧЕСКОЕ МОДЕЛИРОВАНИЕ И ЧИСЛЕННЫЕ МЕТОДЫ

5.1. Модели систем и их предназначение

5.1.1. Понятие модели и цели моделирования

В самом общем виде модель – это упрощенное представление реальной системы, объекта или процесса, созданное для конкретной цели и отражающее существенные, с точки зрения этой цели, черты оригинала.

Процесс построения и использования моделей называется моделированием. Он является основным инструментом познания и управления в науке и инженерии. Цели моделирования чрезвычайно разнообразны и определяют то, какие именно aspects системы будут включены в модель.

Объяснение и понимание. Модель помогает выявить причинно-следственные связи между элементами системы, понять ее структуру и принципы функционирования.

Анализ и прогнозирование. На основе модели можно предсказать поведение системы в будущем при различных входных воздействиях или изменениях параметров, не подвергая риску реальный объект (например, прогноз экономических показателей или последствий изменения климата).

Оптимизация. Модель позволяет найти такие значения управляемых параметров системы, при которых достигается наилучший (оптимальный) результат согласно выбранному критерию (максимизация прибыли, минимизация затрат или времени).

Проектирование и синтез. Модели используются для создания новых систем, проверки их работоспособности и соответствия техническому заданию до этапа физического воплощения (компьютерное проектирование самолетов, чипов, зданий).

Управление. Модели, особенно в реальном времени, являются основой систем автоматического управления сложными объектами (например, модель полета для автопилота).

Обучение. Модели и основанные на них тренажеры (симуляторы) позволяют безопасно и эффективно готовить специалистов для работы со сложными системами (тренажеры пилотов, операторов АЭС).

Ключевой принцип моделирования – компромисс между адекватностью и сложностью. Хорошая модель не стремится быть идеальной копией системы, а должна быть настолько простой, насколько это возможно, но при этом достаточной для достижения поставленной цели.

5.1.2. Классификация моделей систем

Существует множество признаков для классификации моделей. Выбор типа модели напрямую зависит от целей исследования и природы самой системы.

По способу представления (форме):

1. *Материальные (натурные, физические) модели*. Воспроизводят геометрические, физические или функциональные свойства объекта. Это могут быть макеты зданий, модели самолетов в аэродинамической трубе, фантомы в медицине. Их главное преимущество – высокая наглядность, недостаток – высокая стоимость создания и изменения, сложность проведения широкого спектра экспериментов.

2. *Абстрактные (идеальные, знаковые) модели*. Описывают систему с помощью символов, формул, графиков, алгоритмов. Являются основным инструментом теоретических исследований. Делятся на:

   a. *Вербальные (словесные) модели*. Описание системы на естественном языке. Часто являются первоначальной, качественной формой модели.

   b. *Математические модели*. Важнейший класс моделей. Представляют систему в виде совокупности уравнений, неравенств, логических условий и других математических соотношений. Именно этот тип моделей обеспечивает возможность глубокого анализа, точного прогноза и оптимизации.

По учету временного фактора (поведению):

1. *Статические модели*. Описывают состояние системы в определенный момент времени, «снимок» системы. Не учитывают изменение во времени. Пример: уравнение равновесия цен на рынке, модель структуры базы данных.

2. *Динамические модели*. Описывают изменение состояния системы во времени, ее поведение. Именно эти модели используются для прогнозирования и управления. Пример: дифференциальные уравнения движения механической системы, модель роста популяции.

По характеру отражения причинно-следственных связей:

1. *Детерминированные модели*. Однозначно определяют реакцию системы на любое входное воздействие. В них не учитываются случайные факторы. Все связи считаются точно известными.

2. *Стохастические (вероятностные) модели*. Учитывают случайный характер, как входных воздействий, так и внутренних параметров системы. Результаты моделирования носят вероятностный характер. Пример: модели массового обслуживания (очереди), финансовые модели для оценки рисков.

По типу математического аппарата:

1. *Аналитические модели*. Описываются уравнениями, решения которых могут быть получены в виде формул (аналитически). Дают наиболее полное понимание взаимосвязей в системе, но применимы лишь к относительно простым системам.

2. *Имитационные модели.* Представляют собой алгоритм, который воспроизводит процесс функционирования системы во времени с сохранением логической структуры и последовательности событий. Позволяют исследовать чрезвычайно сложные системы, недоступные для аналитического описания (например, работа цеха, транспортный поток, экономика региона). Результатом работы имитационной модели является набор статистических данных о поведении системы, требующий последующего анализа.

По степени агрегирования:

1. *Макромодели*. Система рассматривается как единое целое, выделяются лишь ее глобальные входы и выходы.

2. *Микромодели*. Детально описывается внутренняя структура системы, поведение каждого ее элемента и взаимодействия между ними.

Таким образом, моделирование является универсальным и мощным методом научного исследования. Выбор адекватного типа модели является ключевым этапом любого системного анализа. Правильно построенная модель служит мостом между теоретическим знанием и практической деятельностью, позволяя не только объяснять наблюдаемые явления, но и целенаправленно воздействовать на систему, проектировать новые и оптимизировать существующие сложные объекты, прогнозируя последствия принимаемых решений. Владение методологией моделирования является неотъемлемой компетенцией современного исследователя.

5.2. Аналитическое и имитационное моделирование

5.2.1. Сущность подходов и ключевые различия

В основе математического моделирования лежат два фундаментально различных, но часто взаимодополняющих подхода: аналитический и имитационный. Выбор между ними определяется спецификой системы, целями исследования и требуемой глубиной анализа.

Аналитическое моделирование – это подход, при котором функционирование системы описывается замкнутой системой уравнений (алгебраических, дифференциальных, интегральных и т. д.), решение которых позволяет в общем виде получить характеристики системы как функции ее параметров.

Имитационное моделирование – это подход, при котором процесс функционирования системы воспроизводится пошагово во времени с сохранением логической структуры и последовательности выполняемых операций. Модель является алгоритмом, а эксперимент – его выполнением.

Ключевое различие заключается в **способе получения результата:**

1. Аналитическая модель дает решение в форме формулы.

2. Имитационная модель генерирует массив данных (статистику), который является решением.

Сравнительная характеристика подходов представлена в табл. 5.1.

Таблица 5.1

Сравнение аналитического и имитационного моделирования

Критерий	Аналитическое моделирование	Имитационное моделирование
Сущность подхода	Получение явной функциональной зависимости выходов системы от ее входов и параметров	Пошаговое воспроизведение работы системы во времени
Уровень абстракции	Высокий. Требует сильного упрощения и агрегирования	Низкий. Позволяет учесть множество деталей и сложных взаимосвязей

Окончание таблицы 5.1

Критерий	Аналитическое моделирование	Имитационное моделирование
Учет случайности	Сильно затруднен. Часто требует приближений	Естественен. Легко интегрируются вероятностные распределения
Тип результата	Точное или приближенное решение в общем виде	Статистические оценки искомых характеристик
Глубина анализа	Позволяет провести полный параметрический анализ и найти оптимум	Результаты верны только для конкретного сценария
Требуемые ресурсы	Интеллектуальные (высокая квалификация исследователя)	Вычислительные (мощность CPU/GPU, объем памяти, время)
Область применения	Простые, хорошо формализуемые системы	Сложные, неформализуемые системы со стохастическими элементами

Процесс и особенности

Процесс аналитического моделирования включает:

1. Формализацию задачи и выделение существенных переменных.
2. Составление системы уравнений на основе законов (физических, экономических и пр.).
3. Решение системы уравнений аналитическими или численными методами.
4. Анализ полученного решения: нахождение экстремумов, исследование устойчивости, чувствительности.

Преимущества:

1. *Общность*. Решение в виде формулы показывает, как каждый параметр влияет на результат.
2. *Глубина анализа*. Позволяет доказать оптимальность решения.
3. *Скорость*. Расчет по готовой формуле происходит мгновенно.

Недостатки:

1. *Нереалистичные допущения*. Требуется предположить, например, пуассоновский поток заявок, детерминированное время обслуживания и т. д., что часто далеко от реальности.

2. *Сложность*. Для многих реальных систем составление и решение уравнений невозможно или чрезмерно сложно.

Пример: Классическая модель массового обслуживания M/M/1 (с пуассоновским входным потоком, экспоненциальным временем обслуживания и одним каналом) имеет аналитические формулы для средней длины очереди и времени ожидания.

5.2.2. Имитационное моделирование

Процесс и особенности

Процесс имитационного моделирования (ИМ) представляет собой цикл:

1. *Разработка концептуальной модели*: Формальное описание системы, ее элементов, связей и правил взаимодействия.

2. *Разработка программной модели*: Реализация концептуальной модели на языке программирования (C++, Python) или в специализированной среде (AnyLogic, Arena, GPSS).

3. *Верификация и валидация*:

– верификация («строим модель правильно?»): проверка, соответствует ли программная реализация задуманной концептуальной модели;

– валидация («строим правильную модель?»): проверка, адекватно ли модель отражает реальную систему (например, по историческим данным);

– планирование экспериментов: определение множества сценариев для прогона модели;

– проведение экспериментов и анализ выходных данных. Многократные «прогоны» модели для сбора репрезентативной статистики и ее последующая обработка.

Преимущества:

1. Наглядность: позволяет «увидеть» процесс в действии (анимация, пошаговое выполнение).

2. Гибкость и детализация: можно учесть практически любую сложность и стохастичность системы без упрощающих допущений.

3. Безопасность и экономичность: позволяет тестировать критические и дорогостоящие сценарии без рисков для реальной системы.

Недостатки:

1. Вычислительная сложность: исследование может требовать значительного машинного времени.

2. Статистический характер результатов: требует применения методов статистического анализа, оценки доверительных интервалов.

3. Локальность результатов: результаты справедливы только для проверенных сценариев; обобщение требует дополнительных усилий.

Пример: Имитационная модель логистического склада. Модель учитывает случайный спрос, время доставки, количество погрузчиков, график работы сотрудников, поломки оборудования и позволяет найти конфигурацию, минимизирующую издержки.

5.2.3. Синтез подходов

На практике аналитический и имитационный подходы не исключают, а дополняют друг друга:

1. *Быстрый анализ и проверка гипотез*. Аналитическая модель может быть использована для предварительного, грубого анализа системы и выбора перспективных областей для более детального изучения с помощью имитационной модели.

2. *Верификация имитационной модели*. Результаты имитационного моделирования для упрощенной конфигурации системы можно сверить с данными аналитической модели (если она существует) для повышения доверия к имитатору.

3. *Оптимизация*. Аналитическая модель может задать направление поиска оптимума, в окрестности которого затем проводятся детальные имитационные эксперименты.

Выбор между аналитическим и имитационным моделированием является ключевым этапом проектирования исследования. Аналитическое моделирование – это мощный инструмент для глубокого понимания фундаментальных закономерностей в относительно простых системах. Имитационное моделирование – это практически единственный инструмент для анализа и проектирования сложных, стохастических реальных систем, где аналитические методы бессильны. Современный исследователь должен владеть обоими подходами для эффективного решения широкого круга задач.

5.3 Основные этапы математического моделирования

Математическое моделирование – это не единовременный акт, а циклический процесс, состоящий из последовательности взаимосвязанных этапов. Строгое следование этой методологии позволяет создать адекватную, полезную и эффективную модель. Процесс является итеративным: результаты каждого этапа могут потребовать возврата и корректировки предыдущих.

5.3.1. Последовательность этапов

Постановка задачи и содержательный анализ

Цель: четко определить проблему, которую требуется решить с помощью модели.

Содержание: формулировка целей моделирования, определение границ системы (что включать в модель, а что – нет), выявление существенных переменных (входных, выходных, управляемых, возмущающих), установление требований к модели (точность, быстродействие, форма представления результатов).

Результат: техническое задание на моделирование, содержащее исчерпывающее словесное описание решаемой проблемы.

Формализация и построение концептуальной модели

Цель: перейти от словесного описания к формальному представлению.

Содержание: выбор типа модели (аналитическая, имитационная, статическая, динамическая и т. д.). Описание структуры системы, ее элементов и связей между ними с помощью формальных языков (схем, графиков, диаграмм). Формулировка основных допущений и упрощений.

Результат: концептуальная модель – абстрактное, но структурно полное описание системы без привязки к математическому аппарату.

Выбор математического аппарата и параметризация

Цель: найти математические средства, адекватные концептуальной модели.

Содержание: выбор конкретного математического инструментария: дифференциальные уравнения, теория массового обслуживания, теория игр, регрессионный анализ, методы сетевого планирования и др. Определение и сбор данных для оценки всех параметров модели (например, интенсивности потока заявок, коэффициентов уравнений, функций распределения).

Результат: математическая модель – система уравнений, отношений, алгоритмов, готовая для реализации.

Разработка компьютерной модели (Компьютерная реализация)

Цель: перевести математическую модель на язык, понятный компьютеру.

Содержание: выбор средства реализации (универсальный язык программирования – Python, C++; специализированное ПО – AnyLogic, MATLAB, Simulink; табличные процессоры – Excel). Написание, отладка и тестирование кода.

Результат: работоспособная программная реализация модели.

Верификация модели

Цель: убедиться, что модель реализована правильно. Ответ на вопрос: «Правильно ли мы запрограммировали задуманную модель?»

Содержание: проверка логической целостности кода, отсутствия ошибок программирования, синтаксиса. Проводится методами тестовых прогонов с заранее известным результатом, пошаговой отладки, анализа кода.

Результат: уверенность в том, что программная реализация адекватно отражает математическую модель.

Валидация модели (Адекватность модели)

Цель: убедиться, что модель реализована в правильную модель. Ответ на вопрос: «Правильно ли наша модель отображает реальную систему?»

Содержание: сравнение поведения модели с поведением реальной системы на одном и том же наборе входных данных. Используются методы сравнения с историческими данными, экспертные оценки, статистические тесты (критерий хи-квадрат, Колмогорова–Смирнова). Это самый сложный и часто игнорируемый этап.

Результат: уверенность в том, что модель является адекватным представлением оригинала и ее выходным данным можно доверять.

Планирование и проведение вычислительных экспериментов

Цель: получить необходимые для принятия решений данные с помощью проверенной и валидированной модели.

Содержание: разработка плана экспериментов: определение варьируемых параметров, последовательности их изменения, количества необходимых прогонов для получения статистически значимых результатов. Непосредственное проведение расчетов на ЭВМ.

Результат: массив исходных данных, отражающий реакцию модели на различные воздействия.

Анализ результатов и интерпретация

Цель: извлечь содержательные выводы из сырых данных, полученных в ходе экспериментов.

Содержание: обработка выходных данных модели: статистический анализ, построение графиков, диаграмм, таблиц. Интерпретация численных результатов в терминах исходной предметной области. Формулировка рекомендаций для принятия решений.

Результат: отчет, содержащий выводы, рекомендации и, возможно, прогноз поведения системы.

Принятие решения и использование модели

Цель: применить результаты моделирования на практике.

Содержание: представление результатов лицам, принимающим решения. Внедрение модели в практическую деятельность (например, использование для оперативного планирования или обучения). Если результаты неудовлетворительны, осуществляется возврат на один из предыдущих этапов для уточнения модели.

Результат: принятое управляющее воздействие или скорректированный проект системы.

5.3.2. Итеративность процесса

Важно понимать, что процесс моделирования носит **итерационный характер**. Результаты, полученные на этапах верификации, валидации или анализа, могут выявить:

1. Ошибки в программировании → возврат к этапу 4.
2. Неадекватность математической модели → возврат к этапу 3.
3. Неполноту или ошибочность концептуальной модели → возврат к этапу 2.
4. Некорректную постановку задачи → возврат к этапу 1.
5. Таким образом, модель создается и уточняется в ходе последовательных приближений, каждое из которых повышает ее точность и полезность.

5.4. Прямые и обратные задачи математического моделирования

В основе любой деятельности, связанной с математическим моделированием, лежит решение двух принципиально разных типов задач: прямых и обратных. Их различие фундаментально и определяется направлением логического вывода и целью исследования.

5.4.1. Прямые задачи

Определение. Прямая задача – это задача предсказания *следствий* (выходных характеристик, результатов, поведения системы) при известных *причинах* (входных воздействиях, параметрах модели, начальных условиях).

Логическая схема: Причина → Модель → Следствие.

Сущность: имеется полностью определенная математическая модель (уравнения, алгоритмы, параметры которой известны и фиксированы). Требуется, решив уравнения или просимулировав модель, рассчитать отклик системы на заданные входные данные.

Цель: прогноз, анализ поведения системы, проверка гипотез.

Примеры:

1. **Механика.** Известны силы, действующие на тело (причина), и его масса (параметр модели). Требуется по второму закону Ньютона (модель) рассчитать ускорение и траекторию движения тела (следствие).

2. **Финансы.** Известна сумма первоначальных инвестиций (причина), процентная ставка и период (параметры модели сложного процента). Требуется рассчитать будущую стоимость вклада (следствие).

3. **Теплофизика.** Известны распределение источников тепла в теле и температура на его границах (причина/параметры). Требуется, решив уравнение теплопроводности (модель), найти стационарное распределение температуры внутри тела (следствие).

Особенности. Прямые задачи, как правило, являются корректно поставленными (по Адамару): их решение существует, единственно и устойчиво к малым изменениям входных данных.

5.4.2. Обратные задачи

Определение. Обратная задача – это задача определения *причин*, свойств системы или параметров модели по наблюдаемым *следствиям* (измеренным данным, результатам экспериментов).

Логическая схема: Следствие → Модель → Причина (Параметры).

Сущность. Имеется математическая модель, структура которой известна, но некоторые ее параметры, начальные условия, граничные условия или даже сами уравнения неизвестны. Также имеются экспериментальные данные о поведении системы (ее выходе). Требуется подобрать такие параметры модели, чтобы ее предсказания максимально точно соответствовали экспериментальным данным.

Цель: идентификация, калибровка, диагностика, обнаружение, восстановление картины явления.

Примеры:

1. **Медицина (томография).** Измерены ослабления интенсивности рентгеновских лучей после прохождения через организм (следствие). Требуется восстановить внутреннее распределение плотности тканей (причина/параметры). Прямая задача здесь – рассчитать ослабление лучей для *известного* распределения плотности.

2. **Геофизика.** Измерены колебания земной поверхности (следствие). Требуется определить местоположение и магнитуду подземного толчка (причина) или структуру земных недр (параметры модели).

3. **Калибровка модели.** По данным о прошлых продажах компании (следствие) требуется подобрать коэффициенты эконометрической модели (параметры), чтобы она могла давать точные прогнозы.

Особенности. Обратные задачи чрезвычайно сложны. Они почти всегда являются некорректно поставленными (по Адамару). Чаще всего нарушается условие устойчивости: малые погрешности в измеренных данных (следствиях) могут приводить к сколь угодно большим ошибкам в оценке параметров (причин). Это требует применения специальных методов регуляризации для получения осмысленного решения.

Таблица 5.2

Сравнительная характеристика задач

Критерий	Прямая задача	Обратная задача
Направление	От причины к следствию	От следствия к причине
Цель	Прогноз, анализ	Диагностика, идентификация, калибровка
Постановка	Как правило, корректна	Как правило, не корректна
Сложность	Вычислительная (решить уравнение)	Методологическая (преодолеть некорректность)
Данные	Параметры модели и входные воздействия	Выходные (измеренные) данные системы
Результат	Характеристики состояния системы	Параметры модели, входные воздействия

5.4.3. Взаимосвязь и значение

Прямые и обратные задачи неразрывно связаны (табл. 5.2). Решение обратной задачи всегда опирается на многократное решение прямой задачи. Например, в процессе калибровки модели алгоритм снова и снова варьирует параметры (решает обратную задачу), каждый раз запускает модель с новыми параметрами (решает прямую задачу) и сравнивает результат моделирования с реальными данными.

Итеративный процесс калибровки модели – наглядная иллюстрация этой связи:

1. Выдвигается гипотеза о значениях параметров модели.
2. Решается прямая задача – рассчитывается выход модели.
3. Выход модели сравнивается с экспериментальными данными.
4. На основе рассогласования корректируются параметры (решается обратная задача).
5. Шаги 2–4 повторяются до достижения удовлетворительного совпадения.

Таким образом, владение методами решения как прямых, так и обратных задач является cornerstone современного математического моделирования. Прямые задачи – это инструмент прогнозирования, в то время как обратные задачи – это инструмент познания, позволяющий извлекать информацию о внутренней структуре системы из внешних наблюдений.

5.5. Моделирование стационарных и динамических систем посредством численного дифференцирования и интегрирования

Математические модели многих физических, химических, биологических и экономических процессов формулируются в виде дифференциальных уравнений. Поскольку аналитическое решение таких уравнений возможно лишь для узкого класса задач, на практике широко используются методы численного дифференцирования и интегрирования. Эти методы являются вычислительным ядром для моделирования как стационарных (не зависящих от времени), так и, что особенно важно, динамических систем.

5.5.1. Стационарные системы и численное интегрирование

Стационарная система – это система, состояние которой не изменяется во времени. Ее модель часто описывается краевой задачей для обыкновенных дифференциальных уравнений (ОДУ) или уравнениями в частных производных (УрЧП) эллиптического типа.

Роль численного интегрирования: Многие стационарные задачи сводятся к вычислению определенных интегралов от известных или получаемых в ходе решения функций.

Пример 1: Вычисление общей массы. Пусть известно пространственное распределение плотности вещества $\rho(x, y, z)$ в некотором объеме V. Полная масса вычисляется как тройной интеграл: $\rho(x, y, z)dv$. Для сложных областей интегрирования этот интеграл вычисляется численно, например, методом Монте-Карло или путем разбиения области на малые элементы (метод конечных элементов, МКЭ).

Пример 2: Решение уравнения Пуассона. Уравнение $\Delta\varphi = f$, где Δ – оператор Лапласа) описывает стационарное распределение температуры, потенциала электростатического поля и т. д. Методы его решения (как МКЭ, так и метод конечных разностей, МКР) на этапе формирования матрицы жесткости требуют вычисления интегралов от базисных функций или их произведений.

Таким образом, численное интегрирование является не самостоятельной целью, а важнейшим *вычислительным блоком* в алгоритмах моделирования стационарных состояний.

5.5.2. Динамические системы и численное дифференцирование

Динамическая система – это система, состояние которой эволюционирует во времени. Ее поведение описывается задачами Коши для ОДУ или УрЧП параболического и гиперболического типов.

Роль численного дифференцирования: Производная по времени – это скорость изменения функции. Замена производной ее численной аппроксимацией позволяет «шагать» по времени, вычисляя новое состояние системы на основе предыдущего.

Пусть $f(x)$ – функция, для которой нужно найти производную в заданной точке отрезка $[a; b]$, $F_n(x)$ – интерполяционный многочлен для $f(x)$, построенный на отрезке $[a; b]$. Заменяя $f(x)$

интерполяционным многочленом $F_n(x)$, получим значение производной $f(x)$ на отрезке $[a; b]$ как значение производной интерполяционного многочлена, т. е. примем приближенно

$$f'(x) = F_n'(x).$$

Аналогичным путем можно поступать при нахождении значений производных высших порядков функции $f(x)$.

Полагая, что погрешность интерполирования определяется формулой

$$R_n(x) = f(x) - F_n(x).$$

Получаем подход к оценке погрешности производной $F_n'(x)$:

$$r_n(x) = f'(x) - F_n'(x) = R_n'(x).$$

Т. е. погрешность производной интерполирующей функции равна производной от погрешности этой функции.

Рассмотрим методы численного дифференцирования на основе интерполяционных многочленов Лагранжа и Ньютона.

Интерполяционная формула Лагранжа для равноотстоящих узлов

Применяя для численного дифференцирования на отрезке $[a; b]$ интерполяционный полином, естественно строить на этом отрезке систему равноотстоящих узлов

$$a = x_0,\ x_1,\ x_2,\ \ldots,\ x_n = b,$$

которыми отрезок делится на п равных частей:

$$x_{i+1} - x_i = h = const,$$
$$(i = 0, 1, 2, n - 1).$$

В этом случае шаг интерполирования равен $h = \frac{b-a}{n}$, а интерполяционный многочлен Лагранжа строится на равноотстоящих узлах и имеет более удобный вид. Положим

$$\frac{x-x_0}{h}=t\,.$$

С учетом формулы Лагранжа

$$L_n(x)=\sum_{i=0}^{n}y_i\frac{\Pi_{n+1}(x)}{(x-x_i)\cdot\Pi'_{n+1}(x)}=\Pi_{n+1}(x)\sum_{i=0}^{n}\frac{y_i}{(x-x_i)\cdot\Pi'_{n+1}(x)},$$

получим новые выражения для $\Pi_{n+1}(x)$. Учитывая, что

$$\Pi_{n+1}(x)=(x-x_0)(x-x_i)\cdot\ldots\cdot(x-x_n),$$

последовательно находим:

$$x-x_0=ht,$$
$$x-x_1=x-x_0-h=h(t-1),$$
$$x-x_2=x-x_0-2h=h(t-2),$$

т. е. в общем случае:

$$x-x_i=x-x_0-ih=h(t-i),$$
$$I=0,\,1,\,\ldots,\,n.$$
$$\Pi_{n+1}(x)=h^{n+1}t(t-1)(t-2)\cdot\ldots(t-n).$$

Обозначим

$$t(t-1)(t-2)\cdot\ldots\cdot(t-n)=t^{[n+1]},$$

тогда выражение $\Pi_{n+1}(x)$ примет вид:

$$\Pi_{n+1}(x)=h^{n+1}t^{[n+1]}.$$

Учитывая, что при постоянном шаге имеет место

$$x_i=x_0+ih,\ i=0,\,1,\,\ldots,\,n,$$

последовательно находим:

$$x_i - x_0 = hi,$$
$$x_i - x_1 = x_i - x_0 - h = h(i-1),$$
$$x_i - x_n = x_i - x_0 - nh = h(i-n).$$

Заметим, что ровно *n* строк (i-я отсутствует), причем значения разностей из первых i-строк положительны, а остальных – отрицательны.

$$\Pi'_{n+1}(x_i) = (x_i - x_0)\cdot\ldots\cdot(x_i - x_{i-1})\cdot(x_i - x_{i+1})\cdot\times$$
$$\times\ldots\cdot(x_i - x_n) = h^n i(i-1)\cdot\ldots\cdot 1\cdot(-1)\cdot\ldots\cdot(-(n-i)),$$

т. е.

$$\Pi'_{n+1}(x_i) = h^n i!(n-i)!(-1)^{n-i}.$$

С учетом представлений формула Лагранжа для равноотстоящих узлов принимает вид:

$$L_n(x_0 + th) = \sum_{i=0}^{n} y_i \frac{(-1)^{n-i} t^{[n+1]}}{i!(n-i)!(t-i)}.$$

Численное дифференцирование на основе интерполяционной формулы Ньютона

Запишем для функции *f*(*x*), заданной своими значениями в равноотстоящих узлах $x_0, x_1, x_2, \ldots, x_n$ первый интерполяционный многочлен Ньютона:

$$P_n(x) = P_n(x_0 + th) = y_0 + t\Delta y_0 + \frac{t(t-1)}{2!}\Delta^2 y_0 - \frac{t(t-1)(t-2)}{3!}\Delta^3 y_0 + \ldots +$$
$$+\frac{t(t-1)\cdot\ldots\cdot(t-n+1)}{n!}\Delta^n y_0.$$

Перепишем этот полином, производя перемножение скобок:

$$P_n\left(x_0+th\right)=y_0+t\Delta y_0+\frac{t^2-t}{2}\Delta^2 y_0-\frac{t^3-3t^2+2t}{3!}\Delta^3 y_0+$$

$$+\frac{t^4-6t^3+11t^2-6t}{24}\Delta^4 y_0+\ldots$$

Дифференцируя $P_n\left(x_0+th\right)$ по t, получим аналогично формуле

$$f'\left(x\right)\approx P_n^{'}\left(x_0+th\right)=$$

$$=\frac{1}{h}\left(\Delta y_0+\frac{2t-1}{2!}\Delta^2 y_0-\frac{3t^2-6t+2}{6}\Delta^3 y_0+\frac{2t^3-9t^2+11t-3}{12}\Delta^4 y_0+\ldots.\right).$$

Подобным путем можно получить и производные функции $f(x)$ более высоких порядков. Однако каждый раз, вычисляя значение производной функции $f(x)$ в фиксированной точке x, в качестве x_0 следует брать ближайшее слева узловое значение аргумента.

Формула существенно упрощается, если исходным значением x оказывается один из узлов таблицы. Так как в этом случае каждый узел можно считать начальным, то, принимая $x = x_0$, $t = 0$, получаем:

$$f'\left(x_0\right)=\frac{1}{h}\left(\Delta y_0-\frac{\Delta^2 y_0}{2}+\frac{\Delta^3 y_0}{3}-\frac{\Delta^4 y_0}{4}+\frac{\Delta^5 y_0}{5}\ldots.\right).$$

Эта формула позволяет точно получать значения производных функций, заданных таблично.

Выведем формулу погрешности дифференцирования. Используя формулу применительно к первому интерполяционному многочлену Ньютона, запишем:

$$R_n\left(x\right)=h^{n+1}\cdot\frac{t\left(t-1\right)\left(t-2\right)\cdot\ldots\cdot\left(t-n\right)}{\left(n+1\right)!}f^{(n+1)}\left(\xi\right),$$

где ξ – промежуточное значение между $x_0, x_1, x_2, \ldots, x_n$ и заданной точкой *x*. Предполагая, что *f(x)* дифференцируема *n* + 1 раз, получим для оценки погрешности дифференцирования $r_n(x_i)$ (по аналогии с формулой

$$r_n(x) = R_n'(x) = \frac{h^n}{(n+1)!}\left(f^{(n+1)}(\xi) \cdot \frac{d}{dt} t^{(n+1)} + t^{(n+1)} \cdot \frac{d}{dt}\left[f^{(n+1)}(\xi) \right] \right).$$

Для случая оценки погрешности в узле таблицы получим:

$$R_n'(x) = (-1)^n \frac{h^n}{n+1} f^{(n+1)}(\xi).$$

На практике $f^{(n+1)}(\xi)$ оценивать непросто, поэтому при малых *h* приближенно полагают:

$$f^{(n+1)}(\xi) \approx \frac{\Delta^{n+1} y_0}{h^{n+1}},$$

что позволяет использовать приближенную формулу

$$r_n(x_0) \approx \frac{(-1)^n \cdot \Delta^{n+1} y_0}{h(n+1)}.$$

5.5.3. Задачи численного интегрирования

При вычислении определенного интеграла

$$I1 = \int_a^b f(x)\,dx,$$

где *f*(*x*) – непрерывная на отрезке [*a*; *b*] функция, иногда удается воспользоваться формулой Ньютона–Лейбница:

$$\int_a^b f(x)\,dx = F(b) - F(a).$$

Здесь $F(x)$ – одна из первообразных функции $f(x)$. Однако даже в тех редких случаях, когда первообразную удается явно найти в аналитической форме, не всегда удается довести до числового ответа значение определенного интеграла. Если учесть, что подынтегральная функция задается таблицей или графиком, то интегрирование по формуле не получает широкого применения на практике.

В подобных случаях применяют различные методы численного интегрирования. Формулы, используемые для вычисления однократных интегралов, называют квадратурными формулами.

Прием построения квадратурных формул состоит в том, что подынтегральная функция $f(x)$ заменяется на отрезке $[a; b]$ интерполяционным многочленом, например, многочленом Лагранжа $L_n(x)$, и получается приближенное равенство

$$\int_a^b f(x)\,dx \approx \int_a^b L_n(x)\,dx.$$

Предполагается, что отрезок $[a; b]$ разбит на n частей точками $x_0, x_1, x_2, \ldots, x_n$, наличие которых подразумевается при построении многочлена $L_n(x)$.

Подставляя вместо $L_n(x)$, получим

$$\int_a^b f(x)\,dx \approx \int_a^b \sum_{i=0}^{n} y_i \frac{\Pi_{n+1}(x)}{(x-x_i)\cdot \Pi'_{n+1}(x)}\,dx = \sum_{i=0}^{n} y_i \int_a^b \frac{\Pi_{n+1}(x)}{(x-x_i)\cdot \Pi'_{n+1}(x)}\,dx.$$

Таким образом,

$$\int_a^b f(x)\,dx \approx \sum_{i=0}^{n} y_i A_i,$$

где

$$A_i = \int_a^b \frac{\Pi_{n+1}(x)}{(x-x_i)\cdot \Pi'_{n+1}(x)}\,dx.$$

В формуле коэффициенты A_i не зависят от функции $f(x)$, так как они составлены только с учетом узлов интерполяции; если $f(x)$ – полином степени n.

Квадратурные формулы Ньютона–Котеса

Предполагаем построение на отрезке интегрирования [a; b] системы узлов интерполяции $a = x_0, x_1, x_2, ..., x_n = b$, которыми отрезок делится на n частей. Длина $x_{i+1} - x_i = h$, $i = 1, 2, ..., n-1$ называется шагом интегрирования. Естественно считать, что шаг h постоянен, т. е.

$$h = \frac{b-a}{n}.$$

В этом случае можно применить интерполяционную формулу Лагранжа для равноотстоящих узлов.

$$A_i = \int_{x_0}^{x_n} \frac{(-1)^{n-i} t^{[n+1]}}{i!(n-i)!(t-i)} dx, i = 0,1,2,\ldots,n$$

Перейдем в этом интеграле к переменной t. Из подстановки получаем:

$$dt = \frac{dx}{h},$$

т. е.

$$dx = hdt = \frac{b-a}{n} dt.$$

При $x = x_0$ имеем $t = 0$, а при $x = x_n$ будет

$$t = \frac{x_n - x_0}{h} = n.$$

Тогда

$$A_i = \frac{b-a}{n} \int_{0_0}^{n} \frac{(-1)^{n-i} t^{[n+1]}}{i!(n-i)!(t-i)} dt = (b-a) H_i,$$

где

$$H_i = \frac{1}{n}\int_{0_0}^{n} \frac{(-1)^{n-i}\, t^{[n+1]}}{i!(n-i)!(t-i)}\, dt,\ i = 0,1,2,\ldots,n.$$

Эти числа называют коэффициентами Котеса. Они не зависят от функции $f(x)$, а только от числа точек разбиения. Окончательно получаем следующий вид квадратурных формул формулу Ньютона–Котеса:

$$\int_a^b f(x)dx \approx (b-a)\sum_{i=0}^{n} y_i H_i,$$

дающих на одном участке интегрирования различные представления различного числа n отрезков разбиения.

Формула трапеций

При $n = 1$ из формулы имеем $i = 0;1$:

$$H_0 = -\int_0^1 \frac{t(t-1)}{t}\, dt = -\int_0^1 (t-1)\, dt = \frac{1}{2},$$

$$H_1 = \int_0^1 t dt = \frac{1}{2}.$$

Тогда на отрезке $[x_0; x_1]$ получаем интеграл:

$$\int_{x_0}^{x_1} f(x)dx = (x_1 - x_0)(H_0 y_0 + H_1 y_1) = \frac{h}{2}(y_0 + y_1).$$

Формула дает один из простейших способов вычисления определенного интеграла и называется формулой трапеций. Действительно, при $n = 1$ подынтегральная функция заменяется интерполяционным многочленом Лагранжа первой степени (т. е. линейной функцией), а геометрически это означает, что площадь криволинейной фигуры заменяется площадью трапеции.

Распространяя формулу на все отрезки разбиения, получим общую формулу трапеций для отрезка $[a;b]$:

$$\int_a^b f(x)\,dx = h\left(\frac{y_0}{2} + y_1 + y_2 + \ldots + y_{n-1} + \frac{y_n}{2}\right).$$

Если аналитическое выражение для подынтегральной функции известно, может быть поставлен вопрос об оценке погрешности численного интегрирования.

В этом случае имеется в виду, что

$$\int_a^b f(x)\,dx = \int_a^b L_n(x)\,dx + R_n(f),$$

где $R_n(f)$ – остаточный член квадратурной формулы. Формулу остаточного члена получим вначале для отрезка $[x_0; x_1]$. Имеем:

$$R = \int_{x_0}^{x_1} f(x)\,dx - \frac{h}{2}(y_0 + y_1) = \int_{x_0}^{x_0+h} f(x)\,dx - \frac{h}{2}\big(f(x_0) + f(x_0+h)\big),$$

откуда следует, что естественно рассматривать R как функцию шага h: $R = R(h)$. Заметим, что $R(0) = 0$.

Продифференцируем $R(h)$ по h:

$$R'(h) = \left(\int_{x_0}^{x_0+h} f(x)\,dx\right)' - \frac{1}{2}\big(f(x_0) + f(x_0+h)\big) - \frac{h}{2}\big(f'(x_0+h)\big) =$$

$$= f(x_0+h) - \frac{1}{2}f(x_0+h) - \frac{1}{2}f(x_0) -$$

$$-\frac{h}{2}f'(x_0+h) = \frac{1}{2}\big(f(x_0+h) - f(x_0)\big) - \frac{h}{2}f'(x_0+h).$$

Заметим, что $R'(0) = 0$.

Определим R, последовательно интегрируя $R''(h)$ на отрезке $[0;\ h]$:

$$\int_0^h R''(z)\,dz = R'(h) - R'(0) = R'(h),$$

откуда имеем:

$$R'(h) = \int_0^h R''(z)\,dz = -\frac{1}{2}\int_0^h z f''(x_0 + z)\,dz\,.$$

Применяя обобщенную теорему о среднем, получаем:

$$R'(h) = -\frac{1}{2} f''(\xi_1)\int_0^h z dz = -\frac{h^2}{4} f''(\xi_1),$$

где $\xi_1 \in [x_0; x_0 + h]$ и ξ_1 зависит от h.

Далее

$$\int_0^h R'(z)\,dz = R(h) - R(0) = R(h),$$

откуда с учетом обобщенной теоремы о среднем имеем:

$$R(h) = \int_0^h R'(z)\,dz = \frac{1}{4}\int_0^h z^2 f''(\xi_1)\,dz = -\frac{h^3}{12} f''(\xi),$$

где $\xi_1 \in [x_0; x_0 + h]$.

Таким образом, погрешность метода при интегрировании функции на отрезке $[x_0; x_1]$ имеет величину:

$$R = -\frac{h^3}{12} f''(\xi),\ \ \xi \in [x_0; x_1].$$

Можно показать, что при распространении оценки на весь отрезок интегрирования $[a; b]$ получается формула:

$$R_n = \frac{h^3 n}{12} f''(\xi),\ \ \xi \in [a; b].$$

Учитывая, что $hn = b - a$, найден следующий окончательный вид для оценки погрешности метода интегрирования по формуле трапеций:

$$|R_n| \le M \frac{|b-a|h^2}{12},$$

где

$$M = \max_{x \in [a,b]} |f''(x)|.$$

Формула Симпсона

При $n = 2$ последовательно имеем ($i = 0, 1, 2$):

$$H_0 = -\frac{1}{2}\int_0^2 \frac{t(t-1)(t-2)}{t}dt = \frac{1}{6},$$

$$H_1 = -\frac{1}{2}\int_0^2 t(t-2)dt = \frac{2}{3},$$

$$H_2 = \frac{1}{4}\int_0^2 t(t-1)dt = \frac{1}{6}.$$

Тогда получим на отрезке $[x_0; x_2]$:

$$\int_{x_0}^{x_2} f(x)dx \approx (x_2 - x_0)\sum_{i=0}^{2} H_i y_i = 2h\left(\frac{1}{6}y_0 + \frac{2}{3}y_1 + \frac{1}{6}y_2\right),$$

т. е.

$$\int_{x_0}^{x_2} f(x)dx \approx \frac{h}{3}\left(\frac{1}{6}y_0 + 4y_1 + y_2\right).$$

Геометрически, в соответствии со смыслом интерполяционной формулы Лагранжа при $n = 2$, означает замену подынтегральной функции $f(x)$ параболой, проходящей через точки $M_i(x_i, y_i)$ ($i = 0, 1, 2$).

Если считать, что n – четное ($n = 2m$), то, применяя формулу (последовательно к каждой паре частичных отрезков $[x_{2i-2}; x_{2i}]$. ($i = 1, 2, \ldots, m$), получим:

$$\int_a^b f(x)dx \approx \frac{2h}{3}\left(\frac{y_0 + y_{2m}}{2} + 2y_1 + y_2 + \ldots + 2y_{2m-1}\right).$$

Формула называется формулой Симпсона.

Оценка остаточного члена формулы Симпсона дается формулой:

$$R_n = \frac{(b-a)h^4}{180} f^{'v}(\xi), \ \xi \in [a;b]$$

или

$$|R_n| \le M \frac{(b-a)h^4}{180},$$

где $M = \max\limits_{x\in[a,b]} |f''(x)|$. Как следует из оценки, формула Симпсона, оказывается точной для полиномов до третьей степени включительно (т. к. для этих случаев производная четвертого порядка равна нулю). Формула Симпсона обладает повышенной точностью по сравнению с формулой трапеций. Это означает, что для достижения той же точности, что и по формуле трапеций, в ней можно брать меньшее число отрезков разбиения.

5.6. Методы математической статистики, анализа и обработки данных

5.6.1. Данные это …

Именно так чаще всего выглядит запрос в поисковых системах – «данные это …». Разумеется, мы тоже начнем с этого шага – выясним, что же написано об интересующем нас предмете в словарях. И здесь мы уже столкнемся с первыми трудностями, поскольку определения с первой страницы выдачи не очень подходят для целей статистического анализа данных. Поэтому мы предложим свой вариант.

«Данные – это факты и идеи, представленные в виде, пригодном для анализа средствами математики»

То есть, мы готовы воспользоваться не любой информацией, а только такой, которая изначально пригодна для средств математики или может быть преобразована к пригодности. Например, повествование о произведении «Хождение за три моря» А. Никитина, несомненно, являющееся информацией, для наших целей не может быть признано данными, поскольку к математизированному виду не приводится.

Выборка vs случай

Далее мы должны договориться, что статистический анализ данных всегда имеет дело с некоторыми **выборками.** Анализ отдельного случая тоже может быть методом научного исследования и может иметь определенную научную ценность, однако средства математического анализа данных для него неприменимы.

Рис. 5.1. Выборка и случай

Таким образом, мы всегда будем иметь дело с **выборкой**, которая будет состоять из некоторого количества **случаев** (рис. 5.1). Также всегда будем помнить, что полученные нами результаты относятся исключительно к исследованной выборке, а распространение их на всю Реальность является отдельной задачей.

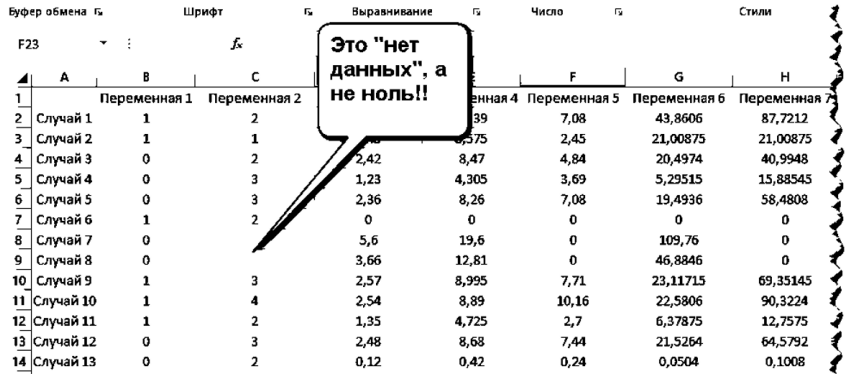
Однородность данных и табличная форма

Итак, мы остановились на том, что данные будут в виде выборки, состоящей из случаев, но теперь должны добавить еще два требования:

1. Все случаи должны описываться одинаковыми наборами данных.

2. Данные выборки должны представляться в табличном виде, где каждая строка – это отдельный случай с набором характеризующих его элементов информации, а каждый столбец – переменная, определяющая содержание элемента.

Предложенные требования не встретят проблем с пониманием, поскольку дословно соответствуют представлению данных в хорошо известном табличном процессоре Excel (рис. 5.2).



	A	B	C	D	E	F	G	H
1		Переменная 1	Переменная 2		енная 4	Переменная 5	Переменная 6	Переменная 7
2	Случай 1	1	2		39	7,08	43,8606	87,7212
3	Случай 2	1	1		575	2,45	21,00875	21,00875
4	Случай 3	0	2	2,42	8,47	4,84	20,4974	40,9948
5	Случай 4	0	3	1,23	4,305	3,69	5,29515	15,88545
6	Случай 5	0	3	2,36	8,26	7,08	19,4936	58,4808
7	Случай 6	1	2	0	0	0	0	0
8	Случай 7	0		5,6	19,6	0	109,76	0
9	Случай 8	0		3,66	12,81	0	46,8846	0
10	Случай 9	1	3	2,57	8,995	7,71	23,11715	69,35145
11	Случай 10	1	4	2,54	8,89	10,16	22,5806	90,3224
12	Случай 11	1	2	1,35	4,725	2,7	6,37875	12,7575
13	Случай 12	0	3	2,48	8,68	7,44	21,5264	64,5792
14	Случай 13	0	2	0,12	0,42	0,24	0,0504	0,1008

Рис. 5.2. Представление данных в таблице

Терминология

В различных источниках и программах применяются разные слова для обозначения случаев и переменных. Следует запомнить правила:

Объект = случай = наблюдение = case

Поле = переменная = variable

В различных контекстах переменная также может называться фактор или предиктор.

Типы данных

Точное и правильное понимание типа используемых данных совершенно необходимо для работы с методами статистического анализа, поскольку для разных типов данных применяются разные методы и критерии. Ошибка в определении типа обязательно поставит под сомнение полученные результаты. К сожалению, в терминологии и классификации здесь тоже есть существенные

разночтения, поэтому представим схему типологии данных в наиболее удобном, на наш взгляд, виде, а ниже расскажем о возможных разночтениях.

Первоначально целесообразно разделить типы данных на **количественные** и **качественные.** Количественные данные изначально характеризуются численно, а качественные говорят о принадлежности к некоторым категориям. Далее основные типы данных делятся на подтипы (рис. 5.3).

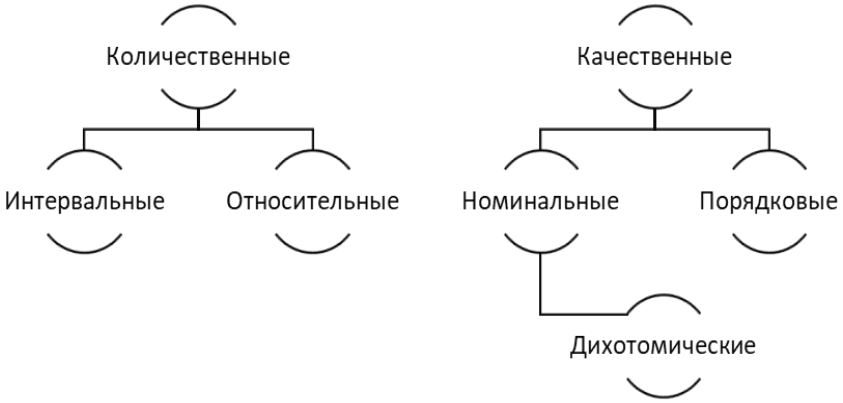


Рис. 5.3. Типы данных

Интервальные данные описывают свойство численно, однако шкала измерения не позволяет оценить отношение величин. Типичный пример – шкала температуры Цельсия, по которой нельзя сказать, что 30 градусов в 3 раза больше, чем 10.

Относительные – данные, для которых шкала начинается с нуля и позволяет записать отношение величин. Пример – шкала температуры Кельвина или длина, измеренная рулеткой.

Номинальные – данные, которые нельзя упорядочить. Пол, национальность, цвет, город и т. д.

Подтипом номинальных данных являются **Дихотомические**, у которых есть только две возможные категории. Это, конечно, мужчины и женщины, а также любой признак наличия-отсутствия.

Порядковые данные также говорят о принадлежности к категориям, но таким, что их можно выстроить по увеличению некоторого

свойства. Например, типы самолетов легкие – средние – тяжелые выстроены по увеличению массы.

Варианты терминологии

Качественные данные также могут называться категориальными и атрибутивными. Дихотомические – бинарными.

Количественные данные иногда делят на непрерывные (те, что могут иметь дробную часть) и дискретные, которые могут принимать только целочисленные значения, например, количество колес.

Что можно делать с данными (рис. 5.4)

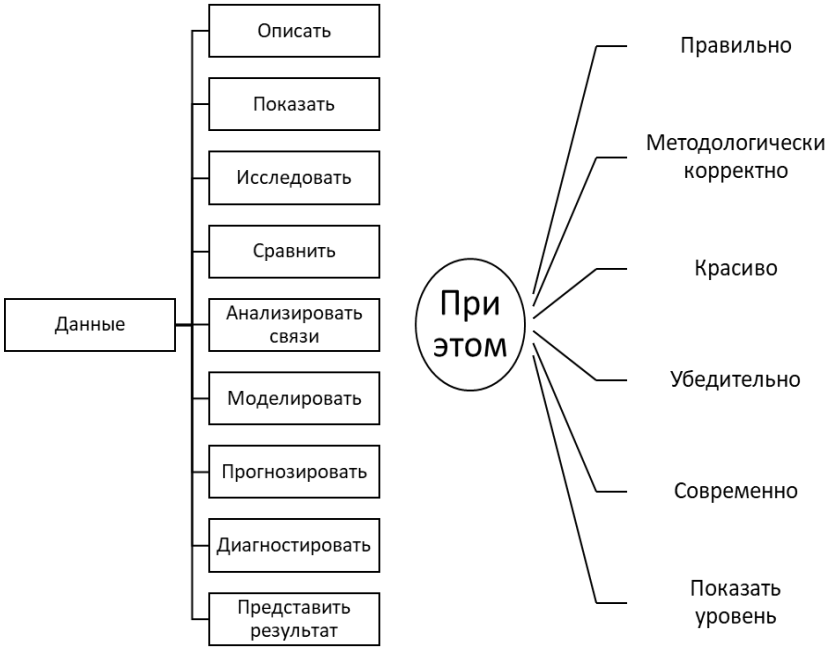


Рис. 5.4. Что можно делать с данными

Преобразование данных

Нередко исследование данных требует их преобразования, причем оно может потребовать изменение типа данных.

Основные варианты возможных преобразований представлены на рис 5.5.

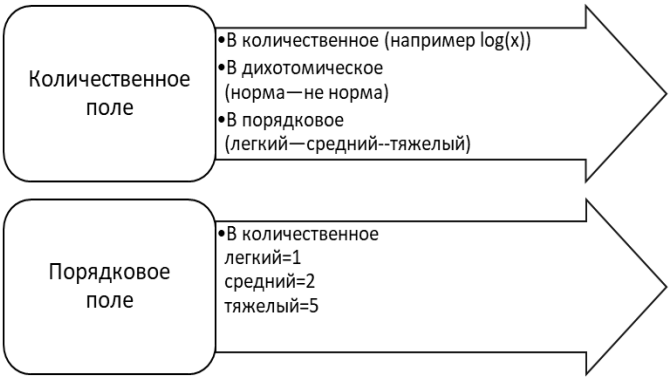


Рис. 5.5. Преобразование типа данных

Количественное поле может быть преобразовано:

– в количественное поле по некоторой формуле, например, логарифма или возведения в квадрат. Применяется для более удобного представления или для поиска линейных зависимостей;

– в дихотомическое. Это всем знакомое поле «норма-не норма»;

– в порядковое. Иногда численные данные являются избыточными и есть смысл выделения категорий в виде порядкового поля.

Порядковое поле можно преобразовать в количественное, просто считая номера категорий числами. Однако может быть, целесообразно сделать «нелинейный» перевод, как показано на рисунке, чтобы точнее отразить степень выраженности свойства.

Подготовка данных к статистическому анализу

Все, что мы производили с данными к этому моменту, было первоначальными этапами подготовки к статистическому анализу.

1. Определились с набором полей (переменных), характеризующих объект (случай). При этом, традиционно первым полем задают номер объекта, а вторым – его название.

2. Создали кодификатор и правила преобразования (кодирования) качественных данных.

3. Осуществили кодирование и получили данные в табличной форме.

4. Обдумали целесообразность преобразования данных и осуществили его при необходимости.

В абсолютном большинстве случаев подготовка данных производится в программе Excel или аналогичных табличных

процессорах Google doc, Open office. Далее следует произвести проверку полученной таблицы.

1. Проверка на простые ошибки. Посмотреть максимальные и минимальные величины в столбце и убедиться, что они в пределах разумного. Обычно это делают с помощью сортировки.

2. Проверить форматы ячеек. Часто ячейки, которые выглядят, как числа, оказываются в текстовом формате, и в пакетах прикладных программ они не будут восприниматься.

3. Убедиться, что при отсутствии данных ячейки не заполнены, и в них не стоят нули.

5.6.2. Генеральная совокупность

Генеральной совокупностью называют все множество объектов, которое мы собираемся исследовать и относительно которого собираемся делать выводы. Это то, что выше мы назвали Реальностью.

Нетрудно заметить, что генеральная совокупность представляет собой субъективное образование. То есть, ее состав напрямую зависит от того, что намерен включать в совокупность исследователь (рис. 5.6).

Рис. 5.6. Генеральная совокупность и выборка

Поэтому исследование должно начинаться именно с определения границ генеральной совокупности. Обычно это делается с помощью **критериев включения** и **критериев исключения.** В практике научных исследований редко говорят о генеральной совокупности, но почти обязательно описывают **материал исследования**, для которого

и указывают критерии включения и исключения. Таким образом, оказывается, что правила включения объектов в выборку для исследования одновременно являются правилами, описывающими границу генеральной совокупности.

После определения критериев включения объектов в исследование производится сбор информации в виде данных об объектах, таким образом формируется исследуемая выборка.

Вариабельность данных

Анализ данных основан на том, что объекты генеральной совокупности имеют различия в свойствах. Несомненно, бывают совокупности объектов, не имеющих различий, например, поверхностное наблюдение говорит, что все императорские пингвины одинаковы. Но такие данные не требуют анализа, поэтому в нашей методологии свойства исследуемых объектов всегда имеют различия.

Вариабельность данных – различия в однородных исследуемых данных или степень этих различий (рис. 5.7).

Величины, которые различны при разных наблюдениях, также называют **случайными** или **стохастическими**. По существу, различия обусловлены некоторыми внутренними или внешними причинами, рассмотрение которых не входит в задачи исследования, поэтому в статистическом анализе стохастичность величины просто считается присущим ей свойством.

Рассматривая причины различий в данных, можно выделить две основные – фактическая собственная вариабельность величины и вариабельность ее измерения.

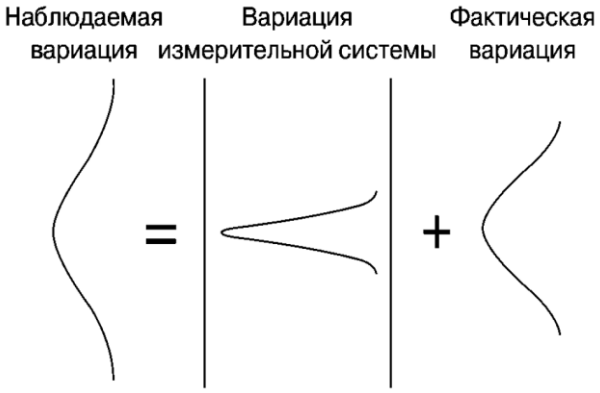


Рис. 5.7. Вариабельность данных

Таким образом, в наблюдении имеются две составляющие, соотношение которых в общем случае может быть произвольным. Однако в практике научных исследований редко встречаются случаи, когда вариабельности измерительной системы и собственной вариабельности величины соизмеримы. Обычно имеют место предельные случаи, когда одна из составляющих намного больше другой. Соответственно различны методы исследования.

В инженерных системах обычно собственная изменчивость величин невелика, а различия определяются случайной ошибкой измерения. Такие процессы и системы называют **детерминированными**, а их исследование сосредоточено на оценке точности измерения.

В сложных системах, биологических, социальных, наоборот, обычно измерения достаточно точны, а изменчивость системы связана множеством неизвестных внутренних причин. Такие системы и описывающие их модели называют ***стохастическими***.

5.6.3. Количественные и качественные данные

Представления количественных и качественных данных в исследуемых выборках совершенно различны.

Количественные данные всегда выражены числом, и эти числа для разных объектов различны. Следовательно, для выборки количественные данные по некоторой переменной представлены в виде набора чисел, причем их количество рано количеству объектов в выборке. Например, рост студентов: 168, 176, 184, 174 и т. д. И совершенно понятно, что в другой выборке будет другой набор чисел.

Качественные данные всегда указывают на принадлежность к категориям. Поэтому для выборки качественные данные представляются как количества объектов, принадлежащих к каждой категории. Например, те же студенты приехали из городов: минск-5, гродно-3, брест-4 и т. д. Здесь уже сумма чисел будет равна количеству объектов в выборке. Кроме абсолютных количеств качественные данные также представляются выборочными долями.

Выборочная доля – отношение количества объектов, принадлежащих к категории к общему количеству объектов. Например, минск-5/20, гродно-3/20 или соответственно 25 % и 15 %.

Тенденции и меры

Нетрудно понять, что оперировать данными как списком чисел неудобно и неперспективно, поэтому в математической статистике введены обобщающие понятия: центральная тенденция и мера рассеяния (табл. 5.3).

Центральная тенденция – число, описывающее всю выборку, то или иное понимание ее центра. Существует несколько величин, характеризующих центральную тенденцию, на что стоит обратить внимание.

Мера рассеяния – число, характеризующее степень различий в данных. Мера рассеяния также имеет несколько вариантов.

Центральная тенденция

Две основные величины, характеризующие центральную тенденцию, – это среднее и медиана.

Среднее – величина, представляющая собой среднее арифметическое всех наблюдений и вычисляемая по формуле. Иногда среднее обозначают буквой μ.

$$\overline{x} = \frac{x_{1+}x_2 + x_3 + \ldots x_n}{n} = \frac{\sum x_i}{n}.$$

Медиана (Me) – это такое число, что половина из элементов выборки больше него, а другая половина меньше. Для нахождения медианы выстраивают все числа по увеличению и находят расположенное посередине.

К центральным тенденциям также относится ***мода*** – величина, на которую приходится максимальное количество наблюдений.

Мера рассеяния

Рассеяние также характеризуется различными способами. Это величина дисперсия и перцентили.

Дисперсия – величина, характеризующая квадратичное отклонение наблюдений от среднего и вычисляемая по формуле

$$s^2 = \frac{\sum_{i=1}^{n}\left(x_i - \overline{x}\right)^2}{n-1}.$$

Величина S называется **среднеквадратическим отклонением. Перцентиль** – это величина, меньше которой попадает заданная доля наблюдений. Для практических целей используются *нижний квартиль* и *верхний квартиль*.

Нижний (LQ) и верхний (UQ) квартили – это 25 %-й перцентиль и 75 %-й перцентиль. Соответственно 25 % наблюдений попадет ниже нижнего квартиля, и 25 % – выше верхнего. Медиана же по существу является 50 %-м перцентилем.

Собственно же характеристикой рассеяния является расстояние от нижнего квартиля до верхнего – **интерквартильный размах**. Иногда пишут интерквартильный диапазон, межквартильный размах.

К мерам рассеяния также относится просто ***размах*** – расстояние от минимального значения до максимального.

Таблица 5.3

Термины и обозначения

Русский	Английский	Обозначение
Среднее	mean	$\overline{X}$ – выборка μ – генеральная совокупность
Дисперсия	variance	s^2 – выборка σ^2 – генеральная совокупность
Стандартное отклонение	standard deviation	*SD*
Стандартная ошибка (среднего)	standard error	*SE*
Медиана (50 % перцентиль)	median	*Me*
Нижний квартиль (25 % перцентиль)	lower quartile	*LQ*
Верхний квартиль (75 % перцентиль)	upper quartile	*UQ*
Размах	range	*R* … Min – Max

5.6.4. Функция распределения. Гистограмма

Наиболее удобное и визуально-понятное представление о характере стохастической количественной величины можно получить с помощью функции распределения, чаще всего имеющей вид гистограммы (рис. 5.8).

Функция распределения (плотность распределения) – функция, определяющая вероятность того, что случайная величина примет соответствующе значение.

Гистограмма – способ представления данных в виде столбчатой диаграммы.

Наш курс ориентирован на применение пакетов прикладных программ, поэтому алгоритм построения гистограммы вручную приведем в самом кратком виде.

Алгоритм построения гистограммы

1. Расчет размаха R из n результатов измерений (размах – это разница между наибольшим X_{max} и наименьшим X_{min} значениями).

2. Определение количества интервалов k по формуле $k = 1 + 3{,}3 \times \times \lg n$. Обычно $6 < k < 20$.

3. Вычисление ширины интервалов гистограммы h

4. Расчет границ интервалов. Границы интервалов выбирают обычно таким образом, чтобы они не совпадали с результатами измерений и крайние интервалы были заполнены.

5. Подсчет числа попаданий результатов в интервалы. Полученные результаты сводят в таблицу.

6. Построение столбчатой гистограммы.

В результате получим столбчатую диаграмму, в которой по горизонтальной оси отложена исследуемая величина, а по вертикальной – количество наблюдений, попавших в соответствующий интервал.

Огибающая столбчатую диаграмму кривая и будет представлять собой функцию плотности распределения вероятностей.

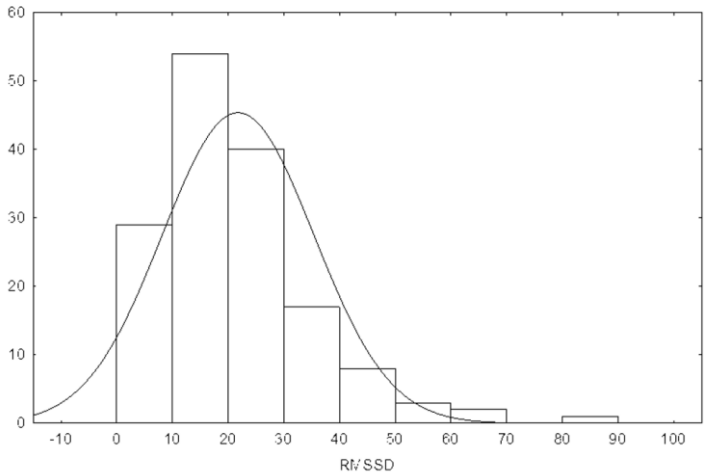


Рис. 5.8. Гистограмма

При построении гистограммы в пакете Statistica огибающая кривая строится автоматически, а над диаграммой выводятся параметры соответствующей функции Гаусса.

Нормально распределенные количественные данные

Количественные данные могут иметь *нормальное распределение* или не иметь его. И это обстоятельство имеет в математической статистике важнейшую роль, поскольку для нормально распределенных данных существуют отдельные точные методы расчетов и дополнительные возможности по их описанию.

Нормальное распределение стохастическая величина имеет в том случае, если ее функция распределения соответствует функции Гаусса

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\bar{x})^2}{2\sigma^2}}.$$

Здесь функция $f(x)$ определяет плотность вероятности, $\bar{x}$ – среднее, σ^2 – дисперсия. Функция Гаусса на гистограмме показана линией, и о ней можно сказать, что она симметрична и куполообразна.

Как узнать, можно ли считать распределение нормальным?

Фактическая гистограмма не обязательно должна точно совпадать с функцией Гаусса, достаточно, чтобы она соответствовала ей приблизительно. Для определения степени соответствия существуют специальные критерии, общее их количество не менее двух десятков. Но для практических целей обычно используют три, по той простой причине, что они непосредственно рассчитываются в программе STATISTICA. Это критерии:

– Шапиро–Уилка;

– Колмогорова–Смиронова;

– Лиллиефорса.

Вывод о нормальности распределения делается по рассчитанному *уровню значимости.* Если уровень значимости $p > 0{,}05$ считается, что значимые отличия от нормальности не обнаружены. Этот факт позволяет считать случайную величину нормально распределенной и применять к ней соответствующие методы, которые называют **параметрические.**

5.6.5. Параметрические свойства нормального распределения

Нормальное распределение описывается функцией, которая основана на параметрах среднего значения и дисперсии (рис. 5.9). Следовательно, функцию распределения можно вычислить, а это дает возможность рассчитывать аналитически вероятность попадания наблюдения в некоторый интервал значений.

Прежде всего, надо знать, какое количество наблюдений попадает в интервалы, кратные среднеквадратическому отклонению.

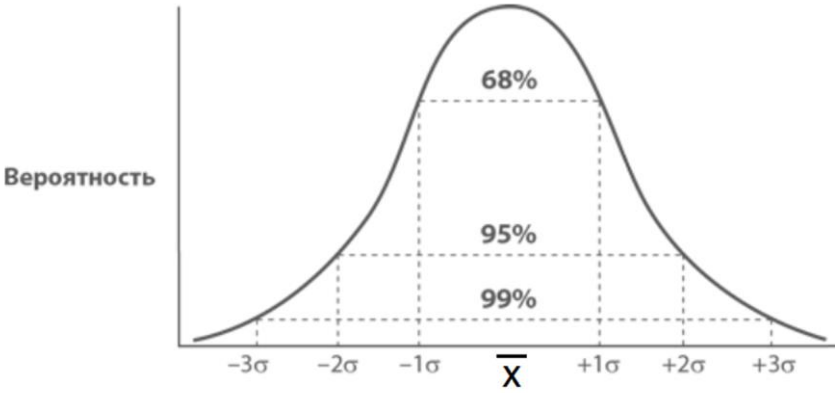


Рис. 5.9. Нормальное распределение

Как видим, в диапазон $\pm\sigma$ попадает 68 % наблюдений, в $\pm 2\sigma$ – 95 %, а в $\pm 3\sigma$ – 99 %.

В то же время можно рассчитать вероятность попадания наблюдения в произвольный диапазон. Для этого рассчитывают величину Z – отношение отклонения значения от среднего к среднеквадратической дисперсии и определяют вероятность по таблицам критерия Z. Однако в нашем курсе такая потребность не предвидится, поэтому детализацию опускаем, а при необходимости она легко может быть восстановлена поиском в сети Интернет.

Уровень значимости

Важную роль в математической статистике играет диапазон значений, выпадающий за $\pm 2\sigma$. Несложно посчитать, что вероятность получения значения $> \overline{x} + 2\sigma$ или $< \overline{x} - 2\sigma$ составляет только 5 %.

Этот факт используют при оценке вероятности ошибки статистического вывода. То есть вывод о некотором различии делают только тогда, когда вероятность ошибки меньше 5 %. Записывают так: $p < 5$ %, и называют эту величину ***уровнем значимости.***

Оценка параметров генеральной совокупности

В большинстве исследований анализ выборки осуществляется для того, чтобы получить знания о генеральной совокупности (рис. 5.10). В генеральной совокупности есть некоторое, неизвестное нам среднее значение величины и есть неизвестная нам дисперсия генеральной совокупности.

Исследуя выборки, мы получаем значения среднего и дисперсии для выборок, однако они не равны величинам генеральной совокупности. Поэтому перенос данных, полученных для выборки на генеральную совокупность, называют **оценкой параметров генеральной совокупности.**

Существует два метода оценки.

Точечная оценка

Этим красивым термином называют простой способ – считать, что среднее генеральной совокупности равно среднему выборки, а дисперсия – дисперсии выборки. Больше здесь добавить нечего, кроме того, что точечные оценки в научном исследовании могут применяться разве только для предварительных оценок или в случаях, когда научные выводы не сделаны.

Интервальная оценка

Интервальная оценка предполагает, что для параметров генеральной совокупности мы найдем некоторый интервал, в котором они находятся.

В то же время мы можем сказать, что параметр генеральной совокупности находится в заданном интервале с некоторой вероятностью. Следовательно, интервальная оценка одновременно содержит размер интервала и вероятность нахождения в нем искомого значения.

Распределение средних значений выборок

Существенную помощь здесь оказал тот факт, что средние значения выборок, извлеченных из этой генеральной совокупности, всегда имеют нормальное распределение.

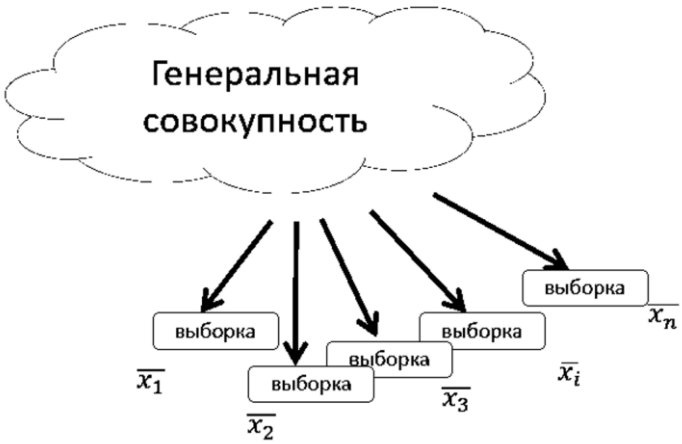


Рис. 5.10. Ошибка среднего

Причем, этот факт имеет место независимо от того, является ли распределение стохастической величины в генеральной совокупности нормальным. Точнее, это справедливо для больших выборок *с* $n > 30$, а для малых выборок применяется распределение Стьюдента, зависящее от *n*.

То есть, если мы извлечем из генеральной совокупности *n* выборок и для каждой из них рассчитаем среднее значение, а затем соберем все эти значения и посчитаем их выборкой, то окажется, что эта выборка всегда распределена нормально, а ее дисперсия в $\sqrt{n}$ раз меньше дисперсии исходных выборок.

Эту дисперсию называют ***ошибка среднего*** и обозначают как *ОС* или *SE* (см. рис. 5.10).

$$SE = \frac{\sigma}{\sqrt{n}}.$$

Известная дисперсия средней величины выборок позволяет сделать обратный вывод в отношении генеральной совокупности – ошибка оценки среднего генеральной совокупности также равна *SE*, следовательно, истинное среднее генеральной совокупности будет отличаться от точечной ошибки не более чем на *SE*.

Некоторое время назад принято было записывать результат исследования в виде $X = \bar{x} \pm SE$, например, рост студентов = 178 ± 3 см. Однако в последнее время такая запись считается некорректной, и предпочтительно просто указать параметры: **рост студентов (*SE*) составил 176 см (3 см).**

Доверительный интервал

В научных исследованиях чаще всего делают интервальную оценку в виде доверительного интервала, обычно – 95 %-й доверительный интервал (рис. 5.11). Это обусловлено тем, что в диапазон ±*SE* попадает только 68 % значений и для научного вывода это недостаточная надежность. Поэтому в большинстве случаев диапазон расширяют до ± 2*SE*, а вероятность правильного вывода становится 95 %, что считается приемлемым.

95 %-й доверительный интервал = 95 % ДИ – интервал, в котором среднее генеральной совокупности находится с вероятностью 95 %.

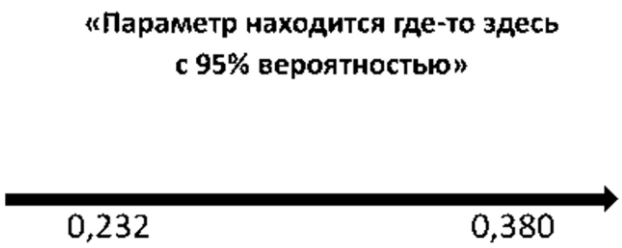


Рис. 5.11. Доверительный интервал

Для различных типов данных существуют соответствующие методы расчета доверительных интервалов.

Результат исследования, в котором оценивался доверительный интервал, записывают так: **рост студентов (95 % ДИ) составил 176 см (от 170 до 182 см).**

Применение компьютеров для статистического анализа

Рассмотрим применение компьютеров для статистического анализа на примере теста на нормальность распределения по критерию Шапиро-Уилка (Shapiro-Wilk), как это обычно делается в статистических пакетах (R, Python, SPSS) или даже вручную (хотя последнее крайне редко):

1. Подготовьте данные:

– убедитесь, что у вас есть выборка числовых данных, нормальность распределения которой вы хотите проверить. Данные должны быть представлены в виде вектора (списка чисел) или столбца в таблице данных;

– ограничение: критерий Шапиро–Уилка надежен для выборок размером примерно от 3 до 5000 наблюдений. Для выборок большего размера часто используют другие.

2. Сформулируйте гипотезы:

– нулевая гипотеза (H_0): Распределение данных соответствует нормальному распределению;

– альтернативная гипотеза (H_0): Распределение данных не соответствует нормальному распределению;

– уровень значимости (α): Выберите порог для принятия решения, обычно α = 0,05 (5 %).

3. Выполните расчет тестовой статистики и p-value:

– На языке R: Используйте функцию shapiro.test().

```
# Пример: Проверка нормальности вектора данных `my_data`
result <- shapiro.test(my_data)
print(result)  # Выведет тестовую статистику W и p-value
```

– В Python (с использованием библиотеки SciPy): используйте функцию scipy.stats.shapiro().

```
python
# Пример: Проверка нормальности массива данных `my_data`
from scipy import stats
W, p_value = stats.shapiro(my_data)
print(f"Статистика W: {W}, p-value: {p_value}")
```

– В SPSS: перейдите в меню Analyze > Descriptive Statistics > Explore... Перенесите переменную в поле Dependent List. Нажмите кнопку Plots... и в открывшемся окне установите галочку Normality plots with tests. Результат теста Шапиро–Уилка будет выведен в таблице Tests of Normality.

4. Интерпретируйте результат:

Если p-value > α (например, > 0,05): нет достаточных оснований отвергнуть нулевую гипотезу. Это означает, что наша выборка не показала статистически значимых отклонений от нормального распределения на выбранном уровне значимости.

Если p-value ≤ α (например, ≤ 0,05): Отвергаем H_0 (нулевую гипотезу) в пользу альтернативной – H_0. Это означает, что имеются статистически значимые свидетельства того, что распределение данных отличается от нормального.

5.7. Системы и пакеты для математических вычислений. Назначение, возможности, примеры применения

5.7.1. Системы математических вычислений. MatLab

Графический интерфейс пользователя и простейшие вычисления

MatLab (англ. Matrix Laboratory – матричная лаборатория), разработанная фирмой MathWorks, Inc.(США), является интерактивной системой для выполнения инженерных, экономических, научных и других расчетов. В MatLab интегрирован мощный математический аппарат, позволяющий решать сложные задачи анализа динамических систем, а также систем управления с предоставлением пользователю эффективных средств графического отображения информации.

Графический интерфейс пользователя MatLab состоит из 4 независимых окон, имеющих следующие названия:

– Command Window (Командное окно) – область ввода команд и вывода получаемых результатов;

– Workspace (Рабочая область) – область отображения всех переменных, используемых в текущем сеансе работы;

– Command History (Окно истории команд) – перечень команд, выполнявшихся ранее;

– Current Directory (Список файлов) – один из элементов панели инструментов (в верхней части экрана), на который указывается папка, которая будет по умолчанию использоваться для всех операций с файлами.

Вид окна можно изменять с помощью команд из меню Desktop, а также используя стандартные возможности управления окнами.

В рабочей области окна Command Window находится строка ввода команд, отмеченная знаком курсора >>, в которой можно вводить числа, имена переменных и знаки операций, составляющих в совокупности выражения. Имена переменных должны начинаться с буквы и состоять из букв, цифр и знаков препинания.

Вид команд, вводимых в командном окне, очень нагляден. Например, если ввести $x = 4 \cdot 3$ и нажать клавишу Enter, то на экран выводится результат. После отображения результатов вычисления в командном окне создается новая строка ввода команд, отмеченная знаком >>. При этом выполненная команда отображается в окне истории команд (Command History), а в рабочей области (Workspace) создается переменная *x*.

Если команда завершается точкой с запятой, то ее результат не выводится на экран. Если при выполнении команды не указана переменная, которой должен быть присвоен результат, то он по умолчанию присваивается переменной *ans*.

Новую команду в командном окне можно вводить только в строке ввода, в которой находится курсор. В эту строку можно копировать команды, выполненные ранее, используя обычные способы копирования. Кроме того, команды, выполненные ранее, можно вызывать в строку ввода из окна истории команд (двойным щелчком или перетаскиванием с помощью мыши), а также нажатиями клавиши «Стрелка вверх».

С переменными можно работать в рабочей области. Двойной щелчок по имени переменной в рабочей области вызывает редактор в виде электронной таблицы, в котором можно изменять значение переменной, строить графики (если переменная представляет собой матрицу) и выполнять ряд других операций. Кроме того, для переменных в рабочей области можно использовать команды контекстного меню, вызываемого правой кнопкой мыши.

Заглавные и строчные буквы в именах переменных различаются. Имеется возможность просмотра краткой подсказки по любой функции. Для этого используется команда *help*. Например, для получения подсказки о функции sin следует в окне команд ввести help sin. Для очистки командного окна используется команда clc.

Представление и отображение данных в MatLab

Основной тип данных, используемый в MatLab, – вещественные числа (тип double). Точность представления – 15 значащих цифр. Для управления форматом представления данных на экране используется команда format. Основные форматы вывода данных на экран следующие: format short – вывод с точностью до четырех значащих цифр после запятой (используется по умолчанию);

format long – полное представление числа. Выбранный формат применяется до тех пор, пока не будет введена команда format с другим форматом.

Значения переменных, вычисленных в течение текущего сеанса работы, сохраняются в специально зарезервированной области оперативной памяти компьютера, называемой рабочей областью. Значения переменных, необходимость хранения которых нецелесообразна, удаляются командой

clear name1 name2 …
Здесь name1, name2, … – имена удаляемых переменных.
Для удаления всех переменных используется команда
Clear

Содержимое рабочей области (т. е. значения всех переменных) можно сохранить на диске, используя команду меню File, Save Workspace As.., или команду save имя файла в командном окне. Файл с содержимым рабочей области сохраняется с расширением .mat. При выходе из MatLab содержимое рабочей области теряется, поэтому прежде, чем завершать сеанс работы в MatLab, необходимо сохранить рабочую область.

Для загрузки сохраненного содержимого рабочей области используется команда меню File, Open или команда load имя файла в командном окне. Имя файла в командах save и load задается по обычным правилам. Если при сохранении или загрузке файла не указывается путь, то используется папка, указанная в поле окна Current Directory.

Все команды, вводимые в MatLab, а также результаты их выполнения, выводимые в командном окне, можно сохранять в текстовом файле. Для этого требуется ввести команду diary имя_файла. Для прекращения вывода команд и результатов в файл используется команда diary off.

Операции с матрицами

Все данные в MatLab рассматриваются как матрицы. Даже обычная переменная представляет собой матрицу размером модуль *x*.

Применяются следующие простейшие способы задания матриц:

– перечислением;

– в виде диапазона значений (с помощью операции «двоеточие»);

– с помощью специальных функций.

Например, требуется задать для дальнейшего применения следующие матрицы:

a = (3 6 5), b = .
Для этого ввести:
a = [3 6 5]
b = [5 7 9; 6 2 1]

Как видно, матрицы вводятся в квадратных скобках. Если матрица двумерная (многомерная), то она вводится по строкам; при вводе в одной строке конец строки обозначается точкой с запятой. Элементы матрицы разделяются пробелами. Вместо пробела в качестве разделителя элементов матриц можно использовать запятую, например:

a = [3, 6, 5]
b = [5, 7, 9; 6, 2, 1]

Вычисления с матрицами

Примеры работы с матрицами

1. Задать матрицу-строку:

```
>>  s1 = [1 3 2]
s1 =
1   3   2
```

2. Задать матрицу-столбец:

```
>> s2 = [2;1;-1]
s2 =
2
1
-1
```

3. Задать одномерную матрицу-строку, содержащую числа от 0 до 10 с шагом 0,1.

В конце команды вставить точку с запятой, чтобы созданная матрица (в рассматриваемом примере из 101 элемента) не выводилась на экран.

```
>> dialp = 0:0.1:10;
```

4. Вычислить скалярное произведение векторов

```
>> a = [1 2 3];
>> b = [3 2 1];
>> a* b'
ans =
10
```

В соответствии с правилами умножения матриц, принятыми в линейной алгебре, можно умножать вектор-строку на вектор столбец, поэтому для вычисления скалярного произведения необходимо, как видно, предварительно транспонировать вектор *b*: «'» – символ транспонирования.

5. Поэлементное умножение векторов

```
>> a=[1 2 3];
>>b=[3 2 1];
>> a·b
ans =
3      4      3
```

6. Создать матрицу

```
>> A=[-1 1 2;3 -1 1; -1 3 4]
A =
-112
   3  -1  1
-1 3 4
```

7. Выделить заданный столбец матрицы

```
>>A(:,1)
ans =
-1
3
-1
```

8. Выделить заданную строку матрицы

```
>>A(2, :)
ans =
3  -1  1
>>
```

8. Выделить определитель матрицы

```
>> det(A)
ans =
10
```

10. Вычислить обратную матрицу

```
>>  inv (A)
ans =
-0.7000    0.2000      0.3000•
-1.3000   -0.2000     0.7000
0.8000     0.2000     -0.2000
```

Построение графиков

Для построения простых графиков вида $y = f(x)$ применяется функция plot (x, y,'строка'), где x и y – матрицы (обычно – одномерные), задающие координаты точек, по которым строится график; 'строка' – набор управляющих символов, задающих вид линии графика (необязателен). Кроме того, в окне графика имеется собственная система меню для настройки его внешнего вида. Можно также построить в одном окне несколько графиков.

Пример. Построить график функции $y = 0{,}25 \cdot x + \sin(x)$-1 для значений от 0 до 10.

1. Получить массив значений переменной 0 от 0 до 10 с шагом 0,1. Для этого ввести: x = 0:0,1:10; (точка с запятой требуется, чтобы на экран не выводились все полученные величины).

2. Получить массив соответствующих значений переменной y: $Y = 0{,}25*x+\sin(x)$-l;.

3. Для построения графика ввести: plot(x, y).

Построение трехмерных графиков

В качестве примера построения трехмерных графиков рассмотрим построение графиков функций $z = f(x, y)$. Такие графики строятся следующим образом:

– задаются диапазоны значений переменных x и y;

– строятся две матрицы значений переменных x и y, составляющие координатную сетку для последующего вычисления функции $z = f(x, y)$ и построения ее графика. Для этого используется функция meshgrid: [x, y] = meshgrid(диапазон_x, диапазон_y); (точка с запятой в конце строки желательна, так как матрицы координатной сетки x и y, получаемые в результате применения функции meshgrid, обычно достаточно велики);

– вычисляются значения функции $z = f(x, y)$ (матрица z);

– строится график функции $z = f(x, y)$: plot3(x, y, z).

Решение алгебраических и дифференциальных уравнений

Функция для решения алгебраических уравнений

Основная функция для решения алгебраических уравнений вида $f(x) = 0$ функция fzero. Она может применяться в двух формах:

fzero('уравнение', начальная_точка) или fzero('уравнение', [a; b]),

где 'уравнение' – левая часть решаемого уравнения $f(x) = 0$ или имя М-файла, реализующего функцию $f(x)$; начальная_точка – значение переменной, в окрестности которого ищется решение; a, b – границы отрезка, на котором ищется решение, при этом величины $f(a)$ и $f(b)$ должны иметь разные знаки. Если уравнение имеет несколько решений, то функция fzero находит лишь одно из них. Другие решения требуется определять, изменяя значения начальной точки или границы отрезка (a или b).

Решение систем алгебраических уравнений

Основная функция для решения систем алгебраических уравнений – функция

fsolve('система_уравнений', начальная_точка),

где 'система уравнений' – имя М-файла (функции), реализующего левые части уравнений системы; начальная_точка – массив, задающий начальную точку для поиска решения.

Для использования функции fsolve решаемую систему уравнений необходимо преобразовать к виду, где правые части уравнений представляют собой нули.

Решение дифференциальных уравнений

Для решения дифференциальных уравнений в MATLAB имеется ряд специальных функций, называемых решателями. Решатель ode55 вызывается следующим образом:

[x, y] = ode55('ду',[$x$min, $x$max], y0);

где 'ду' – имя М-файла (функции), реализующего правую часть системы дифференциальных уравнений; $x$min, $x$max – границы диапазона значений независимой переменной; y0 – массив начальных условий для функций $y1$, $y2$,...,yn.

Выходные параметры решателя ode55 имеют следующий смысл:

x – массив-столбец значений независимой переменной;

y – матрица из n столбцов, представляющих собой наборы значений переменных *y*1, *y*2,..., *yn*, полученные по результатам решения системы дифференциальных уравнений. Каждый столбец содержит набор значений одной переменной.

Поиск экстремумов функций. Решение задач линейного и нелинейного программирования

Поиск экстремумов функций

Для поиска минимумов функций одной переменной $y = f(x)$ используются функция.

fminbnd('функция', a, b),

где 'функция ' – функция $f(x)$, для которой требуется найти минимум, или имя М-файла, реализующего эту функцию; *a*, *b* – границы отрезка, на котором ищется минимум.

Если функция y = $f(x)$ имеет несколько минимумов, то функция fminbnd находят лишь один из них. Другие минимумы требуется определять, изменяя значения *a* или *b*. Если требуется найти максимум функции $y = f(x)$, то необходимо использовать функции MatLab fminbnd, указав в них функцию $f(x)$, умноженную на -1.

Для поиска минимума функций нескольких переменных применяется функция fminsearch:

fminsearch('функция',x0),

где 'функция ' – функция $f(x)$, для которой требуется найти минимум, или имя М-файла, реализующего эту функцию; $x0$ – вектор аргументов, с которого начинается поиск экстремума.

Решение задач линейного программирования

Для решения задач линейного программирования в MATLAB используется функция

linprog(f, a, b, ar, br, xmin, xmax),

где *f* – массив-столбец коэффициентов целевой функции, подлежащей минимизации;

a – матрица коэффициентов ограничений-неравенств, имеющих вид «меньше или равно» (коэффициенты каждого ограничения указываются в отельной строке матрицы);

b – массив-столбец правых частей ограничений-неравенств;

ar – матрица коэффициентов ограничений-равенств (коэффициенты каждого ограничения указываются в отдельной строке матрицы);

br – массив-столбец правых частей ограничений-равенств;

*x*min – массив-столбец ограничений на минимальные значения переменных;

*x*max – массив-столбец ограничений на максимальные значения переменных.

Большой класс практических задач оптимизации может быть сведен к решению задачи линейного программирования.

5.7.2. Моделирование динамических систем в пакете Simulink

Основные сведения о системе Simulink

Simulink – это графическая программная среда на основе MATLAB для моделирования, имитации и анализа многодоменных динамических систем. Ее основной интерфейс – это графический инструмент для построения блок-схем и настраиваемый набор библиотек блоков Система Simulink может применяться для моделирования самых разнообразных объектов и процессов: электрических схем, систем передачи и обработки сигналов, механизмов, тепловых процессов и т. д. Модель в системе Simulink строится в виде набора стандартных блоков, описывающих моделируемый объект или явление. На основе такой модели система Simulink автоматически строит описание объекта моделирования в виде систем дифференциальных уравнений, решает эти системы и отображает характеристики объекта моделирования.

Для начала работы с системой Simulink требуется в командном окне ввести команду simulink. На экран выводится окно библиотек Simulink (Simulink Library Browser).

В состав системы Simulink входят основная библиотека блоков (собственно Simulink) и ряд специализированных библиотек. Для удобства пользования библиотеки Simulink разбиты на группы и подгруппы блоков.

Приведем примеры некоторых из них:

1. Sources (источники) – набор блоков, используемых для имитации источников моделируемых величин, например: Clock (сигнал, имитирующий независимую переменную), Constant (постоянный сигнал), Pulse Generator (импульсный сигнал), Random Number

(случайный сигнал), Sine Wave (синусоидальный сигнал), Step (ступенчатый сигнал), From File (ввод величины из файла), From Workspace (ввод величины из рабочей области MATLAB) и т. д.;

2. Sinks (приемники) – набор блоков, используемых для имитации приема и отображения моделируемых величин, например: Display (отображение числовой величины), Scope (осциллограф), To File (вывод результатов моделирования в файл), To Workspace (вывод результатов моделирования в рабочую область MATLAB) и т. д.;

3. Continuous (непрерывные процессы): Derivative (производная), Integrator (интегрирование), Delay (задержка) и т. д.:

4. Math Operations (математические операции): Add (суммирование и вычитание входных величин), Divide (деление и перемножение входных величин), Gain (умножение на число или матрицу), Sum (то же, что Add), Product (то же, что Divide), Math Function (набор математических функций) и т. д.

Имеется также большой набор специализированных библиотек, например, Communications Blockset (коммуникационные системы), Neural Net-work Toolbox (нейронные сети), SimMechanics (механизмы), SimPow-erSystems (электротехника и электроника) и т. д.

Для создания модели требуется в окне библиотек Simulink выбрать команду File, New, Model. Создается пустое окно модели. Необходимые блоки перетаскиваются из библиотек в окно модели с помощью мыши.

Двойной щелчок мыши по любому из блоков, размещенных в окне модели, вызывает на экран окно параметров выбранного блока. Эти параметры могут представлять собой, например, амплитуду и частоту моделируемого сигнала, сопротивление резистора, формулу математического преобразования и т. д.

Результаты моделирования не только отображаются на экране с помощью соответствующих блоков, но и представляются в виде матриц, которые могут выводиться в рабочую область MATLAB. Это позволяет выполнять их дальнейшую обработку, используя все средства MATLAB. Сохранение файла модели осуществляется командой File, Save. Файл сохраняется под указанным именем с расширением .MDL.

6. МЕТОДЫ ОПТИМИЗАЦИИ И СИСТЕМЫ ПОДДЕРЖКИ ПРИНЯТИЯ РЕШЕНИЙ

6.1. Оптимизация как основной этап вычислительного эксперимента

Вычислительный эксперимент редко ограничивается лишь анализом единственного, «базового» сценария модели. Его истинная сила раскрывается тогда, когда мы начинаем целенаправленно искать наилучший, оптимальный вариант решения поставленной задачи. Этот целенаправленный поиск и составляет суть этапа оптимизации.

Место оптимизации в цикле вычислительного эксперимента

Оптимизация является логическим завершением и кульминацией всего цикла моделирования. Она следует за этапами построения математической модели, разработки вычислительного алгоритма и верификации, и предшествует этапу принятия решений. Без оптимизации модель остается мощным, но пассивным инструментом анализа «что будет, если...». Оптимизация же превращает ее в активный инструмент синтеза, отвечающий на вопрос: «как сделать наилучшим образом?».

Сущность и цель оптимизации

В общем виде оптимизация – это процесс нахождения экстремума (минимума или максимума) некоторой целевой функции (или функций) при заданных ограничениях.

В контексте вычислительного эксперимента:

Целевая функция (критерий оптимальности) – это численная мера эффективности или качества системы, формализованный показатель, который мы хотим улучшить. Ее вычисление, как правило, требует одного или нескольких запусков имитационной или математической модели.

Примеры: себестоимость продукции, время выполнения процесса, прочность конструкции, прибыль предприятия, расход топлива.

Параметры модели (Управляемые переменные) – это внутренние переменные модели, которые мы можем изменять в определенных пределах в поисках optimum.

Примеры: геометрические размеры детали, температура в реакторе, состав сплава, количество обслуживающих аппаратов, тарифы.

Ограничения – это условия, которые сужают область допустимых значений управляемых переменных и определяют, является ли решение вообще реализуемым.

Примеры: физические законы (положительная масса), технические нормы (напряжение не должно превышать предельного), бюджетные рамки (общая стоимость $\leq N$).

Таким образом, цель этапа оптимизации – найти такой набор значений управляемых параметров модели, который доставляет целевой функции экстремальное значение (максимум или минимум) и при этом удовлетворяет всем наложенным ограничениям.

Зачем необходима оптимизация?

Проведение оптимизации продиктовано несколькими ключевыми причинами:

Повышение эффективности. Позволяет найти решения, которые существенно (иногда на десятки процентов) лучше интуитивно выбранных или существующих аналогов.

Систематизация поиска. Человек часто не способен предугадать сложные нелинейные взаимосвязи между десятками параметров модели. Алгоритмы оптимизации проводят системный и exhaustive search в многомерном пространстве параметров.

Объективизация решений. Решение, найденное в результате оптимизации, обосновано математически и не зависит от субъективного мнения лица, принимающего решения.

Выявление скрытых возможностей. Процесс оптимизации может выявить неочевидные, контринтуитивные конфигурации параметров, ведущие к значительному улучшению.

Классификация методов оптимизации

Методы, используемые для решения оптимизационных задач в вычислительном эксперименте, чрезвычайно разнообразны и выбираются в зависимости от свойств модели.

Градиентные методы (напр., метод наискорейшего спуска) – эффективны для гладких функций, используют информацию о производных для движения к экстремуму.

Методы прямого поиска (напр., симплекс-метод Нелдера – Мида) – не требуют вычисления производных, основаны на сравнении значений функции в точках пробного комплекса.

Стохастические методы (напр., генетические алгоритмы, метод роя частиц) – эффективны для многомодальных, негладких или «зашумленных» функций, имитируют природные процессы эволюции или коллективного поведения.

Методы на основе метамоделей (суррогатных моделей) – строят упрощенную аппроксимирующую модель (например, на основе нейронных сетей или радиальных базисных функций) по результатам ограниченного числа дорогостоящих расчетов, а быструю суррогатную модель затем интенсивно используют для оптимизации.

Вывод: оптимизация – это не просто дополнительная опция, а ключевой, неотъемлемый этап современного вычислительного эксперимента. Именно он позволяет перейти от пассивного наблюдения к активному конструированию оптимальных решений, максимально используя потенциал, заложенный в математической модели.

6.2. Модели и постановки задач оптимизации в различных предметных областях

Одной из главных причин widespread adoption оптимизации является ее невероятная универсальность. Абстрактная математическая структура «найти экстремум функции при ограничениях» оказывается удивительно гибкой и может быть наполнена конкретным содержанием из самых разных наук и отраслей. Рассмотрим характерные примеры.

Техника и машиностроение

Задачи оптимизации здесь чаще всего связаны с поиском наилучших конструктивных решений, режимов работы и материалов.

Пример 1: Оптимизация конструкции балки

Цель: минимизировать вес балки прямоугольного поперечного сечения, работающей на изгиб.

Управляемые переменные: высота h и ширина b поперечного сечения балки.

Ограничения:

Прочность: напряжение в балке не должно превышать допустимое ($\sigma \leq [\sigma]$).

Жесткость: прогиб балки не должен превышать допустимый ($f \le [f]$).

Геометрические: переменные должны быть положительными и находиться в пределах стандартного сортамента ($b > 0$, $h > 0$, b, $h \in R$).

Целевая функция: вес (h, b) = $\rho \cdot L \cdot b \cdot h \rightarrow \min$, где ρ – плотность материала, L – длина балки.

Пример 2: Оптимизация теплового двигателя

Цель: максимизировать коэффициент полезного действия (КПД) цикла Карно.

Управляемые переменные: температура нагревателя Тн и температура холодильника Тх.

Ограничения:

Физические: температуры должны быть положительными и выражены в Кельвинах (Тн > 0, Тх > 0).

Логические: температура нагревателя должна быть выше температуры холодильника (Тн > Тх).

Технические: температуры ограничены свойствами материалов (T_н ≤ T_max, Тх ≥ T_min).

Целевая функция: КПД (Тн, Тх) = (1 – Тх / Тн) * 100 % → max.

Экономика, логистика и бизнес

В этих областях оптимизация используется для максимизации прибыли, минимизации издержек и оптимального распределения ресурсов.

Пример 1: Задача об оптимальном раскрое (Cutting Stock Problem)

Цель: минимизировать отходы при раскрое крупных листов материала (стали, стекла, ткани) на заготовки меньшего размера.

Управляемые переменные: количество раз каждого типа раскроя *x_ij*.

Ограничения: необходимое количество заготовок каждого типа i должно быть выполнено.

Целевая функция: суммарные_отходы (*x*_1, *x*_2, ..., *x_n*) → min.

Пример 2: Транспортная задача (Transportation Problem)

Цель: минимизировать стоимость перевозок грузов от поставщиков к потребителям.

Управляемые переменные: объем перевозок *x_ij* от поставщика i к потребителю *j*.

Ограничения:

Все запасы поставщиков a_i должны быть вывезены.

Все запросы потребителей b_j должны быть удовлетворены.

Перевозки не могут быть отрицательными ($x\_ij \geq 0$).

Целевая функция: суммарная_стоимость = $\Sigma\Sigma$ ($c\_ij * x\_ij$) → min, где c_ij – стоимость перевозки единицы груза.

Информатика и Data Science

Здесь оптимизация лежит в основе обучения моделей машинного обучения и построения эффективных алгоритмов.

Пример 1: Обучение линейной регрессии

Цель: найти такие параметры модели w (веса) и b (смещение), которые наилучшим образом описывают данные.

Управляемые переменные: вектор весов w и скаляр b.

Ограничения: часто накладываются регуляризационные ограничения на величину весов ($L1$- или $L2$-регуляризация) для борьбы с переобучением.

Целевая функция: функция_потерь $(w, b) = \Sigma (y\_i - (w*x\_i + b))^2 \rightarrow \min$.

Пример 2: Задача коммивояжера (Traveling Salesman Problem – TSP)

Цель: найти самый короткий маршрут, проходящий через все заданные города по одному разу с возвратом в исходный город.

Управляемые переменные: порядок обхода городов (перестановка).

Ограничения: каждый город должен быть посещен ровно один раз.

Целевая функция: общая_длина_маршрута (перестановка) → min.

Химия и биоинженерия

Пример: Оптимизация химического реактора

Цель: максимизировать выход целевого продукта в химической реакции.

Управляемые переменные: температура T, давление P, концентрация катализатора C_cat, время реакции t.

Ограничения:

Безопасность: температура и давление не должны превышать критических значений для установки ($T < T\_crit$, $P < P\_crit$).

Физико-химические: концентрации реагентов должны быть положительными.

Целевая функция: выход_продукта (T, P, C_cat, t) → max.

Общая схема постановки задачи

Как видно из примеров, несмотря на различие терминов и контекстов, структура задачи остается единой. Предметная область определяет смысл целевой функции и ограничений. Математическая модель формализует этот смысл в виде функций и уравнений.

Вычислительный эксперимент позволяет рассчитать значения этих функций для любых заданных управляемых переменных (часто это самый ресурсоемкий этап).

Метод оптимизации работает с этими расчетами как с «черным ящиком» и ищет экстремум, руководствуясь чисто математическими принципами.

Умение абстрагироваться от предметной специфики, выделить целевую функцию, управляемые переменные и ограничения, а затем применить адекватный математический аппарат для поиска решения является ключевым навыком для современного инженера, ученого и аналитика. Эта унификация подходов позволяет использовать мощные вычислительные методы для решения самых разнообразных прикладных проблем.

6.3. Проекции, размерность данных и способы их уменьшения

Переходя от общих принципов оптимизации к их практической реализации, мы сталкиваемся с фундаментальной проблемой вычислительной математики и анализа данных – проклятием размерности (curse of dimensionality). Это явление описывает резкое усложнение вычислительных задач с ростом числа переменных и измерений. Эта глава посвящена методам борьбы с этим проклятием через проецирование данных в пространства меньшей размерности.

Проклятие размерности: зачем нужно уменьшение размерности?

С ростом количества признаков (измерений) пространство данных становится чрезвычайно разреженным. Это приводит к нескольким ключевым проблемам:

Вычислительная сложность. Объем данных и время работы алгоритмов оптимизации и машинного обучения часто растут экспоненциально или полиномиально высокого порядка от размерности.

Переобучение моделей. Для качественного обучения модели в пространстве высокой размерности требуются объемы данных, которые на практике часто недостижимы. Модель начинает запоминать шум, а не общие закономерности.

Визуализация. Человек не способен непосредственно воспринимать данные в пространстве с размерностью больше трех.

Уменьшение размерности – это процесс преобразования данных из пространства высокой размерности в пространство низкой размерности с целью сохранения максимально возможной информации исходного набора данных или его наиболее важных структур.

Проекция как основной инструмент

Проекция – это математическое отображение точек из пространства высокой размерности $\mathbb{*R*}^n$ на подпространство меньшей размерности $\mathbb{*R*}^*k*$ (где $*k* < ·$).

Проще говоря, это способ «увидеть» многомерный объект под определенным углом, получив его упрощенное, но информативное представление. Классическая аналогия – тень (двумерная проекция) трехмерного объекта.

Все методы уменьшения размерности, так или иначе, используют идею проекции, но differ in how они определяют, какое подпространство является «наилучшим».

Подходы к уменьшению размерности

Методы можно разделить на две большие категории:

Отбор признаков (Feature Selection). Это методы, которые направлены на выбор подмножества наиболее значимых исходных признаков без их преобразования.

Фильтры. Признаки оцениваются на основе их статистических характеристик (дисперсия, корреляция с целевой переменной, взаимная информация) независимо от модели (например, удаление константных признаков).

Врапперы. Признаки выбираются на основе оценки производительности конкретной модели-критика (например, последовательный отбор признаков). Требуют больших вычислительных ресурсов.

Встроенные методы. Отбор происходит в процессе обучения модели (например, использование *L*1-регуляризации (Lasso), которая обнуляет веса у неважных признаков).

Преобразование признаков (Feature Extraction)

Это методы, которые создают новые, более эффективные признаки на основе исходных. Это и есть проекция в чистом виде.

Методы, основанные на сохранении дисперсии (вариации) данных.

Метод главных компонент (PCA – Principal Component Analysis) – ключевой алгоритм. Его цель – найти такие ортогональные направления (главные компоненты) в данных, проекция на которые сохраняет максимальную дисперсию. Первая главная компонента направлена вдоль наибольшей дисперсии, вторая – вдоль следующей наибольшей при условии ортогональности первой, и т. д.

Преимущество: линейность, простота вычислений, гарантированное нахождение оптимального линейного подпространства для задачи сохранения дисперсии.

Недостаток: PCA слеп к классовой принадлежности данных; он сохраняет общую структуру, но не обязательно оптимален для задач классификации.

Методы, основанные на сохранении топологии/расстояний

t-SNE (t-Stochastic Neighbor Embedding) – нелинейный метод, идеально подходящий для визуализации. Он focuses на сохранении попарных расстояний между точками, так что близкие в многомерном пространстве объекты остаются близкими на двумерной карте.

Преимущество: отлично выявляет скопления (кластеры) данных.

Недостаток: результат stochastic, интерпретация расстояний между кластерами может быть неоднозначна, вычислительно сложен.

UMAP (Uniform Manifold Approximation and Projection) – более современный нелинейный метод, также нацеленный на сохранение как глобальной, так и локальной структуры данных. Работает быстрее t-SNE и часто лучше сохраняет глобальную структуру.

Методы, основанные на прогнозной модели

Автоэнкодеры (Autoencoders) – это нейронные сети, которые решают задачу «бутылочного горлышка». Они обучаются восстанавливать входной сигнал, пропуская его через внутренний слой с малым количеством нейронов (кодировщик). Этот внутренний слой и есть нелинейное представление данных в низкоразмерном пространстве.

Связь с вычислительным экспериментом и оптимизацией

Процесс уменьшения размерности сам по себе является задачей оптимизации. PCA оптимизирует (максимизирует) дисперсию проекции.

t-SNE оптимизирует расхождение Кульбака–Лейблера между распределениями расстояний в исходном и целевом пространствах.

Автоэнкодер оптимизирует функцию потерь bocconstruction (например, MSE).

Таким образом, применяя эти методы на подготовительном этапе вычислительного эксперимента, мы оптимизируем сами данные, чтобы последующие этапы (построение моделей, оптимизация параметров) были вычислительно эффективными, точными и наглядными.

Уменьшение размерности – не просто технический прием, а стратегический этап анализа сложных данных. Выбор между отбором признаков, линейными (PCA) и нелинейными (*t*-SNE, UMAP) методами проекции зависит от конкретной задачи: ускорения вычислений, визуализации или повышения точности прогнозной модели.

6.4. Классификация методов минимизации функций

После того как задача оптимизации формально поставлена, ключевым вопросом становится выбор адекватного численного метода для ее решения. Существует огромное множество методов оптимизации, и их эффективность сильно зависит от свойств целевой функции и ограничений. Классификация помогает структурировать этот многообразный ландшафт и сделать осознанный выбор.

Рассмотрим каждую категорию.

По наличию ограничений

Безусловная оптимизация: задачи, в которых отсутствуют какие-либо ограничения на управляемые переменные. Методы ищут экстремум функции по всему пространству.

Примеры – большинство градиентных методов (градиентный спуск, метод Ньютона), методы прямого поиска (Нелдера–Мида).

Условная оптимизация: задачи, в которых на управляемые переменные наложены ограничения (равенства или неравенства). Методы должны искать экстремум только в допустимой области.

Подходы:

Методы штрафных функций. Преобразуют задачу с ограничениями в последовательность задач безусловной оптимизации путем добавления к целевой функции «штрафа» за нарушение ограничений.

Методы проекции. На каждом шаге алгоритма найденную точку проецируют на допустимую область.

Методы возможных направлений. Движение к экстремуму происходит вдоль направлений, остающихся в допустимой области.

Методы Лагранжа. Сводят задачу к нахождению седловой точки функции Лагранжа.

По типу целевой функции и ограничений

Линейное программирование (LP). Целевая функция и все ограничения являются линейными функциями. Класс задач, для которых существуют очень эффективные методы решения (симплекс-метод, метод внутренней точки).

Пример: Транспортная задача.

Нелинейное программирование (NLP). Целевая функция или хотя бы одно из ограничений являются нелинейными. Это наиболее общий и сложный класс.

Пример: Оптимизация конструкции балки, обучение нейронной сети.

Целочисленное (дискретное) программирование (IP). Управляемые переменные могут принимать только целочисленные значения (или дискретные значения из заданного множества).

Пример: Задача коммивояжера (переменные – факт поездки из города i в город j).

По использованию производных (порядку метода)

Эта классификация критически важна для практической реализации, так как производные не всегда доступны или их вычисление дорого.

Методы нулевого порядка (прямые методы, поисковые методы). Используют только значения целевой функции. Не требуют вычисления производных.

Преимущества: универсальность; применимы к зашумленным функциям.

Недостатки: сходимость обычно медленнее, чем у методов высших порядков.

Примеры: Метод деления отрезка пополам (для 1D), метод Нелдера-Мида (симплексный метод), метод роя частиц (PSO).

Методы первого порядка. Используют значения целевой функции и ее первых производных (градиента).

Преимущества: хорошая скорость сходимости, информация о градиенте указывает направление убывания функции.

Недостатки: требуют вычисления градиента.

Примеры: Градиентный спуск (и его модификации – Momentum, Adam), метод сопряженных градиентов.

Методы второго порядка. Используют значения функции, градиента и вторых производных (матрицы Гессе).

Преимущества: наивысшая скорость сходимости (квадратичная), учитывают «кривизну» функции.

Недостатки: требуют вычисления и хранения матрицы Гессе, что вычислительно дорого для больших задач.

Примеры: Метод Ньютона.

По размерности задачи

Одномерная минимизация (поиск на прямой): Поиск экстремума функции одной переменной. Часто является вспомогательной процедурой в многомерных методах для поиска длины шага.

Примеры: Метод золотого сечения, метод парабол.

Многомерная минимизация: Поиск экстремума функции многих переменных. Это основной объект изучения.

По детерминированности

Детерминированные методы: При повторных запусках с одними и теми же начальными условиями выдают абсолютно одинаковый результат. Большинство классических методов (градиентный спуск, Ньютон) являются детерминированными.

Стохастические (вероятностные) методы: Используют элемент случайности при поиске. При повторных запусках могут сходиться к разным (возможно, лучшим) решениям, особенно полезны для задач с большим числом локальных экстремумов.

Примеры: Генетические алгоритмы (GA), метод имитации отжига (Simulated Annealing), метод роя частиц (PSO).

Вывод. Не существует «лучшего» метода оптимизации на все случаи жизни. Выбор метода зависит от ответов на ключевые вопросы:

1. Есть ли ограничения?

2. Являются ли функция и ограничения линейными?

3. Доступны ли производные? Насколько дорого их вычислять?

4. Какова размерность задачи?

5. Имеет ли функция много локальных экстремумов?

6. Что важнее: скорость сходимости или универсальность?

Понимание этой классификации позволяет осознанно подходить к выбору инструмента для решения конкретной прикладной задачи в рамках вычислительного эксперимента.

6.5. Методы условной оптимизации

Большинство практических задач содержат ограничения, вытекающие из физических законов, технических норм, бюджетных рамок или требований к качеству. Методы условной оптимизации предназначены для решения задач, в которых требуется найти экстремум функции при наличии ограничений на управляемые переменные.

$$\min_x f(x) \text{ при условии} \begin{cases} g_i \le 0 \quad i = 1,\dots,m \\ h_j = 0 \quad j = 1,\dots,k \\ x \in D \end{cases}$$

Методы решения можно классифицировать на несколько фундаментальных подходов.

Методы преобразования к безусловной задаче

Идея этих методов – преобразовать исходную задачу с ограничениями в одну или последовательность задач без ограничений.

1. Метод штрафных функций (Penalty Methods):

Суть: к целевой функции добавляется специальная функция штрафа, которая резко возрастает при приближении к границе допустимой области и нарушении ограничений. Минимизируется новая функция:

Преимущества: универсальность, простота реализации.

Недостатки: ухудшение обусловленности задачи при больших или малых значениях параметра, что затрудняет численное решение.

Метод множителей Лагранжа

Суть: формируется функция Лагранжа, которая инкорпорирует ограничения в целевую функцию с помощью множителей Лагранжа

Теория: Для оптимальной точки существуют такие множители, что точка является стационарной точкой функции Лагранжа (выполняются условия Куна–Таккера). Это позволяет свести задачу условной оптимизации к решению системы уравнений.

Применение: Метод особенно эффективен для задач с ограничениями-равенствами.

Методы, учитывающие ограничения в процессе поиска

Эти методы работают непосредственно с допустимой областью, целенаправленно строя последовательность допустимых точек.

1. Метод возможных направлений:

Суть: на каждой итерации в текущей точке определяется возможное направление, которое ведет внутрь допустимой области и одновременно уменьшает целевую функцию. Затем производится шаг вдоль этого направления.

Сложность: нахождение возможного направления само по себе является вспомогательной задачей оптимизации.

Применение: широко используется в линейном и нелинейном программировании.

Метод проекции градиента

Суть: это развитие идеи градиентного спуска. На каждом шаге вычисляется антиградиент (направление наискорейшего спуска). Если шаг вдоль этого направления выводит из допустимой области, то полученная точка проецируется обратно на допустимую область.

Преимущества: относительная простота, если проекция на множество вычисляется легко.

Недостатки: может быть медленным для сложных допустимых множеств, где проекция трудновычислима.

Методы активного множества (для задач с ограничениями-неравенствами)

Суть: идея в том, что в точке оптимума лишь часть ограничений-неравенств является активной. Остальные ограничения можно проигнорировать. Алгоритм iteratively определяет, какое множество ограничений предположительно активно в решении, и решает задачу оптимизации только с этими ограничениями (как с равенствами). Набор активных ограничений пересматривается на каждой итерации.

Применение: лежит в основе многих алгоритмов квадратичного программирования (QP).

Специализированные методы

Линейное программирование (LP): Симплекс-метод и методы внутренней точки (interior-point methods) являются классическими и highly эффективными методами для задач с линейными целевыми функциями и ограничениями.

Целочисленное программирование (IP): Методы ветвей и границ, отсечений (branch and bound, branch and cut) используются для задач, где переменные должны принимать целые значения.

Выбор метода условной оптимизации зависит от характера ограничений (линейные/нелинейные, равенства/неравенства), размерности задачи, требований к точности и вычислительным ресурсам (табл. 6.1).

Таблица 6.1

Сравнительная характеристика методов

Метод	Преимущества	Недостатки	Применимость
Штрафных функций	Простота реализации, универсальность	Плохая обусловленность, медленная сходимость	Общего назначения, хорош для «мягких» ограничений
Множителей Лагранжа	Точность, хорошая сходимость	Сложность решения системы уравнений	В первую очередь для ограничений-равенств
Проекции градиента	Простота, гарантированная допустимость точек	Требует легковычислимой проекции, может «зигзагать»	Множества с простой проекцией (полиэдры, сферы)
Активного множества	Эффективность, точность	Сложность реализации, зависит от начального набора	Задачи с ограничениями-неравенствами

Понимание принципов работы основных подходов позволяет инженеру или исследователю выбрать наиболее адекватный инструмент для проведения вычислительного эксперимента и получения корректных результатов.

6.6. Методы решения вариационных задач

Вариационное исчисление – это раздел математики, который обобщает задачи нахождения экстремумов функций на случай, когда искомый объект является не вектором в конечномерном пространстве, а функцией (кривой, поверхностью и т. д.). Таким образом, мы ищем экстремум функционала – величины, значение которой зависит от выбора всей функции.

Постановка вариационной задачи

Простейшая задача вариационного исчисления ставится следующим образом: найти функцию, доставляющую экстремум (минимум или максимум) функционалу

$$J[y] = \int_a^b F\left(x, y(x), y'(x)\right) dx,$$

при граничных условиях: $y(a) = A$, $y(b) = B$.

Ключевое отличие. В обычной оптимизации мы ищем точку (набор чисел). В вариационной задаче мы ищем целую функцию.

Классические примеры:

Задача о брахистохроне: Определить форму кривой, по которой материальная точка под действием силы тяжести скатится из одной заданной точки в другую за кратчайшее время. (Функционал – время спуска).

Задача о кратчайшем пути: Найти кривую наименьшей длины, соединяющую две точки. (Функционал – длина кривой).

Принцип Гамильтона в механике: Движение механической системы происходит так, что функционал действия $S = \int (T - U)\, dt$ принимает стационарное значение.

Необходимое условие экстремума: уравнение Эйлера–Лагранжа

Основной инструмент решения классических вариационных задач – уравнение Эйлера–Лагранжа. Если функция доставляет экстремум функционалу, то она необходимо удовлетворяет уравнению Эйлера–Лагранжа.

Это дифференциальное уравнение второго порядка (в общем случае нелинейное), которое и определяет искомую функцию.

Прямые методы вариационного исчисления (Метод Ритца)

Уравнение Эйлера–Лагранжа не всегда можно решить аналитически. Прямые методы позволяют найти приближенное решение вариационной задачи, сводя бесконечномерную задачу к конечномерной задаче оптимизации.

Идея метода Ритца: искомая функция аппроксимируется конечной линейной комбинацией известных пробных (базисных) функций, удовлетворяющих граничным условиям:

$$y_n(x) = f_0(x) + \sum_{i=1}^{n} c_i f_i(x).$$

Связь с оптимальным управлением (Принцип максимума Понтрягина)

Вариационное исчисление естественным образом развилось в теорию оптимального управления. Здесь, помимо траектории $y(x)$, появляется управление $u(x)$, которое мы можем выбирать, чтобы минимизировать функционал.

Постановка задачи: найти управление $u(t)$ и траекторию $x(t)$, которые доставляют минимум функционалу:

$$J = \int_{t_0}^{t_1} F(t, x(t), u(t))dt.$$

Численные методы для вариационных задач

На практике большинство вариационных задач и задач оптимального управления решаются численно. Основные подходы:

Метод конечных элементов (МКЭ): развитие идей метода Ритца, где базисные функции (например, кусочно-линейные «шапочки») определяются на дискретной сетке. Широко используется для

решения задач, описываемых дифференциальными уравнениями в частных производных.

Методы стрельбы: задача с граничными условиями преобразуется в задачу с начальными условиями, и параметры подбираются так, чтобы решение удовлетворяло граничным условиям на другом конце.

Динамическое программирование (уравнение Беллмана): особенно эффективно для дискретных по времени задач оптимального управления. Принцип оптимальности: оптимальная стратегия обладает тем свойством, что whatever the initial state and initial decision are, the remaining decisions must constitute an optimal policy with regard to the state resulting from the first decision.

Вывод. Вариационное исчисление и теория оптимального управления предоставляют мощный аппарат для решения задач, где решение – это не число, а функция или процесс. От необходимых условий в виде дифференциальных уравнений Эйлера–Лагранжа до прямых методов типа Ритца и принципа максимума Понтрягина – эти методы позволяют находить оптимальные формы, траектории и стратегии управления в самых разных областях: от физики и техники до экономики и биологии.

6.7. Системы поддержки принятия решений

Главной особенностью информационной технологии поддержки принятия решений является качественно новый метод организации взаимодействия человека и компьютера. Выработка решения, что является основной целью этой технологии, происходит в результате итерационного процесса, в котором участвуют:

– система поддержки принятия решений в роли вычислительного звена и объекта управления;

– человек как управляющее звено, задающее входные данные и оценивающее полученный результат вычислений на компьютере.

Окончание итерационного процесса происходит по воле человека. В этом случае можно говорить о способности информационной системы совместно с пользователем создавать новую информацию для принятия решений.

Дополнительно к этой особенности информационной технологии поддержки принятия решений можно указать еще ряд ее отличительных характеристик:

– ориентация на решение плохо структурированных задач;

– сочетание традиционных методов доступа и обработки компьютерных данных с возможностями математических моделей и методами решения задач на их основе;

– направленность на непрофессионального пользователя компьютера;

– высокая адаптивность, обеспечивающая возможность приспосабливаться к особенностям имеющегося технического и программного обеспечения, а также требованиям пользователя.

Информационная технология поддержки принятия решений может использоваться на любом уровне управления. Кроме того, решения, принимаемые на различных уровнях управления, часто должны координироваться. Поэтому важной функцией и систем, и технологий является координация лиц, принимающих решения, как на разных уровнях управления, так и на одном уровне.

Основные компоненты. Рассмотрим структуру системы поддержки принятия решений, а также функции составляющих ее блоков, которые определяют основные технологические операции (рис. 6.1).

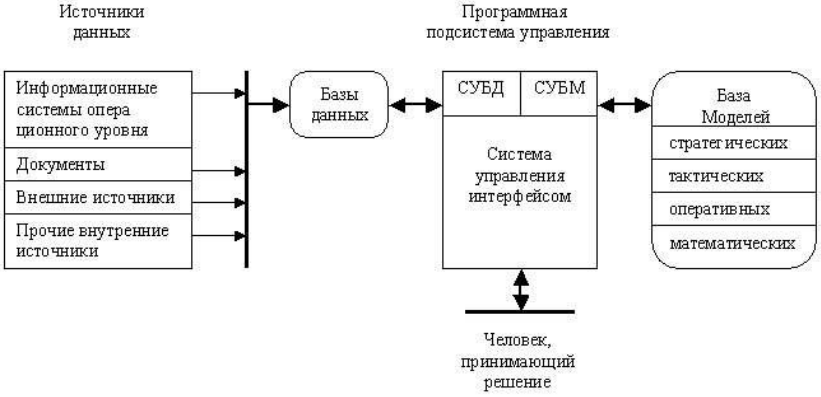


Рис. 6.1. Основные компоненты информационной технологии поддержки принятия решений

В состав системы поддержки принятия решений входят три главных компонента: база данных, база моделей и программная подсистема, которая состоит из системы управления базой данных (СУБД), системы управления базой моделей (СУБМ) и системы управления интерфейсом между пользователем и компьютером.

База данных играет в информационной технологии поддержки принятия решений (СППР) важную роль. Данные могут использоваться непосредственно пользователем для расчетов при помощи математических моделей.

Помимо данных об операциях фирмы для функционирования системы поддержки принятия решений требуются и другие внутренние данные, например данные о движении персонала, инженерные данные и т. п., которые должны быть своевременно собраны, введены и поддержаны.

Важное значение, особенно для поддержки принятия решений на верхних уровнях управления, имеют данные из внешних источников. В числе необходимых внешних данных следует указать данные о конкурентах, национальной и мировой экономике. В отличие от внутренних внешние данные обычно приобретаются у специализирующихся на их сборе организаций.

В системах поддержки принятия решения база моделей состоит из стратегических, тактических и оперативных моделей, а также математических моделей в виде совокупности модельных блоков, модулей и процедур, используемых как элементы для их построения (рис. 6.2).

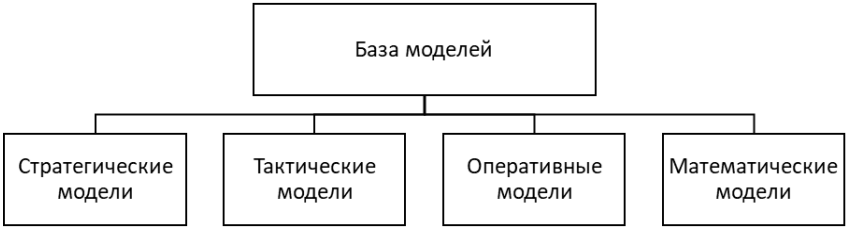


Рис. 6.2. База моделей

Значительно возросла за последнее время популярность визуального интерфейса. С помощью манипулятора «мышь» пользователь выбирает представленные ему на экране в форме картинок объекты и команды, реализуя таким образом свои действия.

В ближайшие годы следует ожидать использования в качестве языка сообщений человеческого голоса. Сейчас эта форма применяется в системе поддержки принятия решений сферы финансов, где в процессе генерации чрезвычайных отчетов голосом поясняются причины исключительности той или иной позиции.

Знания пользователя – это то, что пользователь должен знать, работая с системой. К ним относятся не только план действий, находящийся в голове у пользователя, но и учебники, инструкции, справочные данные, выдаваемые компьютером.

6.8. Понятие об экспертных системах и эвристических процедурах

Большой прогресс среди компьютерных информационных систем отмечен в области разработки экспертных систем, основанных на использовании искусственного интеллекта. Экспертные системы дают возможность менеджеру или специалисту получать консультации экспертов по любым проблемам, о которых этими системами накоплены знания.

Под искусственным интеллектом обычно понимают способности компьютерных систем к таким действиям, которые назывались бы интеллектуальными, если бы исходили от человека.

Решение специальных задач требует специальных знаний. Однако не каждая компания может себе позволить держать в своем штате экспертов по всем связанным с ее работой проблемам или даже приглашать их каждый раз, когда проблема возникла. Главная идея использования технологии экспертных систем заключается в том, чтобы получить от эксперта его знания и, загрузив их в память компьютера, использовать всякий раз, когда в этом возникнет необходимость. Являясь одним из основных приложений искусственного интеллекта, экспертные системы представляют собой компьютерные программы, трансформирующие опыт экспертов в какой-либо области знаний в форму эвристических правил (эвристик). Эвристики не гарантируют получения оптимального результата с такой же

уверенностью, как обычные алгоритмы, используемые для решения задач в рамках технологии поддержки принятия решений. Однако часто они дают в достаточной степени приемлемые решения для их практического использования. Все это делает возможным использовать технологию экспертных систем в качестве советующих систем.

Сходство информационных технологий, используемых в экспертных системах и системах поддержки принятия решений, состоит в том, что обе они обеспечивают высокий уровень поддержки принятия решений. Однако имеются три существенных различия. Первое связано с тем, что решение проблемы в рамках систем поддержки принятия решений отражает уровень ее понимания пользователем и его возможности получить и осмыслить решение. Технология экспертных систем, наоборот, предлагает пользователю принять решение, превосходящее его возможности. Второе отличие указанных технологий выражается в способности экспертных систем пояснять свои рассуждения в процессе получения решения. Очень часто эти пояснения оказываются более важными для пользователя, чем само решение. Третье отличие связано с использованием нового компонента информационной технологии – знаний.

Основными компонентами информационной технологии, используемой в экспертной системе, являются: интерфейс пользователя, база знаний, интерпретатор, модуль создания системы (рис. 6.3).

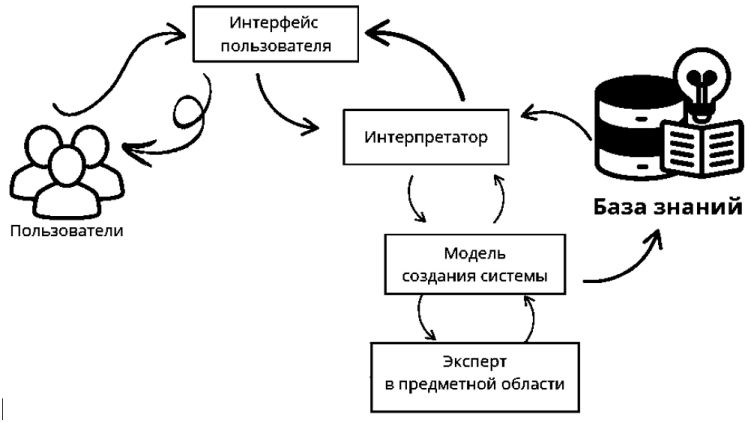


Рис. 6.3. Компоненты экспертной системы

Интерфейс пользователя. Менеджер (специалист) использует интерфейс для ввода информации и команд в экспертную систему и получения выходной информации из нее. Команды включают в себя параметры, направляющие процесс обработки знаний. Информация обычно выдается в форме значений, присваиваемых определенным переменным.

Менеджер может использовать четыре метода ввода информации: меню, команды, естественный язык и собственный интерфейс.

Технология экспертных систем предусматривает возможность получать в качестве выходной информации не только решение, но и необходимые объяснения.

Различают два вида объяснений:

– объяснения, выдаваемые по запросам. Пользователь в любой момент может потребовать от экспертной системы объяснения своих действий;

– объяснения полученного решения проблемы. После получения решения пользователь может потребовать объяснений того, как оно было получено. Система должна пояснить каждый шаг своих рассуждений, ведущих к решению задачи. Хотя технология работы с экспертной системой не является простой, пользовательский интерфейс этих систем является дружественным и обычно не вызывает трудностей при ведении диалога.

База знаний содержит факты, описывающие проблемную область, а также логическую взаимосвязь этих фактов. Центральное место в базе знаний принадлежит правилам. Правило определяет, что следует делать в данной конкретной ситуации, и состоит из двух частей: условие, которое может выполняться или нет, и действие, которое следует произвести, если выполняется условие.

Все используемые в экспертной системе правила образуют систему правил, которая даже для сравнительно простой системы может содержать несколько тысяч правил.

Все виды знаний в зависимости от специфики предметной области и квалификации проектировщика (инженера по знаниям) с той или иной степенью адекватности могут быть представлены с помощью одной либо нескольких семантических моделей. К наиболее распространенным моделям относятся логические, продукционные, фреймовые и семантические сети.

Интерпретатор – это часть экспертной системы, производящей в определенном порядке обработку знаний, находящихся в базе знаний. Технология работы интерпретатора сводится к последовательному рассмотрению совокупности правил (правило за правилом). Если условие, содержащееся в правиле, соблюдается, то выполняется определенное действие, и пользователю предоставляется вариант решения его проблемы.

Кроме того, во многих экспертных системах вводятся дополнительные блоки: база данных, блок расчета, блок ввода и корректировки данных. Блок расчета необходим в ситуациях, связанных с принятием управленческих решений. При этом важную роль играет база данных, где содержатся плановые, физические, расчетные, отчетные и другие постоянные или оперативные показатели. Блок ввода и корректировки данных используется для оперативного и своевременного отражения текущих изменений в базе данных.

Модуль создания системы служит для создания набора (иерархии) правил. Существует два подхода, которые могут быть положены в основу модуля создания системы: использование алгоритмических языков программирования и использование оболочек экспертных систем.

Для представления базы знаний специально разработаны языки Лисп и Пролог, хотя можно использовать и любой известный алгоритмический язык.

Оболочка экспертных систем представляет собой готовую программную среду, которая может быть приспособлена к решению определенной проблемы путем создания соответствующей базы знаний. В большинстве случаев использование оболочек позволяет создавать экспертные системы быстрее и легче в сравнении с программированием.

6.9. Обзор и характеристика стандартных пакетов программ анализа данных

Теоретические знания по оптимизации, статистике и численным методам находят свое практическое воплощение в программном обеспечении. Современный вычислительный эксперимент

немыслим без использования специализированных пакетов и сред, которые предоставляют готовые, оптимизированные реализации алгоритмов, средства визуализации и управления данными. Данный раздел дает обзор наиболее популярных и мощных инструментов.

Классификация пакетов

Пакеты для анализа данных можно условно разделить на несколько категорий:

Универсальные среды научных вычислений и языка программирования.

Статистические пакеты с акцентом на готовые решения и GUI.

Специализированные пакеты для визуализации и интерактивного анализа.

Пакеты для глубокого обучения и ИИ.

Системы управления базами данных (СУБД) и средства для работы с Big Data.

Универсальные среды научных вычислений

Эти инструменты являются наиболее гибкими и мощными, так как позволяют не только применять готовые функции, но и программировать собственные алгоритмы.

Python + NumPy/SciPy/Pandas/Matplotlib/Scikit-learn:

Характеристика: не единый пакет, а экосистема свободно распространяемых (open-source) библиотек. Фактический стандарт в научном сообществе и индустрии для анализа данных и машинного обучения.

Сильные стороны: гибкость и универсальность: от скриптов для быстрой проверки гипотез до создания complex production-систем. Огромное сообщество и богатейший набор библиотек для любых задач: NumPy (вычисления с массивами), SciPy (научные вычисления и оптимизация), Pandas (анализ и обработка табличных данных), Matplotlib/Seaborn (визуализация), Scikit-learn (классические ML алгоритмы), TensorFlow/PyTorch (глубокое обучение).

Хорошая интеграция с другими инструментами и платформами.

Слабые стороны: требует умения программировать; по скорости выполнения отдельных алгоритмов может уступать высоко оптимизированным коммерческим пакетам.

R + tidyverse:

Характеристика: Свободная среда для статистических вычислений и графики, особенно сильна в статистическом моделировании, анализе и создании публикационных графиков.

Сильные стороны: невероятное богатство статистических методов. Огромное количество пакетов на CRAN для самых узкоспециализированных задач.

Превосходные возможности для визуализации (пакеты ggplot2, lattice).

Концепция tidyverse (dplyr, tidyr)提供了 очень удобный и последовательный синтаксис для обработки данных.

Слабые стороны: синтаксис и идеология языка R менее интуитивны для программистов, пришедших из других языков; не так силен в задачах, выходящих за рамки статистики (например, создание веб-приложений).

Julia:

Характеристика: современный высокопроизводительный язык научных вычислений, создан с целью совместить скорость языков типа C с простотой использования Python.

Сильные стороны: высокая скорость выполнения благодаря JIT-компиляции.

Читаемый синтаксис, похожий на Python.

Хорошая поддержка параллельных и распределенных вычислений «из коробки».

Слабые стороны: моложе Python и R, поэтому сообщество и количество библиотек пока меньше, хотя быстро растут.

MATLAB:

Характеристика: проприетарная среда для численных вычислений, моделирования и анализа данных.

Сильные стороны: очень удобный и оптимизированный движок для операций с матрицами и линейной алгебры.

Широкий набор Toolboxes (пакетов расширений) для специфических областей (финансы, обработка сигналов, контроль систем).

Интуитивно понятный язык, легкий для освоения инженерами.

Отличная встроенная среда разработки (IDE).

Слабые стороны: высокая стоимость; менее гибкий по сравнению с Python; в основном сфокусирован на численных методах, а не на работе с данными и ML.

Wolfram Mathematica:

Характеристика: уникальная проприетарная система компьютерной алгебры.

Сильные стороны: символьные вычисления – сильнейшая сторона.

Интегрированные curated data (например, данные о странах, химических элементах).

Высокое качество визуализаций и интерактивных возможностей.

Единый, мощный язык.

Слабые стороны: очень высокая стоимость; менее распространена в data science по сравнению с Python/R; свой уникальный язык.

Статистические пакеты с GUI

IBM SPSS Statistics:

Характеристика: коммерческий пакет для статистического анализа, широко используется в социальных науках, маркетинге.

Сильные стороны: очень удобный графический интерфейс (GUI); мощные средства для проведения стандартных статистических тестов; хорошие возможности для работы с опросами.

Слабые стороны: ограниченные возможности для программирования и создания кастомных анализа; стоимость.

Stata / SAS:

Характеристика: мощные коммерческие пакеты, исторически популярные в экономике, эпидемиологии, госуправлении.

Сильные стороны: огромный арсенал проверенных временем статистических методов; высокая надежность; хорошая документация.

Слабые стороны: стоимость; проприетарные языки; менее гибкие для современных задач ML и визуализации.

Системы управления базами данных (СУБД) и Big Data

SQL (SQLite, PostgreSQL, MySQL, etc.): Язык для работы с реляционными базами данных. Критически важно для извлечения и первичной агрегации данных.

Apache Spark: Фреймворк для распределенной обработки больших данных. Предоставляет APIs на Python, R, Scala и Java для выполнения сложных аналитических задач на кластерах компьютеров.

Критерии выбора инструмента

Выбор пакета зависит от конкретных задач:

Тип задачи: статистический анализ (R), машинное обучение (Python), инженерное моделирование (MATLAB), символьные вычисления (Mathematica).

Требования к производительности: высокая скорость (Julia, C++), распределенные вычисления (Spark).

Стоимость и лицензирование: бюджет проект (open-source vs. проприетарный).

Необходимость интеграции: существующая IT-инфраструктура предприятия.

Аудитория: кто будет использовать результаты? (Для отчетов с графикой – R ggplot2, для встраивания алгоритма в веб-сервис – Python).

Вывод. Современный аналитик или исследователь должен владеть не одним, а несколькими инструментами. Комбинация Python/R + SQL является на сегодняшний день наиболее востребованной и мощной связкой для решения подавляющего большинства задач анализа данных и вычислительного эксперимента. Понимание возможностей и ограничений каждого пакета позволяет выбрать правильный инструмент для каждой конкретной задачи.

6.10. Искусственный интеллект, нейронные сети, эволюционные вычисления и теория нечетких множеств

Классические методы оптимизации и анализа данных, рассмотренные ранее, often сталкиваются с limitations при решении сложных, плохо формализуемых задач, характеризующихся большими объемами данных, нелинейностью, зашумленностью и неполнотой информации. Для решения таких задач были разработаны подходы, вдохновленные биологическими и когнитивными системами. Они образуют ядро современных направлений искусственного интеллекта (ИИ).

Искусственный интеллект (ИИ) как мета-область

Искусственный интеллект (ИИ) – это широкая область компьютерных наук, целью которой является создание машин и систем, способных выполнять задачи, требующие человеческого интеллекта. К таким задачам относятся:

1. Обучение (Learning): приобретение знаний и навыков на основе данных и опыта.
2. Рассуждение (Reasoning): логический вывод и принятие решений на основе правил и фактов.
3. Восприятие (Perception): анализ и интерпретация сенсорных данных (зрение, речь).
4. Взаимодействие (Interaction): коммуникация с человеком и окружающей средой.
5. Методы, описанные ниже, являются инструментами для достижения этих целей.

Нейронные сети и глубокое обучение

Искусственные нейронные сети (ИНС) – это математические модели, а также их программные или аппаратные реализации, построенные по принципу организации и функционирования биологических нейронных сетей (сетей нервных клеток живого организма).

Базовый элемент – нейрон: принимает входные сигналы, умножает их на веса (силу синаптических связей), суммирует, добавляет смещение (bias) и пропускает через нелинейную функцию активации (e.g., Sigmoid, ReLU). Это позволяет сети обучаться сложным нелинейным закономерностям.

Архитектура: нейроны организованы в слои (входной, скрытые, выходной). Сложность модели определяется количеством скрытых слоев и нейронов в них.

Обучение: процесс настройки весов и смещений нейронов на основе обучающих данных. Основной алгоритм обучения – обратное распространение ошибки (backpropagation), который представляет собой частный случай градиентного спуска, applied к многослойной структуре.

Глубокое обучение (Deep Learning): Подобласть машинного обучения, связанная с использованием многослойных (глубоких) нейронных сетей. Глубина позволяет сети создавать иерархические

представления данных (например, на первом слое распознаются края, на следующем – паттерны, затем части объектов и, наконец, целые объекты).

Применение: Распознавание изображений и речи, генерация текста и музыки, машинный перевод, беспилотные автомобили.

Эволюционные вычисления

Это семейство методов глобальной оптимизации, основанных на принципах естественной эволюции: наследовании, мутации, отборе и рекомбинации. Они работают не с одним решением, а с популяцией возможных решений (особей).

Генетические алгоритмы (ГА): наиболее известный представитель.

Кодирование: Решения (особи) кодируются в виде хромосом (часто в виде битовых строк или векторов чисел).

Оценка приспособленности: каждая особь оценивается с помощью целевой функции (fitness function).

Отбор: наиболее приспособленные особи получают больше шансов передать свои «гены» следующему поколению.

Кроссовер (Рекомбинация): две родительские особи обмениваются частями своих хромосом, создавая потомков.

Мутация: случайное изменение некоторых «генов» в хромосоме для внесения нового генетического материала и предотвращения сходимости к локальному оптимуму.

Сильные стороны: не требуют вычисления градиента, менее чувствительны к локальным оптимумам, хорошо работают с дискретными и смешанными переменными.

Применение: оптимизация сложных систем (расписания, топология антенн), настройка гиперпараметров ML-моделей.

Теория нечетких множеств и нечеткая логика

Классическая булева логика оперирует только двумя состояниями: True (1) и False (0). Теория нечетких множеств расширяет эту концепцию, позволяя элементу принадлежать множеству частично (со степенью принадлежности от 0 до 1).

Нечеткое множество: определяется функцией принадлежности $\mu(x) \rightarrow [0, 1]$, которая указывает степень принадлежности элемента *x* к данному множеству.

Пример: Множество «Высокая температура». Для 36,6 °C $\mu = 0{,}1$, для 38,5 °C $\mu = 0{,}7$, для 40,0 °C $\mu = 1{,}0$.

Нечеткая логика: использует нечеткие множества для формирования нечетких правил вида ЕСЛИ (предпосылка), ТО (заключение).

Пример правила для кондиционера: ЕСЛИ температура = высокая, ТО мощность = максимальная.

Применение: нечеткие логические контроллеры в бытовой технике (стиральные машины, кондиционеры), системы принятия решений в условиях неопределенности, экспертные системы (табл. 6.2).

Таблица 6.2

Сравнительная характеристика и синергия методов

Метод	Ключевая идея	Сильные стороны	Слабые стороны
Нейронные сети	Аппроксимация сложных нелинейных функций	Высокая точность на больших данных, универсальность	«Черный ящик», требует много данных и вычислительных ресурсов
Эволюционные вычисления	Поиск решений через эволюции	Глобальная оптимизация, не требует градиента	Медленная сходимость, стохастичность, параметры настройки
Нечеткая логика	Моделирование человеческой логики и неопределенности	Интерпретируемость, работа с неточными данными	Создание правил часто требует экспертных знаний

Эти методы не являются взаимоисключающими. Часто их комбинируют для создания гибридных интеллектуальных систем:

Нейро-нечеткие системы: нейронная сеть обучается на данных, чтобы автоматически формировать и настраивать функции принадлежности и правила нечеткой логики. Это сочетает способность НС к обучению с интерпретируемостью нечетких систем.

Генетические алгоритмы для настройки НС: ГА используются для оптимизации архитектуры нейронной сети (число слоев, нейронов) или гиперпараметров, что избавляет исследователя от ручного перебора.

Вывод. Современный искусственный интеллект представляет собой богатый арсенал методов, основанных на различных принципах. Выбор подхода зависит от конкретной задачи: наличия данных, требований к интерпретируемости результатов, необходимости глобальной оптимизации или работы с нечеткими понятиями. Понимание основ нейронных сетей, эволюционных вычислений и нечеткой логики позволяет инженеру и исследователю выбирать и комбинировать наиболее подходящие инструменты для проведения сложного вычислительного эксперимента и решения прикладных задач.

7. ПРИМЕНЕНИЕ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ В СЕЛЬСКОМ ХОЗЯЙСТВЕ

7.1. Пакеты специальных прикладных программ для обработки данных в сельском хозяйстве

Современное сельское хозяйство перестало быть исключительно «тракторной» отраслью. Сегодня это высокотехнологичная сфера, где ключевую роль играют данные и их анализ. Для сбора, обработки и интерпретации этих данных используются специализированные программные комплексы. Эти пакеты позволяют перейти от интуитивного управления к точному земледелию (precision farming), основанному на количественных показателях и прогнозах.

Специализированное программное обеспечение (ПО) для АПК можно условно разделить на несколько ключевых категорий в зависимости от решаемых задач.

1. Геоинформационные системы (ГИС) и платформы точного земледелия

Это ядро цифровизации сельского хозяйства. Данные программы интегрируют спутниковые снимки, данные с датчиков на технике, почвенные пробы и агрономические наблюдения в единую интерактивную карту полей.

Примеры ПО:

OneSoil: популярная платформа, предлагающая веб-приложение и мобильное приложение для мониторинга состояния посевов по спутниковым снимкам, создания карт продуктивности и зонирования полей.

SAP Agro: мощное корпоративное решение для управления всем циклом производства – от планирования посевов и контроля выполнения полевых работ до учета урожая и анализа экономической эффективности.

«Агросигнал» – отечественная платформа, предоставляющая услуги мониторинга вегетации, прогнозирования урожайности, контроля хода полевых работ и скаутинга (фиксации проблем на полях).

Решаемые задачи:

– создание карт неоднородности полей;

– планирование дифференцированного внесения удобрений и средств защиты растений (СЗР);

– мониторинг развития культур и выявление участков со стрессом (недостаток влаги, питательных веществ, болезни);

– контроль выполнения полевых работ в режиме реального времени.

2. Программы для управления сельскохозяйственным предприятием (ERP-системы)

Эти системы автоматизируют бизнес-процессы и управление ресурсами предприятия. Они часто интегрируются с ГИС-платформами для получения полной картины: от состояния поля до финансового результата.

Примеры ПО:

1С:ERP Агропромышленный комплекс 2: специализированное решение на базе платформы 1С, охватывающее управление растениеводством, животноводством, переработкой, складом, финансами и логистикой.

AgriXP (Россия): облачная система для управления растениеводством, учета техники, складского учета (ГСМ, семена, удобрения) и формирования отчетности.

Решаемые задачи:

– планирование посевных кампаний и бюджета;

– учет материально-технических ресурсов (семена, удобрения, ГСМ, запчасти);

– управление парком техники (учет отработанных моточасов, планирование ТО);

– расчет себестоимости продукции и анализ экономической эффективности.

3. Программы для работы с данными с сельскохозяйственной техники (API-системы)

Современные комбайны и тракторы генерируют огромные массивы данных в формате ISOXML. Для их обработки и анализа используются специальные программы.

Примеры ПО:

MyJohnDeere Operations Center / CLAAS 365 / AgroConsole Connect (от CNH): облачные платформы самих производителей техники. Они позволяют просматривать данные о выполненных операциях, анализировать расход топлива, урожайность и составлять отчеты.

SMS Advanced (от AGCO): desktop-приложение для сложного агрономического анализа данных об урожайности, построения карт внесения и планирования севооборотов.

Решаемые задачи:

– импорт и конвертация данных с накопителей с техники;

– анализ качества выполнения полевых работ (например, контроль за перекрытиями при посеве или внесении удобрений);

– создание карт урожайности;

– формирование заданий для дифференцированного внесения материалов.

4. Специализированные приложения и сервисы

Для животноводства: программы для управления фермой (учет поголовья, контроль надоев, рационы кормления, ветеринария), например, «1С: Управление свиноводческим предприятием», Cattle360.

Для агрономического моделирования: программы для прогнозирования урожайности, развития болезней и вредителей на основе погодных данных (например, модели в платформе «Агросигнал»).

Мобильные приложения для скаутинга: приложения, позволяющие агроному непосредственно в поле отмечать на карте проблемные участки, делать фотографии и привязывать их к координатам (например, OneSoil Scouting, Fieldview Cab).

Использование пакетов прикладных программ позволяет сельскохозяйственным предприятиям перейти на качественно новый уровень управления. Интеграция данных из разных источников (спутники, датчики, техника, человек) в единую цифровую среду дает возможность принимать обоснованные, своевременные и экономически эффективные решения, снижая риски и повышая продуктивность аграрного производства.

7.2. Разработка и использование моделей для решения прикладных задач в сельском хозяйстве

Собранные с помощью технологий точного земледелия данные являются ценным сырьем. Однако сами по себе они не дают готовых решений. Ключевым этапом становится математическое и компьютерное моделирование – создание цифровых «двойников» реальных агрономических процессов. Эти модели позволяют не просто констатировать факты, но и предсказывать будущие состояния системы и оценивать последствия управленческих решений.

Модели в сельском хозяйстве можно классифицировать по их целевому назначению.

Типы моделей и решаемые задачи

1. **Фенологические модели**

Описывают развитие сельскохозяйственных культур в зависимости от факторов среды, прежде всего – температуры и влажности.

Решаемая задача: прогнозирование наступления критических фаз развития растения (всходы, кущение, цветение, созревание).

Практическое применение: оптимизация сроков посева. Модель рассчитывает наиболее вероятную дату наступления урожайной спелости, позволяя избежать рисков ранних заморозков или засухи.

Планирование полевых работ: точное прогнозирование фазы цветения позволяет оптимально запланировать внесение средств защиты растений или полив.

Прогноз урожайности: на основе пройденных фаз развития и накопленной биомассы модель может корректировать ожидаемый yield.

2. **Модели роста и продуктивности растений**

Это более сложные модели, которые учитывают большее число переменных: интенсивность солнечной радиации (ФАР), влажность почвы, содержание питательных веществ, плотность посева и др.

Решаемая задача: прогнозирование биомассы и конечной урожайности культуры.

Практическое применение: расчет нормы внесения удобрений. Модель может симулировать отклик растения на разные дозы азота, фосфора и калия, помогая подобрать экономически и экологически оптимальную норму.

Управление орошением: интеграция с данными метеостанций и датчиков влажности почвы позволяет моделировать водный баланс поля и давать рекомендации по времени и объему полива для предотвращения стресса у растений.

3. **Модели распространения болезней и вредителей** (Эпифитотиологии)

Прогнозируют вспышки заболеваний и активность вредителей на основе погодных условий (температура, влажность воздуха, наличие осадков) и фенофазы растения-хозяина.

Решаемая задача: оценка риска возникновения эпифитотии (массового распространения болезни).

Практическое применение: система принятия решений для обработок. Модель определяет «окна уязвимости» культуры и «благоприятные периоды» для патогена. Это позволяет применять пестициды не по календарному принципу, а только тогда, когда риск экономического ущерба превышает cost обработки, что является основой интегрированной защиты растений (IPM).

4. **Гидрологические и эрозионные модели**

Моделируют движение воды и питательных веществ в почве, прогнозируют процессы водной и ветровой эрозии.

Решаемая задача: оценка воздействия сельскохозяйственной деятельности на окружающую среду и плодородие почв.

Практическое применение: противоэрозионная организация территории. Моделирование помогает определить оптимальное размещение лесополос, буферных зон и гидротехнических сооружений.

Предотвращение вымывания нитратов: модель может показать, когда и в каком количестве внесенные удобрения могут достигнуть грунтовых вод, что позволяет скорректировать график и нормы их внесения.

Процесс разработки и внедрения моделей

Постановка задачи: определение конкретной прикладной проблемы (например, «спрогнозировать урожайность пшеницы в хозяйстве с точностью до 10 % за 30 дней до уборки»).

Сбор и подготовка данных: использование исторических данных (погода, урожайность, агротехнологии) и данных в реальном времени (спутники, датчики). Это самый критичный этап – качество модели напрямую зависит от качества и релевантности данных.

Выбор или разработка алгоритма: можно использовать как классические статистические модели (регрессионный анализ), так и современные методы машинного обучения (Machine Learning) и искусственного интеллекта (AI).

Например: Нейронные сети для прогнозирования урожайности по спутниковым снимкам.

Алгоритмы классификации для автоматического обнаружения сорняков или болезней по изображениям с дронов.

Валидация и калибровка: модель тестируется на независимых данных (например, по данным прошлого года, которое не использовалось при обучении). Параметры модели подстраиваются под конкретные условия региона или даже поля.

Внедрение и использование: модель интегрируется в пользовательский интерфейс (например, в виде модуля в платформе точного земледелия) и выдает рекомендации агроному в понятной форме: «Вероятность развития септориоза на поле № 5 в ближайшие 5 дней превышает 70 %. Рекомендуется провести обработку фунгицидом X».

Математическое моделирование превращает сельское хозяйство из ремесла в точную науку. Оно позволяет перенести риски с реальных полей в виртуальную среду, где можно протестировать десятки сценариев и выбрать наилучший. Развитие этого направления напрямую связано с прогрессом в области больших данных (Big Data) и искусственного интеллекта, что открывает путь к созданию полностью автономных «цифровых агрономов» – систем, способных самостоятельно принимать и реализовывать решения по управлению агроценозами.

7.3. Постановка эксперимента и автоматизация обработки данных. Компьютерное зрение, анализ текста, временные ряды

Современный агроном-исследователь или технологическая компания в АПК сталкивается не с отдельными разрозненными данными, а с их непрерывными потоками. Ключевая задача – организовать их сбор, структурировать и автоматически извлекать ценную информацию, минимизируя ручной труд. Этот процесс начинается с корректной постановки эксперимента и заканчивается применением передовых методов анализа.

Постановка эксперимента в цифровом сельском хозяйстве

Цифровой эксперимент – это не просто пробное внесение удобрений на участке. Это строгий план сбора данных, при котором можно однозначно оценить эффект от того или иного воздействия.

Определение гипотезы и переменных: четкое формулирование, что мы проверяем (например, «Применение биопрепарата X увеличивает урожайность кукурузы на 5 %»). Определение независимых переменных (доза препарата, срок внесения) и зависимых (урожайность, содержание хлорофилла, NDVI).

Планирование экспериментального поля: разбивка на делянки (поля-участники), включая контрольную группу (где воздействие

не применяется) и несколько опытных групп (с разным уровнем воздействия). Для исключения влияния неоднородности почвы используются методы рандомизации и повторности.

Выбор инструментов и метрик: определение, какие именно данные и с помощью каких sensors будут собираться (спутниковые снимки, дроны с мультиспектральными камерами, датчики на технике, отбор почвенных и растительных проб).

Протоколирование: все действия (дата посева, дата обработки, погодные условия) должны фиксироваться в цифровом виде, чтобы потом их можно было коррелировать с результатами.

Результат: чистые, структурированные и релевантные данные, готовые для автоматизированной обработки.

Автоматизация обработки данных: ключевые технологии

Собранные данные часто представляют собой изображения, текстовые отчеты и последовательности измерений. Для их анализа применяются специализированные методы искусственного интеллекта.

1. Компьютерное зрение (Computer Vision)

Наиболее бурно развивающееся направление в АПК. Алгоритмы учатся «видеть» и интерпретировать визуальную информацию.

Задачи и применение:

Мониторинг состояния посевов: автоматический расчет вегетационных индексов (например, NDVI, NDRE) по мультиспектральным снимкам со спутников и дронов для оценки здоровья растений.

Диагностика болезней и вредителей: мобильное приложение, где агроном фотографирует лист, а нейросеть в реальном времени определяет вид заболевания или дефицита питательных веществ и дает рекомендации по лечению.

Счетчик растений и оценка густоты стояния: анализ изображений с дронов на ранних стадиях вегетации для оценки всхожести и принятия решения о подсеве.

Сегментация сорняков: создание карт распространения сорняков на поле. Это позволяет роботу-пропольщику или опрыскивателю с компьютерным зрением точечно уничтожать сорняки, экономя до 90 % гербицидов.

Оценка качества урожая: анализ изображений зерна на ленте комбайна или фруктов на сортировочной линии для автоматического определения повреждений, калибра и спелости.

Анализ текста (Text Mining / NLP)

В сельском хозяйстве генерируется множество неструктурированных текстовых данных: протоколы обследований, метеосводки, научные статьи, инструкции к препаратам, новости рынка.

Задачи и применение:

Извлечение сущностей: автоматическое структурирование данных из бумажных агрономических журналов. Например, система сама находит в тексте записи: «[Дата]: [Поле № 5], [обработано] [гербицидом «А»], [норма] – [0,5 л/га]» и заносит в цифровую базу данных.

Классификация запросов: автоматическая маршрутизация вопросов от агрономов в техподдержку компаний-поставщиков.

Анализ новостей: мониторинг новостей и соцсетей для прогнозирования цен на рынке сельхозпродукции на основе тональности публикаций.

Чат-боты и умные помощники: создание систем, которые на основе анализа огромного массива документации (ГОСТы, регламенты, научные базы данных) могут отвечать на сложные вопросы агрономов в диалоговом режиме.

3. Анализ временных рядов (Time Series Analysis)

Это основа прогнозирования в сельском хозяйстве. Практически все данные – это временные ряды: показания метеостанций, динамика вегетационных индексов, котировки на бирже, данные датчиков на животных.

Задачи и применение:

Прогнозирование урожайности: анализ временного ряда индекса NDVI за несколько лет в сочетании с погодными данными позволяет построить точную модель для предсказания yield.

Прогноз погоды: сверхкраткосрочный и краткосрочный прогноз для конкретного поля на основе исторических данных локальной метеостанции.

Предиктивная аналитика для животноводства: анализ данных с носимых датчиков коров (активность, температура, жвачка) позволяет выявить аномалии и предсказать начало болезни или наилучшее время для осеменения за несколько дней до появления видимых симптомов.

Прогнозирование рыночных цен: анализ временных рядов цен прошлых лет, объемов предложения для построения ценовых моделей.

Современное сельское хозяйство стало IT-индустрией, где побеждает тот, кто наиболее эффективно собирает и анализирует данные. Постановка строгих цифровых экспериментов – источник качественных данных. А инструменты искусственного интеллекта, такие как компьютерное зрение, анализ текста и временных рядов, – это фабрики по превращению этих данных в готовые управленческие решения и полностью автоматизированные процессы. Синтез этих подходов ведет к созданию «цифровых двойников» ферм – самообучающихся систем, способных автоматически управлять агропроизводством с минимальным вмешательством человека.

7.4. Принятие решений на основе данных. Функция потерь, байесовский и минимаксный подходы, метод последовательного анализа

Собранные данные и построенные модели сами по себе бесполезны, если они не приводят к обоснованному и экономически эффективному управленческому решению. Агроном ежедневно стоит перед выбором: обрабатывать поле или нет, поливать сегодня или завтра, какой гибрид выбрать. Теория статистических решений предоставляет формальный аппарат для making these choices оптимальным образом в условиях неопределенности.

Функция потерь (Loss Function) – Критерий оптимальности

Ключевая концепция – формализация цены ошибки. В сельском хозяйстве последствия неправильного решения – это прямые финансовые убытки (потеря урожая, напрасные затраты на ресурсы).

Функция потерь $L(a, \theta)$ – это функция, которая количественно оценивает ущерб от принятия действия a в том случае, если истинное состояние природы (погода, наличие болезни, реальная урожайность) равно θ.

Пример:

Решение (a): Опрыскивать поле фунгицидом против септориоза.

Состояние природы (θ):

θ_1: Вспышка болезни действительно произойдет.

θ_2: Вспышки болезни не будет.

Функция потерь:

L (не опрыскивать, θ_2) = Потеря 50 % урожая → ущерб 500 тыс. руб.

L (не опрыскивать, θ_1) = Ущерба нет → 0 руб.

L (опрыскивать, θ_1) = Затраты на обработку → ущерб 50 тыс. руб. (спасли урожай).

L (опрыскивать, θ_1) = Напрасные затраты на обработку → ущерб 50 тыс. руб.

Цель – выбрать действие *a*, которое минимизирует ожидаемые потери.

Подходы к принятию решений в условиях неопределенности

Байесовский подход (Bayesian Approach)

Этот подход требует наличия априорных вероятностей P(θ) наступления каждого состояния природы. Эти вероятности могут быть основаны на исторических данных, экспертных оценках или прогнозах моделей.

Байесовское решающее правило: выбрать действие *a*, для которого ожидаемая потеря (байесовский риск) минимальна.

$$R(a) = \Sigma\ [\ L(a, \theta\_i) * P(\theta\_i)\ ] \text{ для всех возможных } \theta\_i.$$

Продолжение примера. Допустим, модель прогнозирует вероятность вспышки септориоза $P(\theta_1) = 0{,}3$ (30 %). Следовательно, $P(\theta_1) = 0{,}7$.

Ожидаемые потери для действия «опрыскивать»:

R (опрыскивать) = *L* опрыскивать, θ_1 * 0,3 + *L* (опрыскивать, θ_1) *0,7 = 50 * 0,3 + 50 * 0,7 = 50 тыс. руб.

Ожидаемые потери для действия «не опрыскивать»:

R (не опрыскивать) = *L* (не опрыскивать, θ_2) * 0,3 + *L* (не опрыскивать, θ_2) * 0,7 = 500 * 0,3 + 0 * 0,7 = 150 тыс. руб.

Вывод. Байесовское решение – опрыскивать, так как оно минимизирует ожидаемые потери (50 тыс. руб. против 150 тыс. руб.).

Минимаксный подход (Minimax Approach)

Это консервативный, пессимистичный подход. Он применяется, когда невозможно оценить вероятности состояний природы (например, для нового вредителя нет исторических данных) или когда последствия ошибки катастрофичны. Принимающий решение стремится минимизировать максимально возможные потери.

Минимаксное решающее правило: для каждого действия a найти наихудший сценарий (максимальные потери), а затем выбрать действие с наименьшим значением этого наихудшего исхода.

Выбрать *a*, при котором min по *a* [max по θ *L*(*a*, θ)].

Продолжение примера:

Для действия «опрыскивать»: max потеря = 50 тыс. руб. (в любом случае мы тратим деньги).

Для действия «не опрыскивать»: max потеря = 500 тыс. руб. (если болезнь ударит).

Вывод. Минимаксное решение – опрыскивать, так как оно ограничивает максимально возможные потери суммой 50 тыс. руб. против 500 тыс. руб.

Метод последовательного принятия решений (Sequential Decision Making)

Сельское хозяйство – не единовременный акт, а длительный процесс, в котором решения принимаются по шагам на основе поступающей новой информации. Это основа адаптивного управления.

Суть: в момент времени t мы принимаем решение a_t, основанное на всей доступной к этому моменту информации I_t (погода, данные скаутинга, спутниковые снимки). В результате мы получаем новые данные $I\_\{t+1\}$ и пересматриваем наши оценки (например, апостериорные вероятности в байесовском подходе), чтобы принять решение $a\_\{t+1\}$.

Пример последовательности решений для управления орошением:

t_1 (начало сезона): на основе типа почвы и долгосрочного прогноза погоды (априорные данные) составляется предварительный план поливов.

t_2 (через неделю): получены новые данные: датчики влажности почвы показывают быстрое высыхание, а метеостанция не зафиксировала ожидавшихся осадков. Решение: внести коррективы в план, провести полив на 3 дня раньше запланированного.

t_3 (после полива): данные датчиков подтвердили, что влажность почвы вернулась к оптимальному уровню. Спутниковый снимок показывает хорошее состояние посевов. Решение: вернуться к мониторингу, следующий полив запланировать на основе прогноза испарения.

... и т. д.

Этот процесс является практической реализацией Байесовского обновления: априорные представления (план) постоянно обновляются новыми данными, что приводит к уточнению апостериорных вероятностей и, как следствие, к более точным решениям.

Теория принятия решений формализует работу агронома, переводя ее из области интуиции в область вычислений. Выбор между байесовским и минимаксным подходами зависит от доступности данных

и отношения к риску. Последовательный анализ позволяет гибко и адаптивно управлять агроценозом в течение всего сезона. Интеграция этих методов в современные платформы точного земледелия – это создание систем поддержки принятия решений (DSS – Decision Support System), которые действуют как «советник агронома», просчитывая экономические последствия каждого возможного выбора и рекомендуя оптимальную стратегию.

Исследование операций и задачи искусственного интеллекта в сельском хозяйстве

Завершающим этапом цифровизации АПК является переход от поддержки единичных решений к оптимальному управлению всей производственной системой в целом. Этот переход обеспечивается синтезом классических методов исследования операций и передовых задач искусственного интеллекта.

7.5. Исследование операций (Operations Research – *OR*)

Исследование операций – это дисциплина, занимающаяся разработкой и применением аналитических методов для принятия оптимальных решений. В сельском хозяйстве задачи *OR* часто связаны с ограниченными ресурсами (земля, техника, вода, финансы, время) и направлены на поиск лучшего способа их использования.

Ключевые классы задач и методы:

Задачи оптимизации:

Линейное и нелинейное программирование: оптимизация состава кормовых смесей для животноводства (минимизация стоимости при заданных питательных свойствах), расчет оптимального плана внесения удобрений (максимизация урожайности при ограниченном бюджете).

Пример: Целевая функция: минимизировать стоимость кормовой смеси. Ограничения: содержание протеина не менее 18 %, клетчатки не более 10 %. Переменные: доля каждого компонента (кукуруза, соевый шрот, жмых и т. д.).

Задачи распределения и назначения:

Транспортная задача: оптимальное прикрепление нескольких хозяйств к элеваторам или пунктам переработки с минимизацией логистических затрат.

Задача назначения: оптимальное распределение парков сельхозтехники по полям с учетом их производительности, типа почвы и сроков агротехнических окон.

Задачи планирования и упорядочения (Scheduling):

Составление графиков работ: оптимальное планирование последовательности полевых работ (вспашка, посев, опрыскивание, уборка) для всего предприятия с учетом прогноза погоды и доступности техники. Решается методами теории расписаний и симуляции.

Задачи управления запасами (Inventory Management):

Определение оптимального объема складских запасов семян, СЗР, запчастей и ГСМ, чтобы минимизировать затраты на хранение, но избежать простоя техники из-за их отсутствия (Модель управления запасами с фиксированным интервалом времени или фиксированным объемом заказа).

Задачи сетевого планирования:

Построение модели CPM (Critical Path Method) для управления крупными проектами, например, строительством животноводческого комплекса или реконструкцией мелиоративной системы. Позволяет выявить критические операции, от которых зависит общий срок выполнения всего проекта.

7.6. Задачи искусственного интеллекта (ИИ) в АПК

В то время как OR фокусируется на оптимальности, ИИ фокусируется на интеллектуальном анализе сложных данных и автономности.

Ключевые направления:

Представление знаний и экспертные системы.

Создание баз знаний, кодирующих опыт лучших агрономов и зоотехников. Экспертная система может играть роль консультанта, задавая уточняющие вопросы о состоянии поля (тип почвы, культура, фаза развития, симптомы проблемы) и выдавая диагностику и рекомендации по лечению.

Машинное обучение (ML) и интеллектуальный анализ данных (Data Mining).

Предсказательное моделирование: прогноз урожайности, цен на рынке, вспышек заболеваний на основе исторических данных (см. раздел 5.2).

Кластеризация: Автоматическое выявление однородных зон на поле по данным многолетней урожайности и почвенным картам для дифференцированного управления.

Аномалия-детектирование: Автоматическое обнаружение сбоев в работе доильного аппарата по данным датчиков или выявление больных животных в стаде по видеопотоку.

Нейросетевые и глубокое обучение (Deep Learning):

Обработка изображений (Computer Vision): Мониторинг состояния посевов, идентификация сорняков, оценка качества продукции (см. раздел 5.3).

Обработка естественного языка (NLP): Автоматизация анализа научной литературы, новостей рынка и нормативных документов.

Интеллектуальная робототехника:

Создание автономных агроботов для прополки, точечного внесения удобрений, сбора урожая фруктов и овощей. Эти роботы объединяют компьютерное зрение для навигации и идентификации объектов и машинное обучение для принятия решений.

Синтез OR и ИИ: Когнитивное управление агробизнесом:

Современный тренд – не противопоставление, а интеграция методов OR и ИИ для создания самообучающихся систем управления.

Гибридные системы: ИИ-алгоритм (например, нейросеть) прогнозирует урожайность на каждом поле. Это прогнозное значение передается в оптимизационную OR-модель, которая рассчитывает оптимальный план уборочной кампании, маршруты движения комбайнов и грузовиков, а также график загрузки зерносушилок и складов.

Обучение с подкреплением (Reinforcement Learning). Это область ИИ, которая напрямую решает задачи последовательного принятия решений. Агент (например, система управления поливом) учится методом проб и ошибок, какое действие (поливать/не поливать) в каком состоянии (влажность почвы, прогноз погоды, фаза растения) приводит к максимальному «вознаграждению» (максимальный урожай при минимальном расходе воды). Таким образом, система не просто оптимизирует разовый полив, а самостоятельно вырабатывает оптимальную долгосрочную стратегию.

Цифровые двойники (Digital Twins): Создание виртуальной копии всего предприятия (поля, фермы, логистической цепи), которая непрерывно обновляется данными с IoT-датчиков. На этом двойнике можно с помощью OR- и ИИ-методов проводить тысячи симуляций,

тестируя различные сценарии управления («что, если будет засуха?», «что, если цены на удобрения вырастут?») и, выбирая наилучший без риска для реального объекта.

Эволюция применения ИТ в сельском хозяйстве прошла путь от простой автоматизации учета к созданию сложных киберфизических систем. Исследование операций – математический аппарат для нахождения оптимальных решений в условиях ограниченности ресурсов. Искусственный интеллект – инструменты для извлечения знаний из данных и автономной работы в условиях неопределенности и изменчивой среды. Их синтез ведет к становлению агроинформатики как самостоятельной дисциплины и к революции в управлении агропроизводством, которая повышает его эффективность, устойчивость и предсказуемость на качественно новом уровне.

7.7. Экспертиза и неформальные процедуры в условиях цифровой трансформации сельского хозяйства

Несмотря на мощь формальных моделей, алгоритмов и систем искусственного интеллекта, процесс принятия решений в сельском хозяйстве никогда не будет полностью формализован. Это связано с уникальной сложностью, непредсказуемостью и открытостью агроэкосистем. Внезапные заморозки, новые штаммы патогенов, сложное социально-экономическое поведение рынка – все это создает ситуации, где строгие алгоритмы бессильны без человеческого опыта, интуиции и неформальных процедур.

Роль экспертизы (экспертного знания)

Экспертное знание в сельском хозяйстве – это глубокое, часто неявное (tacit) знание, накопленное в результате многолетнего личного опыта работы в конкретных условиях (регионе, хозяйстве, на конкретных полях).

Формы проявления экспертизы:

Интуитивное принятие решений: способность опытного агронома предчувствовать проблему (например, вспышку болезни) по едва уловимым признакам (оттенок цвета поля, вид росы по утрам, поведение насекомых), которые невозможно оцифровать и заложить в модель.

Работа с «мягкими» данными: умение интерпретировать неструктурированную информацию: разговоры с соседями-аграриями, новости о фитосанитарной обстановке в соседнем регионе, доверие к тому или иному поставщику семян.

Учет локального контекста: эксперт знает нюансы, неизвестные глобальным моделям: «этот низменный участок всегда вымокает весной», «на том склоне всегда вымерзает озимая», «это поле любит мой дед, он с ним сроднился».

Валидация и интерпретация данных моделей: эксперт выступает как «последняя инстанция». Алгоритм может выдать прогноз о высоком риске болезни, но агроном, зная, что сорт устойчив, а погода стоит сухая, может проигнорировать этот сигнал или скорректировать рекомендацию.

Неформальные процедуры принятия решений

Это устоявшиеся в сообществе практики, которые не описаны в официальных регламентах, но эффективно работают на местах.

Советы бывалых: передача знаний «из рук в руки», от старшего поколения агрономов младшему, в обход формальных инструкций.

Коллективное обсуждение проблем («полевые советы»): собрания агрономов, главных инженеров и директора в критический момент (перед уборкой, при угрозе заморозков), где решение вырабатывается в ходе дискуссии, а не спускается сверху алгоритмом.

«Правила большого пальца» (Heuristics): простые практические правила, выработанные опытным путем.

Пример: «Если за ночь выпало меньше 5 мм дождя, а днем обещают +25 °С, то поливать не стоит – вода не дойдет до корней».

Пример: «Если кукуруза на восходе солнца стоит с закрученными листьями, а к 10 утра расправляет их – влаги хватает. Если не расправляет – нужен полив».

Синтез формального и неформального: Со-интеллект (Collective Intelligence)

Современный подход заключается не в противопоставлении искусственного интеллекта и человеческого, а в их симбиозе.

Обучение моделей на основе экспертных знаний: процесс активного обучения (Active Learning), когда ИИ-модель, не уверенная в своих прогнозах, сама запрашивает разметку или оценку у эксперта-агронома. Тем самым она учится на его опыте.

Объяснимая искусственная интеллектуальная система (XAI – Explainable AI): модель не просто выдает ответ «обработать поле», но и показывает, на основании каких данных и признаков она приняла это решение («потому что на соседних полях уже зафиксирована болезнь, потому что влажность превысила пороговое значение X»). Это позволяет эксперту проверить «логику» алгоритма и либо довериться ему, либо отвергнуть его выводы на основе своего опыта.

Экспертные системы гибридного типа: системы, которые кодируют не только формальные правила («если влажность > Y, то…»), но и эвристики, полученные от лучших специалистов отрасли. Пользователь такой системы видит не только голую рекомендацию, но и ее обоснование на языке, понятном агроному: «Система рекомендует отложить посев на 2 дня, так как модель прогнозирует похолодание. Кроме того, Иван Петрович из соседнего хозяйства в подобной ситуации в прошлом году поступил так же и получил лучшую всхожесть».

Цифровая трансформация сельского хозяйства – это не слепая автоматизация и вытеснение человека. Это усиление человеческого интеллекта с помощью технологий. Формальные модели и алгоритмы берут на себя рутинную обработку больших данных, расчет вариантов и выявление скрытых паттернов. Человек-эксперт привносит в эту систему здравый смысл, интуицию, способность к обобщению в условиях неопределенности и ответственность за конечный результат.

Наиболее эффективные решения рождаются на стыке цифрового и аналогового, формального и неформального, глобального данных и локального контекста. Успех современного агробизнеса будет определяться не тем, кто соберет больше данных, а тем, кто сумеет построить наиболее эффективный диалог между человеком и машиной, где каждый делает то, что у него получается лучше всего. Таким образом, будущее АПК – это не «фермы без людей», а высокотехнологичные фермы, управляемые высококвалифицированными и технологически подкованными экспертами.

7.8. Автоматизация проектирования в сельском хозяйстве

Это использование специализированного программного обеспечения (САПР, CAD) для создания цифровых моделей и проектов объектов агропромышленного комплекса.

Ключевые применения:

Проектирование ферм и животноводческих комплексов: разработка планировки помещений, систем вентиляции, кормления и навозоудаления.

Проектирование мелиоративных систем и полей орошения: расчет уклонов, размера полей, проектирование каналов и трубопроводов.

Проектирование сельскохозяйственной техники: создание 3D-моделей и чертежей новых узлов и агрегатов.

Землеустройство: разработка проектов землепользования, севооборотов, размещения лесополос.

Связь с другими технологиями: современное проектирование тесно интегрировано с ГИС (для привязки к местности) и BIM (информационным моделированием зданий), что позволяет управлять всем жизненным циклом объекта – от проектирования до эксплуатации.

7.9. Искусственный интеллект и распознавание образов

Сельское хозяйство переживает цифровую революцию, и ключевыми драйверами этой трансформации являются искусственный интеллект (ИИ) и распознавание образов (Computer Vision). Вместе они создают основу для точного земледелия, делая его более эффективным, предсказуемым и устойчивым.

Как это работает?

Специальные алгоритмы машинного обучения, и особенно глубокого обучения (глубокие нейронные сети), обучаются на огромных массивах данных – тысячах и миллионах изображений. Алгоритм выявляет скрытые закономерности и учится классифицировать новые, ранее не виданные данные.

Источники данных: спутники: обеспечивают регулярный мониторинг больших площадей.

Беспилотники (дроны): дают данные высокого разрешения для детального анализа.

Даталогические датчики на технике: фиксируют данные об урожайности, состоянии почвы в реальном времени.

Стационарные камеры и роботы: используются в теплицах и на фермах.

Ключевые применения в растениеводстве:

1. Мониторинг состояния посевов

Алгоритмы анализируют мультиспектральные и гиперспектральные снимки для расчета вегетационных индексов (например, NDVI). Это позволяет:

выявлять участки стресса у растений (недостаток влаги, питательных веществ) еще до того, как это станет видно невооруженным глазом;

создавать карты неоднородности полей.

2. Диагностика болезней и вредителей

Системы компьютерного зрения на основе ИИ могут с высокой точностью определять:

заболевания растений (фитофтороз, мучнистая роса, ржавчина) по пятнам на листьях;

наличие и тип вредителей;

дефицит питательных элементов (азота, калия, фосфора) по изменению цвета листвы.

Практическое применение: мобильные приложения, где агроном просто фотографирует лист на смартфон и мгновенно получает диагноз и рекомендации по лечению.

3. Определение сорняков и точечная обработка (селективное внесение СЗР)

ИИ-системы, установленные на опрыскивателях или специальных роботах, в реальном времени:

различают культурное растение и сорняк;

принимают решение о точечном внесении гербицида только на сорняк.

Результат: экономия средств защиты растений (до 90 %), снижение пестицидной нагрузки на почву и окружающую среду.

4. Прогнозирование урожайности

Модели ИИ анализируют комплекс данных: историческую урожайность, погодные условия, состояние вегетации, данные с датчиков. Это позволяет с высокой точностью предсказать урожайность до начала уборочной кампании, что критически важно для логистики, хранения и продаж.

Ключевые применения в животноводстве:

1. мониторинг здоровья и поведения животных

Системы компьютерного зрения следят за каждым животным в стаде, анализируя:

Поведение: отслеживание активности, времени приема пищи и лежания. Снижение активности может сигнализировать о начале болезни.

Внешний вид: обнаружение хромоты, повреждений на теле, изменение кондиции тела.

Физиологические показатели: с помощью тепловизоров можно выявлять маститы на ранних стадиях.

2. Прецизионное животноводство

Оптимизация кормления: системы анализируют, сколько корма потребляет каждое животное, и автоматически подбирает рацион для максимизации продуктивности.

Выявление охоты: алгоритмы по поведенческим паттернам точно определяют оптимальное время для осеменения.

Преимущества и будущее

Повышение эффективности: оптимизация использования ресурсов (воды, удобрений, кормов, топлива).

Снижение затрат: роботизация и точечное внесение материалов экономят деньги.

Устойчивое развитие: минимизация воздействия на окружающую среду за счет сокращения химикатов.

Предсказуемость: возможность прогнозировать результаты и риски.

Будущее за полностью автономными фермами, где системы на основе ИИ будут самостоятельно принимать решения: роботы-агрономы будут высаживать, мониторить и убирать урожай, а системы машинного зрения на фермах – обеспечивать благополучие животных без постоянного вмешательства человека.

СПИСОК ЛИТЕРАТУРЫ

1. Статистические методы анализа и планирования эксперимента : пособие / Н. Г. Серебрякова, А. П. Мириленко. – Минск : БГАТУ, 2022. – 104 с.

2. Проектирование сельскохозяйственной техники. Курсовое проектирование : учебно-методическое пособие / сост.: П. В. Авраменко [и др.]. – Минск : БГАТУ, 2022. – 88 с.

3. Эргономика и дизайн. Практикум: учебно-методическое пособие / сост. : В. Н. Еднач [и др.]. – Минск : БГАТУ, 2023. – 140 с.

4. Эргономика производственных систем : пособие /сост. : Н. Г. Серебрякова, Т. В. Молош, Е. И. Подашевская. – Минск : БГАТУ, 2021.

5. Компьютерные информационные технологии : [электронный учебно-методический комплекс] / сост. : Н. Г. Серебрякова, А. П. Мириленко. – Минск : БГАТУ, 2025. – Текст : электронный.

6. Информатика : [электронный учебно-методический комплекс] /сост.: Н. Г. Серебрякова, А. П. Мириленко. – Минск : БГАТУ, 2025. – Текст : электронный.

7. Автоматизация инженерных расчетов при проектировании сельскохозяйственной техники : [электронный учебно-методический комплекс] / сост. : А. П. Мириленко, Н. Г. Серебрякова. – Минск : БГАТУ, 2024. – Текст : электронный.

8. Информационные технологии : [электронный учебно-методический комплекс] / сост. : Н. Г. Серебрякова [и др.]. – Минск : БГАТУ, 2020. – Текст : электронный.

9. Интеллектуальные технические системы в животноводстве : [электронный учебно-методический комплекс] / сост.: Н. Г. Серебрякова, А. П. Мириленко, Е. И. Подашевская. – Минск : БГАТУ, 2020. – Текст : электронный.

10. Серебрякова, Н. Г. Историко-философские истоки современной инженерной деятельности / Н. Г. Серебрякова, И. А. Серебряков // Современное профессиональное образование. – 2025. – № 6. – С. 115–121.

11. Серебрякова, Н. Г. Применение концептуально-ориентированных опорных конспектов в инженерном образовании / Н. Г. Серебрякова // Стандарты и мониторинг в образовании. – 2024. – Т. 12, № 5. – С. 27–35. – DOI 10.12737/1998-1740-2024-12-5-27-35.

Учебное издание

Серебрякова Наталья Григорьевна,
Сапун Оксана Леонидовна,
Мириленко Андрей Петрович

ОСНОВЫ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ

Пособие

Ответственный за выпуск *Н. Г. Серебрякова*
Редактор *Г. В. Анисимова*
Компьютерная верстка *В. Л. Невдах*
Дизайн обложки *Д. О. Михеевой*

Подписано в печать 29.12.2025. Формат 60×84 1/16.
Бумага офсетная. Ризография.
Усл. печ. л. 15,58. Уч.-изд. л. 12,18. Тираж 99 экз. Заказ 632.

Издатель и полиграфическое исполнение:
учреждение образования
«Белорусский государственный аграрный технический университет».
Свидетельство о государственной регистрации издателя, изготовителя,
распространителя печатных изданий
№ 1/359 от 09.06.2014.
№ 2/151 от 11.06.2014.
Пр-т Независимости, 99–1, 220012, Минск.