Дальневосточный государственный университет

Институт автоматики и процессов управления

КЛЕЩЁВ А.С.

ОСНОВЫ АНАЛИЗА И ФОРМАЛИЗАЦИИ ИНФОРМАЦИИ

Курс лекций

2005

**ВВЕДЕНИЕ**

На протяжении своей истории программирование последовательно столкнулось с тремя главными проблемами: кодирования, проектирования и анализа.

С проблемой кодирования программирование столкнулось в самом начале своего развития. Она состояла в том, что создание компьютерных программ оказалось весьма трудной задачей. Положение усугублялось тем, что в то время компьютеры имели память очень малого объема, а само кодирование могло выполняться только в цифровом коде. Успехи в трех направлениях способствовали решению этой проблемы: радикальное увеличение оперативной и внешней памяти компьютеров; развитие языков и систем программирования и других инструментальных средств; выделение этапа кодирования в самостоятельный процесс, отдельный от других этапов разработки программ.

С проблемой проектирования программирование столкнулось тогда, когда размер памяти компьютеров стал достаточным для разработки программ большого размера. Она состояла в том, что создание больших программ оказалось весьма трудной задачей. Положение усугублялось тем, что большие программы требовали для своего кодирования большого числа исполнителей, которые должны были руководствоваться каким-либо единым документом – проектом разрабатываемой программы. К этому добавилась необходимость сопровождать разработанные программы, причем сопровождение, как правило, должно было осуществляться не разработчиками, а специалистами организации, где программа использовалась. Это также требовало проектной документации. Наконец, решение вопросов защиты прав заказчиков и покупателей компьютерных программ потребовало создания стандартов на процесс разработки программ и сопровождающей их документации. Успехи в решении проблемы проектирования связаны с развитием специальной дисциплины – технологии программирования.

С проблемой анализа программирование столкнулось тогда, когда возникла потребность массовой разработки нестандартных компьютерных программ, отвечающих потребностям многих пользователей. Она состояла в том, что создание программ, полезных многим пользователям, оказалось весьма трудной задачей. Оказалось, что программа является тем полезней, чем больше востребованных ее пользователями знаний в нее заложено. Решение проблемы анализа потребовало создания специальной дисциплины – аналитики, основной задачей которой является выделение всех составных частей системы знаний, необходимых для разработки той или иной наукоемкой программы, а также представление их в таком виде, чтобы их можно было использовать при проектировании и кодировании программы, т.е. их математизация и формализация. Настоящий курс посвящен основам анализа и формализации информации.

1. ИНФОРМАЦИЯ

**Информатика** – это область деятельности, связанная с передачей, хранением и обработкой информации. Поэтому, прежде всего, необходимо договориться о том, что далее будет обозначаться термином "информация".

1.1. Информация о материальном мире

Мы привыкли считать, что мы живем в материальном мире, который воздействует на наши органы чувств, в результате чего мы получаем достоверную информацию об этом материальном мире, и поэтому хорошо знаем, как он устроен. Рассмотрим на примере органа слуха, какую информацию о материальном мире мы получаем с его помощью.

Как известно из акустики, звук представляет собой колебания давления воздуха в диапазоне от нескольких герц до 20 килогерц. Звуковые волны распространяются в воздушной среде и взаимодействуют с органом слуха, точнее с базиллярной мембраной. Последняя, как известно из физиологии органов слуха, имеет резонансные частоты в том же диапазоне, причем резонансная частота базиллярной мембраны плавно меняется вдоль ее длины. При взаимодействии звуковых волн с базиллярной мембраной последняя начинает колебаться, причем амплитуда этих колебаний различна в разных точках мембраны. В результате амплитуды вдоль длины мембраны соответствуют спектру звуковых волн, изменяющемуся во времени. Из нейрофизиологии известно, что по площади мембраны расположены рецепторы, которые преобразуют амплитуду колебаний мембраны в электрические сигналы. Из электрофизиологии известно, что каждый сигнал представляет собой последовательность импульсов одинаковой формы, причем интервалы между импульсами зависят от амплитуды колебаний мембраны в точке расположения рецептора. Эти электрические сигналы передаются по нервным волокнам в мозг, где происходит их дальнейшая обработка. Такова упрощенная схема начальных этапов восприятия звука человеком.

Теперь зададимся вопросом, что же воспринимает человек с помощью органа слуха? Никто без специальной подготовки (в области акустики или электрофизиологии) не может нарисовать график звуковых волн, которые он слышит, график изменения спектра этих волн во времени или вид электрического сигнала, формируемого некоторым рецептором. Никто (за исключением особо выдающихся мнемонистов) не может буквально (как магнитофон) воспроизвести часовую лекцию. Однако человек, который внимательно слушал эту лекцию, знает язык, на котором она была прочитана, и имеет необходимую подготовку в той области знаний, к которой относится тематика лекции, сможет своими словами пересказать основные идеи этой лекции. Если же хотя бы одно из этих условий не выполнено, пересказ становится невозможным или неудовлетворительным. Этот пример показывает, что при воздействии речевого сигнала на орган слуха человек может при определенных условиях воспринять некоторые идеи, причем эти идеи либо совпадают с теми, которые уже были известны этому человеку, либо сконструированы из таких идей. Эти идеи могут быть вербальными (имеющими языковую форму) или невербальными (ощущениями) – например, ощущениями тембра и громкости голоса.

Итак, мы приходим к следующей общей схеме восприятия материального мира человеком с помощью органов чувств – при воздействии материального мира на органы чувств человек воспринимает некоторые идеи, которые некоторым (возможно весьма сложным) образом связаны с исходным воздействием и идеями, уже известными этому человеку. Проведенный умозрительный анализ позволяет дать следующее толкование термина "информация". **Информация** – это идеи, которые могут находиться только в сознании человека.

1.2. Прикладные задачи

С помощью органов чувств мы можем воспринимать информацию только об очень ограниченной части материального мира в настоящем, знаем только часть информации о прошлом и не знаем информации о будущем. Однако повседневная жизнь заставляет нас задавать себе и другим вопросы о неизвестных нам аспектах материального мира, ответами на которые являются некоторые идеи, т.е. некоторая информация. Таким образом, мы постоянно испытываем потребность в информации.

Каждый работающий человек принимает участие в той или иной **профессиональной деятельности**. Если основным результатом такой деятельности является изменение материального мира (создание материальных вещей, энергии и т.п.), то будем называть такую деятельность **материальной**. Если же основным результатом такой деятельности являются идеи, то будем называть ее **информационной**. Любая материальная деятельность включает в себя и некоторую информационную. Таким образом, значительная (и постоянно растущая) часть работающего населения занята в информационной деятельности.

Любая информационная деятельность есть некоторая система решения взаимосвязанных **задач**, называемых **прикладными**. Мы будем различать задачи, связанные с передачей, хранением и обработкой информации. Сами эти термины будут уточнены ниже. При передаче информации одному лицу требуются некоторые идеи, известные другому лицу; необходимо сделать их известными тому, кому они требуются. При хранении информации в будущем могут потребоваться некоторые идеи, известные в настоящем; необходимо сделать их доступными в будущем. При обработке информации требуются новые идеи, которые определенным образом связаны с известными идеями; необходимо найти эти новые идеи.

Прикладные задачи, связанные с передачей и хранением информации, состоят в перемещении известных идей в пространстве и времени. Прикладные задачи, связанные с обработкой информации, состоят в поиске неизвестных идей на основе известных.

1.3. Механизмы формирования идей

Выше было отмечено, что при восприятии материального мира человеком с помощью органов чувств в его сознании формируются идеи, связанные некоторым образом с ранее известными этому человеку идеями. Укажем основные механизмы возникновения этих ранее известных идей. Во-первых, это врожденная способность к запоминанию и обобщению ощущений. Например, увидев некоторого человека, мы узнаем его при дальнейших встречах, даже если он по-другому одет, изменил прическу и постарел. Во-вторых, это взаимодействие с окружающими в повседневной жизни, в результате чего связывается вместе вербальная и невербальная информация, и формируются общие для этого окружения восприятие материального мира и совокупность отвлеченных идей. В-третьих, общее и профессиональное обучение дает человеку запас профессиональных идей. Наконец, наука с помощью экспериментов, наблюдений и сознательного обобщения формирует научную картину материального мира и обнаруживает законы его функционирования. Каждый человек располагает своим запасом вербальных и невербальных идей, что оказывает влияние на его индивидуальное восприятие материального мира.

При решении задач используются иные механизмы возникновения идей. Искомая в задаче информация может быть получена в результате **наблюдения, эксперимента или измерения** с помощью приборов. Такое получение информации называется **эмпирическим познанием**. Однако в задачах, связанных с обработкой информации, основными механизмами получения информации являются инсайт и рассуждение. **Инсайт** (озарение, внутреннее видение) – это угадывание информации, необходимой в задаче. Любая творческая задача решается только с помощью инсайта (иначе она не называется творческой). Инсайт происходит мгновенно и поэтому не поддается изучению. Получение информации с помощью инсайта называется **интуитивным познанием**. **Рассуждение** – это последовательность шагов, на каждом из которых по некоторой известной к этому шагу информации формируется некоторая новая информация. Если на последнем шаге рассуждения формируется информация, необходимая в задаче, то мы говорим, что решили задачу с помощью этого рассуждения. На каждом шаге рассуждения используется некоторая идея о взаимосвязи идей (например, идеи о причинной связи событий, законах природы и общества ит.п.), и информация, известная к этому шагу. В результате выполнения шага получается некоторая новая информация. Поэтому рассуждения как последовательности идей могут быть предметом изучения. Получение информации с помощью рассуждений называется **рациональным познанием**.

1.4. Формальные рассуждения

Основным свойством информации является ее правильность. Информация является **правильной**, если она соответствует материальному миру. Если информация получена с помощью органов чувств, то она может быть неправильной из-за иллюзии (были восприняты не те идеи, которые соответствуют воздействию материального мира на органы чувств). Если информация получена с помощью приборов, то она может быть неправильной из-за неисправности приборов или неправильного их применения. Если информация получена с помощью инсайта, то она может быть неправильной из-за ненадежности механизма инсайта. Наконец, если информация получена с помощью рассуждения, то она может быть неправильной, если рассуждение неправильно. Если информация является неправильной и на ее основе принимаются некоторые (даже правильные) решения о действиях в материальном мире, то результаты этих действий могут быть далеки от ожидаемых. Поэтому для любой деятельности (повседневной и профессиональной) важно, чтобы информация, участвующая в этой деятельности, была правильной.

Иллюзия, неисправность приборов и неправильное их применение встречаются достаточно редко и не представляют для нас интереса. Вера в правильность идей, полученных с помощью инсайта, относится либо к области интуитивного познания, либо мистики. Такая идея может, в свою очередь, вызывать инсайт, что типично для идей, относящихся к искусству, либо быть пророчеством, тогда вера в ее правильность связана с авторитетом лица, выдвинувшего эту идею (пророка). Долгое время определение правильности рассуждений также относилось к области интуитивного познания (такие рассуждения получили название интуитивных). Аристотель поставил следующий вопрос – можно ли среди всех возможных рассуждений выделить такие, правильность которых можно было бы установить методами рационального познания, основываясь на их форме, а не на содержании? Такие рассуждения получили название **формальных**.

Разработав правила логики, Аристотель впервые ввел в человеческую практику полуформальные рассуждения. Рассуждение называется **полуформальным**, если методами рационального познания невозможно установить его правильность, но можно установить его неправильность. Этот тип рассуждений нашел свое первое применение в юриспруденции. Полуформальные рассуждения, основанные на логике Аристотеля, используются, как правило, для обоснования правильности идей, полученных с помощью инсайта или иным путем. В судебном заседании обвинитель утверждает виновность обвиняемого, а адвокат – его невиновность. И тот, и другой должны провести рассуждение, обосновывающее правильность соответствующей идеи о виновности или невиновности. Судья должен следить за правильностью их рассуждений. Если он сочтет какое-либо из этих рассуждений неправильным, он должен отвергнуть соответствующую идею.

Решение задач в ходе профессиональной деятельности с помощью правильных формальных рассуждений позволяет получать правильные результаты, а с помощью полуформальных - обнаруживать неправильность выдвинутых идей.

1.5. Профессиональные идеи

Идеи, так или иначе связанные с какой-нибудь профессиональной деятельностью, будем называть **профессиональными**. Остальные идеи будем называть обыденными. Профессиональные идеи будут основным предметом изучения в этом курсе.

В множестве всех профессиональных идей можно выделить подмножества, которые называются **предметными областями**. Каждая профессиональная идея относится к некоторой предметной области. Сложная предметная область может содержать идеи из разных предметных областей.

Каждая профессиональная деятельность связана с некоторой предметной областью в том смысле, что все идеи, связанные с этой профессиональной деятельностью, относятся к этой предметной области. Однако с одной и той же предметной областью обычно связано несколько видов профессиональной деятельности – производственной, образовательной, научной. Все эти виды деятельности могут использовать общие идеи, относящиеся к этой предметной области. Таким образом, предметная область не определяет профессиональную деятельность, связанную с этой предметной областью, однозначно.

Главной характеристикой предметной области является используемая в ней совокупность терминов - **терминология**. **Термин** – это слово или словосочетание, имеющее в предметной области специальный смысл (часто отличный от общепринятого, если это слово или словосочетание используется в обычной жизни). Терминология однозначно характеризует предметную область. Понимание профессиональных идей, относящихся к предметной области, без понимания смысла ее терминологии невозможно.

Другой характеристикой предметной области может быть ее **профессиональный** **диалект** – язык, на котором выражаются профессиональные идеи, относящиеся к этой предметной области. Диалект, по сравнению с обычным языком, содержит специальные обороты, имеющие в предметной области специальный смысл.

Наконец, предметная область может характеризоваться **специальными обозначениями**. Обычно специальные обозначения являются **формальными**, т.е. имеются точные правила построения этих специальных обозначений, а также правила извлечения из них смысла. Формальные обозначения позволяют исключить неоднозначность в понимании идей, представленных с их помощью. Примером специальных обозначений могут служить химические формулы. Часто для формальных специальных обозначений определяются и правила их **формальных преобразований**, отличные от правил логики, т.е. такие формальные рассуждения, которые позволяют по идеям, заданным с помощью специальных обозначений, получать новые идеи в тех же обозначениях, связанные с исходными идеями определенным образом.

1.6. Выводы

Информация – это идеи в сознании людей. Задача – это запрос на получение информации. Рассуждение – это рациональный метод решения задачи. Формальные рассуждения позволяют установить правильность получаемой с их помощью информации рациональными методами. Профессиональные идеи – это идеи, связанные с той или иной профессиональной деятельностью.

2. МАТЕМАТИКА

Математика является предметной областью, с которой связана научная и образовательная профессиональная деятельность. В то же время математика предлагает новые по сравнению с логикой, правильные формальные рассуждения, которые можно использовать при решении задач. Производственная деятельность, связанная с математикой, состоит в том, чтобы сделать эти правильные формальные рассуждения применимыми для профессиональной деятельности в разных предметных областях, отличных от математики. Поэтому связь математики и информатики нуждается в более подробном рассмотрении.

2.1. Теоретическая математика

Теоретическая или чистая математика изучает точные, абстрактные и отвлеченные идеи. Идея является **точной**, если для нее дано определение и существует рациональный способ проверки соответствия идеи этому определению. Идея является **абстрактной**, если она не содержит никаких деталей, кроме своей сути. Наконец, идея является **отвлеченной**, если она не относится к материальному миру. Идеи, изучаемые в теоретической математике, как правило, выдвигаются выдающимися математиками. Важность идеи существенно зависит от авторитета того, кто ее выдвинул. В теоретической математике существует два способа выдвижения точной идеи – с помощью системы аксиом и с помощью определения.

С помощью систем аксиом вводятся первичные идеи, обозначаемые неопределяемыми терминами. **Неопределяемый термин** обозначает идею, которая обладает всеми свойствами, выраженными с помощью системы аксиом. В классических аксиоматических системах первичные идеи являются абстракциями идей, относящихся к материальному миру. В таких системах аксиома считается правильной, потому что каждый, кто понимает аксиому, интуитивно понимает, что она правильна и в применении к соответствующим идеям, относящимся к материальному миру. В современных аксиоматических системах первичные идеи являются отвлеченными. В таких системах аксиома правильна, потому что так договорились математики.

С помощью определений вводятся производные идеи, обозначаемые определяемыми терминами. **Определяемый термин** обозначает идею, которая соответствует определению этого термина. Определение может вводить новую идею, используя уже определенную идею, обозначенную неопределяемым или определяемым термином и наделяя ее новыми свойствами, явно сформулированными в определении. Возможен и другой случай, когда новая идея определяется как некоторая композиция уже определенных идей. Определение всегда считается правильным.

**Математический мир** состоит из идей (математических объектов), обозначенных математическими терминами (определяемыми и неопределяемыми). Таким образом, математический мир существует только в сознании людей, имеющих математическое образование, и дан им через свое описание – системы аксиом и определений. Описание математического мира задает его единственным образом. Остальные утверждения о свойствах математического мира, существующие в теоретической математике, являются теоремами и проблемами. **Теорема** – это утверждение о свойствах математического мира, про которое известно, что оно правильно. Последнее означает, что, обозрев весь математический мир, математик непосредственно убеждается, что утверждение теоремы правильно в каждом ее частном случае, т.е. относительно всех математических объектов, к которым это утверждение оказывается применимым. Таким образом, теорема не входит в описание математического мира и не добавляет новых свойств этому миру. Она является догадкой математиков о свойствах математического мира, недоступных их взору, получается с помощью инсайта и лишь явно утверждает то, что неявно содержится в аксиомах и определениях. **Проблема** – это утверждение о свойствах математического мира, про которое неизвестно, правильно оно или нет. Проблемы обычно формулируются выдающимися математиками, а затем остальные пытаются установить, является проблема теоремой или нет.

Поскольку математический мир содержит бесконечное множество математических объектов, обозреть его весь практически невозможно. Поэтому в современной математической практике правильность утверждений о свойствах математического мира устанавливается с помощью специальных рассуждений, которые получили название интуитивных доказательств. Утверждение о свойствах математического мира считается теоремой, если для него придумано интуитивное доказательство. **Интуитивное доказательство** является полуформальным рассуждением. Как полуформальное рассуждение интуитивное доказательство должно удовлетворять определенным правилам, принятым в математике, в том числе и правилам логики Аристотеля. Если какое-либо из этих правил нарушено, то интуитивное доказательство считается неправильным. С другой стороны в интуитивное доказательство могут входить и содержательные рассуждения, правильность которых устанавливается методами интуитивного познания. Правильность всего интуитивного доказательства может быть установлена только методами интуитивного познания. Наоборот, неправильность такого доказательства может быть установлена методами рационального познания, либо с помощью логических (обнаружение нарушения правил рассуждения или неправильного их применения), либо содержательных (построение опровергающего примера) рассуждений.

Описания неопределяемых терминов, аксиомы, определения, теоремы, проблемы и интуитивные доказательства в математической практике представляются на **математическом диалекте**. Математический диалект является расширяемым языком. Новые термины вводятся с помощью описаний и определений. Понимание специальных оборотов математического диалекта базируется на знании математиками естественного языка (в учебниках математики смысл основных оборотов разъясняется для новичков).

Кроме того, математический диалект позволяет вводить в него специальные обозначения и способы формальной записи математических идей (математические выражения). **Специальные обозначения** могут быть локальными (действующими в некоторой части математического описания) и глобальными (действующими во всей математике). Локальное обозначение, как правило, обозначает произвольную математическую идею, обладающую указанными при введении этого обозначения свойствами. Глобальное обозначение играет ту же роль, что и математический термин. Способы **формальной записи математических идей** образуют формальный подъязык математического диалекта. Для таких способов записи на математическом диалекте вводятся точные правила построения математических выражений и извлечения из них смысла (позволяющие выразить те же идеи на математическом диалекте без использования этих способов записи).

Для математических выражений на математическом диалекте вводятся правила их формального преобразования. **Формальные преобразования** – это формальные рассуждения, позволяющие по одним математическим выражениям получать другие математические выражения, связанные с исходными определенным образом. Наиболее важным классом формальных преобразований являются **эквивалентные преобразования**, которые позволяют получать разные формальные представления одних и тех же идей.

Как всякая предметная область, теоретическая математика рассматривает и задачи, причем все они относятся только к классу задач обработки информации. Формулировка математической задачи должна обладать следующим свойством: существует общий метод, позволяющий проверить для любых допустимых входных данных задачи, что некоторые данные являются ее выходными данными. Относительно этих задач теоретическая математика ставит ряд стандартных вопросов (проводит **исследование** этих **задач**), ответы на которые должны иметь форму доказанных теорем. Основные из этих вопросов – имеет ли задача решение (ответ - теорема о существовании решения задачи); если имеет, то сколько их (ответ - теорема о единственности решения задачи, или о мощности множества решений задачи); может ли способ решения задачи быть выражен математическим выражением; если да, то каким. Если ответ на последний вопрос найден, то это математическое выражение определяет правильное формальное рассуждение, с помощью которого может быть решена эта задача. Типичным примером таких задач являются задачи решения различных типов уравнений.

Дальнейшая возможность расширения применения формальных рассуждений в теоретической математике связана с возникновением и развитием математической логики. В математической логике была построена **математическая модель математической практики** – исчисление предикатов первого порядка. В ней в качестве моделей математического мира рассматриваются алгебраические системы, в качестве модели математического диалекта – язык исчисления предикатов первого порядка, в качестве моделей описания математического мира – логические теории (множества нелогических аксиом), в качестве модели способов рассуждения – логические аксиомы и правила вывода исчисления предикатов первого порядка, в качестве моделей полного доказательства теорем – формальные доказательства в исчислении предикатов первого порядка.

**Язык исчисления предикатов первого порядка** является формальным, но нерасширяемым языком – он имеет точные правила синтаксиса (построения логических теорий) и семантики (вычисления значений термов и формул). Однако множество неопределяемых понятий, выразимых на этом языке, фиксировано при определении языка. Этими неопределяемыми понятиями являются множество (задается неявно), объект (элемент этого множества, задается неявно), функция на множестве (задается неявно, число ее аргументов также задается неявно), предикат на множестве (задается неявно, число его аргументов также задается неявно), математический термин (неявно задается и связывается со значением), пропозициональные связки и логические кванторы, переменная (вводится с помощью квантора), аппликация функции и предиката.

Прагматика (способ применения) языка исчисления предикатов первого порядка позволяет формализовать (представлять в виде термов и формул языка) математические идеи. **Алгебраическая система** моделирует математический мир и определяет смысл (интерпретацию) математических терминов, множество которых (сигнатура) входит в логическую теорию. Сама **логическая теория** определяет множество своих **моделей**, т.е. таких алгебраических систем, относительно которых истинны все нелогические аксиомы этой теории. Таким образом, логическая теория (модель описания математического мира) может задавать несколько (как правило, бесконечно много) моделей этого мира. Предложение языка исчисления предикатов первого порядка называется **теоремой** логической теории, если оно истинно во всех моделях этой теории.

**Исчисление предикатов первого порядка** содержит два конечных множества - логических аксиом и правил вывода, представленных формально на некотором метаязыке. Только эти логические аксиомы и правила вывода можно использовать при построении формального доказательства. Само **формальное доказательство** есть конечная последовательность предложений, каждое из которых является либо нелогической аксиомой, либо результатом применения логической аксиомы, либо значением заключения правила вывода, значения посылок которого предшествуют этому предложению в доказательстве. Таким образом, формальное доказательство является моделью полностью формального рассуждения, поскольку его правильность может быть проверена формальными методами (на основе определения формального доказательства). Теорема Геделя о полноте исчисления предикатов первого порядка утверждает, что для любой теоремы существует формальное доказательство, и любое предложение языка, для которого существует формальное доказательство, является теоремой.

Применение исчисления предикатов первого порядка в математической практике могло бы состоять в переводе описания математического мира и математических знаний (теорем) на язык исчисления предикатов и в построении формальных доказательств по интуитивным. Несмотря на то, что формальное доказательство допускает проверку его правильности формальными методами, а интуитивное доказательство не допускает такой проверки, формальные доказательства не нашли применения в математической практике, а применение формальных рассуждений в теоретической математике ограничивается формальными преобразованиями математических выражений и интуитивными доказательствами (полуформальными рассуждениями). Основные причины этого состоят в бедности языка исчисления предикатов первого порядка и в бедности самого исчисления. Бедность языка ведет к тому, что математические идеи, сравнительно просто выражаемые на математическом диалекте, на языке исчисления предикатов часто выражаются громоздкими конструкциями. Бедность исчисления ведет к тому, что формальные доказательства по размеру значительно превосходят интуитивные и существенно отличаются от них в содержательном плане.

2.2. Прикладная математика

Прикладная математика изучает математическими методами точные и абстрактные (математические) идеи, являющиеся абстракциями профессиональных идей. **Метод прикладной математики** состоит в замене прикладных задач похожими на них математическими задачами, чтобы затем исследовать эти математические задачи математическими методами.

При такой замене прикладной задачи похожей на нее математической задачей собираются все идеи, относящиеся к прикладной задаче, и для каждой такой идеи ищется похожая на нее, математическая идея. Сходство состоит в том, что математическая идея является абстракцией профессиональной идеи. Этот процесс перехода от профессиональных идей к математическим будем называть **математизацией идей**.

При математизации идей устанавливается соответствие между терминологией предметной области, использованной при описании математизируемых профессиональных идей, и математической терминологией. Каждому термину предметной области ставится в соответствие некоторый математический объект или класс математических объектов в качестве значения этого термина, а профессиональные идеи формулируются с помощью этих терминов на математическом диалекте. Примерами предметных областей, где метод прикладной математики давно и с успехом применяется, являются физика и техника.

Математизация идей ведет к построению описания **математизированного мира**, похожего на представления о материальном мире, принятые в предметной области. Математизированный мир строится из математических объектов математического мира, введенного в теоретической математике. Свойства этих объектов уточняются в описании математизированного мира. Тексты на профессиональном диалекте служат поясняющими комментариями к описанию математизированного мира на математическом диалекте. В некоторых предметных областях (например, в физике и технике) специалисты предметной области имеют достаточное математическое образование, чтобы самим строить и использовать описание математизированного мира. В других предметных областях использование метода прикладной математики требует совместной работы экспертов предметной области и прикладных математиков.

После того, как описание математизированного мира построено, математическая постановка прикладной задачи формулируется на математическом диалекте с помощью тех же терминов в виде математической задачи. Исследование задач прикладной математики проводится так же, как и исследование задач теоретической математики. При этом в прикладной математике формальные методы и рассуждения используются в тех же случаях, что и в теоретической.

Если найден метод решения математической задачи, то решение прикладной задачи состоит в математизации исходных данных прикладной задачи, решении математической задачи с помощью этого метода и в **дематематизации** результата решения математической задачи. Последнее означает, что для математической идеи, являющейся результатом решения математической задачи, находится похожая на нее профессиональная идея, которая и рассматривается как результат решения прикладной задачи. Если каждый раз оказывается, что результат решения прикладной задачи, полученный таким образом, является правильным (соответствует материальному миру), то математизация считается **адекватной**. Таким образом, математизация профессиональных идей и задач является частью профессиональной деятельности в предметных областях, отличных от математики.

2.3. Вычислительная математика

Выше было отмечено, что единственной формой, в которой рассматриваются методы решения математических задач в теоретической и прикладной математике, являются математические выражения. Однако метод решения не всякой математической задачи может быть представлен в такой форме. Вычислительная математика изучает методы решения математических задач прикладной математики в иной форме – в форме алгоритмов.

**Алгоритм** есть такое описание некоторого класса рассуждений на математическом диалекте, что по этому описанию математик может единственным образом выполнить эти рассуждения для любых допустимых для этого алгоритма входных данных. Если входные данные алгоритма совпадают с входными данными некоторой математической задачи, а его выходные данные – с выходными данными этой задачи, то алгоритм может рассматриваться как описание метода решения этой математической задачи.

Относительно алгоритмов вычислительная математика ставит ряд стандартных вопросов (проводит **исследование алгоритмов**), ответы на которые должны иметь форму доказанных теорем. Основные из этих вопросов - при любых ли допустимых исходных данных рассуждения, описываемые алгоритмом, заканчиваются за конечное число шагов (ответ - теорема об остановке алгоритма); при любых ли допустимых исходных данных результат работы алгоритма есть результат решения задачи для этих же входных данных (ответ - теорема о **корректности алгоритма**); как зависит число шагов рассуждения, описываемого алгоритмом, от его исходных данных (ответ - теорема о сложности вычислений). Использование только корректных алгоритмов является обязательным условием адекватной математизации при решении прикладных задач.

Если алгоритм является корректным относительно некоторой математической задачи, то все рассуждения, описываемые этим алгоритмом, являются правильными и формальными. Это означает, что в этом случае математик может не думать о том, как решать задачу; если он будет правильно выполнять рассуждения, описываемые алгоритмом, он всегда получит правильный результат. Если же он ошибся при выполнении рассуждений, описываемых алгоритмом, но протоколировал свои рассуждения, он имеет возможность найти тот шаг рассуждений, на котором он совершил ошибку, и исправить ее. Итак, вычислительная математика разделяет проблему использования формальных рассуждений при решении математических задач на две части – теорема о корректности алгоритма обосновывается с помощью интуитивного доказательства; правильность выполнения рассуждений, описываемых алгоритмом, может быть проверена с помощью протоколирования этих рассуждений.

2.4. Выводы

Теоретическая математика устанавливает свойства отвлеченных математических идей с помощью инсайта и подтверждает их правильность с помощью интуитивных доказательств. Прикладная математика предлагает метод математизации профессиональных идей и прикладных задач, и исследование их математическими методами как часть профессиональной деятельности. Вычислительная математика в качестве методов решения задач прикладной математики вводит и исследует математическими методами алгоритмы – классы правильных формальных рассуждений, используемые в профессиональной деятельности.

3. ИНФОРМАЦИОННЫЕ СИСТЕМЫ

3.1. Программирование

**Компьютер** – это техническое устройство для выполнения **программ** – алгоритмов, записанных на **языке программирования**. Такой язык всегда является формальным. Если алгоритм, записанный на математическом диалекте, может выполнить только математик, то программу может выполнить как программист (человек, который знает формальный язык, на котором написана программа), так и компьютер. Если математик может ошибиться при выполнении алгоритма, а программист – при выполнении программы, то, если компьютер исправен, он всегда выполнит программу правильно. Другим преимуществом компьютера является высокая скорость выполнения программ. Хотя при выполнении программы в компьютере происходят некоторые электронные и механические процессы, программисты считают, что компьютеры выполняют рассуждения, определяемые программами. Мы будем стоять на той же точке зрения. Если программа представляет алгоритм решения прикладной задачи, то такая **программа** называется **прикладной**, а область деятельности, связанная с разработкой таких программ, называется **прикладным программированием**.

Прикладные программы, как правило, записываются на **языках программирования высокого уровня**. Такой язык, как и любой язык программирования, является формальным, а его операционная семантика (правила выполнения программ) описывается на математическом диалекте в терминах его грамматики и вычислительного процесса. Программа, написанная на языке программирования высокого уровня, как правило, не может быть выполнена компьютером непосредственно, однако для таких языков существуют специальные программы (системы программирования), с помощью которых компьютеры могут выполнять такие программы.

**Программирование** можно рассматривать как деятельность по переводу алгоритмов в программы, т.е. по переводу алгоритмов с математического диалекта на язык программирования. Чем выше уровень языка программирования, тем более программа на этом языке похожа на исходный алгоритм.

**Правильность программы** может быть установлена лишь методами интуитивного познания в результате сопоставления программы и алгоритма и понимания того, что соответствующие части алгоритма и программы определяют одни и те же рассуждения. **Неправильность программы** может быть установлена методами рационального познания с использованием постановки математической задачи, метод решения которой описывает алгоритм. Установление неправильности программы называется ее **испытаниями**. Если результат выполнения программы при некоторых входных данных не совпадает с выходными данными этой математической задачи при этих входных данных, то программа считается неправильной. Если в результате испытаний неправильность программы не установлена, то она считается правильной до тех пор, пока ее неправильность не будет установлена в ходе ее использования (продолжающихся испытаний). Таким образом, для обеспечения возможности испытаний программы необходима математическая постановка задачи, алгоритм решения которой эта программа представляет.

3.2. Информационная система

**Информационная система** – это программная система (взаимосвязанная совокупность оборудования, прикладных программ, информационных компонент и пользователей или операторов), предназначенная для автоматизации некоторой профессиональной деятельности. Основными характеристиками информационной системы, определяющими ее свойства, являются функциональность и качество.

**Функциональность** – это совокупность постановок прикладных задач (передачи, хранения и обработки информации) профессиональной деятельности, которые решаются с помощью прикладных программ, входящих в информационную систему, а также совокупность описаний служебных функций. Все постановки задач обработки информации профессиональной деятельности должны быть математизированы для обеспечения возможности исследования этих математических задач, разработки и исследования алгоритмов их решения, а также проведения испытаний программ, представляющих эти алгоритмы. Функциональность информационной системы определяется в результате **анализа** автоматизируемой профессиональной деятельности. Кроме прикладных задач профессиональной деятельности, которые определяют **основные** функции информационной системы, функциональность обычно включает и ряд **служебных** функций – задач, решение которых необходимо для обеспечения требуемого качества информационной системы.

**Качество** – это набор свойств информационной системы, не зависящих от автоматизируемой профессиональной деятельности. Эти свойства определяются международными и национальными стандартами. Примерами таких свойств являются надежность, простота использования, эффективность, легкость сопровождения и перенесения на другое оборудование. Качество программ изучается в курсе «Технология программирования» и других курсах технологического цикла. Качество информационной системы всегда является компромиссом между потребностями заказчика (покупателей) и его (их) платежеспособностью. Оно определяется в результате сбора и анализа требований заказчика (покупателей).

Результаты анализа профессиональной деятельности и требований заказчика (покупателей) являются исчерпывающей информацией для проектирования информационной системы.

3.3. Анализ профессиональной деятельности

Анализ профессиональной деятельности выполняется аналитиком с помощью экспертов предметной области. Он включает три вида работ: систематизацию профессиональных идей, их перевод на язык, понятный аналитику, математизацию и формализацию этих идей.

**Систематизация** – это составление полного списка и классификация элементов этого списка. **Систематизация профессиональных идей** включает систематизацию прикладных задач, терминологии, законов предметной области, оборотов профессионального диалекта и специальных обозначений. Каждая прикладная задача должна иметь формулировку и быть отнесена к одному из трех классов – передачи, хранения или обработки информации. Формулировки прикладных задач содержат термины предметной области; с них начинается систематизация терминов. Каждому термину с помощью экспертов дается возможно более точное определение. Определение связывает определяемый термин с другими терминами, которые также включаются в систематизацию терминов. Кроме определений, связи между терминами устанавливаются законами предметной области. Законы могут содержать термины, еще не попавшие в список терминов. В формулировки прикладных задач, определений терминов и законов предметной области могут входить обороты профессионального диалекта и специальные обозначения. Они систематизируются в специальных списках. Полнота и правильность списков систематизированных профессиональных идей проверяется экспертами.

Как указывалось выше, терминология, профессиональный диалект и специальные обозначения предметной области затрудняют понимание аналитиком систематизированных профессиональных идей. Для того, чтобы эти идеи были правильно поняты аналитиком, необходим их перевод на понятный ему язык. Минимально необходимым является: использование в определениях в качестве неопределяемых таких терминов, которые известны и аналитику, и экспертам; сопоставление всем специальным оборотам профессионального диалекта их аналогов, понятных и аналитику, и экспертам; составление точных правил построения и извлечения смысла для всех специальных обозначений. Перевод на понятный аналитику язык остальных систематизированных профессиональных идей необходим для удобства использования систематизированного описания. Правильность перевода проверяется экспертами на основе интуитивного познания.

После того как профессиональные идеи систематизированы и переведены на понятный аналитику язык, они должны быть математизированы. В результате должно быть построено такое математизированное описание предметной области, в терминах которого возможно заменить все задачи обработки информации профессиональной деятельности соответствующими математическими задачами. За правильность математизации ответственность несет аналитик. Правильность математизации устанавливается на основе интуитивного познания. Дальнейшая работа аналитика ведется математическими методами. Она состоит в исследовании математических задач, разработке и исследовании алгоритмов для их решения.

Последним этапом работы является формализация математизированного описания предметной области средствами подходящего формального языка. Формализация состоит в переводе математических идей на этот формальный язык на основе его семантики и прагматики. Формальное описание должно обязательно содержать неформальный, содержательный комментарий, часть которого понятна эксперту, а другая – математику.

3.4. Результаты анализа профессиональной деятельности

С результатами анализа профессиональной деятельности работают три типа специалистов: эксперты предметной области, аналитики и проектировщики. Эксперты являются носителями профессиональных идей, аналитики – это специалисты по получению профессиональных идей от экспертов, их систематизации, математизации и формализации; проектировщики должны получить от аналитиков правильное, полное и понятное им описание основных функций информационной системы на профессиональном и математическом диалектах, а также на формальном языке.

Эксперт владеет терминологией предметной области, профессиональным диалектом и специальными обозначениями. Он знает все профессиональные идеи, но не может их систематически перечислить. Эксперты отвечают за правильность профессиональных идей и за правильность понимания их аналитиком. Для этого они работают с той частью результатов анализа профессиональной деятельности, где содержатся систематизированные профессиональные идеи, представленные на профессиональном диалекте и в системе понятий предметной области.

Проектировщик работает с математизированным и формализованным описанием предметной области, математическими задачами и алгоритмами для их решения. Он также использует описания на профессиональном диалекте в качестве комментариев к математизированному и формальному описаниям. Терминология, профессиональный диалект и специальные обозначения предметной области используются им также для описания пользовательского интерфейса информационной системы. Проектировщики отвечают за правильность воплощения функций информационной системы в ее проекте.

Аналитик работает со всеми результатами анализа профессиональной деятельности. Он должен получить от экспертов и систематизировать все профессиональные идеи, необходимые для математизированного описания предметной области, представить эти идеи и прикладные задачи в понятном для экспертов виде, получить само математизированное описание и формулировки математических задач, провести необходимую математическую работу – исследование математических задач, разработку и исследование алгоритмов для их решения, а также формализовать математизированное описание. Аналитик отвечает за правильность всех результатов анализа профессиональной деятельности, а также за то, чтобы эти результаты были однозначно и правильно поняты проектировщиком. Для этого аналитики и используют **метод формализации**, состоящий в переводе математизированного описания предметной области и постановок математических задач на формальный язык спецификаций.

3.5. Выводы

Программирование состоит в переводе алгоритмов с математического диалекта на язык программирования. Основными свойствами информационной системы, важными для ее проектирования, являются функциональность и качество. Анализ профессиональной деятельности, в ходе которого определяется функциональность информационной системы, состоит в систематизации профессиональных идей, переводе их на язык, понятный аналитику, и их математизации. Результаты анализа должны быть понятны эксперту и проектировщику. Для обеспечения правильности и точности результатов анализа используется метод их формализации.

4. МЕТОД МОДЕЛИРОВАНИЯ

В этом разделе будет рассмотрен вопрос о том, что такое сходство между идеями.

4.1. Модели

**Объект моделирования** – это любой объект, явление или система, существующие в материальном мире или в сознании человека и данные этому человеку как совокупность идей об этом объекте. **Модель** – это любой другой объект или система, существующие в материальном мире или в сознании человека, данные этому человеку как совокупность идей об этом объекте и связанные с объектом моделирования определенным образом. Модель (как объект или система) становится **моделью** объекта моделирования, если между ними установлено соответствие, удовлетворяющее следующим условиям:

- для объекта моделирования выделены некоторые свойства (идеи), которые далее будут называться **существенными**;

- установлено однозначное соответствие между существенными свойствами объекта моделирования и некоторыми свойствами его модели (свойства модели также являются некоторыми идеями; будем говорить, что свойства модели, соответствующие существенным свойствам объекта моделирования, **представляют** эти существенные свойства в модели);

- определено, в каком случае модель является **адекватной** объекту моделирования.

Будем говорить, что совокупность идей, представляющих существенные свойства объекта моделирования, **похожа** на совокупность идей, представляющих эти существенные свойства в модели.

Примером модели некоторого участка земной поверхности является его топографическая карта. Объектом моделирования является этот участок земной поверхности. Моделью является топографическая карта. На карте имеется таблица условных обозначений, которые определяют свойства земной поверхности, принятые как существенные, и свойства карты, которые представляют эти существенные свойства объекта моделирования в модели (на карте). Карта используется для решения некоторых задач (например, ориентирования на местности). Модель (карта) является адекватной объекту моделирования (участку земной поверхности), если она позволяет решать эти задачи правильно.

Как правило, модель проще, чем объект моделирования, поскольку не все свойства объекта моделирования рассматриваются как существенные, а свойства модели, не представляющие существенных свойств объекта моделирования, обычно игнорируются. Более того, чем проще модель, тем она удобнее для использования.

Таким образом, модель – это отношение сходства между идеями о существенных свойствах объекта моделирования и идеями о некоторых свойствах его модели (как объекта).

4.2. Два способа использования моделей

Модели используются, главным образом, для достижения двух различных целей: для изучения объектов моделирования с помощью их моделей и для построения объектов моделирования по их моделям.

В первом случае для объекта моделирования строится или выбирается его модель, после чего изучение существенных свойств объекта моделирования заменяется изучением этих свойств, представленных в его модели. Этот способ использования моделей широко распространен в науке. При изучении какого-либо природного явления (например, биоценоза) изучаются его существенные свойства (популяции, образующие биоценоз, их кормовая база, взаимосвязи между популяциями типа хищник-жертва и другие, влияние на биоценоз климатических флуктуаций, антропогенных воздействий и т.п.). Далее строится компьютерная модель этого природного явления, в которой представлены все эти существенные свойства. Эта модель позволяет заменить изучение свойств этого природного явления, недоступных наблюдению (например, развития этого природного явления в будущем), изучением свойств его модели, представляющих недоступные для наблюдения свойства объекта моделирования. Решение задач прогноза с использованием компьютерной модели этого природного явления позволяет определить свойства этой модели, представляющие существенные свойства природного явления в будущем. Если по прошествии времени, когда это будущее настанет, результаты прогноза будут подтверждены, и так будет происходить всегда, компьютерная модель природного явления является адекватной. Таким образом, в этом случае для изучения объекта моделирования его модель строится.

При разработке новых лекарств необходимо проводить их клинические испытания, прежде чем внедрять их в медицинскую практику. Однако проведение таких испытаний на людях сопряжено с большим риском и может быть разрешено только в том случае, если есть достаточная уверенность, что такие испытания не принесут вреда здоровью испытуемых. Для того чтобы получить такую уверенность, в качестве моделей испытуемых выбираются животные. При этом изучается вопрос о том, в какой мере те свойства человеческого организма, на которые новое лекарство оказывает воздействие, представлены в организме этих животных. На ранних этапах разработки лекарств для их испытания могут использоваться сравнительно дешевые виды животных, являющиеся не слишком адекватными моделями (например, белые мыши). Когда с помощью таких испытаний наиболее опасные свойства лекарства будут устранены, для дальнейших испытаний выбираются все более адекватные модели – собаки, низшие обезьяны и, наконец, приматы. Если результаты испытаний лекарства на животных будут подтверждены результатами испытаний на людях, то животные будут считаться адекватными моделями людей при решении данной проблемы. Таким образом, при изучении воздействия новых лекарственных веществ в качестве моделей человеческого организма выбираются уже существующие объекты - животные.

Не менее часто этот способ использования моделей применяется и в практической деятельности. Например, некоторое предприятие общественного питания хочет улучшить обслуживание посетителей и снизить свои непроизводительные затраты. Для этого оно может построить компьютерную модель процесса обслуживания посетителей на своем предприятии, используя результаты изучения этого процесса. Затем эта модель может изменяться таким образом, чтобы моделировать различные изменения в системе обслуживания на этом предприятии – увеличение или уменьшение обслуживающего персонала в различное время дня, установка нового оборудования и т.п. На этих измененных моделях могут быть изучены характеристики обслуживания посетителей, расходы и прибыль предприятия и т.п. Только после такого изучения на моделях может быть приняты решения о реорганизации процесса обслуживания посетителей. Если практические результаты этой реорганизации совпадут с прогнозируемыми, модель будет считаться адекватной.

Искусство является одним из способов интуитивного познания мира. Оно заменяет существенные свойства мира, трудные для изучения с помощью непосредственного наблюдения и научных методов, например, нравы, события и психологию личностей прошлого и настоящего, представлением этих свойств в моделях – произведениях искусства, например, карикатурах. Сущность карикатуры – преувеличение. Именно таким способом автор карикатуры представляет в ней существенные свойства явления и сообщает об этом зрителю. Однако карикатуры отличаются особенной правдивостью (модели являются адекватными моделируемому явлению) не вопреки, а именно благодаря своей преувеличенности. Преувеличивая, художник обнажает самую суть явления, отбрасывая все покровы, способные ввести в заблуждение. Самый близорукий взгляд видит, о чем идет речь, самый неповоротливый ум понимает внутреннюю тайну явления. Подобным образом и другие жанры искусства позволяют нам познавать жизнь. Адекватность таких моделей определяется интуитивно.

Философия предлагает умозрительные модели (философские системы) устройства мира и отражения его в сознании человека, используемые как структуры, в рамках которых происходит последующее научное или интуитивное изучение более конкретных вопросов. В этих моделях постулируются некоторые существенные свойства объекта моделирования. Например, Аристотель утверждал, что существование всех конечных и преходящих вещей является движением от бытия в возможности к бытию в действительности, что мир существует вечно, а движение всех вещей – это непрекращающееся движение, что источником этого движения является неподвижный, вечный Перводвигатель, действие которого состоит в переводе конечных вещей из бытия в возможности к бытию в действительности. Эпикур утверждал, что вселенная всецело материальна и состоит из материи и пустоты, ничто не возникает из ничего и не уничтожается в ничто, т.е. вселенная не знает ни прироста, ни убыли, что отклонение атомов является причиной свободы воли во вселенной, которая в остальных отношениях причинно детерминирована. Проверка адекватности таких моделей весьма проблематична, что не уменьшает их познавательной ценности для человечества.

Другой класс умозрительных моделей мира предлагает религия. Например, буддизм утверждает, что все явления в мире причинно взаимосвязаны, что души живых существ включены в цикл последовательных воплощений, что все поступки живых существ создают позитивную или негативную карму, которая на основе закона действия и возмездия влияет на будущие жизни, но нравственное поведение, дисциплина ума и мудрость могут вырвать душу из цикла последовательных воплощений и, тем самым, избавить ее от страданий. Проверка адекватности таких моделей еще более затруднительна.

В случае использования моделей для построения объектов моделирования по их моделям определяются существенные свойства, которыми должен обладать объект моделирования, затем строится модель, в которой представлены эти существенные свойства, и после этого по этой модели создается объект моделирования, обладающий этими существенными свойствами. Этот способ использования моделей обычно называется проектированием и планированием. Существенные свойства объекта моделирования выражаются в виде требований к объекту проектирования или планирования. Проект является моделью некоторого объекта, который предстоит создать (например, нового холодильника или самолета). Если создание опытного образца покажет, что он является работоспособным и удовлетворяет требованиям к объекту проектирования, то проект является адекватной моделью объекта проектирования. План является моделью некоторой последовательности действий, которую предстоит выполнить, чтобы достичь заданной цели. Если в результате выполнения этой последовательности действий в соответствии с планом цель будет достигнута, то план является адекватной моделью этой последовательности действий.

4.3. Материальные модели

Модель является **материальной**, если она является объектом или системой материального мира. В приведенных выше примерах материальными моделями являются топографическая карта, животные в клинических испытаниях новых лекарств, произведения искусства. При этом животные уже существуют в материальном мире и лишь выбираются в качестве моделей, а топографические карты и произведения искусства специально создаются, чтобы служить моделями. Однако не все свойства материальных моделей могут быть доступными для изучения, что определяется природой этих объектов. Наиболее ярким из уже приведенных примеров моделей, трудных для изучения, являются животные. Кроме того, создание, изучение и использование материальных моделей может потребовать значительных финансовых затрат. Особенно дорогим может оказаться создание произведений искусства, а также изучение и использование животных в качестве моделей. Наконец, использование материальных моделей может быть связано с нежелательным воздействием (возможно, разрушительным) на объекты материального мира. Например, при клинических испытаниях новых лекарств могут гибнуть подопытные животные. Все эти отрицательные свойства материальных моделей создают ограничения на их использование.

4.4. Идеальные модели

Модель является **идеальной**, если она состоит из идей, т.е. существует в сознании человека. Примерами идеальных моделей могут служить философские и религиозные системы. В отличие от материальных, идеальные модели не имеют ограничений с точки зрения доступности для изучения их собственных свойств. Однако свойства идеальных моделей часто могут иметь неоднозначное, иногда даже субъективное толкование. Как правило, затраты на работу с идеальными моделями (затраты на интеллектуальную деятельность) существенно меньше, чем затраты на работу с материальными моделями. Использование идеальных моделей не связано с воздействием на объекты материального мира.

4.5. Компьютерные модели

**Компьютерные модели** - это программы. Они обладают некоторыми свойствами материальных и идеальных моделей. Как и в случае материальных (но в отличие от идеальных) моделей, не все свойства компьютерных моделей могут быть доступными для изучения, так как это изучение связано с работой компьютера, происходящей в пространстве (памяти компьютера) и во времени; в силу этого в ряде случаев сложность вычислений может стать препятствием в исследовании некоторых свойств компьютерной модели. Свойства компьютерных моделей почти всегда имеют однозначное толкование.

Как и в случае идеальных (но в отличие от материальных) моделей, использование компьютерных моделей, как правило, не ведёт к воздействию на объекты материального мира. Создание, изучение и использование компьютерных моделей обычно требует больших финансовых затрат, чем работа с идеальными моделями: помимо затрат на разработку и сопровождение программ, требуются еще затраты на их эксплуатацию, приобретение и эксплуатацию оборудования, энергию и т.п. Все компьютерные модели являются конечными. Поэтому они часто представляют существенные свойства объектов моделирования лишь приближенно.

4.6. Математические модели

Модель является **математической**, если она есть объект математики. Математические модели являются подклассом идеальных моделей. Свойства математических моделей всегда толкуются однозначно. Работа с математическими моделями связана, как правило, с минимальными материальными затратами, но требует специальной подготовки (математического образования) и интеллектуальных усилий, вплоть до чрезмерно высоких. Сами математические модели имеют минимальную сложность среди всех возможных адекватных моделей одного и того же объекта моделирования, так как в силу абстрактности идей, их составляющих, не имеют других свойств, кроме тех, с помощью которых представлены существенные свойства этого объекта моделирования.

4.7. Связи между объектом моделирования, его компьютерной и математической моделями

Если компьютерная и математическая модели одного и того же объекта моделирования, представляют одни и те же его существенные свойства, то математическую модель этого объекта моделирования можно рассматривать и как математическую модель компьютерной модели этого объекта. При использовании компьютерных моделей обычно для объекта моделирования строится его математическая модель, а затем по этой модели создается компьютерная модель, которая будет компьютерной моделью и объекта моделирования, представляющей те же его существенные свойства, что и представленные в математической модели.

4.8. Модели в жизненном цикле информационной системы

При разработке информационной системы объектом моделирования является профессиональная деятельность, подлежащая автоматизации. Систематизация информации об объекте моделирования (определение существенных свойств профессиональной деятельности) и построение его модели составляет **этап анализа** в жизненном цикле информационной системы.

В настоящее время существует три основных подхода к анализу профессиональной деятельности. Первый ориентирован на построение процедурной модели профессиональной деятельности как системы преобразования, хранения и передачи информации. Такая модель представляет существенные свойства профессиональной деятельности, выполняемой людьми (до автоматизации). Второй подход – это объектно-ориентированный анализ задач обработки информации профессиональной деятельности. Строится процедурная модель профессиональной деятельности в виде системы взаимодействующих объектов. Эти два подхода к анализу изучаются в курсе "Технология программирования". Третий подход состоит в построении декларативной модели предметной области, в которой происходит профессиональная деятельность, построении и исследовании математических спецификаций задач профессиональной деятельности в терминах математической модели предметной области, построении и исследовании методов решения этих задач. В этом случае существенными свойствами профессиональной деятельности считаются предметная область, совокупность задач, решение которых возлагается на информационную систему, и методы решения этих задач. В настоящем курсе начинается изучение третьего подхода к анализу профессиональной деятельности. Это изучение продолжается в курсе "Системы искусственного интеллекта".

Как уже говорилось, математическая модель профессиональной деятельности является одновременно и математической моделью информационной системы, автоматизирующей эту профессиональную деятельность. Таким образом, математическая модель информационной системы состоит из математической модели предметной области, спецификаций задач профессиональной деятельности и описания методов их решения.

В процессе разработки информационной системы (в ее жизненном цикле) последовательно создаются как минимум три модели автоматизируемой деятельности: математическая модель информационной системы (математическая модель автоматизируемой деятельности), проект информационной системы (идеальная модель информационной системы) и код информационной системы (компьютерная модель автоматизируемой деятельности).

Математическая модель информационной системы представляет существенные свойства предметной области, задач профессиональной деятельности и методов их решения. При этом в математической модели информационной системы отсутствует какая бы то ни было информация о качестве этой системы. Проект информационной системы разрабатывается на основе ее математической модели и требований заказчика. Проект информационной системы содержит все необходимые детали реализации системы в форме, удобной для разработки ее кода, но сам проект программой еще не является. Код информационной системы разрабатывается на основе проекта этой системы.

Модели информационной системы, разрабатываемые в ее жизненном цикле (до ее кода), являются средством борьбы с различными типами сложностей, возникающих в ходе разработки информационной системы: сложностью профессиональной деятельности и предметной области, сложностью достижения необходимого уровня качества информационной системы, вытекающей из требований пользователя, сложностью кодирования и отладки больших программных систем, необходимостью привлечения к разработке многих разработчиков и т.п. Усилия, затрачиваемые на создание моделей информационной системы, должны "окупаться": приводить к уменьшению числа ошибок в коде программы, сокращать сроки и стоимость его получения, повышать его надежность, модифицируемость и т.п.

Различные модели автоматизируемой деятельности требуют различной квалификации их разработчиков. В связи с этим часто различают аналитиков (прикладных математиков, инженеров знаний и т.п.), проектировщиков, кодировщиков и испытателей информационных систем. Кодировщики не имеют дела непосредственно с математическими моделями информационных систем, тогда как деятельность аналитиков состоит в их разработке, проектировщиков - в разработке проектов информационных систем на основе их математических моделей и требований заказчика, а испытателей – установление неправильности информационных систем относительно их функциональности и качества. Поэтому не только аналитики, но и проектировщики, и испытатели должны иметь достаточную подготовку, чтобы пользоваться математическими моделями информационных систем.

4.9. Выводы

Модель – это такой объект, для которого определено, как в нем представлены существенные свойства объекта моделирования. Модели используются для изучения объектов моделирования и для построения объектов моделирования по моделям. Материальные модели – это объекты материального мира. Идеальные модели – это системы, построенные из идей. Компьютерные модели – это программы. Математические модели – это математические объекты. Математическая модель объекта моделирования может служить математической моделью компьютерной модели этого объекта моделирования. В жизненном цикле информационной системы строятся как минимум три модели автоматизируемой деятельности – математическая (результат анализа), идеальная (результат проектирования) и компьютерная (результат кодирования).

5. ПЕРЕДАЧА, ХРАНЕНИЕ И ОБРАБОТКА ИНФОРМАЦИИ

В этом разделе рассматривается вопрос о том, как информация, которая существует только в сознании людей в виде идей, может передаваться, храниться и обрабатываться вне человеческого сознания.

5.1. Информация и сообщение

**Информация** – это объект идеального мира - идеи, которые могут существовать только в сознании человека. **Сообщение** – это объект или процесс материального мира, который используется в качестве адекватной модели информации. Например, сообщением является красный, желтый или зеленый свет светофора, который используется в качестве модели информации для пешеходов о разрешении или запрещении перехода через улицу и в качестве модели информации для водителей транспортных средств о разрешении или запрещении движения транспорта по улице. В этом примере информация, моделируемая сообщениями светофора, появляется в виде идей в сознании пешеходов и водителей как реакция на восприятие этих сообщений. Два сообщения **эквивалентны**, если они адекватно моделируют одну и ту же информацию. Например, можно установить отношения эквивалентности между сообщениями светофора и жестами (сообщениями) регулировщика движения. **Интерпретация** сообщения – это процесс перехода от сообщения к информации (отображение сообщений на информацию). В приведенных примерах - это процесс осознания смысла сообщений светофора или регулировщика. Таким образом, интерпретация – это процесс, происходящий в сознании человека. **Правило интерпретации** сообщения – это способ перехода от сообщения к информации. В приведенных выше примерах - это использование знаний о том, что означают сообщения светофора или жесты регулировщика. **Язык** – это множество сообщений, имеющих одно и то же правило интерпретации. Действительно, язык – это, во-первых, множество сообщений, устроенных в соответствии с правилами (грамматикой) языка (например, множество текстов на русском языке) и, во-вторых, правила получения смысла (интерпретации) этих текстов. **Кодирование** информации – это процесс перехода от информации к сообщению, т.е. процесс, обратный интерпретации.

5.2. Передача информации

**Передача сообщения** – это процесс перемещения этого сообщения из одной части пространства в другую. **Среда** передачи сообщений – это часть пространства, обладающая определенными физическими свойствами, через которую осуществляется передача сообщений. Среда может быть одномерной (например, металлические провода для передачи сообщений с помощью электричества), двумерной (например, поверхность воды при передаче сообщений с помощью волн на водной поверхности) и трехмерной (например, воздушное пространство при передаче акустических сообщений). **Сигнал** - это физический процесс, протекающий в среде и используемый для передачи сообщений. В приведенных выше примерах это распространение электричества по проводнику, волн по водной поверхности и акустических волн в воздухе. **Параметр** сигнала – это изменяющееся во времени свойство сигнала, значения которого на некоторых отрезках времени представляют передаваемое сообщение. В приведенных примерах параметром сигнала могут быть амплитуда и частота сигнала. **Дискретный сигнал** – это такой сигнал, параметр которого может принимать только конечное множество различных значений. Например, светофор использует дискретные световые сигналы для передачи сообщений, причем параметром сигнала является длина световой волны. **Дискретное сообщение** – это сообщение, которое передается дискретным сигналом. Сообщения, передаваемые светофором, являются дискретными. **Передача информации** от отправителя к получателю состоит в кодировании этой информации отправителем, передаче сообщения, полученного в результате кодирования, получателю и интерпретации получателем сообщения, полученного им в результате передачи. Например, отправитель хочет передать некоторую информацию получателю. Для этого он пишет письмо (кодирует эту информацию в виде письменного сообщения на языке, известном получателю), передает это письмо посыльному, который доставляет его получателю, а получатель интерпретирует это письмо с помощью правила интерпретации языка, на котором написано это письмо, и таким образом получает переданную ему информацию.

5.3. Хранение информации

**Хранение** **сообщения** – это передача этого сообщения из одного момента времени в другой, более поздний. **Носитель** - это часть пространства, обладающая определенными физическими свойствами, с помощью которого осуществляется хранение сообщений. Например, носителем может быть поверхность листа бумаги, обладающая цветом, с помощью которого могут храниться сообщения. **След** – это неоднородность в пространстве носителя, возникающая в результате взаимодействия носителя и некоторого сигнала, сохраняющаяся некоторое время после своего возникновения и используемая для хранения сообщений. Например, следом является письменное сообщение, т.е. неоднородность цвета на поверхности бумажного листа, которая возникает на ней как результат соприкосновения конца грифеля карандаша с поверхностью этого листа в процессе передачи сообщения с помощью сигнала - движения руки, в которой находится карандаш. Параметром этого сигнала является положение конца карандаша относительно листа бумаги, изменяющееся во времени. Сообщение хранится до тех пор, пока сохраняется этот лист бумаги, и на нем сохраняется след карандаша. **Параметр** следа – это изменяющееся в пространстве носителя свойство следа, значения которого в некоторых областях носителя представляют хранимое сообщение. Для письменного сообщения на листе бумаги параметром следа является форма линий. **Дискретное сообщение** – это сообщение, которое хранится в виде следа, параметр которого может принимать только конечное множество различных значений. Письменные сообщения являются дискретными. **Хранение информации** – это процесс, состоящий из кодирования этой информации, хранения сообщения, полученного в результате кодирования, и последующей интерпретации этого сообщения. Примером хранения информации может служить обычай строителей зданий оставлять послания потомкам: они кодируют свое послание на листе бумаги и закладывают его в специальный тайник. Через много лет (веков) реставраторы этого здания обнаруживают тайник с посланием в нем и интерпретируют это послание.

5.4. Знаки

**Графемы** – это элементы, из которых строятся письменные дискретные сообщения некоторого языка. Примерами графем являются буквы, цифры, знаки препинания, нотные знаки и т.п. **Фонемы** – это элементы, из которых строятся устные дискретные сообщения некоторого естественного языка. **Знаки** – это элементы, из которых строится какое-либо множество дискретных сообщений. Графемы и фонемы являются примерами знаков. Знаками являются и цветовые сигналы, передаваемые светофором. **Набор знаков** – это конечное множество знаков. Примером может служить набор из трех знаков светофора. **Алфавит** – это упорядоченный набор знаков. Примерами алфавитов являются русский и латинский алфавиты, алфавит десятичных цифр. **Двоичный набор знаков** – это набор, состоящий из двух знаков. Он является минимальным по количеству знаков в наборе. Примером является набор знаков, состоящий из точки и тире, – набор знаков азбуки Морзе. **Бит** – это элемент двоичного набора знаков. Таким образом, точка и тире являются битами.

5.5. Коды и кодирование

Далее дискретные сообщения будут рассматриваться как конечные последовательности знаков из некоторого набора (или алфавита). **Длина** дискретного сообщения – это количество знаков в нём. Например, длина сообщения "длина" равна 5. **Слово** – это наименьшая часть сообщения, имеющая интерпретацию. Например, сообщение "Слово – это наименьшая часть сообщения, имеющая интерпретацию." состоит из 7 слов, причем знаки "-", "," и "." не являются словами и не входят в другие слова. **Двоичное сообщение** – это дискретное сообщение, построенное из битов. Примером является двоичное сообщение ".-.".

**Код** – это отображение (и его результат) одного множества сообщений на другое, либо отображение различной информации на некоторое множество сообщений. Примером отображения одного множества сообщений на другое может служить таблица соответствия между названиями и символами химических элементов: "водород – H, гелий – He, литий – Li, бериллий – Be, бор – B, углерод – C, азот – N, кислород – O, фтор – F, неон – Ne и т.д.". Примером отображения информации на множество сообщений является соответствие между информацией о возможности или невозможности перехода через улицу и сообщениями светофора. **Кодирование** – это процесс получения кода как результата отображения. **Двоичный код** – это отображение (и его результат) набора сообщений или информации на некоторый набор двоичных сообщений. Примером может служить представление русских текстов и передаваемой ими информации средствами азбуки Морзе. Любая компьютерная модель может рассматриваться как двоичное сообщение.

5.6. Обработка информации

**Обработка информации** – это получение некоторой неизвестной информации по некоторой известной. Обработка информации **всегда** происходит в сознании человека. Например, врач должен по известной ему информации о результатах обследования больного и известной ему информации о закономерностях проявления заболеваний получить неизвестную ему информацию о диагнозе больного. **Содержательная постановка задачи обработки информации** – это содержательное описание известной и неизвестной информации, а также взаимосвязи между неизвестной и известной информацией. В приведенном выше примере эта содержательная постановка задачи определяет, что диагноз больного - это такое известное врачу заболевание, закономерности проявления которого соответствуют результатам обследования больного. **Математическая постановка** з**адачи обработки информации** состоит в замене ее математической задачей обработки сообщений. **Обработка сообщений** - это кодирование известной информации, получение таким образом известного сообщения, кодирование неизвестной информации, получение некоторого неизвестного сообщения и установление взаимосвязей между этими сообщениями. Поскольку, в отличие от информации, сообщения являются объектами реального мира, обработка сообщений **всегда** требует некоторой работы в реальном мире. Например, приведенная выше задача обработки информации может быть заменена следующей задачей обработки сообщений. Врач использует компьютер. Он кодирует известные ему результаты обследования больного, а также его диагноз в виде специальных сообщений, кодирует медицинские знания и явным образом описывает взаимосвязь диагноза и результатов обследования больного. **Алгоритм обработки сообщений** – это способ получения неизвестного сообщения по известному. В данном примере – это программа компьютерной диагностики заболеваний. При замене обработки информации обработкой сообщений интерпретация известного сообщения должна давать известную информацию, а интерпретация неизвестного сообщения - неизвестную информацию. Например, коды, набираемые с помощью клавиатуры, должны интерпретироваться как результаты обследования больного. Код на экране дисплея, получаемый после завершения работы программы, должен интерпретироваться как диагноз больного.

**Диаграмма обработки сообщений** состоит из четырех элементов - неизвестной и известной информации, а также неизвестного и известного сообщений. Эти четыре элемента связаны четырьмя связями – правило кодирования информации связывает известную информацию с известным сообщением, правило интерпретации связывает неизвестное сообщение с неизвестной информацией; содержательная постановка задачи связывает известную информацию с неизвестной; математическая постановка задачи и алгоритм ее решения неявно и явно связывают известное сообщение с неизвестным.

Диаграмма обработки сообщений должна быть **коммутативной** - путь от известной информации через правило ее кодирования к известному сообщению, далее к неизвестному сообщению через алгоритм обработки сообщений и затем к неизвестной информации через правило интерпретации неизвестного сообщения должен приводить к той же информации, что и путь от известной информации к неизвестной через содержательную постановку задачи. Таким образом, при замене задачи обработки информации задачей обработки сообщений мыслительный процесс – обработка информации заменяется работой в реальном мире – обработкой сообщений, а также двумя другими мыслительными процессами – кодированием известной информации и интерпретацией неизвестного сообщения. Такая замена имеет смысл только в том случае, когда обработка информации имеет более высокую стоимость из-за более высокой интеллектуальной сложности, чем суммарная стоимость обработки сообщений, интерпретации неизвестного сообщения и кодирования известной информации.

Основная проблема программирования – это замена задач обработки информации задачами обработки сообщений, т.е. построение для известной и неизвестной информации таких сообщений (кодирование информации) и правил их интерпретации (в общем случае - языков), а также алгоритмов обработки сообщений (программ), чтобы получаемые диаграммы обработки сообщений оказывались коммутативными. В этом случае обработку сообщений выполняет компьютер.

Алгоритм обработки сообщений должен быть выполнимым (в частности, допускать реализацию на компьютере). Это значит, что он должен описывать способ рассуждений. Алгоритм обработки сообщений (рассуждение) и все определяемые им шаги обработки должны представлять собой конечные последовательности элементарных шагов обработки. Каждый элементарный шаг обработки должен быть непосредственно выполнимым (человеком, компьютером или другим исполнителем). При этом алгоритм обработки определяет формальные рассуждения. Такие рассуждения обладают некоторыми важными для практики обработки сообщений свойствами, зависящими от исполнителя - временем и памятью, требуемыми для обработки сообщений. Не менее важным является и возможность представления алгоритмов обработки сообщений дискретными сообщениями. Это дискретное сообщение управляет работой компьютера при обработке сообщений, либо правило интерпретации этого дискретного сообщения дает информацию, необходимую для выполнения этого правила обработки сообщений человеком.

5.7. Выводы

Информация может быть получена в результате интерпретации сообщения, а сообщение – в результате кодирования информации. Передача информации от отправителя к получателю состоит в кодировании этой информации отправителем, передаче сообщения, полученного в результате кодирования, получателю и интерпретации получателем сообщения, полученного им в результате передачи. Хранение информации – это процесс, состоящий из кодирования этой информации, хранения сообщения, полученного в результате кодирования, и последующей интерпретации этого сообщения. Знаки – это элементы, из которых строится дискретное сообщение. Код – это отображение одного множества сообщений на другое. Обработка информации – это процесс, состоящий в кодировании известной информации, обработке известного сообщения, полученного в результате кодирования, и интерпретация неизвестного сообщения, полученного в результате обработки.

Задание 1 (по теме "Передача, хранение и обработка информации")

Придумать пример профессиональной деятельности.

План ответа

1. Содержательное описание профессиональной деятельности.

2. Задачи передачи, хранения и обработки информации, из которых состоит эта деятельность.

3. Для каждой задачи передачи информации указать отправителя, адресата, содержательное описание передаваемой информации, описание сообщений, с помощью которых эта информация передается, диаграмму передачи информации.

4. Для каждой задачи хранения информации указать цель хранения, содержательное описание хранимой информации, описание сообщений, с помощью которых эта информация хранится, диаграмму хранения информации.

5. Для каждой задачи обработки информации указать описание известной и неизвестной информации, содержательную постановку задачи, описание известных и неизвестных сообщений, математическую постановку задачи, алгоритм решения задачи обработки сообщений, диаграмму обработки информации.

**II. ВЕЛИЧИНЫ, ОНТОЛОГИИ И ЗНАНИЯ**

Ключевым для дальнейшего изложения является понятие вербализуемой информации. Рассмотрим некоторую идею (информацию) *i*. Она является вербализуемой, если выполнены следующие условия:

для этой информации существует конечный набор терминов *T = {t1, …, tm}*  и множество возможных значений этих терминов *V*;

идея *i* может быть закодирована (представлена) сообщением, которое есть отображение множества *T* в множество *V*.

Иными словами, информация является вербализуемой, если ее можно представить сообщением, которое есть некоторое соответствие (таблица) между конечной совокупностью терминов и их значений. Такое представление информации будем называть вербальным. Если *r* вербальное представление информации *i*, то *r(t1) = v1, …, r(tm) = vm*. Чтобы уменьшить в записи количество скобок, будем записывать то же самое следующим образом: *r.t1 = v1, …, r.tm = vm*.

Особенностью вербального представления информации является то, что элементы множеств *V* и *T* имеют прагматику и семантику – профессионалы используют их в определенных контекстах, понимают их в этих контекстах и вкладывают в них определенный смысл. Поэтому вербальное представление информации весьма экономично для профессионалов – в нем используется только соответствие между обозначениями (терминами и значениями), а смысл этих обозначений для профессионалов считается известным. Никаких других синтаксических особенностей такое представление информации не имеет. Далее будет рассматриваться только вербализуемая информация и ее вербальное представление. Чтобы такое представление информации могли понимать (интерпретировать) и непрофессионалы (например, аналитики и проектировщики), должен быть явно определен смысл терминов и значений.

Смысл значения - элемента множества *V* определяется через его связи с другими значениями: аргументом каких операций и отношений это значение может быть (возможно совместно с другими значениями), и в результате выполнения каких операций над другими значениями оно может быть получено. Чтобы определить смысл значений таким образом, вводятся понятия величин – подмножеств множества *V*, с которыми связываются определенные наборы операций и отношений. Каждое значение принадлежит некоторой величине, которая и определяется его смысл.

Смысл терминов из *T* неявно определяется с помощью концептуализации, использующей терминологию *T*, - множества всех вербальных представлений, имеющих смысл, с одним и тем же набором терминов *T*. Явно же смысл этих терминов определяется с помощью онтологии, представляющей эту концептуализацию.

С использованием такого определения смысла значений из *V* и терминов из *T* вводится понятие системы знаний о некоторой бесконечной совокупности вербализуемой информации *I*. Система знаний явно определяет такую совокупность информации как подмножество концептуализации.

На основе этих понятий вводятся основы онтолого-ориентированного анализа информации.

6. ВЕЛИЧИНЫ И ИХ ПРЕДСТАВЛЕНИЕ В МАТЕМАТИЧЕСКИХ И КОМПЬЮТЕРНЫХ МОДЕЛЯХ

6.1. Величины

Множество всех значений *V* является объединением подмножеств, которые образуют так называемые величины. Каждая величина характеризуется некоторой конечной совокупностью функций и операций (функций от двух аргументов), обозначаемых специальными терминами и выполняемых над значениями этой величины, в результате чего возникают новые значения этой же величины (свойство замкнутости). Кроме функций и операций величина характеризуется конечной совокупностью нефункциональных отношений, в которых значения этой величины могут находиться между собой. Отношения также обозначаются специальными терминами. Величины могут быть простыми или сложными.

**Простая величина** - это система, состоящая из конечной совокупности операций, функций и отношений (обозначаемых специальными терминами), а также множества тех и только тех значений, которые могут быть аргументами этих операций, функций и отношений, причем результат применения любой из этих операций или функций к этим аргументам принадлежит этому же множеству (свойство замкнутости).

**Сложная величина** – это система, состоящая из конечной совокупности операций, функций и отношений (обозначаемых специальными терминами), а также конечной совокупности (простых и/или сложных) величин, элементы которых могут быть аргументами этих операций, функций и отношений, причем для каждого аргумента каждой операции, функции и отношения определено, элементом какой из этих величин он может быть, а для результата применения любой из этих операций или функций к этим значениям аргументов определено, какой из этих величин он принадлежит (свойство замкнутости).

Смысл обозначений операций, функций и отношений, связанных с некоторой величиной, определяется явно в терминах математики при определении этой величины. Смысл обозначений операций и функций позволяет выполнять соответствующие им действия над значениями, входящими в эту величину, и получать в результате этих действий новые значения той же величины, т.е. задавать связи между значениями этой величины операционально. Смысл обозначений отношений, связанных с некоторой величиной позволяет устанавливать, что некоторые значения, входящие в эту величину, находятся в этих отношениях, т.е. позволяет устанавливать связи между значениями этой величины операционально. Смысл значений, входящих в некоторую величину, определяется этой величиной – каждое значение может быть аргументом или результатом тех операций, функций и отношений, которые допускает для этого значения величина. Выделение величин в множестве значений *V* позволяет связывать семантику тех или иных действий и отношений над значениями с семантикой самих этих значений. Выполнение операций и функций над значениями и установление отношений между значениями величин являются элементарными (неделимыми) действиями при обработке вербализуемой информации.

При таком способе определения смысл обозначений операций, функций и отношений, а также значений, связанных с некоторой величиной, не зависит от вербализуемой информации, вербально представляемой с помощью этих значений, а определяется в терминах математики, т.е. является, в определенной степени, универсальным (в той же степени, в какой смысл математических терминов можно считать универсальным). При этом часть этого смысла теряется: именно, теряется связь смысла значений с вербализуемой информацией, необходимая при вербальном кодировании этой информации и при интерпретации вербального представления, а также теряется соответствие между преобразованием вербализуемой информации и преобразованием вербальных представлений этой информации. Эта часть смысла относится к прагматике величин.

Среди всех возможных величин можно выделить множество **стандартных величин**, которые часто используются в ходе профессиональной деятельности. Такими величинами являются размерные и скалярные величины, величины множеств и отображений, объединенные и структурные величины, а также величины последовательностей. Кроме стандартных величин в той или иной профессиональной деятельности могут использоваться **нестандартные величины** других типов.

6.2. Отношения и функции

Теперь от объектов предметной области перейдем к рассмотрению математических объектов, которые будут моделями объектов предметной области в математических моделях предметных областей.

**Множество** – это совокупность математических объектов, объединённых вместе. **Декартово произведение** *n* множеств – это множество всех возможных кортежей длины *n*, в каждом из которых *i*-й элемент принадлежит *i*-му множеству-сомножителю в декартовом произведении для всех *i* от *1* до *n*. **Проекцией** называется функция двух аргументов, первым из которых является целое число *i* в диапазоне от *1* до *n*, а вторым – элемент декартова произведения n множеств. Значением проекции является *i*-й элемент второго аргумента. *n*-местное **отношение** есть подмножество декартова произведения *n* множеств. *m*-ая **декартова степень** – это декартово произведение множества самого на себя *m* раз. *m*-местное **отношение на множестве** – это подмножество *m*-ой декартовой степени множества. **Частичная функция** – это однозначное соответствие между подмножеством декартова произведения *n* множеств (**областью определения** *n*-местной функции) и некоторым множеством (**областью значений** функции). **Функция** – это однозначное соответствие между декартовым произведением множеств и некоторым множеством. **Операция** – это двухместная функция. **Значение функции** – это результат соответствия при фиксированных значениях аргументов. **Частичная функция на множестве** – это частичная функция, областью определения которой является декартова степень. **Функция на множестве** – это функция, областью определения которой является декартова степень. **Операция на множестве** – это двухместная функция на множестве. **Частичная операция** – это частичная двухместная функция. **Предикат** – это функция с логическими значениями. Совокупность кортежей из области определения предиката, на которых предикат имеет значение "истина", называется **графиком** этого предиката. Очевидно, что график любого предиката является отношением.

6.3. Алгебраическая система

**Моносортная алгебраическая система** – это тройка, состоящая из **носителя** - некоторого множества, некоторой конечной (возможно пустой) совокупности (частичных) операций и функций на носителе, и некоторой конечной совокупности отношений на носителе. **Многосортная алгебраическая система** – это тройка, состоящая из **носителя** - конечной совокупности (моносортных и/или многосортных) алгебраических систем (**сортов**), некоторой конечной (возможно пустой) совокупности (частичных) операций и функций, и некоторой конечной совокупности отношений, причем для каждого аргумента каждой функции и отношения определено, элементы какого сорта (из носителя) могут быть значениями этого аргумента и какому сорту (из носителя) принадлежат результаты каждой функции.

**Реляционная алгебраическая система** – это алгебраическая система, совокупность операций и функций которой пуста. Совокупность отношений любой алгебраической системы содержит отношения **равенства** и **неравенства**. Любой элемент носителя алгебраической системы равен самому себе и не равен любому другому элементу носителя. **Алгебра** – это алгебраическая система, совокупность отношений которой не содержит других отношений, кроме равенства и неравенства. **Частичная алгебраическая система** – это алгебраическая система, у которой совокупность операций и функций содержит только частичные операции и функции.

Моносортная алгебраическая система *A1* является **подсистемой** моносортной алгебраической системы *A2*, если выполнены следующие условия:

- носитель *A1* является подмножеством носителя *A2*;

- между совокупностями операций и функций алгебраических систем *А1* и *А2* существует взаимно однозначное соответствие, при котором каждая операция и функция в *A1* есть сужение соответствующей операции или функции в *A2* на носитель *A1*;

- между совокупностями отношений алгебраических систем *А1* и *А2* существует взаимно однозначное соответствие, при котором каждое отношение в *A1* есть такое подмножество соответствующего отношения в *A2*, которое состоит из таких кортежей этого отношения, элементами которых являются элементы носителя *A1*.

Многосортная алгебраическая система *A1* является **подсистемой** многосортной алгебраической системы *A2*, если выполнены следующие условия:

- между сортами, образующими носители *A1* и *A2*, существует взаимно однозначное соответствие, такое, что любой сорт из носителя *A1* является подсистемой соответствующего сорта из носителя *A2*;

- между совокупностями операций и функций алгебраических систем *А1* и *А2* существует взаимно однозначное соответствие, при котором каждая операция и функция в *A1* есть сужение соответствующей операции или функции в *A2* на носитель *A1*;

- между совокупностями отношений алгебраических систем *А1* и *А2* существует взаимно однозначное соответствие, при котором каждое отношение в *A1* есть такое подмножество соответствующего отношения в *A2*, которое состоит из таких кортежей этого отношения, элементами которых являются элементы носителя *A1*.

Моносортные алгебраические системы являются математическими моделями простых величин, а многосортные – сложных. Алгебраическая система *A* является **адекватной** **математической моделью величины** *M*, если носитель *A* является адекватной математической моделью множества значений, образующих величину *M*, совокупность операций и функций *A* является адекватной математической моделью совокупности операций и функций *M*, а совокупность отношений *A* – адекватной математической моделью совокупности отношений *M*. Множество *S1* математических объектов является **адекватной** **математической моделью множества значений** *S2*, если существует однозначное соответствие между *S2* и *S1*, при котором соответствующий элемент *S1* является адекватной математической моделью значения из *S2*. Совокупность операций и функций *F1* алгебраической системы является **адекватной** **математической моделью совокупности операций и функций** *F2* величины, если между *F2* и *F1* имеется взаимно-однозначное соответствие, при котором соответствующая операция или функция из *F1* является адекватной математической моделью операции или функции из *F2*. Математическая операция или функция *f1* является **адекватной** **математической моделью операции или функции** *f2* величины, если при условии, что значениями аргументов операции или функции *f2* являются произвольные значения, а значениями аргументов операции или функции *f1* являются их адекватные математические модели, значение *f2* является адекватной математической моделью значения *f1*. Совокупность отношений *R1* алгебраической системы является **адекватной** **математической моделью совокупности отношений** *R2* величины, если между *R2* и *R1* имеется взаимно-однозначное соответствие, при котором соответствующее отношение из *R1* является адекватной математической моделью отношения из *R2*. Математическое отношение *r1* является **адекватной** **математической моделью отношения** *r2* величины, если между множествами их кортежей существует взаимно-однозначное соответствие, при котором соответствующие элементы соответствующего кортежа из *r1* являются адекватными математическими моделями элементов кортежа из *r2*.

6.4. Типы данных и их математические модели

Теперь мы переходим к компьютерным моделям величин. Напомним, что математические объекты являются математическими моделями и значений, и их компьютерных моделей.

**Примитивный тип данных** – это множество компьютерных объектов вместе с набором операций, функций и предикатов, которые можно выполнять над этими объектами, причем результат выполнения каждой операции или функции над элементами типа данных принадлежит этому же типу данных. **Непримитивный тип данных** – это конечная совокупность (примитивных и/или непримитивных) типов данных вместе с набором операций, функций и предикатов, которые можно выполнять над элементами этих типов данных, причем для каждого аргумента каждой операции, функции и предиката определено, элемент какого типа данных может быть значением этого аргумента, а результат выполнения каждой операции или функции над этими значениями аргументов принадлежит определенному типу данных из этой совокупности.

Моносортные алгебраические системы являются математическими моделями примитивных типов данных, а многосортные – непримитивных. Алгебраическая система *A* является **адекватной** **математической моделью типа данных** *T*, если носитель *A* является адекватной математической моделью множества компьютерных объектов, образующих тип данных *T*, совокупность операций и функций *A* является адекватной математической моделью совокупности операций и функций *T*, а совокупность отношений *A* – адекватной математической моделью совокупности предикатов *T*. Множество *S1* математических объектов является **адекватной** **математической моделью множества компьютерных объектов** *S2*, если существует однозначное соответствие между *S2* и *S1*, при котором соответствующий элемент *S1* является адекватной математической моделью объекта информатики из *S2* Совокупность операций и функций *F1* алгебраической системы является **адекватной** **математической моделью совокупности операций и функций** *F2* типа данных, если между *F2* и *F1* имеется взаимно-однозначное соответствие, при котором соответствующая операция или функция из *F1* является адекватной математической моделью операции или функции из *F2*. Математическая операция или функция *f1* является **адекватной** **математической моделью операции или функции** *f2* типа данных, если при условии, что значениями аргументов операции или функции *f2* являются произвольные объекты типа данных, а значениями аргументов операции или функции *f1* являются их адекватные математические модели, значение *f2* является адекватной математической моделью значения *f1*. Совокупность отношений *R1* алгебраической системы является **адекватной** **математической моделью совокупности предикатов** *R2* типа данных, если между *R2* и *R1* имеется взаимно-однозначное соответствие, при котором соответствующее отношение из *R1* является адекватной математической моделью предиката из *R2*. Математическое отношение *r1* является **адекватной математической моделью предиката** *r2* типа данных, если между графиком *r2* и множеством кортежей *r1* существует взаимно-однозначное соответствие, при котором соответствующие элементы соответствующего кортежа из *r1* являются адекватными математическими моделями элементов кортежа графика *r2*.

6.5. Представление величин в математических и компьютерных моделях

Легко видеть, что определения величин, алгебраических систем и типов данных очень похожи. Разница между ними состоит лишь в том, какие объекты (значения, математические объекты или компьютерные объекты), а также операции, функции и предикаты (отношения) над какими объектами входят в эти определения.

Поэтому математическими моделями величин являются алгебраические системы, а компьютерными – типы данных. Учитывая ранее сказанное, алгебраические системы являются также математическими моделями типов данных.

При анализе информации должны быть определены все величины, значения которых используются для вербального представления этой информации. Величины или их подмножества являются объемами понятий предметной области (см. следующую главу). Система величин должна быть обоснована: для каждой величины должны быть указаны те понятия, объемы которых совпадают с этой величиной или ее подмножеством. Если для некоторых понятий нет таких величин, то это значит, что не все величины, необходимые для представления этой информации найдены. Если для некоторой величины таких понятий нет, то это значит, что эта величина не связана с представлением этой информации. Таким образом, анализ системы понятий должен предшествовать анализу системы величин, а результаты анализа системы величин должны быть обоснованы результатами анализа системы понятий.

7. КОНЦЕПТУАЛИЗАЦИИ И ОНТОЛОГИИ

7.1 Концептуализация

Термины, с которыми приходится сталкиваться аналитику, могут быть связаны с величинами: обозначать операции, функции и отношения величин, а также другие их свойства. Термины из множества терминов *T* в определении вербализуемой информации будем называть **названиями понятий**. Каждое название понятия в вербальном представлении имеет смысл, причем смыслы названий разных понятий могут зависеть друг от друга. Наиболее полно (но неявно) смысл названий понятий, использованных в вербальном представлении информации, может быть представлен подмножеством *C* множества всех возможных вербальных представлений с одним и тем же множеством названий понятий *T*, элементы которого (как сообщения) имеют интерпретацию, т.е. несут некоторую информацию (имеют смысл, представляют некоторые идеи). Это множество *C* получило название **концептуализации**. Элементы этого множества будем называть **элементами концептуализации**. Таким образом, в концептуализации смысл названий понятий определяется с помощью всех тех вербальных представлений, которые представляют какую-либо информацию.

Концептуализация характеризуется множеством названий понятий *T*, а также множеством элементов концептуализации (таких вербальных представлений, которые как сообщения несут некоторую информацию). Поскольку величины, которым принадлежат значения понятий в вербальном представлении, являются, как правило, бесконечными величинами, концептуализация обычно является бесконечным множеством вербальных представлений и не может быть задана их перечислением. Будем называть такие концептуализации **нетривиальными**. В то же время каждый элемент любой концептуализации является отображением с конечной областью определения и поэтому, как конечный объект, может быть определен полностью.

7.2. Онтология

**Онтология** есть явная спецификация нетривиальной концептуализации. **Спецификация** есть интенсиональное определение некоторого (бесконечного) множества на декларативном языке, позволяющее на основе этого описания для любой сущности из заданной совокупности определить, является эта сущность элементом этого множества или нет. Отсюда следует, что онтология есть такое определение концептуализации, которое для каждого вербального представления позволяет установить, является это представление элементом концептуализации или нет (т.е. несет это вербальное представление как сообщение некоторую информацию или нет). Таким образом, онтология определяет смысл названий понятий явно. Далее будем считать, что каждая онтология имеет **название** – некоторый термин.

Как уже говорилось, математика как наука, которая заботится о точном определении смысла названий своих понятий, предложила два способа определения смысла названий понятий – с помощью систем аксиом и с помощью явных определений. С помощью системы аксиом точно определяется смысл некоторой совокупности названий (первичных) понятий. Очевидно, этот способ определения смысла является наиболее общим, но достаточно трудным для понимания людьми. На множестве остальных названий (производных) понятий математика определяет такой частичный порядок, что смысл каждого названия производного понятия точно определяется через смысл предшествующих ему в этом частичном порядке названий понятий. Названия первичных понятий предшествуют названиям всех производных. Этот частичный порядок нарушается только специальными правилами рекурсивных определений. Явное определение смысла одного названия понятия легче понимается людьми и более удобно для использования, чем неявное определение смысла совокупности названий понятий с помощью системы аксиом.

И в случае других концептуализаций онтологию предпочтительно строить из явных **определений** названий понятий. При этом возникают следующие вопросы: какие понятия использовать в качестве первичных и как определять названия этих первичных понятий. Из определения понятия «спецификация» следует, что в качестве названий первичных понятий должны использоваться термины, в которых описан декларативный язык спецификации, на котором представлена онтология. Если в качестве языка спецификации используется профессиональный диалект, то этот профессиональный диалект, как правило, не описывается (поскольку является диалектом естественного языка). В этом случае названия первичных понятий выделить практически невозможно. Поэтому так представленные онтологии предназначены для их восприятия и понимания только профессионалами, владеющими профессиональным диалектом, на котором описана онтология. Если же в качестве языка спецификации используется некоторый искусственный язык, то такой язык описывается в некоторых терминах. В этом случае онтология предназначена для понимания и использования специалистами, владеющими этим искусственным языком спецификации (но не профессионалами, владеющими профессиональным диалектом), а вопрос о том, как определены термины, в которых описан этот язык, остается открытым (решается вне онтологии и вне описания искусственного языка). Возможен еще один вариант, когда в качестве первичных (неопределяемых) названий понятий выступают термины, введенные при определении стандартных и нестандартных величин. В этом случае онтология может быть понята и использована и профессионалами, и аналитиками, которые знают эти величины, а вопрос о том, как определены термины, в которых описаны эти величины, остается открытым (решается вне онтологии и вне описания этих величин). Наконец, величины могут быть заменены их математическими моделями, а онтология может быть описана в терминах этих моделей, т.е. фактически в математических терминах. В этом случае для понимания и использования онтологии необходимо математическое образование. Математические же термины, в которых описана онтология, точно определены в математике. Когда речь будет идти об онтологиях, в качестве первичных понятий будут использоваться понятия, связанные с величинами. Когда же речь пойдет о моделях онтологий и их свойствах, в качестве первичных понятий будут использоваться математические понятия.

Из-за того, что смыслы названий понятий могут быть связаны между собой весьма сложным образом (что должно следовать из свойств конкретной концептуализации), онтология, построенная только из явных определений названий понятий, как правило, задает смысл этих названий понятий неточно. Для уточнения смысла названий понятий в таких онтологиях к явным определениям добавляются **онтологические соглашения**, описывающие необходимые взаимосвязи между смыслами названий разных понятий.

Определения названий понятий и представление онтологических соглашений только в терминах концептуализации может оказаться весьма громоздким и трудным для понимания. Поэтому обычно в онтологии вводится ряд названий понятий, отсутствующих в концептуализации, но позволяющих сделать другие определения и онтологические соглашения онтологии более обозримыми и понятными. Такие названия понятий и сами понятия, ими обозначаемые, будем называть **вспомогательными**. В противоположность к ним, названия понятий из множества *T* будем называть **основными**.

Таким образом, онтология состоит из трех конечных множеств: множества (возможно пустого) названий других онтологий (модулей), являющихся частями данной онтологии; непустого множества определений названий понятий, не входящих в онтологии, названия которых являются элементами первого множества; множества (возможно пустого) онтологических соглашений для описания взаимосвязей между понятиями онтологии. Все названия понятий концептуализации должны иметь определения. Никакое название понятия не может иметь в онтологии двух определений. Определения понятий могут содержать термины, связанные с величинами, и названия понятий.

Каждое определение названия основного понятия вводит **термин**, обозначающий это понятие в онтологии, и **объем** этого понятия, т.е. множество значений некоторой величины, которые могут быть значениями этого понятия в элементах концептуализации. Объем каждого понятия либо совпадает с какой-либо величиной, либо является ее подмножеством.

Каждое определение названия вспомогательного понятия вводит **термин**, обозначающий это понятие в онтологии, и либо **значение** этого понятия (конкретное значение некоторой величины), либо способ вычисления значения этого понятия по значениям других понятий онтологии.

Каждое **онтологическое соглашение** задает связь между названиями понятий онтологии и представляет собой некоторое утверждение, содержащее термины системы понятий и термины, связанные с величинами. Онтологические соглашения задают дополнительные (по сравнению с определениями понятий) ограничения на объемы понятий.

Будем называть множество всех элементов концептуализации, которые соответствуют онтологии, **концептуализацией, определяемой онтологией**. Элемент концептуализации, определяемой онтологией, соответствует онтологии, если выполнены следующие условия:

значение каждого термина, обозначающего основное понятие онтологии, в этом элементе концептуализации принадлежит объему этого понятия, определенному в онтологии;

значение каждого термина, обозначающего вспомогательное понятие онтологии, в этом элементе концептуализации равно либо значению этого понятия, определенному в онтологии, либо результату вычисления с использованием способа вычисления, определенного для этого понятия в онтологии, причем значения других терминов, обозначающих понятия, участвующие в этом вычислении, берутся из этого же элемента концептуализации;

каждое онтологическое соглашение онтологии является истинным, если в нем каждый термин, обозначающий понятие, заменить его значением в этом элементе концептуализации.

Концептуализация *CO*, определяемая онтологией *O* , представляет смысл всех названий понятий, определенных в онтологии, в то время как исходная концептуализация *C* представляет смысл только названий понятий из множества *T*. **Редукция** концептуализации *CO* есть множество таких элементов (отображений), каждый из которых получается из некоторого элемента *CO* сужением области определения этого отображения до множества *T*. Если для исходной концептуализации *C* построена онтология *O*, то редукция концептуализации, определяемой онтологией *O*, не обязательно будет совпадать с исходной концептуализацией *C*. Будем называть **онтологию адекватной** исходной концептуализации, если эта исходная концептуализация является подмножеством редукции концептуализации, определяемой этой онтологией. Также будем называть **онтологию точной**, если эти две концептуализации совпадают.

Нетривиальная концептуализация задает смысл названий понятий неявно, через бесконечное множество своих элементов, которое не может быть явно представлено. Такая концептуализация существует только в сознании носителей информации и соответствует интуитивному пониманию смысла названий понятий этими носителями информации. Разные носители информации не могут узнать на основе своих концептуализаций, одинаково ли они понимают смысл одних и тех же названий понятий. Напротив, онтология определяет смысл названий понятий явно, через явную спецификацию этого бесконечного множества с помощью конечного описания. Онтология может изучаться методами рационального познания. Носители информации могут сравнивать свои онтологии и по результатам этого сравнения устанавливать, одинаково или нет они понимают смысл одних и тех же названий понятий.

7.3. Прикладная логическая теория

**Прикладная логическая теория** с именем *Т* есть пара *<TS, SS>*, где *TS* есть конечное множество (возможно пустое) имен других прикладных логических теорий, а *SS* есть конечное множество (возможно пустое) предложений. Прикладные логические теории используются в качестве математических моделей онтологий. Имя *T* прикладной логической теории совпадает с именем моделируемой онтологии. Множество *TS* представляет множество названий других онтологий (возможно пустое), являющихся частями данной онтологии. Множество *SS* представляет множество определений остальных понятий онтологии (не входящих в онтологии, имена которых являются элементами первого множества) и множество онтологических соглашений.

Любая прикладная логическая теория *Т = <TS, SS>* по определению эквивалентна прикладной логической теории *<∅, SS'>*, где *SS'* есть результат следующего процесса. Обозначим *ts(T) = TS*, *ss(T) = SS*. Положим *TS1 = ts(T)*, *SS1 = ss(T)*. Для каждого *i = 1, 2, …* положим *TSi+1 = (∪(t ∈ TSi) ts(t))*, *SSi+1 = SSi ∪ (∪(t ∈ TSi) ss(t))*. Если *TSn = ∅* на текущем шаге *n*, то *SS' = SSn*. Прикладная логическая теория *<∅, SS'>* называется **редукцией** теории *<TS, SS>*. Переход от прикладной логической теории к ее редукции показывает, каким образом другие онтологии используются в качестве частей данной онтологии.

7.4. Язык прикладной логики

Прикладные логические теории будут представляться на **языке прикладной логики**. Язык прикладной логики является расширяемым и может рассматриваться как язык всей математики с унифицированным синтаксисом. Язык состоит из **ядра**, **стандартного расширения** и произвольного числа **специализированных расширений**. Каждое расширение имеет **название**. Стандартное расширение называется *ST*. **Имя** прикладной логической теории имеет вид *T(E1, E2,…, Ek)*, где *Т* – имя теории, а *E1, E2,…, Ek* – названия расширений языка, использованных для представления теории. Средства ядра могут быть использованы в представлении любой теории.

Язык допускает три типа предложений – **описание сорта имени**, **описание значения имени** и **ограничение на интерпретацию имен**. Предложения первого типа (в упрощенном варианте) имеют вид *сорт n: t*, где *n* есть имя, а *t* – терм, значением которого является множество. Это множество называется **сортом** этого имени. Каждое имя в прикладной логической теории может иметь не более одного сорта. Каждое описание сорта имени сопоставляет имени *n* его сорт – значение терма *t*. Описание сорта имени представляет определение основного понятия, обозначенного термином *n*. Терм *t* представляет объем определяемого понятия. Предложения второго типа (в упрощенном варианте) имеют вид *n ≡ t*, где *n* есть имя, а *t* – терм Значение этого терма называется **значением** этого имени. Каждое имя в прикладной логической теории может иметь не более одного значения. Каждое описание значения имени сопоставляет имени *n* его значение – значение терма *t*. Описание значения имени представляет определение вспомогательного понятия, обозначенного термином *n*. Терм *t* представляет значение определяемого понятия. Ограничение на интерпретацию имен в упрощенном варианте является формулой. Оно представляет онтологическое соглашение.

В общем случае каждое **предложение** состоит из множества описаний переменных и тела. **Множество описаний переменных** есть возможно пустое множество описаний переменных *(v1: t1) (v2: t2)… (vm: tm)*, где *(vi: ti)* – **описание переменной**, *vi* – **переменная** (некоторое имя, не совпадающее ни с одним именем понятия в прикладной логической теории), *ti* – терм, имеющий значением некоторое множество (область возможных значение переменной) для всех *i = 1,…, m*. Если переменная не связана в терме или формуле, то она является **свободной** в этом терме или в этой формуле. Если переменная связана в терме или формуле, то она **связана** и в предложении, содержащем этот терм или эту формулу. Все переменные *v1, v2,…, vm* попарно различны и являются свободными переменными предложения. Других свободных переменных предложение не имеет.

Тело предложения зависит от его типа. Для определения сорта имени тело предложения имеет вид *сорт t1: t2*, где *t1* есть терм (зависящий от переменных *v1, v2,…, vm*), имеющий значением имя, а *t2* – терм (возможно зависящий от переменных *v1, v2,…, vm*), значением которого является множество. Для определения значения имени тело предложения имеет вид *t1 ≡ t2*, где *t1* есть терм (зависящий от переменных *v1, v2,…, vm*), имеющий значением имя, а *t2* – терм (возможно зависящий от переменных *v1, v2,…, vm*). Для ограничения на интерпретацию имен тело предложения является формулой (зависящей от переменных *v1, v2,…, vm*).

Ядро языка, а также его расширения определяют возможные типы термов и формул. Определим здесь два вида **термов** ядра языка:

*n*, где *n* есть имя;

*v*, где *v* есть переменная.

Определим также некоторые виды **формул** ядра языка прикладной логики:

*¬ f*, где *f* – формула, причем *¬ f* истинна тогда и только тогда, когда ложна *f*;

*f1 & f2*, где *f1* и *f2* – формулы, причем *f1 & f2* истинна тогда и только тогда, когда истинны и *f1*, и *f2*;

*f1 ∨ f2*, где *f1* и *f2* – формулы, причем *f1 ∨ f2* истинна тогда и только тогда, когда истинна хотя бы одна из формул *f1*, и *f2*;

*f1 ⇒ f2*, где *f1* и *f2* – формулы, причем *f1 ⇒ f2* истинна тогда и только тогда, когда либо *f1*, ложна, либо *f2* истинна;

*f1 ⇔ f2*, где *f1* и *f2* – формулы, причем *f1 ⇔ f2* истинна тогда и только тогда, когда либо и *f1*, и *f2* обе истинны, либо и *f1*, и *f2* обе ложны.

Определим здесь также некоторые виды **формул** стандартного расширения языка прикладной логики:

*t1 = t2*, где *t1* и *t2* – термы, причем *t1 = t2* истинно тогда и только тогда, когда значения термов *t1* и *t2* равны;

*t1 ≠ t2*, где *t1* и *t2* – термы, причем *t1 ≠ t2* истинно тогда и только тогда, когда значения термов *t1* и *t2* не равны.

7.5. Семантика языка прикладной логики

**Семантика** языка прикладной логикиопределяет множество допустимых функций интерпретации имен, определенных в прикладной логической теории. **Функция интерпретации** α сопоставляет каждому имени, вводимому прикладной логической теорией, значение этого имени. Таким образом, функция интерпретации совпадает с вербальным представлением информации. Множество имен, вводимых прикладной логической теорией, может быть разбито на два непересекающихся подмножества: множество однозначно интерпретируемых имен и множество неоднозначно интерпретируемых имен. Имя является **однозначно интерпретируемым**, если выполнено одно из следующих условий:

- прикладная логическая теория не определяет ни сорта, ни значения для имени *n*; в этом случае для любой функции интерпретации *α* имеет место *α(n) = n*;

- прикладная логическая теория определяет значение *e* имени *n* и это значение не зависит от интерпретации других имен; в этом случае для любой функции интерпретации *α* имеет место *α(n) = e*;

- прикладная логическая теория определяет значение *e* имени *n* и это значение однозначно определяется интерпретацией других имен.

Все остальные имена являются **неоднозначно интерпретируемыми**. Они моделируют основные понятия онтологии. Для каждого такого имени прикладная логическая теория определяет сорт *s* этого имени, но не определяет его значения. В этом случае каждая допустимая функция интерпретации определяет значение этого имени и должна удовлетворять ограничению *α(n) ∈ s*.

Функция интерпретации *α* является **допустимой** для прикладной логической теории, если все предложения этой теории имеют смысл для этой функции интерпретации. Прикладная логическая теория является **семантически корректной**, если для нее существует допустимая функция интерпретации. Сужение допустимой функции интерпретации на множество неоднозначно интерпретируемых имен называется **моделью** этой **теории**. Множество моделей прикладной логической теории моделирует множество элементов концептуализации, определяемой моделью онтологии. Любая функция интерпретации прикладной логической теории может быть явно представлена с помощью другой прикладной логической теории, содержащей определения значений всех имен, введенных исходной прикладной логической теорией.

7.6. Семантика предложений языка прикладной логики

Префикс предложения определяет множество допустимых подстановок вместо свободных переменных этого предложения. Если префикс пуст, то и множество допустимых подстановок пусто. Обозначим *Jαθ(t)* – значение терма *t* при функции интерпретации *α* и подстановке *θ*, а *Jαθ(f)* – значение формулы *f* при функции интерпретации *α* и подстановке *θ*. Для префикса *(v1: t1) (v2: t2)… (vm: tm)* **множество допустимых подстановок** есть множество всех подстановок вида *θ = (v1 / c1, v2 / c2,…, vm / cm)*, где *c1 ∈ Jαθ(t1), c2 ∈ Jαθ(t2), …, cm ∈ Jαθ(tm)*.

Описание сорта имени, тело которого имеет вид *сорт t1****:*** *t2*, **имеет смысл**, если для каждой допустимой подстановки *Jαθ(t1)* является именем, а *Jαθ(t2)* – непустым множеством. Это **предложение имеет** следующую **семантику**: имя *Jαθ(t1)* имеет сорт *Jαθ(t2)*, т.е. функция интерпретации *α* является допустимой, если *α(Jαθ(t1)) ∈ Jαθ(t2)*.

Описание значения имени, тело которого имеет вид *t1* ***≡*** *t2*, **имеет смысл**, если для каждой допустимой подстановки *Jαθ(t1)* является именем, а *Jαθ(t2)* существует. Это предложение имеет следующую **семантику**: имя *Jαθ(t1)* имеет значение *Jαθ(t2)*, т.е. функция интерпретации *α* является допустимой, если *α(Jαθ(t1)) = Jαθ(t2)*.

Ограничение на интерпретацию имен, телом которого является формула *f*, **имеет смысл**, если для каждой допустимой подстановки *Jαθ(f)* существует. Это предложение имеет следующую **семантику**: функция интерпретации *α* является допустимой, если для каждой допустимой подстановки *θ* имеет место *Jαθ(f) = истина*.

7.7. Семантика некоторых термов и формул языка прикладной логики

Определим **семантику термов и формул**, введенных в разделе 7.4:

*Jαθ(n) = α(n)*, где *n* – имя;

*Jαθ(v)* есть значение *v* при подстановке *θ*, где *v* – переменная;

*Jαθ(¬ f) ⇔ ¬ Jαθ(f)*;

*Jαθ(f1 & f2) ⇔ Jαθ(f1) & Jαθ(f2)*;

*Jαθ(f1 ∨ f2) ⇔ Jαθ(f1) ∨ Jαθ(f2)*;

*Jαθ(f1 ⇒ f2) ⇔ Jαθ(f1) ⇒ Jαθ(f2)*;

*Jαθ(f1 ⇔ f2) ⇔ Jαθ(f1) ⇔ Jαθ(f2)*;

*Jαθ(t1 = t2) ⇔ Jαθ(t1) = Jαθ(t2)*;

*Jαθ(t1 ≠ t2) ⇔ Jαθ(t1) ≠ Jαθ(t2)*.

7.8. Описание идентификаторов в языках программирования

Большинство современных языков программирования содержит конструкцию "**описание идентификаторов**". Эта конструкция связывает в программе идентификатор с некоторым типом данных. Каждый идентификатор программы может принадлежать не более чем одному типу данных. Описание идентификаторов может рассматриваться как компьютерная модель определения основных понятий онтологии. Разница состоит в том, что, как правило, описание идентификатора связывает идентификатор с типом данных, а не с его подмножеством. Моделирование объемов понятий, обозначенных идентификаторами, определений вспомогательных понятий и онтологических соглашений выполняется другими операторами программы.

7.9. Представление онтологии в математических и компьютерных моделях

Легко видеть, что определения онтологии, прикладной логической теории и описаний идентификаторов похожи друг на друга. Разница между ними состоит лишь в том, какие объекты (значения величин, математические объекты или компьютерные объекты) входят в эти определения. Поэтому в математических моделях онтологии представляются прикладными логическими теориями, а в компьютерных – описаниями идентификаторов. Учитывая ранее сказанное, прикладные логические теории представляют также описания идентификаторов в математических моделях (когда объектом моделирования является компьютерная модель).

7.10. Состояние памяти

**Состояние памяти** – это соответствие между всеми идентификаторами программы и их значениями на некотором шаге вычислительного процесса. Значением каждого идентификатора является объект, принадлежащий типу данных этого идентификатора. Объекты различных типов вычисляются в программах с помощью выражений. Эти объекты сопоставляются идентификаторам в качестве их значений с помощью операторов присваивания.

8. СИСТЕМЫ ЗНАНИЙ

8.1. Система знаний

Теперь рассмотрим такую совокупность информации *I*, что каждая идея *i ∈ I* является вербализуемой, причем при представлении всех идей *i ∈ I* в виде элементов концептуализации используется одно и то же множество названий понятий *T*. Концептуализацию *C* будем называть **адекватной** для совокупности информации *I*, если любая идея *i ∈ I* может быть представлена как элемент концептуализации *C*. Однако не любой элемент концептуализации *C* будет представлять информацию, входящую в совокупность информации *I*. Совокупность информации *I*, содержащую бесконечное множество идей, будем называть **нетривиальной**. Нетривиальная совокупность информации не может быть представлена перечислением элементов концептуализации.

**Система знаний** о некоторой нетривиальной совокупности информации есть явная спецификация этой совокупности информации в терминах, определенных в онтологии, представляющей эту концептуализацию, и в терминах, в которых определена сама эта онтология. Отсюда следует, что система знаний есть такое определение нетривиальной совокупности информации, которое для каждой идеи из этой совокупности, представленной как элемент концептуализации, позволяет установить, является эта идея элементом этой совокупности информации или нет.

Система знаний может быть представлена в виде совокупности утверждений (законов) в терминах онтологии и терминах величин. Такие утверждения по форме (но не по содержанию!) ничем не отличаются от онтологических соглашений. Сама эта совокупность утверждений должна содержать ссылку на онтологию, в терминах которой она представлена. Кроме утверждений в эту совокупность могут входить определения вспомогательных (но не основных!) понятий, для упрощения записи утверждений. Элемент концептуализации, определяемой примитивной онтологией, соответствует системе знаний, если он соответствует онтологии и каждое утверждение этой системы знаний является истинным при условии, что в нем каждый термин, обозначающий основное понятие, заменен его значением в этом элементе концептуализации. Любое вербальное представление информации, соответствующее онтологии, для которого нарушено хотя бы одно утверждение из системы знаний, считается не принадлежащим к совокупности информации *I*.

Будем называть множество идей, представленных как элементы редукции концептуализации, определяемой онтологией, и которые соответствуют системе знаний, **совокупностью информации, определяемой системой знаний**. Если для исходной совокупности информации *I* построена система знаний *K*, то совокупность информации, определяемая системой знаний *K*, не обязательно будет совпадать с исходной совокупностью информации *I*. Будем называть систему знаний **адекватной** исходной совокупности информации, если эта исходная совокупность информации является подмножеством совокупности информации, определяемой этой системой знаний. Также будем называть систему знаний **точной**, если эти две совокупности информации совпадают.

Разница между онтологией и системой знаний состоит в том, что первая определяет смысл названий основных понятий концептуализации, а вторая, используя этот смысл, но не изменяя его, определяет нетривиальную совокупность информации как часть концептуализации. Разным совокупностям информации соответствуют разные системы знаний о них, но при этом онтология, в терминах которой представлены эти системы знаний, может быть одной и той же. Нетривиальная совокупность информации существует только в сознании носителей этой совокупности информации и соответствует интуитивному представлению о ней. Разные носители этой совокупности информации не могут узнать, одинаково ли они представляют себе эту совокупность. Напротив, система знаний определяет эту совокупность явно, через явную конечную спецификацию этого бесконечного множества в терминах концептуализации. Система знаний может изучаться методами рационального познания. Носители информации могут сравнивать свои системы знаний и по результатам этого сравнения устанавливать, одинаково ли они представляют себе эту совокупность информации.

8.2. Модель системы знаний

Модель системы знаний есть прикладная логическая теория, в которой множество имен других прикладных логических теорий не пусто и содержит название модели онтологии, в терминах которой представлена система знаний, а множество предложений состоит только из ограничений на интерпретацию имен и описаний значений имен. Ограничения на интерпретацию имен являются моделями законов, а описания значений имен – определениями вспомогательных понятий.

9. РАЗМЕРНЫЕ ВЕЛИЧИНЫ И ИХ ПРЕДСТАВЛЕНИЕ В МАТЕМАТИЧЕСКИХ И КОМПЬЮТЕРНЫХ МОДЕЛЯХ

9.1. Размерные величины

Одним из способов получения информации является измерение. **Измерение** – это получение информации о некоторой характеристике посредством сравнения ее с эталоном. Значение, которое получается при этом, называется размерным значением.

**Размерное значение** – это пара, состоящая из числа и обозначения эталона измерения. Обозначение эталона называется размерностью. **Размерность** представляется алгебраическим одночленом (произведением степеней первичных размерностей). **Первичная размерность** – это некоторый термин. **Безразмерная размерность** – это любая первичная размерность в нулевой степени.

Операции над размерными значениями обладают следующими свойствами. Если *R1* и *R2* – числа, а *s* – размерность, то

*(R1, s) + (R2, s) = (R1+ R2, s),*

*(R1, s) − (R2, s) = (R1− R2, s).*

Таким образом, все размерные значения с одной и той же размерностью образуют **простую размерную величину** относительно операций сложения и вычитания. В качестве **обозначения** этой размерной величины используется размерность образующих ее размерных значений. Кроме того, между размерными значениями такой величины имеют место следующие отношения.

*(R1, s) = (R2, s) ⇔ R1 = R2,*

*(R1, s) ≠ (R2, s) ⇔ R1 ≠ R2,*

*(R1, s) < (R2 ,s) ⇔ R1 < R2,*

*(R1, s) ≤ (R2, s) ⇔ R1 ≤ R2,*

*(R1, s) ≥ (R2, s) ⇔ R1 ≥ R2,*

*(R1, s) > (R2, s) ⇔ R1 > R2.*

Если *s1* и *s2* – размерности, то

*(R1,s1)\* (R2,s2) = (R1\* R2, s1\*s2),*

*(R1,s1) / (R2,s2) = (R1 / R2, s1 \* s2-1).*

Таким образом, все размерные значения произвольных размерностей образуют **сложную размерную величину**.

Например, измеряемыми характеристиками тела могут быть его длина, ширина и высота, а также его объем. Значениями первых трех характеристик тела в вербальном представлении информации о них являются положительные размерные значения, принадлежащие размерной величине *м* (метры), а значение последней характеристики – положительное размерное значение, принадлежащее размерной величине *м3* (кубические метры).

Чтобы установить, в какие простые размерные величины входят размерные значения, необходимо выяснить, какие измерения проводились при получении вербализуемой информации, и какие эталоны использовались в этих измерениях. Кроме того, как указывалось выше, обоснование системы размерных величин ссылается на систему размерных понятий, использованную при вербальном представлении этой информации, где эти величины или их подмножества являются объемами размерных понятий.

9.2. Алгебраическая система чисел

**Алгебраическая система чисел** определяется следующим образом: это моносортная алгебраическая система, носителем которой является множество всех вещественных чисел и которая замкнута относительно арифметических операций и отношений.

Если *(R, s)* есть размерное значение, где *R* – число, а *s* – размерность, то *R* является математической моделью *(R, s)*. Кроме того, арифметические операции и отношения являются математическими моделями соответствующих (имеющих то же обозначение) операций и отношений над размерными значениями. Строго говоря, алгебраическая система чисел не является адекватной моделью системы размерных величин, как это следует из определений, данных выше. Однако на практике именно алгебраическая система чисел традиционно используется в качестве математической модели системы размерных величин.

Переход от размерной величины к ее математической модели - алгебраической системе чисел ("обезразмеривание") состоит в "отбрасывании" размерностей при условии, что все соотношения между размерными понятиями (онтологические соглашения и знания) в системе понятий корректны, т.е. соответствуют формулам раздела 9.1. Например, в системе размерных понятий, включающей объем, ширину, высоту и длину тела, необходимо проверить, что размерности в правой и левой частях равенства, задающего формулу объема, совпадают. Результаты такой проверки обосновывают процесс обезразмеривания.

9.3. Математическая модель целого типа данных

**Математической моделью целого типа данных** является моносортная алгебраическая система, у которой носитель – это конечное множество всех целых чисел, не превосходящих по модулю максимального целого числа, представимого в компьютере, и которая замкнута относительно частично определённые операций сложения, вычитания и умножения, а также арифметических отношений. Арифметические операции определены для тех операндов, для которых их результат, полученный в алгебраической системе чисел, представим в компьютере, и которые являются сужением арифметических операций алгебраической системы чисел на область определенности этих частично определённых операций. Математическая модель целого типа данных является подсистемой алгебраической системы чисел. Сам целый тип данных является примитивным.

9.4. Математическая модель вещественного типа данных

**Математической моделью вещественного типа данных** является алгебраическая система, у которой носитель состоит из нуля и конечного множества всех рациональных чисел вида *m \* 2 ↑ p* таких, что *|m \* 2 ↑ p| ≤ e1*, где *p = ...,−2, −1, 0,1,2,...*, а *|m| = (1/2 + n \* e2)*, где *n* *= 0,1,2,..., (1 / (2 \* e2) − 1)*; здесь *р* называется **порядком** числа, *m* – его **мантиссой**, *e1* - максимальное вещественное число, представимое в компьютере, а *e2* - разность между единицей и ближайшим к единице вещественным числом, меньшим единицы, представимым в компьютере; эта алгебраическая система замкнута относительно частично определенных арифметических операций и отношений. Эти операции определены для тех операндов, для которых их результат, полученный в алгебраической системе чисел, представим в компьютере, и являются суперпозицией арифметических операций алгебраической системы чисел и операций округления (операции округления определяются устройством компьютера). Вещественный тип данных является примитивным.

9.5. Размерные понятия

Основное понятие называется **размерным**, если его объем есть простая размерная величина или ее подмножество. Вспомогательное понятие называется **размерным**, если его значение есть размерное значение. В примере раздела 9.1 размерными понятиями, определяемыми в онтологии размерных характеристик тел, являются объем (объем этого понятия совпадает с подмножеством положительных размерных значений размерной величины *м3*), ширина, длина и высота (объемы этих понятий совпадают с подмножеством положительных размерных значений размерной величины *м*) тела. Онтологическими соглашениями могут быть: длина тела есть длина наибольшего ребра описанного вокруг этого тела прямоугольного параллелепипеда наименьшего объема; ширина тела есть длина ребра этого параллелепипеда, не превосходящая длины тела; высота тела есть длина ребра этого параллелепипеда, не превосходящая ширины тела; объем тела не превосходит объема описанного вокруг этого тела прямоугольного параллелепипеда наименьшего объема. Если рассматривается подмножество концептуализации, относящееся только к прямоугольным параллелепипедам, то знания об этом подмножестве состоят из соотношения между объемом прямоугольного параллелепипеда и его шириной, длиной и высотой. Онтология, в которую входят только размерные понятия, образует **онтологию размерных понятий**.

9.6. Термы языка прикладной логики, значениями которых являются числовые множества

**Термом стандартного расширения языка прикладной логики, значение которого есть числовое множество**, является *R*, причем значение терма *Jαθ(R)* - носитель алгебраической системы чисел.

**Термами специализированного расширения ИНТЕРВАЛЫ, значениями которых суть числовые множества**, являются:

*R[t1, t2]*, где *t1* и *t2* – термы, значениями которых являются числа, причем значением терма *Jαθ(R[t1, t2])* является подмножество носителя алгебраической системы чисел, элементы которого заключены между значениями термов *Jαθ(t1)* и *Jαθ(t2)*, включая их самих; значение определяемого терма существует, если значение терма *Jαθ(t1)* не больше, чем значение терма *Jαθ(t2)*;

*R(t1, t2]*, где *t1* и *t2* – термы, значениями которых являются числа, причем значением терма *Jαθ(R(t1, t2])* является подмножество носителя алгебраической системы чисел, элементы которого заключены между значениями термов *Jαθ(t1)* и *Jαθ(t2)*, исключая *Jαθ(t1)* и включая *Jαθ(t2)*; значение определяемого терма существует, если значение терма *Jαθ(t1)* меньше, чем значение терма *Jαθ(t2)*;

*R[t1, t2)*, где *t1* и *t2* – термы, значениями которых являются числа, причем значением терма *Jαθ(R[t1, t2))* является подмножество носителя алгебраической системы чисел, элементы которого заключены между значениями термов *Jαθ(t1)* и *Jαθ(t2)*, исключая *Jαθ(t2)* и включая *Jαθ(t1)*; значение определяемого терма существует, если значение терма *Jαθ(t1)* меньше, чем значение терма *Jαθ(t2)*;

*R(t1, t2)*, где *t1* и *t2* – термы, значениями которых являются числа, причем значением терма *Jαθ(R(t1, t2))* является подмножество носителя алгебраической системы чисел, элементы которого заключены между значениями термов *Jαθ(t1)* и *Jαθ(t2)*, исключая их самих; значение определяемого терма существует, если значение терма *Jαθ(t1)* меньше, чем значение терма *Jαθ(t2)*;

*R[t, ∞)*, где *t* – терм, значением которого является число, причем значением терма *Jαθ(R[t, ∞))* является подмножество носителя алгебраической системы чисел, элементы которого не меньше значения терма *Jαθ(t)*;

*R(t, ∞)*, где *t* – терм, значением которого является число, причем значением терма *Jαθ(R(t, ∞))* является подмножество носителя алгебраической системы чисел, элементы которого больше значения терма *Jαθ(t)*;

*R(-∞, t]*, где *t* – терм, значением которого является число, причем значением терма *Jαθ(R(-∞, t])* является подмножество носителя алгебраической системы чисел, элементы которого не больше значения терма *Jαθ(t)*;

*R(-∞, t)*, где *t* – терм, значением которого является число, причем значением терма *Jαθ(R(-∞, t))* является подмножество носителя алгебраической системы чисел, элементы которого меньше значения терма *Jαθ(t)*;

*I*, причем значением терма *Jαθ(I)* является подмножество всех целых чисел носителя алгебраической системы чисел;

*I[t1, t2]*, где *t1* и *t2* – термы, значениями которых являются целые числа, причем значением терма *Jαθ(I[t1, t2])* является подмножество множества целых чисел, элементы которого заключены между значениями термов *Jαθ(t1)* и *Jαθ(t2)*, включая их самих; значение определяемого терма существует, если значение терма *Jαθ(t1)* не больше, чем значение терма *Jαθ(t2)*;

*I[t, ∞)*, где *t* – терм, значением которого является целое число, причем значением терма *Jαθ(I[t, ∞))* является подмножество множества целых чисел, элементы которого не меньше значения терма *Jαθ(t)*;

*I(-∞, t]*, где *t* – терм, значением которого является целое число, причем значением терма *Jαθ(I(-∞, t])* является подмножество множества целых чисел, элементы которого не больше значения терма *Jαθ(t)*.

Термы языка прикладной логики, значениями которых являются множества чисел, позволяют сопоставлять именам, определяемым прикладной логической теорией, эти множества чисел в качестве их сортов с помощью описаний сортов имен.

Прикладная логическая теория, моделирующая онтологию размерных характеристик тел, имеет следующий вид.

Онтология размерных характеристик тел(ST, ИНТЕРВАЛЫ):

{}

***сорт*** *длина: R(0, ∞)*

***сорт*** *высота: R(0, ширина]*

***сорт*** *ширина: R(0, длина]*

***сорт*** *объем: R(0, длина \* ширина \* высота]*

9.7. Термы языка прикладной логики, значениями которых являются числа, и формулы, представляющие отношения над числами

**Термами стандартного расширения языка прикладной логики, значениями которых суть числа**, являются:

числовые константы, изображаемые общепринятым способом, причем их значениями являются изображаемые ими числа (целочисленные константы изображают целые числа);

*t1 + t2*, где *t1* и *t2* – термы, значениями которых являются числа, причем *Jαθ(t1 + t2) = Jαθ(t1) + Jαθ(t2)*;

*t1 - t2*, где *t1* и *t2* – термы, значениями которых являются числа, причем *Jαθ(t1 - t2) = Jαθ(t1) - Jαθ(t2)*;

*t1 \* t2*, где *t1* и *t2* – термы, значениями которых являются числа, причем *Jαθ(t1 \* t2) = Jαθ(t1) \* Jαθ(t2)*;

*t1 / t2*, где *t1* и *t2* – термы, значениями которых являются числа, причем *Jαθ(t1 / t2) = Jαθ(t1) / Jαθ(t2)*; значение *Jαθ(t1 / t2)* существует, если *Jαθ(t2)* не равно нулю;

*t1 ↑ t2*, где *t1* и *t2* – термы, значениями которых являются числа, причем *Jαθ(t1 ↑ t2) = Jαθ(t1) ↑ Jαθ(t2)*; значение *Jαθ(t1 ↑ t2)* существует, если существует степень числа, равного *Jαθ(t1)*, с показателем, равным *Jαθ(t2)*.

**Формулами стандартного расширения языка прикладной логики для сравнения термов с числовыми значениями** являются:

*t1 > t2*, где *t1* и *t2* - термы, значениями которых являются числа, причем *Jαθ(t1 > t2) ⇔ Jαθ(t1) > Jαθ(t2)*;

*t1 < t2*, где *t1* и *t2* - термы, значениями которых являются числа, причем *Jαθ(t1 < t2) ⇔ Jαθ(t1) < Jαθ(t2)*;

*t1 ≤ t2*, где *t1* и *t2* - термы, значениями которых являются числа, причем *Jαθ(t1 ≤ t2) ⇔ Jαθ(t1) ≤ Jαθ(t2)*;

*t1 ≥ t2*, где *t1* и *t2* - термы, значениями которых являются числа, причем *Jαθ(t1 ≥ t2) ⇔ Jαθ(t1) ≥ Jαθ(t2)*.

Моделью системы знаний о размерных характеристиках прямоугольных параллелепипедов в терминах модели онтологии примера в разделе 9.6, является прикладная логическая теория, имеющая следующий вид.

Знания о размерных характеристиках прямоугольных параллелепипедов(ST):

{Онтология размерных характеристик тел(ST, ИНТЕРВАЛЫ)}

*объем = длина \* высота \* ширина*

9.8. Описания идентификаторов целого и вещественного типов в программе

**Идентификаторы целого и вещественного типов** моделируют размерные понятия. В свою очередь, математическими моделями и тех, и других являются имена, вводимые прикладной логической теорией, сортами которых являются множества целых и вещественных чисел.

9.9. Размерные значения в вербальном представлении информации, числа в качестве значений имен в математических моделях, значения идентификаторов целого и вещественного типов в состоянии памяти

Значениями размерных понятий в вербальном представлении информации являются размерные значения. Например, с задачей о вычислении объема прямоугольного параллелепипеда по его ширине, высоте и длине может быть связана вербализуемая информация, в которой длина параллелепипеда равна *5м*, ширина – *7м*, высота – *1м*, а объем – *35м3*. В математических моделях вербализуемой информации значения размерных понятий моделируются **числовыми значениями имен**.

Термы, значениями которых являются числа, позволяют сопоставлять именам, определяемым прикладной логической теорией, эти числовые значения с помощью описаний значений имен. Например, модель вербализуемой информации, приведенной в начале этого раздела, может быть представлена следующей прикладной логической теорией:

*длина ≡ 5*

*ширина ≡ 7*

*высота ≡ 1*

*объем ≡ 35*.

Значения целого и вещественного типов в программах вычисляются с помощью арифметических выражений. Значения размерных понятий представляются в состояниях памяти значениями идентификаторов числовых типов.

Задание N2 (по теме "Размерные величины")

Придумать пример вербализуемой информации, представляемой с помощью размерных значений.

План ответа

1. Вербализуемая информация и ее вербальное представление.

2. Размерные величины, элементы которых использованы для представления этой информации.

3. Онтология, в терминах которой представлена эта вербализуемая информация.

4. Модель этой онтологии на языке прикладной логики.

5. Содержательное описание подмножества концептуализации, определяемой онтологией, к которому принадлежит вербализуемая информация.

6. Система знаний об этом подмножестве в терминах онтологии.

7. Модель этой системы знаний на языке прикладной логики.

8. Обоснование процесса "обезразмеривания".

9. Прикладная задача в терминах онтологии, постановка которой включает систему знаний (что дано, что надо найти).

10. Математическая постановка этой задачи в терминах модели онтологии (что дано, что надо найти). Аргументы в пользу того, что задача имеет решение. Сколько этих решений?

11. Алгоритм решения этой задачи. Аргументы в пользу того, что алгоритм находит все решения и не находит лишних решений.

12. Программа (на любом ЯП, **но с комментариями**!), реализующая этот алгоритм.

13. Как в этой программе моделируются размерные величины, размерные понятия и знания.

14. Пример состояния памяти на шаге завершения программы, выполняемой с исходными данными, соответствующими вербализуемой информации.

10. СКАЛЯРНЫЕ ВЕЛИЧИНЫ И ИХ ПРЕДСТАВЛЕНИЕ В МАТЕМАТИЧЕСКИХ И КОМПЬЮТЕРНЫХ МОДЕЛЯХ

10.1. Скалярные величины

Еще одним способом получения информации является идентификация. **Идентификация** – это узнавание объекта или значения его свойства наблюдателем, непосредственно (с помощью органов чувств) или с помощью экспертизы. При идентификации информация об идентифицированном объекте представляется скалярным значением. **Скалярное значение** - это термин, предназначенный для обозначения этой информации.

Скалярные значения образуют простые **скалярные величины**, замкнутые относительно отношений равенства и неравенства.

Например, размерными понятиями при вербальном представлении информации в платежных ведомостях за тепло могут быть оплата за тепло (объем - неотрицательные значения величины *руб*), жилая площадь (объем - положительные значения величины *м2*), тариф (объем - неотрицательные значения величины *руб \* м -2*), а скалярным понятием – время года (объем - множество скалярных идентифицируемых значений зима, весна, лето, осень), от которого зависит тариф.

Чтобы установить, в какие скалярные величины входят скалярные значения, необходимо выяснить, какие идентификации проводились при получении вербализуемой информации. Кроме того, как указывалось выше, обоснование скалярных величин ссылается на систему скалярных понятий, использованную при вербальном представлении этой информации, где эти величины или их подмножества являются объемами скалярных понятий.

10.2. Алгебраические системы скалярных значений

Каждая **алгебраическая система скалярных значений** определяется следующим образом: - это моносортная алгебраическая система, носителем которой является некоторое конечное множество скалярных значений, и которая замкнута относительно отношений равенства и неравенства. Алгебраическая система скалярных значений является одновременно и (вырожденной) реляционной системой, и (вырожденной) алгеброй.

10.3. Модели скалярных типов данных

**Модель** любого **скалярного типа данных** совпадает с подходящей алгебраической системой скалярных значений. Каждый скалярный тип данных является примитивным.

10.4. Скалярные понятия

**Понятие** называется **скалярным**, если его объем состоит из скалярных значений одной и той же величины. В примере раздела 10.1 скалярным является основное понятие "*время года*", объем которого совпадает со скалярной величиной, состоящей из четырех скалярных значений – *зима, весна, лето, осень*.

Кроме того, в онтологию, в терминах которой может быть представлена вербализуемая информация этого примера, входят основные понятия "*оплата*", «*площадь*» и "*тариф*". Онтологическое соглашение устанавливает связь между этими понятиями (оплата равна произведению тарифа на площадь). Если рассматривается информация о платежных ведомостях за тепло в конкретном регионе (например, в Приморском крае) и в конкретном году (например, в 2005), то система знаний может содержать определение значения размерного понятия «*тариф*» в терминах онтологии следующим образом: если *время года* – *зима*, то *100* *руб \* м -2*, если *время года* – *весна*, то *70 руб \* м -2*, если *время года* – *лето*, то *0 руб \* м -2*, если *время года* – *осень*, то *50 руб \* м -2*.

10.5. Терм языка прикладной логики для экстенсионального задания множеств

**Экстенсиональное задание множества** – это перечисление всех его элементов. Очевидно, что экстенсионально могут быть заданы только конечные множества. **Термом стандартного расширения** языка прикладной логики для экстенсионального задания множеств, является *{t1,…, tm}*, где *t1,…, tm* – термы, причем значением терма *Jαθ({t1,…, tm})* является множество, состоящее из элементов *Jαθ(t1),…, Jαθ(tm)*. Термы языка прикладной логики для экстенсионального задания множеств, позволяют сопоставлять именам, определяемым прикладной логической теорией, множества имен в качестве их сортов с помощью описаний сортов имен.

10.6. Терм языка прикладной логики для представления скалярных значений

**Термом ядра** языка прикладной логики для представления скалярных значений является такое имя *n*, которое не имеет ни сорта, ни значения; значением этого терма является само имя *n*, т.е. *Jαθ(n) = n*.

10.7. Условный терм языка прикладной логики

Для задания на языке прикладной логики определений и утверждений, в которых используются скалярные значения, часто удобным оказывается **условный терм стандартного расширения** языка, который имеет вид */(f1 ⇒ t1)…(fm ⇒ tm)/. Jαθ(/(f1 ⇒ t1)…(fm ⇒ tm)/)* существует, если существует и единственно такое натуральное *i* от *1* до *m*, что *Jαθ(fi)* истинно, все *Jαθ(f1), …, Jαθ(fi-1), Jαθ(fi+1), …, Jαθ(fm)* ложны и существует *Jαθ(ti)*. В этом случае *Jαθ(/(f1 ⇒ t1)…(fm ⇒ tm)/) = Jαθ(ti)*.

Прикладная логическая теория, моделирующая онтологию квитанций оплаты за тепло, имеет следующий вид.

Онтология квитанций оплаты за тепло(ST, Интервалы)

{}

***сорт*** *оплата: R[0, ∞)*

***сорт*** *площадь: R(0, ∞)*

***сорт*** *тариф: R[0, ∞)*

***сорт*** *время года: {зима, весна, лето, осень}*

*оплата = тариф \* площадь.*

Прикладная логическая теория, моделирующая систему знаний о тарифе по оплате за тепло в Приморском крае в 2005 году, имеет следующий вид.

# Знания о тарифе по оплате за тепло в Приморском крае в 2005 году (ST)

# 

# {Онтология квитанций оплаты за тепло (ST, Интервалы)}

*тариф ≡ /(время года = зима ⇒ 100) (время года = весна ⇒ 70) (время года = лето ⇒ 0) (время года = осень ⇒ 50)/*.

10.8. Описания идентификаторов скалярных типов в программе

**Идентификаторы скалярных типов** моделируют скалярные понятия. В свою очередь, математическими моделями и тех, и других являются имена, вводимые прикладной логической теорией, сортами которых являются конечные множества имен.

10.9. Скалярные значения в вербальном представлении информации, имена в качестве значений имен в логических моделях, значения идентификаторов скалярных типов в состоянии памяти

Значениями скалярных понятий в вербальном представлении информации являются скалярные значения. Например, с задачей о вычислении оплаты за тепло в зависимости от площади и времени года, может быть связана вербализуемая информация (квитанция), в которой *оплата* равна *10000 руб*, *тариф* равен *100* *руб \* м -2*, *площадь* равна *100* *м2,*а *время года* - *зима*. В математических моделях вербализуемой информации значения скалярных понятий моделируются именами. Например, модель вербализуемой информации, приведенной в начале этого раздела, может быть представлена следующей прикладной логической теорией:

*оплата ≡ 10000*

*тариф ≡ 100*

*площадь ≡ 100*

*время года ≡ зима*

Значения скалярных понятий представляются в состояниях памяти элементами скалярных типов данных.

Задание N3 (по теме "Скалярные величины")

Придумать пример вербализуемой информации, представляемой с помощью размерных и скалярных значений.

План ответа

1. Вербализуемая информация и ее вербальное представление.

2. Размерные и скалярные величины, элементы которых использованы для представления этой информации.

3. Онтология, в терминах которой представлена эта вербализуемая информация.

4. Модель этой онтологии на языке прикладной логики.

5. Содержательное описание подмножества концептуализации, определяемой онтологией, к которому принадлежит вербализуемая информация.

6. Система знаний об этом подмножестве в терминах онтологии.

7. Модель этой системы знаний на языке прикладной логики.

8. Обоснование процесса "обезразмеривания".

9. Прикладная задача в терминах онтологии, постановка которой включает систему знаний (что дано, что надо найти).

10. Математическая постановка этой задачи в терминах модели онтологии (что дано, что надо найти). Аргументы в пользу того, что задача имеет решение. Сколько этих решений?

11. Алгоритм решения этой задачи. Аргументы в пользу того, что алгоритм находит все решения и не находит лишних решений.

12. Программа (на любом ЯП, **но с комментариями**!) реализующая этот алгоритм.

13. Как в этой программе моделируются размерные и скалярные величины, размерные и скалярные понятия и знания.

14. Пример состояния памяти на шаге завершения программы, выполняемой с исходными данными, соответствующими вербализуемой информации.

11. КОНЕЧНЫЕ МНОЖЕСТВА И ИХ ПРЕДСТАВЛЕНИЕ В МАТЕМАТИЧЕСКИХ И КОМПЬЮТЕРНЫХ МОДЕЛЯХ

11.1. Величины конечных множеств

Размерные и скалярные значения не имеют внутренней структуры. Напротив, значения, рассматриваемые далее, имеют внутреннюю структуру, т.е. они состоят из других значений, организованных некоторым образом в новое значение. Простейшей структурой является множество. Значение называется **конечным множеством**, если оно состоит из конечного числа значений, принадлежащих некоторой величине. **Величиной конечных множеств** называется множество всех конечных подмножеств некоторой величины. Каждая величина конечных множеств является **сложной величиной**, которая помимо множества конечных множеств содержит **базу** - величину, которой принадлежат все элементы этих множеств. Величина конечных множеств **замкнута** относительно операций объединения, пересечения и разности множеств, функции вычисления мощности множества, а также отношений принадлежности элемента множеству и включения одного множества в другое.

Например, в вербальном представлении информации о составе студенческой группы в новом учебном году, ее составе в прошедшем учебном году, составе отчисленных из нее студентов, восстановленных в нее студентов и переведенных в нее студентов из других ВУЗов значениями названий понятий состав студенческой группы в новом учебном году, ее состав в прошедшем учебном году, состав отчисленных из нее студентов, восстановленных в нее студентов и переведенных в нее студентов из других ВУЗов являются конечные множества.

Чтобы установить, какие величины конечных множеств используются при вербальном представлении информации, необходимо выяснить, информация о каких объектах с внутренней структурой должна быть представлена, и какие из этих объектов имеют структуру множества. Кроме того, как указывалось выше, обоснование системы величин конечных множеств ссылается на систему понятий, использованную при вербальном представлении этой информации, где эти величины или их подмножества являются объемами понятий.

11.2. Алгебраические системы конечных множеств

Каждая **алгебраическая система конечных множеств** определяется как многосортная алгебраическая система, носитель которой состоит из базы - некоторой алгебраической системы, множества всех конечных подмножеств носителя базы. Алгебраическая система конечных множеств **замкнута** относительно теоретико-множественных операций объединения, пересечения и разности, функции вычисления мощности, а также отношений принадлежности и включения.

11.3. Модели типов данных конечных и разреженных множеств

Модель любого типа данных **конечных множеств** совпадает с подходящей алгебраической системой конечных множеств, носитель базы которой есть конечное множество. Модель любого типа данных **разреженных множеств** совпадает с подходящей алгебраической системой конечных множеств, носитель базы которой есть бесконечное множество. Каждый тип данных конечных и разреженных множеств является **непримитивным**.

11.4. Понятия, соответствующие конечным множествам

**Понятие** называется **соответствующим конечным множествам**, если его объем совпадает с некоторой величиной конечных множеств или является подмножеством такой величины. В примере раздела 11.1 понятиями, соответствующими конечным множествам, являются состав группы в текущем году (объем этого понятия совпадает с множеством всех конечных подмножеств множества всех студентов), состав отчисленных (объем этого понятия совпадает с множеством всех подмножеств значения понятия " состав группы в текущем году"), состав восстановленных (объем этого понятия совпадает с множеством всех конечных подмножеств множества всех студентов), состав переведенных (объем этого понятия совпадает с множеством всех конечных подмножеств множества всех студентов), и состав группы в следующем году (объем этого понятия совпадает с множеством всех конечных подмножеств множества всех студентов). В данном примере онтологическими соглашениями могут быть соотношения между значениями этих понятий.

11.5. Термы языка прикладной логики, значениями которых являются множества конечных множеств

**Термом стандартного расширения** языка прикладной логики, значение которого есть множество конечных множеств, является *{}t*, где *t* – терм, значением которого является множество, причем *Jαθ({}t)* существует, если *Jαθ(t)* есть множество, *Jαθ({}t)* есть множество всех конечных подмножеств множества *Jαθ(t)*. Кроме того, **термом специализированного расширения** "ИНТЕРВАЛЫ" языка прикладной логики, значение которого есть множество конечных множеств целых чисел, является *[]I*, причем *Jαθ([]I)* есть множество всех конечных целочисленных интервалов. Термы языка прикладной логики, значениями которых являются множества конечных множеств, позволяют сопоставлять именам, определяемым прикладной логической теорией, эти множества множеств в качестве их сортов с помощью описаний сортов имен.

11.6. Термы языка прикладной логики, значениями которых являются множества, термы и формулы, связанные с множествами

**Термом ядра** языка прикладной логики, значением которого является множество, является *N*, причем *Jαθ(N)* есть множество всех имен. Имена используются в качестве терминов (для обозначения понятий и скалярных значений), в качестве переменных и в качестве обозначений объектов предметной области.

**Термами стандартного расширения** языка прикладной логики, значения которых суть множества, являются:

*∅*, причем *Jαθ(∅)* есть пустое множество;

*{(v1: t1) (v2: t2)… (vm: tm) f}* – квантор интенсиональности, где *(v1: t1) (v2: t2)… (vm: tm)* есть множество описаний переменных, а *f* – формула языка прикладной логики, переменные *v1, v2, …, vm* связаны в терме *{(v1: t1) (v2: t2)… (vm: tm) f}*, причем *Jαθ({(v1: t1) (v2: t2)… (vm: tm) f})* есть множество всех таких кортежей элементов, что их подстановки вместо переменных *v1, v2, …, vm* являются допустимыми для множества описаний переменных *(v1: t1) (v2: t2)… (vm: tm)* и при этих подстановках в формулу *f* значение этой формулы истинно; **интенсиональное задание множества** – это формирование множества из элементов другого множества, обладающих некоторым общим свойством; синтаксический оператор, который преобразует множество описаний переменных и терм или формулу, зависящие от этих переменных, в терм или формулу, не зависящие от этих переменных, называется **квантором**; квантор интенсиональности преобразует множество описаний переменных и формулу в терм, не зависящий от этих переменных;

*{(v1: t1) (v2: t2)… (vm: tm) t}* – квантор преобразования множества, где *(v1: t1) (v2: t2)… (vm: tm)* есть множество описаний переменных, *t* – терм языка прикладной логики, переменные *v1, v2, …, vm* связаны в терме *{(v1: t1) (v2: t2)… (vm: tm) t}*, причем *Jαθ({(v1: t1) (v2: t2)… (vm: tm) t})* есть множество всех таких значений терма *t*, которые получаются при всех допустимых для множества описаний переменных *(v1: t1) (v2: t2)… (vm: tm)* подстановках в терм *t*; квантор преобразования множеств преобразует множество описаний переменных и терм в терм, не зависящий от этих переменных;

*t1 ∪ t2*, где *t1* и *t2* – термы, значения которых – множества, причем *Jαθ(t1 ∪ t2) = Jαθ(t1) ∪ Jαθ(t2)*;

*t1 ∩ t2*, где *t1* и *t2* – термы, значения которых – множества, причем *Jαθ(t1 ∩ t2) = Jαθ(t1) ∩ Jαθ(t2)*;

*t1 \ t2*, где *t1* и *t2* – термы, значения которых – множества, причем *Jαθ(t1 \ t2) = Jαθ(t1) \ Jαθ(t2)*;.

Кроме того, термами стандартного расширения являются:

*μ(t)*, где *t* – терм, имеющий значением конечное множество, причем *Jαθ(μ(t))* есть мощность множества *Jαθ(t)*;

*(ι(v1: t1) (v2: t2)… (vm: tm) f)* – йота-оператор (квантор), где *(v1: t1) (v2: t2)… (vm: tm)* есть множество описаний переменных, а *f* – формула языка прикладной логики, переменные *v1, v2, …, vm* связаны в терме *(ι(v1: t1) (v2: t2)… (vm: tm) f)*, причем *Jαθ((ι(v1: t1) (v2: t2)… (vm: tm) f))* есть такой кортеж элементов, что их подстановка вместо переменных *v1, v2, …, vm* является допустимой для множества описаний переменных *(v1: t1) (v2: t2)… (vm: tm)* и при подстановке этого кортежа в формулу *f* значение этой формулы истинно; значение терма существует, если такой элемент единственен; йота-оператор преобразует множество описаний переменных и формулу в терм, не зависящий от этих переменных.

**Формулами стандартного расширения** языка прикладной логики являются:

*t1 ∈ t2*, где *t1* и *t2* – термы, значением *t2* является множество, причем *Jαθ(t1 ∈ t2) ⇔ Jαθ(t1) ∈ Jαθ(t2)*;

*t1 ∉ t2*, где *t1* и *t2* – термы, значением *t2* является множество, причем *Jαθ(t1 ∉ t2) ⇔ Jαθ(t1) ∉ Jαθ(t2)*;

*t1 ⊂ t2*, где *t1* и *t2* – термы, значениями которых являются множества, причем *Jαθ(t1 ⊂ t2) ⇔ Jαθ(t1) ⊂ Jαθ(t2)*;

*t1 ⊆ t2*, где *t1* и *t2* – термы, значениями которых являются множества, причем *Jαθ(t1 ⊆ t2) ⇔ Jαθ(t1) ⊆ Jαθ(t2)*;

*t1 ⊄ t2*, где *t1* и *t2* – термы, значениями которых являются множества, причем *Jαθ(t1 ⊄ t2) ⇔ Jαθ(t1) ⊄ Jαθ(t2)*.

С помощью термов, значениями которых являются множества, в **специализированном расширении** МАТЕМАТИЧЕСКИЕ КВАНТОРЫ языка прикладной логики вводятся следующие **термы**:

*(Σ(v1: t1) (v2: t2)… (vm: tm) t)* – квантор суммирования, где *(v1: t1) (v2: t2)… (vm: tm)* есть множество описаний переменных, переменные *v1, v2, …, vm* связаны в терме *(Σ(v1: t1) (v2: t2)… (vm: tm) t)*, *t* – терм, имеющий числовые значения, причем *Jαθ((Σ(v1: t1) (v2: t2)… (vm: tm) t))* равно сумме значений *Jαθ(t)* при всех допустимых для множества описаний переменных *(v1: t1) (v2: t2)… (vm: tm)* подстановках в терм *t*; квантор суммирования преобразует множество описаний переменных и терм в терм, не зависящий от этих переменных;

*(∏(v1: t1) (v2: t2)… (vm: tm) t)* – квантор произведения, где *(v1: t1) (v2: t2)… (vm: tm)* есть множество описаний переменных, переменные *v1, v2, …, vm* связаны в терме *(∏(v1: t1) (v2: t2)… (vm: tm) t)*, *t* – терм, имеющий числовые значения, причем *Jαθ((∏(v1: t1) (v2: t2)… (vm: tm) t))* равно произведению значений *Jαθ(t)* при всех допустимых для множества описаний переменных *(v1: t1) (v2: t2)… (vm: tm)* подстановках в терм *t*; квантор произведения преобразует множество описаний переменных и терм в терм, не зависящий от этих переменных;

*(∪(v1: t1) (v2: t2)… (vm: tm) t)* – квантор объединения, где *(v1: t1) (v2: t2)… (vm: tm)* есть множество описаний переменных, переменные *v1, v2, …, vm* связаны в терме *(∪(v1: t1) (v2: t2)… (vm: tm) t)*, *t* – терм, имеющий значениями множества, причем *Jαθ((∪(v1: t1) (v2: t2)… (vm: tm) t))* равно объединению значений *Jαθ(t)* при всех допустимых для множества описаний переменных *(v1: t1) (v2: t2)… (vm: tm)* подстановках в терм *t*; квантор объединения преобразует множество описаний переменных и терм в терм, не зависящий от этих переменных;

*(∩(v1: t1) (v2: t2)… (vm: tm) t)* – квантор пересечения, где *(v1: t1) (v2: t2)… (vm: tm)* есть множество описаний переменных, переменные *v1, v2, …, vm* связаны в терме *(∩(v1: t1) (v2: t2)… (vm: tm) t)*, *t* – терм, имеющий значениями множества, причем *Jαθ((∩(v1: t1) (v2: t2)… (vm: tm) t))* равно пересечению значений *Jαθ(t)* при всех допустимых для множества описаний переменных *(v1: t1) (v2: t2)… (vm: tm)* подстановках в терм *t*; квантор пересечения преобразует множество описаний переменных и терм в терм, не зависящий от этих переменных.

Кроме того, в **специализированном расширении** МАТЕМАТИЧЕСКИЕ КВАНТОРЫ языка прикладной логики вводятся следующие **формулы**:

*(&(v1: t1) (v2: t2)… (vm: tm) f)* – квантор конъюнкции, где *(v1: t1) (v2: t2)… (vm: tm)* есть множество описаний переменных, переменные *v1, v2, …, vm* связаны в формуле *(&(v1: t1) (v2: t2)… (vm: tm) f)*, *f* – формула, причем *Jαθ((&(v1: t1) (v2: t2)… (vm: tm) t))* есть конъюнкция значений *Jαθ(f)* при всех допустимых для множества описаний переменных *(v1: t1) (v2: t2)… (vm: tm)* подстановках в формулу *f*; квантор конъюнкции преобразует множество описаний переменных и формулу в формулу, не зависящую от этих переменных;

*(∨(v1: t1) (v2: t2)… (vm: tm) f)* – квантор дизъюнкции, где *(v1: t1) (v2: t2)… (vm: tm)* есть множество описаний переменных, переменные *v1, v2, …, vm* связаны в формуле *(∨(v1: t1) (v2: t2)… (vm: tm) f)*, *f* – формула, причем *Jαθ((∨(v1: t1) (v2: t2)… (vm: tm) t))* есть дизъюнкция значений *Jαθ(f)* при всех допустимых для множества описаний переменных *(v1: t1) (v2: t2)… (vm: tm)* подстановках в формулу *f*; квантор конъюнкции преобразует множество описаний переменных и формулу в формулу, не зависящую от этих переменных.

Прикладная логическая теория, моделирующая онтологию студенческих групп, имеет следующий вид.

Студенческие группы(ST)

***сорт*** *группа в текущем году: {}N*

***сорт*** *отчисленные: {}группа в текущем году*

***сорт*** *восстановленные: {}N*

***сорт*** *переведенные: {}N*

***сорт*** *группа в следующем году: {}N*

*группа в следующем году = (группа в текущем году \ отчисленные) ∪ восстановленные ∪ переведенные*

*группа в текущем году ∩ восстановленные = ∅*

*группа в текущем году ∩ переведенные = ∅*

*восстановленные ∩ переведенные = ∅*

11.7. Описания идентификаторов типов данных конечных и разреженных множеств в программе

Идентификаторы типов данных конечных и разреженных множеств моделируют понятия предметной области, соответствующие конечным и разреженным множествам. В свою очередь, математическими моделями и тех, и других являются имена, вводимые прикладной логической теорией, сортами которых являются множества конечных множеств.

11.8. Множества в вербальном представлении информации, множества в качестве значений имен в логических моделях, значения идентификаторов типов конечных и разреженных множеств в состоянии памяти

Значениями понятий, соответствующих множествам, в вербальном представлении информации являются множества. В математических моделях вербализуемой информации такие значения также моделируются множествами. Термы, значениями которых являются множества, позволяют сопоставлять именам, определяемым прикладной логической теорией, эти множества в качестве значений с помощью описаний значений имен.

Значения понятий, соответствующих множествам, представляются в состояниях памяти значениями идентификаторов типов конечных и разреженных множеств.

Задание N4. (по теме "Величины конечных множеств")

Придумать пример вербализуемой информации, представляемой с помощью различных значений, в том числе конечных множеств.

План ответа

1. Вербализуемая информация и ее вербальное представление.

2. Величины конечных множеств и другие величины, элементы которых использованы для представления этой информации.

3. Онтология, в терминах которой представлена эта вербализуемая информация.

4. Модель этой онтологии на языке прикладной логики.

5. Содержательное описание подмножества концептуализации, определяемой онтологией, к которому принадлежит вербализуемая информация.

6. Система знаний об этом подмножестве в терминах онтологии.

7. Модель этой системы знаний на языке прикладной логики.

8. Прикладная задача в терминах онтологии, постановка которой включает систему знаний (что дано, что надо найти).

9. Математическая постановка этой задачи в терминах модели онтологии (что дано, что надо найти). Аргументы в пользу того, что задача имеет решение. Сколько этих решений?

10. Алгоритм решения этой задачи. Аргументы в пользу того, что алгоритм находит все решения и не находит лишних решений.

11. Программа (на любом ЯП, **но с комментариями**!) реализующая этот алгоритм.

12. Как в этой программе моделируются величины конечных множеств и другие величины, понятия и знания.

13. Пример состояния памяти на шаге завершения программы, выполняемой с исходными данными, соответствующими вербализуемой информации.

12. ОТОБРАЖЕНИЯ И ИХ ПРЕДСТАВЛЕНИЕ В МАТЕМАТИЧЕСКИХ И КОМПЬЮТЕРНЫХ МОДЕЛЯХ

12.1. Величины отображений

Значение называется **отображением**, если информация о нем может быть представлена как однозначное отображение некоторой величины в другую величину. Таким образом, отображения можно рассматривать и как значения, и как функциональными отношениями между значениями. **Величиной отображений** называется множество всех отображений с одними и теми же областями определения и значений. Каждая величина отображений является **сложной величиной**, которая помимо множества отображений содержит две величины – область определения и область значений. Сложная величина отображений **замкнута** относительно операции аппликации. **Аппликация** является операцией, первым операндом которой является отображение, а вторым – объект из области определения этого отображения. Результатом аппликации является значение отображения (первого операнда), когда его аргумент равен второму операнду.

Например, при вербальном представлении информации о совокупности тел могут быть использованы следующие величины отображений: множество всех отображений множества тел в размерную величину см3 (подмножество этой величины образует объем понятия объем), множество всех отображений множества тел в множество всех веществ (подмножество этой величины образует объем понятия вещество) и множество всех отображений множества тел в размерную величину *г* (подмножество этой величины образует объем понятия масса).

Чтобы установить, какие величины отображений используются при вербальном представлении информации, необходимо выяснить, информация о каких объектах с внутренней структурой должна быть представлена, и какие из этих объектов имеют структуру отображения. Кроме того, как указывалось выше, обоснование системы величин отображений ссылается на систему понятий, использованную при вербальном представлении этой информации, где эти величины или их подмножества являются объемами понятий.

12.2. Алгебраические системы отображений

Каждая **алгебраическая система отображений** определяется как многосортная алгебраическая система, носитель которой состоит из области определения - алгебраической системы и области значений –алгебраической системы, а также множества всех (всюду определенных или частичных) отображений области определения в область значений. Алгебраическая система отображений **замкнута** относительно операции аппликации.

12.3. Модели типов данных процедур, конечных, разреженных и ссылочных массивов

Модель любого типа данных **процедур** совпадает с подходящей алгебраической системой всюду определенных отображений. Модель любого типа данных **конечных массивов** совпадает с подходящей алгебраической системой всюду определенных отображений, у которых область определения и область значения являются конечными множествами. Модель любого типа данных **разреженных массивов** совпадает с подходящей алгебраической системой частично определенных отображений, у которых область определения является бесконечным множеством, а области определенности – конечными множествами. Модель любого типа данных **ссылочных массивов** совпадает с подходящей алгебраической системой отображений, у которых область значений является бесконечным множеством. Каждый тип данных процедур, конечных, разреженных и ссылочных массивов является **непримитивным**.

12.4. Понятия, соответствующие отображениям

**Понятие** называется **соответствующим отображениям**, если его объем состоит из отображений с одними и теми же областями определения и значений. В примере раздела 12.1 понятиями, соответствующими отображениям, являются объем (объем этого понятия совпадает с множеством всех отображений множества тел на размерную величину *см3*), вещество (объем этого понятия совпадает с множеством всех отображений множества тел на множество веществ), масса (объем этого понятия совпадает с множеством всех отображений множества тел на размерную величину *г*).

Система знаний о совокупностях тел, состоящих из некоторого подмножества веществ, может содержать соотношение, определяющее значение понятия «*вещества*», определение вспомогательного размерного понятия «*плотность*», значение которого определяется в терминах онтологии как отображение (таблица)множества веществ в размерную величину *г\*см –3*, а также соотношения, определяющие способ вычисления массы тела и массы совокупности тел.

12.5. Термы языка прикладной логики, значениями которых являются множества отображений

**Термом ядра** языка прикладной логики, значение которого есть множество отображений, является *t1 → t2*, где *t1* и *t2* – термы, значения которых - множества, причем *Jαθ(t1 → t2)* существует, если *Jαθ(t1)* и *Jαθ(t2)* суть множества; *Jαθ(t1 → t2)* есть множество всех отображений множества *Jαθ(t1)* в множество *Jαθ(t2)*. Термы языка прикладной логики, значениями которых являются множества отображений, позволяют сопоставлять именам, определяемым прикладной логической теорией, эти множества отображений в качестве их сортов с помощью описаний сортов имен.

Кроме того, **термом** **ядра** языка прикладной логики является *t → L*, где *t* – терм, значение которого - множество, причем *Jαθ(t → L)* существует, если *Jαθ(t)* есть множество; *Jαθ(t → L)* есть множество всех предикатов, определенных на множестве *Jαθ(t)*. Отображения, принадлежащие значению терма *t1 → t2*, где *t2* не есть *L*, называются функциями.

Также **термом ядра** языка прикладной логики является *(× t1,…, tk)*, где *t1,…, tk* – термы, значения которых множества, причем *Jαθ((× t1,…, tk))* существует, если *Jαθ(t1), …, Jαθ(tk)* суть множества; *Jαθ((× t1,…, tk))* есть декартово произведение множеств *Jαθ(t1), …, Jαθ(tk)*. Использование декартова произведения множеств в качестве области определения отображений позволяет задавать термы, значениями которых являются множества многоместных отображений (функций и предикатов).

**Термом стандартного расширения** языка прикладной логики является *<t1,…, tk>*, где *t1,…, tk* – термы, причем *Jαθ(<t1,…, tk>)* есть кортеж, составленный из *Jαθ(t1),…, Jαθ(tk)*. Также **термами стандартного расширения** языка прикладной логики являются:

*π(t1, t2)*, где *t1* – терм, значением которого является натуральное число, не большее *k*, а *t2* – терм, значением которого является кортеж из *k* компонентов, причем *Jαθ(π(t1, t2))* есть проекция кортежа *Jαθ(t2)* с номером *Jαθ(t1)*;

*length(t)*, где *t* – терм, значением которого является кортеж, причем *Jαθ(length(t))* есть число компонент в кортеже *Jαθ(t)*.

Термы, значениями которых являются кортежи, позволяют сопоставлять именам, определяемым прикладной логической теорией, эти кортежи в качестве значений с помощью описаний значений имен.

12.6. Термы языка прикладной логики, значениями которых являются отображения, термы и формулы, связанные с отображениями

**Термами стандартного расширения** языка прикладной логики, значения которых суть отображения, являются:

*(λ (v1: t1)… (vm: tm) t)* – ламбда-квантор для функций, где *(v1: t1)… (vm: tm)* есть множество описаний переменных, *t* – терм, переменные *v1, …, vm* связаны в терме *(λ (v1: t1)… (vm: tm) t)*, причем *Jαθ((λ (v1: t1)… (vm: tm) t))* существует, если *Jαθ(t1), …, Jαθ(tm)* суть множества; *Jαθ((λ (v1: t1)… (vm: tm) t))* есть функция от аргументов *v1, …, vm*; если *a1 ∈ Jαθ(t1), …, am ∈ Jαθ(tm)*, то значение этой функции от аргументов *a1, …, am* есть *Jαθ(t)├v1*|*a1, …, vm*|*am┤*, т.е. значение терма *t* при замене в нем переменных *v1, …, vm* на значения *a1, …, am* соответственно;

*(λ (v1: t1)… (vm: tm) f)* – ламбда-квантор для предикатов, где *(v1: t1)… (vm: tm)* есть множество описаний переменных, *f* – формула, переменные *v1, …, vm* связаны в терме *(λ (v1: t1)… (vm: tm) f)*, причем *Jαθ((λ (v1: t1)… (vm: tm) f))* существует, если *Jαθ(t1), …, Jαθ(tm)* суть множества; *Jαθ((λ (v1: t1)… (vm: tm) f))* есть предикат от аргументов *v1, …, vm*; если *a1 ∈ Jαθ(t1), …, am ∈ Jαθ(tm)*, то значение этого предиката от аргументов *a1, …, am* есть *Jαθ(f)├v1*| *a1, …, vm*| *am┤*, т.е. значение формулы *f* при замене в ней переменных *v1, …, vm* на значения *a1, …, am* соответственно.

**Термом ядра** языка является *t(t')*, где *t* и *t'* – термы, если значением терма *t* является функция, причем *Jαθ(t(t'))* существует, если *Jαθ(t)*, является функцией, принадлежащей множеству *t1 → t2*, а *Jαθ(t') ∈ t1*; *Jαθ(t(t'))* есть значение функции *Jαθ(t)* от аргумента *Jαθ(t')*. Кроме того, т**ермом ядра** языка является *t(t1,…, tk)*, где *t* и *t1,…, tk* – термы, если значением терма *t* является функция, причем *Jαθ(t(t1,…, tk))* существует, если *Jαθ(t)*, является функцией, принадлежащей множеству *(× t1,…, tk) → t'*, а *Jαθ(t1) ∈ t1,…, Jαθ(tk) ∈ tk*; *Jαθ(t(t1,…, tk))* есть значение функции *Jαθ(t)* от аргументов *Jαθ(t1),…, Jαθ(tk)*.

**Формулой ядра** языка является *t(t')*, где *t* и *t'* – термы, если значением терма *t* является предикатом, причем *Jαθ(t(t'))* существует, если *Jαθ(t)*, является предикатом, принадлежащим множеству *t1 → L*, а *Jαθ(t') ∈ t1*; *Jαθ(t(t'))* есть значение предиката *Jαθ(t)* от аргумента *Jαθ(t')*. Кроме того, **формулой ядра** языка является *t(t1,…, tk)*, где *t* и *t1,…, tk* – термы, если значением терма *t* является предикат, причем *Jαθ(t(t1,…, tk))* существует, если *Jαθ(t)*, является предикатом, принадлежащим множеству *(× t1,…, tk) → L*, а *Jαθ(t1) ∈ t1,…, Jαθ(tk) ∈ tk*; *Jαθ(t(t1,…, tk))* есть значение предиката *Jαθ(t)* от аргументов *Jαθ(t1),…, Jαθ(tk)*.

Термы, значениями которых являются отображения, позволяют сопоставлять именам, определяемым прикладной логической теорией, эти отображения в качестве значений с помощью описаний значений имен.

Прикладная логическая теория, моделирующая онтологию размерных и скалярных характеристик совокупности тел, имеет следующий вид.

Онтология характеристик совокупности тел(ST, Интервалы)

{}

***сорт*** *тела: {}N*

***сорт*** *объем: тела → R(0, ∞)*

***сорт*** *масса: тела → R(0, ∞)*

***сорт*** *вещества: {}N*

***сорт*** *вещество: тела → вещества*

***сорт*** *масса совокупности тел: R(0, ∞)*

***сорт*** *плотность: вещества → R(0, ∞))*

*масса = (λ (v: тела) плотность(вещество(v)) \* объем(v))*

*масса совокупности тел = (Σ (v: тела) масса(v))*

# Прикладная логическая теория, моделирующая систему знаний о характеристиках совокупности тел, состоящих из железа, меди, серебра и золота, имеет следующий вид.

# Знания о характеристиках совокупности тел, состоящих из железа, меди, серебра и золота(ST)

# {Онтология характеристик совокупности тел(ST, Интервалы)}

*вещества = {железо, медь, серебро, золото}*

*плотность ≡ (λ (v: вещества) /( v = железо ⇒ 7.9) (v = медь ⇒ 8.4) (v = серебро ⇒ 10.5) (v = золото ⇒ 19.3)/)*

12.7. Описания идентификаторов типов данных массивов в программе

Идентификаторы процедур и типов данных конечных, разреженных и ссылочных массивов моделируют понятия, соответствующие отображениям. В свою очередь, математическими моделями и тех, и других являются имена, вводимые прикладной логической теорией, сортами которых являются множества отображений.

12.8. Отображения в вербальном представлении информации, отображения в качестве значений имен в логических моделях, значения идентификаторов процедур и типов конечных, разреженных и ссылочных массивов в состоянии памяти

Значениями понятий, соответствующих отображениям, в вербальном представлении информации являются отображения. В математических моделях вербализуемой информации такие значения также моделируются отображениями.

Значения понятий, соответствующих отображениям, представляются в состояниях памяти значениями идентификаторов процедур (кодами тел процедур) и типов конечных, разреженных и ссылочных массивов.

Задание N5. (по теме "Величины отображений")

Придумать пример вербализуемой информации, представляемой с помощью различных значений, в том числе отображений.

План ответа

1. Вербализуемая информация и ее вербальное представление.

2. Величины отображений и другие величины, элементы которых использованы для представления этой информации.

3. Онтология, в терминах которой представлена эта вербализуемая информация.

4. Модель этой онтологии на языке прикладной логики.

5. Содержательное описание подмножества концептуализации, определяемой онтологией, к которому принадлежит вербализуемая информация.

6. Система знаний об этом подмножестве в терминах онтологии.

7. Модель этой системы знаний на языке прикладной логики.

8. Прикладная задача в терминах онтологии, постановка которой включает систему знаний (что дано, что надо найти).

9. Математическая постановка этой задачи в терминах модели онтологии (что дано, что надо найти). Аргументы в пользу того, что задача имеет решение. Сколько этих решений?

10. Алгоритм решения этой задачи. Аргументы в пользу того, что алгоритм находит все решения и не находит лишних решений.

11. Программа (на любом ЯП, **но с комментариями**!) реализующая этот алгоритм.

12. Как в этой программе моделируются величины отображений и другие величины, понятия и знания.

13. Пример состояния памяти на шаге завершения программы, выполняемой с исходными данными, соответствующими вербализуемой информации.

13. ОБЪЕДИНЕННЫЕ ВЕЛИЧИНЫ

13.1. Объединённые величины

**Объединённая величина** есть объединение величин из некоторой конечной совокупности других величин. Объединенная величина является **сложной величиной**, которая помимо объединения множеств объектов величин, входящих в совокупность, содержит сами эти величины. Эта сложная величина **замкнута** относительно **отношения принадлежности объекта исходной величине**. Это отношение имеет место между объектом объединенной величины и одной из объединяемых величин, если объект входит в эту величину.

Например, при вербальном представлении информации об объемах тел правильной формы могут использоваться понятия тела, значением которого является конечное множество тел правильной формы, состоящее из шаров, кубов и прямоугольных параллелепипедов (объединенная величина), объем, значением которого является отображение множества тел в положительные значения размерной величины *см3*, шары, значением которого является конечное множество шаров, кубы, значением которого является конечное множество кубов, прямоугольные параллелепипеды, значением которого является конечное множество прямоугольных параллелепипедов, радиус, значением которого является отображение множества шаров в положительные значения размерной величины *см*, длина ребра, значением которого является отображение множества кубов в положительные значения размерной величины *см*, длина, значением которого является отображение множества прямоугольных параллелепипедов в положительные значения размерной величины *см*, ширина, значением которого является отображение множества прямоугольных параллелепипедов в положительные значения размерной величины *см*, высота, значением которого является отображение множества прямоугольных параллелепипедов в положительные значения размерной величины *см*.

13.2. Объединенные алгебраические системы

Каждая **объединенная алгебраическая система** определяется как многосортная алгебраическая система, носитель которой состоит из некоторой конечной совокупности алгебраических систем, а также объединения носителей всех алгебраических систем этой совокупности. Алгебраическая система **замкнута** относительно **отношения принадлежности элемента объединения носителей носителю одной из объединяемых алгебраических систем**. Оно имеет место между таким элементом объединения носителей совокупности и носителем одной из объединяемых систем, что этот элемент принадлежит и этому носителю.

13.3. Модели объединенных типов данных

Модель любого **объединенного типа данных** совпадает с подходящей объединенной алгебраической системой. Каждый объединенный тип данных является **непримитивным**. Отношение принадлежности объекта объединенного типа одному из объединяемых типов отсутствует в большинстве языков программирования и реализуется за счет использования специального представления объектов этих типов.

13.4. Понятия, соответствующие объединенным величинам

**Понятие** называется **соответствующим объединенной величине**, если его объем состоит из объектов этой объединенной величины.

Онтология, определяющая понятия, используемые в вербальном представлении примера раздела 13.1, может содержать следующие определения: объем понятия «*тела*» есть множество всех конечных подмножеств множества тел, а объем понятия «*объем*» есть множество всех отображений значения понятия «*тела*» в множество положительных значений величины *см3*. Кроме того, объем понятия «*шары*» есть множество всех конечных подмножеств шаров, объем понятия «*кубы*» есть множество всех конечных подмножеств кубов, а объем понятия «*прямоугольные параллелепипеды*» есть множество всех конечных подмножеств прямоугольных параллелепипедов, объем понятия «*радиус*» есть множество всех отображений значения понятия «*шары*» в множество положительных значений величины *см*, объем понятия «*длина ребра*» есть множество всех отображений значения понятия «*кубы*» в множество положительных значений величины *см*, объем понятия «*длина*» есть множество всех отображений значения понятия «*прямоугольные параллелепипеды*» в множество положительных значений величины *см*, объем понятия «*ширина*» есть множество всех отображений значения понятия «*прямоугольные параллелепипеды*» в множество положительных значений величины *см*, а объем понятия «*высота*» есть множество всех отображений значения понятия «*прямоугольные параллелепипеды*» в множество положительных значений величины *см*. Онтологические соглашения могут устанавливать, что для одного и того же прямоугольного параллелепипеда его длина не меньше ширины, а ширина не меньше высоты.

Система знаний, где тела могут быть только шарами, кубами или прямоугольными параллелепипедами, содержит равенство, определяющее значение понятия «*тела*» как объединение значений понятий «*шары*», «*кубы*» и «*прямоугольные параллелепипеды*», а также равенство, определяющее для значение понятия «*объем*» способ его вычисления в зависимости от формы тела.

13.5. Модель системы понятий, соответствующих объединенным величинам

Средств языка прикладной логики, введенных выше, достаточно, чтобы задавать объединенные сорта - модели определений понятий, соответствующих объединенным величинам.

Прикладная логическая теория, моделирующая онтологию рассмотренного выше примера, имеет следующий вид.

Онтология объемов тел (ST, Интервалы)

{}

***сорт*** *шары: {}N*

***сорт*** *радиус: шары → R(0, ∞)*

***сорт*** *кубы: {}N*

***сорт*** *длина ребра: кубы → R(0, ∞)*

***сорт*** *прямоугольные параллелепипеды: {}N*

***сорт*** *длина: прямоугольные параллелепипеды → R(0, ∞)*

***сорт*** *ширина: прямоугольные параллелепипеды → R(0, ∞)*

***сорт*** *высота: прямоугольные параллелепипеды → R(0, ∞)*

*(v: прямоугольные параллелепипеды) длина(v) ≥ ширина(v) &*

*& ширина(v) ≥ высота(v)*

***сорт*** *тела: {}N*

***сорт*** *объем: тела → R(0, ∞)*

Прикладная логическая теория, моделирующая систему знаний рассмотренного выше примера, имеет следующий вид.

Система знаний объемов тел правильной формы(ST, Интервалы)

{Онтология объемов тел, имеющих форму шара, куба или прямоугольного параллелепипеда(ST, Интервалы)}

*тела = шары ∪ кубы ∪ прямоугольные параллелепипеды*

*pi ≡ 3.1415*

*объем = (λ(v: тела) /(v ∈ шары ⇒ (4 / 3) \* (радиус(v) ↑ 3) \* pi) (v ∈ кубы ⇒ длина ребра(v) ↑ 3) (v ∈ прямоугольные параллелепипеды ⇒ длина(v) \* ширина(v) \* высота(v))/)*

13.6. Описания идентификаторов объединенных типов данных в программе

Идентификаторы объединенных типов данных моделируют понятия предметной области, соответствующие объединенным величинам. В свою очередь, математическими моделями и тех, и других являются имена, сорта которых вводятся прикладной логической теорией способом, указанным в предыдущем разделе.

13.8. Объекты объединенных величин в вербальном представлении информации, значения объединенных сортов в качестве значений имен в логических моделях, значения идентификаторов объединенных типов данных в состоянии памяти

Значениями понятий, соответствующих объединенным величинам, в вербальном представлении информации являются объекты объединенных величин. В логических моделях этих ситуаций они моделируются значениями имен объединенных сортов, а в состоянии памяти – значениями идентификаторов объединенных типов данных.

Задание N6. (по теме "Объединенные величины")

Придумать пример вербализуемой информации, представляемой с помощью различных значений, в том числе элементов объединенных величин.

План ответа

1. Вербализуемая информация и ее вербальное представление.

2. Объединенные величины и другие величины, элементы которых использованы для представления этой информации.

3. Онтология, в терминах которой представлена эта вербализуемая информация.

4. Модель этой онтологии на языке прикладной логики.

5. Содержательное описание подмножества концептуализации, определяемой онтологией, к которому принадлежит вербализуемая информация.

6. Система знаний об этом подмножестве в терминах онтологии.

7. Модель этой системы знаний на языке прикладной логики.

8. Прикладная задача в терминах онтологии, постановка которой включает систему знаний (что дано, что надо найти).

9. Математическая постановка этой задачи в терминах модели онтологии (что дано, что надо найти). Аргументы в пользу того, что задача имеет решение. Сколько этих решений?

10. Алгоритм решения этой задачи. Аргументы в пользу того, что алгоритм находит все решения и не находит лишних решений.

11. Программа (на любом ЯП, **но с комментариями**!) реализующая этот алгоритм.

12. Как в этой программе моделируются величины отображений и другие величины, понятия и знания.

13. Пример состояния памяти на шаге завершения программы, выполняемой с исходными данными, соответствующими вербализуемой информации.

14. СТРУКТУРНЫЕ ЗНАЧЕНИЯ И ИХ ПРЕДСТАВЛЕНИЕ В МАТЕМАТИЧЕСКИХ И КОМПЬЮТЕРНЫХ МОДЕЛЯХ

14.1. Структурные величины

**Структурным значением** называется информация, которая представлена как совокупность значений атрибутов. Каждый **атрибут** имеет свое **название** (термин) и **область значений** - подмножество некоторой величины. Таким образом, каждое структурное значение можно рассматривать как вербальное представление некоторой информации. **Структурной величиной** называется множество структурных значений, имеющих одну и ту же структуру, т.е. одну и ту же совокупность атрибутов. Каждый атрибут можно рассматривать как функцию, которая отображает структурную величину на область значений этого атрибута.

Каждая структурная величина является **сложной** величиной, которая помимо значений этой структурной величины содержит области значений всех атрибутов и **замкнута** относительно аппликации всех атрибутов как функций.

Величина называется **прямым потомком** некоторой структурной величины, если она или ее подмножество является областью значений одного или нескольких атрибутов этой структурной величины. Величина называется **потомком** некоторой структурной величины, если она либо является прямым потомком этой структурной величины, либо существует потомок этой структурной величины, чьим прямым потомком она является. Структурная величина называется **прямо рекурсивной**, если она имеет прямого потомка, который является объединением этой структурной величины и нескольких других, среди которых есть нерекурсивные величины. Структурная величина называется **косвенно рекурсивной**, если она имеет непрямого потомка, который является объединением этой структурной величины и нескольких других, среди которых есть нерекурсивные величины. Прямо или косвенно рекурсивная величина называется **рекурсивной**. Остальные структурные величины называются **нерекурсивными**.

Например, вербальное представление информации о совокупности известных предков человека содержит значения объединения прямо рекурсивных структурных величин - множества всех мужчин и всех женщин, атрибутами которых являются отец и мать, областями значений которых являются объединенные величины, состоящие соответственно из множества мужчин и множества женщин, объединенных со значением неизвестен.

14.2. Алгебраические системы категорий

Каждая **алгебраическая система категорий** определяется следующим образом: - это многосортная алгебра, носитель которой состоит из *n* алгебраических систем (областей значений атрибутов) и множества, являющегося областью определения *n* отображений (атрибутов) в носители этих алгебраических систем, и которая замкнута относительно аппликаций этих отображений.

Алгебраическая система называется **прямым потомком** некоторой алгебраической системы категорий, если она является одной из областей значений атрибутов. Алгебраическая система называется **потомком** некоторой алгебраической системы категорий, если она либо является прямым потомком этой алгебраической системы категорий, либо существует потомок этой алгебраической системы категорий, чьим прямым потомком она является. Алгебраическая система категорий называется **прямо рекурсивной**, если она имеет прямого потомка, который является объединением этой алгебраической системы категорий и нескольких других, среди которых есть нерекурсивные алгебраические системы. Алгебраическая система категорий называется **косвенно рекурсивной**, если она имеет непрямого потомка, который является объединением этой алгебраической системы категорий и нескольких других, среди которых есть нерекурсивные алгебраические системы. Прямо или косвенно рекурсивная алгебраическая система категорий называется **рекурсивной**. Остальные алгебраические системы категорий называются **нерекурсивными**.

14.3. Модель типа записей

Модель любого **типа записей** совпадает с подходящей алгебраической системой категорий. Каждый тип записей является **непримитивным**. Аппликации атрибутов моделируют **операцию выборки**. Значения атрибутов называются **полями**. Тип записей называется **рекурсивным** (**прямо рекурсивным**, **косвенно рекурсивным**), если моделирующая его алгебраическая система является рекурсивной (прямо рекурсивной, косвенно рекурсивной).

14.4. Структурные понятия

Понятие называется **структурным**, если его объем состоит из объектов структурной величины. В примере раздела 14.1 структурным понятием является человек. Объем этого понятия совпадает со значением вспомогательного понятия люди, значением которого является объединение множеств - значений понятий мужчины и женщины.

14.5. Модель структурной величины в языке прикладной логики

Терм языка прикладной логики, моделирующий структурную величину, называется категорией. Пусть значениями термов *t1, …, tk* являются множества, а *s1, …, sm* – имена. Тогда термом (категорией) является *(s1 → t1, …, sm → tm)*, причем *Jαθ((s1 → t1, …, sm → tm))* существует, если *Jαθ(t1), …, Jαθ(tm)*  суть множества; *Jαθ((s1 → t1, …, sm → tm))* есть категория, т.е. множество объектов, которые образуют общую область определения отображений *s1, …, sm* с областями значений *Jαθ(t1), …, Jαθ(tm)*, соответственно.

Прикладная логическая теория, моделирующая онтологию рассмотренного выше примера, имеет следующий вид.

Онтология предков человека (ST, Интервалы, Категории)

{}

***сорт*** *мужчины: {}N*

***сорт*** *женщины: {}N*

*люди ≡ мужчины ∪ женщины*

*(v: люди)* ***сорт*** *v:**(мать → (женщины ∪ {неизвестен}), отец → (мужчины ∪ {неизвестен}), предки → {}люди)*

*(v: люди) предки(v) = /(отец(v) = неизвестен ⇒ ∅), (отец(v) ≠ неизвестен ⇒ {отец(v)} ∪ предки(отец(v)))/ ∪ /(мать(v) = неизвестен ⇒ ∅), (мать(v) ≠ неизвестен ⇒ {мать(v)} ∪ предки(мать(v)))/*

Заметим, что последнее онтологическое соглашение определяет атрибут предки как (прямо) рекурсивный.

14.6. Описания идентификаторов типов записей в программе

Идентификаторы типов записей моделируют структурные понятия. В свою очередь, математическими моделями и тех, и других являются имена, сорта которых вводятся прикладной логической теорией способом, указанным в предыдущем разделе.

14.9. Структурные значения в вербальном представлении информации, значения сортов категорий в качестве значений имен в логических моделях, значения идентификаторов типов записей в состоянии памяти

Значениями структурных понятий в ситуациях являются структурные значения. В логических моделях этих ситуаций они моделируются значениями сортов категорий, а в состоянии памяти – значениями идентификаторов типа записей. Значения понятий, соответствующих структурным величинам, представляются в состояниях памяти значениями идентификаторов типов записей.

Задание N 7 (по теме "Рекурсивные структурные величины")

Придумать пример вербализуемой информации, представляемой с помощью различных значений, в том числе структурных значений рекурсивных величин.

План ответа

1. Вербализуемая информация и ее вербальное представление.

2. Рекурсивные структурные величины и другие величины, элементы которых использованы для представления этой информации.

3. Онтология, в терминах которой представлена эта вербализуемая информация.

4. Модель этой онтологии на языке прикладной логики.

5. Содержательное описание подмножества концептуализации, определяемой онтологией, к которому принадлежит вербализуемая информация.

6. Система знаний об этом подмножестве в терминах онтологии.

7. Модель этой системы знаний на языке прикладной логики.

8. Прикладная задача в терминах онтологии, постановка которой включает систему знаний (что дано, что надо найти).

9. Математическая постановка этой задачи в терминах модели онтологии (что дано, что надо найти). Аргументы в пользу того, что задача имеет решение. Сколько этих решений?

10. Алгоритм решения этой задачи. Аргументы в пользу того, что алгоритм находит все решения и не находит лишних решений.

11. Программа (на любом ЯП, **но с комментариями**!) реализующая этот алгоритм.

12. Как в этой программе моделируются рекурсивные структурные величины и другие величины, понятия и знания.

13. Пример состояния памяти на шаге завершения программы, выполняемой с исходными данными, соответствующими вербализуемой информации.

15. КОНЕЧНЫЕ ПОСЛЕДОВАТЕЛЬНОСТИ И ИХ ПРЕДСТАВЛЕНИЕ В МАТЕМАТИЧЕСКИХ И КОМПЬЮТЕРНЫХ МОДЕЛЯХ

15.1. Конечные последовательности

Значение называется **конечной последовательностью**, если оно состоит из конечного числа значений, принадлежащих некоторой величине и расположенных в определенном порядке. **Величиной конечных последовательностей** называется множество всех конечных последовательностей, составленных из значений некоторой величины. Каждая величина конечных последовательностей является **сложной величиной**, которая помимо множества конечных последовательностей содержит **базу** - величину, которой принадлежат все элементы этих последовательностей. Величина конечных последовательностей **замкнута** относительно операций конкатенации (сцепления двух последовательностей), следующего и предыдущего элементов последовательности, функций головы (отбрасывания последнего элемента), хвоста (отбрасывания первого элемента), первого и последнего элемента, вычисления длины последовательности, а также отношений подпоследовательности, совпадения с началом и совпадения с концом последовательности.

Например, в вербальном представлении информации о программе на языке МИЛАН значением единственного термина *программа* является конечная последовательность символов – синтаксически правильная программа на языке МИЛАН.

Чтобы установить, какие величины конечных последовательностей используются при вербальном представлении информации, необходимо выяснить, информация о каких объектах с внутренней структурой должна быть представлена, и какие из этих объектов имеют структуру последовательности. Кроме того, как указывалось выше, обоснование системы величин конечных последовательностей ссылается на систему понятий, использованную при вербальном представлении этой информации, где эти величины или их подмножества являются объемами понятий.

15.2. Алгебраические системы конечных последовательностей

Каждая **алгебраическая система конечных последовательностей** определяется как многосортная алгебраическая система, носитель которой состоит из базы - некоторой алгебраической системы, а также множества всех конечных последовательностей, составленных из элементов носителя базы. Алгебраическая система конечных последовательностей **замкнута** относительно операций конкатенации, следующего и предыдущего элементов, функций головы, хвоста и вычисления длины последовательности, а также отношений подпоследовательности, совпадения с началом и совпадения с концом последовательности.

15.3. Модели типов данных конечных последовательностей

Модель любого типа данных **конечных последовательностей** совпадает с подходящей алгебраической системой конечных последовательностей. Каждый тип данных конечных последовательностей является **непримитивным** и содержит бесконечное множество значений.

15.4. Понятия, соответствующие конечным последовательностям

**Понятие** называется **соответствующим конечным последовательностям**, если его объем совпадает с некоторой величиной конечных последовательностей или является подмножеством такой величины. В примере раздела 15.1 понятием, соответствующим конечным последовательностям, является *программа* (объем этого понятия совпадает с множеством всех синтаксически правильных программ на языке МИЛАН). В данном примере множество онтологических соглашений может быть представлено грамматикой языка МИЛАН:

<программа> ::= начало<идентификатор><последовательность операторов>конец<идентификатор>

<последовательность операторов> ::= <оператор> ⏐ <оператор>;<последовательность операторов>

<оператор> ::= <присваивание> ⏐ <условный оператор> ⏐ <цикл> ⏐ <обмен>

<присваивание> ::= <идентификатор>:=<выражение>

<условный оператор> ::= если<логическое выражение>то<последовательность операторов>все ⏐ если<логическое выражение>то<последовательность операторов>иначе<последовательность операторов>все

<цикл> ::= пока<логическое выражение> цк<последовательность операторов>кц

<обмен> ::= ввод<идентификатор> ⏐ вывод<выражение>

<выражение> ::= <фактор> ⏐ <фактор> + <выражение>

<фактор> ::= <первичное> ⏐ <фактор>\*<первичное>

<первичное> ::= <идентификатор> ⏐ <константа> ⏐ (<выражение>)

<логическое выражение> ::= <выражение>=<выражение> ⏐<выражение>≠<выражение>

<константа> ::= <цифра> ⏐ <константа><цифра>

<цифра> ::= 0 ⏐ 1 ⏐ 2 ⏐…⏐ 9

<идентификатор> ::= <буква> ⏐ <идентификатор><буква>

<буква> ::= А ⏐ Б ⏐…⏐ Я

15.5. Термы языка прикладной логики, значениями которых являются множества конечных последовательностей

**Термом специализированного расширения** «ПОСЛЕДОВА- ТЕЛЬНОСТИ» языка прикладной логики, значение которого есть множество конечных последовательностей, является *seq t*, где *t* – терм, значением которого является множество, причем *Jαθ(seq t)* существует, если *Jαθ(t)* есть множество, *Jαθ(seq t)* есть множество всех конечных последовательностей, составленных из элементов множества *Jαθ(t)*. Термы языка прикладной логики, значениями которых являются множества конечных последовательностей, позволяют сопоставлять именам, определяемым прикладной логической теорией, эти множества последовательностей в качестве их сортов с помощью описаний сортов имен.

15.6. Термы языка прикладной логики, значениями которых являются последовательности, термы и формулы, связанные с последовательностями

**Термами специализированного расширения** «ПОСЛЕДОВА- ТЕЛЬНОСТИ» языка прикладной логики, значения которых суть последовательности или их элементы, являются:

*Λ*, причем *Jαθ(Λ)* есть пустая последовательность;

*«t1 … tm»*, причем *Jαθ(«t1 … tm»)* – последовательность, составленная из элементов *Jαθ(t1), …, Jαθ(tm)*;

*t1 || t2*, где *t1* и *t2* – термы, значения которых – последовательности, причем *Jαθ(t1 || t2) = Jαθ(t1) || Jαθ(t2)* (конкатенация);

*next(t1, t2)*, где *t1* и *t2* – термы, значение *t1* – последовательность, а *t2* – элемент этой последовательности (кроме последнего); *Jαθ(next(t1, t2))* есть элемент последовательности *Jαθ(t1)*, следующий за элементом *Jαθ(t2)*;

*prev(t1, t2)*, где *t1* и *t2* – термы, значение *t1* – последовательность, а *t2* – элемент этой последовательности (кроме первого); *Jαθ(next(t1, t2))* есть элемент последовательности *Jαθ(t1)*, предшествующий элементу *Jαθ(t2)*;

*head(t)*, где *t*– терм, значением которого является последовательность; *Jαθ(head(t))* есть последовательность *Jαθ(t)*, из которой удален последний элемент;

*tail(t)*, где *t*– терм, значением которого является последовательность; *Jαθ(tail(t))* есть последовательность *Jαθ(t)*, из которой удален первый элемент;

*first(t)*, где *t*– терм, значением которого является последовательность; *Jαθ(first(t))* есть первый элемент последовательности *Jαθ(t)*;

*last(t)*, где *t*– терм, значением которого является последовательность; *Jαθ(last(t))* есть последний элемент последовательности *Jαθ(t);*

*π(t1, t2)*, где *t1* и *t2* – термы, значение *t2* – последовательность, а *t1* – целое число, не меньшее 1 и не большее числа элементов последовательности *t2*; *Jαθ(π(t1, t2))* есть элемент последовательности *Jαθ(t2)*, имеющий номер *Jαθ(t1).*

Кроме того, термом этого специализированного расширения является:

*length(t)*, где *t* – терм, имеющий значением конечную последовательность, причем *Jαθ(μ(t))* есть длина последовательности *Jαθ(t)*.

**Формулами стандартного расширения** языка прикладной логики являются:

*sub(t1, t2)*, где *t1* и *t2* – термы, значениями которых являются последовательности, причем *Jαθ(sub(t1, t2))* истинно тогда и только тогда, когда *Jαθ(t1)* есть подпоследовательность *Jαθ(t2)*;

*beg(t1, t2)*, где *t1* и *t2* – термы, значениями которых являются последовательности, причем *Jαθ(beg(t1, t2))* истинно тогда и только тогда, когда *Jαθ(t1)* совпадает с началом последовательности *Jαθ(t2)*;

*end(t1, t2)*, где *t1* и *t2* – термы, значениями которых являются последовательности, причем *Jαθ(end(t1, t2))* истинно тогда и только тогда, когда *Jαθ(t1)* совпадает с концом последовательности *Jαθ(t2)*.

Кроме того, в **специализированном расширении** МАТЕМАТИЧЕСКИЕ КВАНТОРЫ языка прикладной логики вводятся следующие **формулы**:

*(∀(v1: t1) (v2: t2)… (vm: tm) f)* – квантор всеобщности, где *(v1: t1) (v2: t2)… (vm: tm)* есть множество описаний переменных (здесь последовательности рассматриваются как упорядоченные множества), переменные *v1, v2, …, vm* связаны в формуле *(∀(v1: t1) (v2: t2)… (vm: tm) f)*, *f* – формула, причем *Jαθ((∀(v1: t1) (v2: t2)… (vm: tm) t))* есть конъюнкция значений *Jαθ(f)* при всех допустимых для множества описаний переменных *(v1: t1) (v2: t2)… (vm: tm)* подстановках в формулу *f*;

*(∃(v1: t1) (v2: t2)… (vm: tm) f)* – квантор существования, где *(v1: t1) (v2: t2)… (vm: tm)* есть множество описаний переменных (и здесь последовательности рассматриваются как упорядоченные множества), переменные *v1, v2, …, vm* связаны в формуле *(∃(v1: t1) (v2: t2)… (vm: tm) f)*, *f* – формула, причем *Jαθ((∃(v1: t1) (v2: t2)… (vm: tm) t))* есть дизъюнкция значений *Jαθ(f)* при всех допустимых для множества описаний переменных *(v1: t1) (v2: t2)… (vm: tm)* подстановках в формулу *f*.

Прикладная логическая теория, моделирующая онтологию синтаксиса языка МИЛАН, имеет следующий вид.

Синтаксис языка МИЛАН(ST, ПОСЛЕДОВАТЕЛЬНОСТИ)

***сорт*** *программа: программы*

*программы ≡ {(v1: идентификаторы)(v2: последовательности операторов) «начало» || v1 || v2 || «конец» || v1}*

*последовательности операторов ≡ {(v1: операторы)(v2: последовательности операторов) v1 || «;» || v2}*

*операторы ≡ присваивания ∪ условные операторы ∪ циклы ∪ обмены*

*присваивания ≡ {(v1: идентификаторы)(v2: выражения) v1 || «:=» || v2}*

*условные операторы ≡ {(v1: логические выражения)(v2: последовательности операторов) «если» || v1 || «то» || v2 || «все»} ∪ {(v1: логические выражения)(v2: последовательности операторов) (v3: последовательности операторов) «если» || v1 || «то» || v2 || «иначе» || v3 || «все»}*

*циклы ≡ {(v1: логические выражения)(v2: последовательности операторов) «пока» || v1 || «цк» || v2 || «кц»}*

*обмены ≡ {(v1: идентификаторы) «ввод» || v} ∪ {(v: выражения) «вывод» || v}*

*выражения ≡ факторы ∪ {(v1: выражения)(v2: факторы) v1 || «+» || v2}*

*факторы ≡ первичные ∪ {(v1: факторы)(v2: первичные) v1 || «\*» || v2}*

*первичные ≡ идентификаторы ∪ константы ∪ {(v: выражения) «(» || v || «)»}*

*логические выражения ≡ {(v1: выражения)(v2: выражения) v1 || «=» || v2} ∪ {(v1: выражения)(v2: выражения) v1 || «≠» || v2}*

*константы ≡ цифры ∪ {(v1: константы)(v2: цифры) v1 || v2}*

*цифры ≡ {«0», «1», «2», …, «9»}*

*идентификаторы ≡ буквы ∪ {(v1: идентификаторы)(v2: буквы) v1 || v2}*

*буквы ≡ {«А», «Б», «В», …, «Я»}*

Задание N8. (по теме "Величины конечных последовательностей")

Придумать пример вербализуемой информации, представляемой с помощью различных значений, в том числе конечных последовательностей.

План ответа

1. Вербализуемая информация и ее вербальное представление.

2. Величины конечных последовательностей и другие величины, элементы которых использованы для представления этой информации.

3. Онтология, в терминах которой представлена эта вербализуемая информация.

4. Модель этой онтологии на языке прикладной логики.

5. Содержательное описание подмножества концептуализации, определяемой онтологией, к которому принадлежит вербализуемая информация.

6. Система знаний об этом подмножестве в терминах онтологии.

7. Модель этой системы знаний на языке прикладной логики.

8. Прикладная задача в терминах онтологии, постановка которой включает систему знаний (что дано, что надо найти).

9. Математическая постановка этой задачи в терминах модели онтологии (что дано, что надо найти). Аргументы в пользу того, что задача имеет решение. Сколько этих решений?

10. Алгоритм решения этой задачи. Аргументы в пользу того, что алгоритм находит все решения и не находит лишних решений.

11. Программа (на любом ЯП, **но с комментариями**!) реализующая этот алгоритм.

12. Как в этой программе моделируются величины конечных последовательностей и другие величины, понятия и знания.

13. Пример состояния памяти на шаге завершения программы, выполняемой с исходными данными, соответствующими вербализуемой информации.

16. НЕСТАНДАРТНЫЕ ВЕЛИЧИНЫ И ИХ ПРЕДСТАВЛЕНИЕ В МАТЕМАТИЧЕСКИХ И КОМПЬЮТЕРНЫХ МОДЕЛЯХ

16.1. Нестандартные величины

**Нестандартной величиной** называется любая величина, не совпадающая ни с одной из стандартных величин. Нестандартная величина может быть простой или сложной. Набор стандартных величин является универсальным, т.е. позволяет адекватно представить все свойства онтологий и систем знаний. Однако в некоторых случаях использование стандартных величин (их объектов, операций, функций и отношений) ведет к появлению громоздких конструкций в спецификациях онтологий и систем знаний. Чтобы такие спецификации были как можно более естественными, и используются нестандартные величины. Определение каждой нестандартной величины строится в соответствии с общим определением величины.

Чтобы установить, какие нестандартные величины используются при вербальном представлении информации, необходимо выяснить, все ли используемые при этом значения относятся к стандартным величинам и, если это не так, выделить нестандартные значения, установить для них операции и отношения и определить нестандартные величины, которым они принадлежат. Кроме того, как указывалось выше, обоснование нестандартных величин ссылается на систему понятий, использованную при вербальном представлении этой информации, где эти величины или их подмножества являются объемами понятий.

16.2. Алгебраические системы, моделирующие нестандартные величины

Каждая нестандартная величина моделируется некоторой алгебраической системой, как это имеет место для любых величин.

16.3. Модели типов данных конечных последовательностей

Каждая нестандартная величина моделируется некоторым классом (абстрактным типом данных).

16.4. Понятия, соответствующие конечным последовательностям

**Понятие** называется **соответствующим нестандартной величине**, если его объем совпадает с некоторой нестандартной величиной или является подмножеством такой величины.

16.5. Моделирование нестандартных величин в языке прикладной логики

Чтобы специфицировать средствами языка прикладной логики онтологии и системы знаний, в описании которых используются нестандартные величины, необходимо предварительно описать соответствующие расширения языка прикладной логики, т.е. определить синтаксис и семантику термов и формул, использующих модели операций, функций и отношений нестандартных величин.

17. ОСНОВЫ ОНТОЛОГИЧЕСКОГО АНАЛИЗА ВЕРБАЛЬ-НОЙ ИНФОРМАЦИИ

17.1. Цель онтологического анализа вербальной информации

Анализ информации является первым этапом разработки любой прикладной программы. Анализируется всегда некоторая совокупность информации, а именно та, которая связана с обработкой информации, выполняемой этой прикладной программой. Как правило, эта совокупность является бесконечной (нетривиальной). **Целью анализа** совокупности информации является поиск явного и наиболее точного описания этой совокупности. Целью анализа совокупности вербальной информации является поиск адекватной концептуализации для этой совокупности и построение явного и наиболее точного описания этой совокупности в терминах этой концептуализации. Основным недостатком такого анализа является то, что концептуализация является объектом интуитивного познания, поэтому понимание и однозначное толкование результатов такого анализа зависят от интуитивного понимания концептуализации и смысла ее терминов. **Целью онтологического анализа** совокупности вербальной информации является поиск адекватной концептуализации для этой совокупности, построение адекватной онтологии для этой концептуализации, и построение системы знаний в терминах этой концептуализации, явно и наиболее точно описывающей эту совокупность. При этом онтология и система знаний являются ключом к пониманию и однозначному толкованию результатов онтологического анализа.

17.2. Поиск адекватной концептуализации

Первым этапом онтологического анализа совокупности вербальной информации является поиск адекватной концептуализации для этой совокупности. Идеи, входящие в анализируемую совокупность, являются профессиональными. Аналитик должен попросить эксперта, участвующего в анализе, сформировать возможно более полный список терминов, используемых для представления этой информации, а также представительный список идей, входящих в анализируемую бесконечную совокупность. Далее аналитик с помощью эксперта пытается представить идеи из этого списка в виде отображений уже выделенных терминов в множество некоторых значений. Составляется список уже использованных значений. В процессе этой работы списки терминов, значений и идей могут пополняться. Аналитик нестрого определяет смысл используемых терминов и значений, а также принципы представления с их помощью анализируемой информации. Самостоятельную часть этого этапа составляет анализ списка значений. Каждое значение должно быть отнесено к некоторой величине, стандартной или нестандартной. Составляется список всех использованных величин, отдельно определяются все нестандартные величины. Если все идеи из списка уже представлены как элементы концептуализации, величины выделены, а смысл всех терминов и принципы представления с их помощью информации понятны аналитику, то можно считать, что этот этап успешно завершен и адекватная концептуализация найдена.

17.3. Построение адекватной онтологии

Вторым этапом онтологического анализа является построение для найденной концептуализации адекватной онтологии. Для этого аналитик с помощью эксперта должен построить определения всех терминов концептуализации, используя в этих определениях термины, связанные с величинами, и термины концептуализации, уже получившие определения в онтологии. Особой аккуратности требуют рекурсивные определения. Поиск ошибок в определениях онтологии может быть выполнен с помощью полученного на предыдущем этапе списка идей, представленных как элементы этой концептуализации. Если значение некоторого названия понятия в представлении некоторой идеи выходит за пределы объема этого понятия, определенного в онтологии, то определение этого понятия неправильно. Наконец, особую трудность представляет формулировка онтологических соглашений. Некоторые онтологические соглашения могут быть предложены экспертом, однако нет никаких надежд на то, что таким образом будут выделены все соглашения. Более систематическим способом их выделения является составление с помощью эксперта списка бессмысленных сообщений, представимых в форме отображений множества названий понятий в множество всех значений, но не входящих в концептуализацию (сообщений, не имеющих интерпретации). Попытка составления такого списка и обсуждение с экспертом причин, по которым эти сообщения не входят в концептуализацию, могут привести к формулировке новых онтологических соглашений. Если же оказывается, что такое сообщение согласуется с онтологией, то либо некоторые онтологические соглашения должны быть уточнены, либо должны быть сформулированы новые онтологические соглашения. После того, как онтология построена, она формализуется с помощью некоторого языка спецификации (например, языка прикладной логики).

17.4. Построение системы знаний

Третьим этапом онтологического анализа является построение системы знаний, возможно более точно определяющей анализируемую информацию. Система знаний строится в той же форме, что и система онтологических соглашений. Заключительная часть этого этапа анализа состоит в поиске ошибок в системе знаний (и онтологии). Проверяется, что каждая идея из списка идей, представленных как элементы концептуализации, согласуется с системой знаний. Затем с помощью эксперта составляется список идей, которые входят в концептуализацию (имеют смысл и, тем самым, согласуются с онтологией), но не входят в анализируемую совокупность информации. Если оказывается, что такие идеи согласуются с системой знаний, то система знаний должна быть уточнена. После того, как система знаний построена, она формализуется с помощью некоторого языка спецификации (например, языка прикладной логики).

Система величин, онтология и система знаний, а также их модели являются основными результатами онтолого-ориентированного анализа нетривиальной совокупности вербальной информации.

**III. СПЕЦИФИКАЦИИ ПРЕДМЕТНЫХ ОБЛАСТЕЙ, ЗАДАЧ, АЛГОРИТМОВ, ПРОГРАММ И ИСЧИСЛЕНИЙ**

**Спецификация** – это интенсиональное определение потенциально бесконечного множества на декларативном языке. **Потенциально бесконечное множество** – это либо бесконечное множество, либо подмножество бесконечного множества, не все элементы которого известны. Множество, все элементы которого известны, конечно и не нуждается в спецификации, так как оно может быть определено экстенсионально - перечислением всех его элементов. **Декларативный язык** – это язык, который предназначен для определения потенциально бесконечных множеств интенсионально - посредством задания общих свойств их элементов. Декларативные языки называются также **языками спецификаций**. Семантика каждого декларативного языка фиксирует, из элементов какого типа строятся определяемые с помощью этого языка множества. Примером декларативного языка является язык прикладной логики. Спецификацией на этом языке является прикладная логическая теория. Прикладная логическая теория определяет множество функций интерпретации, заданных на конечном множестве имен, вводимых этой теорией.

Если язык спецификации спроектирован и определен таким образом, что он предназначен для определения множеств сущностей только одного типа, то вопрос о том, что специфицируется на этом языке, не возникает. Если же язык спецификации является проблемно-независимым, то спецификация должна сопровождаться комментариями, которые устанавливают содержательную трактовку элементов специфицируемого множества и представление свойств этих элементов в спецификации. Язык прикладной логики является примером проблемно-независимого языка спецификации.

18. СПЕЦИФИКАЦИЯ ПРЕДМЕТНОЙ ОБЛАСТИ

18.1. Действительность

Как уже говорилось, предметную область можно представлять себе как совокупность прикладных задач, которые в ней решаются. Вербализованную информацию, которая должна быть учтена при решении этих прикладных задач, будем называть **ситуацией**.

Каждая ситуация содержит информацию о конечном множестве объектов и об отношениях между ними, имевших место в конечной области пространства и на конечном промежутке времени. Отказ от предположения о конечноcти множеcтва объектов в cитуации приводит к рассмотрению таких задач, при решении которых необходимо получать, обрабатывать, либо генерировать информацию о беcконечном множеcтве объектов. Отказ от предположения о том, что ситуация может иметь место только на конечном промежутке времени, ведет к необходимости обрабатывать информацию, относящуюся к сколь угодно далекому прошлому. И то, и другое не реализуемо на компьютерах. Конечное множество объектов может размещаться только в конечной области пространства. Выделение объектов и отношений между ними в cитуации определяетcя cпоcобноcтями человека и являетcя прерогативой cпециалиcтов предметной области.

Будем говорить, что множество не связанных между собой всех возможных в предметной области ситуаций (имевших место в прошлом, имеющих место в настоящем, а также возможных в будущем) образуют **действительность** в этой предметной области. Действительность является нетривиальной совокупностью информации (поскольку все ситуации действительности не могут быть известны).

Раccмотрим приемлемоcть представления о действительности как совокупности отдельных ситуаций для cпециалиcтов предметных областей. Эти cпециалиcты, как вcякие люди, предcтавляют себе дейcтвительноcть как единое целое во времени, проcтранcтве и многообразии других форм, но ограниченное их возможноcтями получать информацию о дейcтвительноcти в различных ее аcпектах. Вмеcте c тем профеccиональная деятельноcть этих cпециалиcтов протекает лишь в отдельные отрезки времени. Более того, она, как правило, cоcтоит в решении отдельных задач. Отдельная cитуация как раз и cоответcтвует информации, которая рассматривается при решении задач. Поэтому разделение дейcтвительноcти в рамках предметной области на отдельные cитуации можно cчитать приемлемым для любой профеccиональной деятельноcти. Примером разделения дейcтвительноcти на cитуации может cлужить прием врачом больных в поликлинике: c каждым новым больным, пришедшим на прием, у врача возникают новые задачи диагностики или мониторинга состояния и лечения (новая cитуация). Даже по отношению к операторам непрерывных производcтв правомерно cчитать, что каждое их дежурcтво cвязано c новой cитуацией. Отказ от раccмотрения дейcтвительноcти как множеcтва cитуаций ведет к концепции задач cо cколь угодно глубоким иcпользованием информации о прошлом в их постановке.

Менее приемлемым может оказатьcя предположение о неcвязанноcти различных cитуаций между cобой. Человек обладает cпоcобноcтью запоминать cитуации и поэтому может иcпользовать информацию о прошлых cитуациях при анализе новой по меньшей мере двумя cпоcобами:

- находить cреди прошлых, уже проанализированных, cитуаций те, которые похожи на новую, и иcпользовать информацию о найденных cитуациях при анализе новой (поиcк прецедентов);

- иcпользовать информацию о прошлых cитуациях для корректировки cвоих знаний (накопление опыта).

Первый cпоcоб может иметь лишь ограниченное применение при решении cерьезных и ответcтвенных задач, поcкольку информация о новой cитуации, как правило, не полна, ее cходcтво c прошлыми cитуациями может быть поверхноcтным, а выводы, cделанные на оcнове информации о похожих прошлых cитуациях, могут оказатьcя неверными в новой cитуации.

По отношению ко второму cпоcобу допуcтимой являетcя cледующая точка зрения: в течение периода времени, пока не накоплено такого количеcтва прошлых cитуаций, которое доcтаточно для обоcнованной корректировки знаний, можно cчитать, что информация о прошлых cитуациях не иcпользуетcя при анализе новых; поcле того, как знания cкорректированы, образуютcя новые знания о предметной области - они дейcтвуют до cледующей корректировки знаний и т.д. Таким образом, предположение о неcвязанноcти cитуаций может быть приемлемо локально, на протяжении тех отрезков времени, пока знания не корректируютcя, т.е. когда завиcимоcтью от прошлых cитуаций можно пренебречь.

18.2. Онтология предметной области

Если информация о действительности вербализуема, то концептуализацию информации о действительности предметной области будем называть **концептуализацией** этой **предметной области**. Онтологию, представляющую концептуализацию предметной области, будем называть **онтологией предметной области**.

Онтология предметной области состоит из определений терминов, обозначающих основные и вспомогательные понятия для представления ситуаций, и онтологических соглашений - **ограничений целостности ситуаций**. Ситуация соответствует онтологии предметной области, если выполнены следующие условия:

значение каждого термина, обозначающего основное понятие онтологии, в этой ситуации принадлежит объему этого понятия, определенному в онтологии;

значение каждого термина, обозначающего вспомогательное понятие онтологии, в этой ситуации равно либо значению этого понятия, определенному в онтологии, либо результату вычисления с использованием способа вычисления, определенного для этого понятия в онтологии, причем значения других терминов, обозначающих понятия, участвующие в этом вычислении, берутся из этой же ситуации;

каждое ограничение целостности ситуаций онтологии является истинным, если в нем каждый термин, обозначающий понятие, заменить его значением в этой ситуации.

Будем называть множество всех ситуаций, которые соответствуют онтологии предметной области, **концептуализацией, определяемой онтологией предметной области**. Будем называть **онтологию предметной области адекватной** концептуализации предметной области, если эта концептуализация предметной области является подмножеством концептуализации, определяемой этой онтологией предметной области. Также будем называть **онтологию предметной области точной**, если эти две концептуализации совпадают.

18.3. Математическая модель спецификации предметной области

**Математической моделью онтологии предметной области** является прикладная логическая теория. Определение каждого основного понятия представляется в этой модели описанием сорта имени. Определение каждого вспомогательного понятия представляется в этой модели описанием значения имени. В предложениях этих двух типов именами являются термины – названия соответствующих понятий предметной области. Онтологические соглашения предметной области представляются предложениями третьего типа – ограничениями на интерпретацию имен. Если прикладная логическая теория есть модель онтологии понятий предметной области, то каждая модель этой теории трактуется как модель некоторой ситуации этой предметной области.

18.4. Система знаний предметной области

**Система знаний предметной области** (система знаний о действительности, явно представляющая действительность и выделяющая ее как собственное подмножество концептуализации предметной области) представляется в виде совокупности утверждений в терминах онтологии предметной области. Ситуация соответствует системе знаний предметной области, если она соответствует онтологии предметной области и каждое утверждение этой системы знаний является истинным при условии, чтов нем каждый термин, обозначающий понятие, заменен его значением в этой ситуации.

18.5. Зачем нужны онтология и система знаний предметной области, а также их модели

Постановка любой прикладной задачи обычно включает описание множества исходных данных, множества искомых результатов и описание отношения между ними. Для того чтобы решать прикладную задачу на компьютере, необходимо заменить постановку исходной задачи постановкой соответствующей математической задачи. Как связана постановка исходной прикладной задачи с постановкой математической задачи? В постановке исходной прикладной задачи описания возможных наборов входных и выходных данных заданы в терминах предметной области, к которой отноcитcя прикладная задача. Поэтому в поcтановке математичеcкой задачи должны присутствовать формальные определения понятий (определения их объемов), которые эти термины обозначают. Что каcаетcя опиcания отношения между входными и выходными данными, то здеcь возможны два cлучая. В первом cлучае опиcание этого отношения входит в поcтановку иcходной прикладной задачи (в формальном или неформальном виде). Тогда формальное предcтавление этого опиcания задает описание этого отношения в поcтановке математичеcкой задачи. В это описание могут входить термины, которые не были использованы при опиcания входных и выходных данных. Тогда в поcтановке математичеcкой задачи должны присутствовать и формальные определения понятий, которые эти новые термины обозначают. Во втором cлучае поcтановка иcходной прикладной задачи не cодержит опиcания отношения между входными и выходными данными или содержит неполное описание этого отношения. Тогда опиcание этого отношения cтроитcя на оcнове онтологических соглашений и системы знаний (законов) предметной области. При этом возможны cледующие варианты.

В первом варианте в опиcании отношения учаcтвует лишь небольшая чаcть вcех законов предметной области. Тогда опиcанием этого отношения в поcтановке математичеcкой задачи являетcя формальное описание всех онтологических соглашений, а также этих законов вместе с формальными определениями всех понятий, обозначенных терминами, входящими в эти соглашения и законы. Поиcк вcех таких законов в предметной области может оказатьcя для аналитика веcьма непроcтым.

Во втором варианте в опиcании отношения учаcтвует достаточно большое количество законов (или даже вcе законы) предметной области. Тогда опиcанием этого отношения в поcтановке математичеcкой задачи являетcя формальное описание определений всех понятий, онтологических соглашений и законов предметной области. Выявление большого количества законов (особенно всех законов) обычно предcтавляет для аналитика значительные трудноcти.

Наконец, возможен вариант, когда программа предназначена для решения cразу неcкольких различных прикладных задач в одной и той же предметной области. Чем больше таких задач, тем больше законов учаcтвует в опиcании отношений между входными и выходными данными вcех этих задач. Поэтому в этом варианте предпочтительнее предcтавить вcе законы предметной области, а не опиcывать отношение для каждой задачи.

Таким образом, во вcех вариантах второго cлучая возникает cамоcтоятельная компонента постановки математичеcкой задачи, называемая обычно моделью предметной области. В первом варианте она предcтавлена опиcанием всех онтологических соглашений и только тех законов и определений понятий, которые необходимы для опиcания отношения между входными и выходными данными. Поэтому такая модель предметной области завиcит от решаемой задачи. В поcледних двух вариантах модель предметной области предcтавлена опиcанием вcех онтологических соглашений, законов и определений понятий и потому не завиcит от решаемых задач.

Решать задачу на компьютере можно только при наличии ее математической постановки. В сложных предметных областях трудность получения математической постановки задачи связана с необходимостью учесть в этой постановке большое количество информации, относящейся к предметной области. В соответствии с принципом "разделяй и властвуй" следует сначала представить всю эту информацию в виде математической модели предметной области, а затем уже ставить задачу в терминах этой модели.

18.6. Почему модель предметной области есть спецификация ее действительности

Рассмотрим предметную область как объект моделирования. С этой точки зрения наиболее существенным ее свойством является действительность, т.е. та часть реального мира, информация о которой используется при решении задач профессиональной деятельности. Действительность можно рассматривать как бесконечное множество не зависящих друг от друга ситуаций, каждая из которых представлена информацией, связанной с возможным решением конкретных задач профессиональной деятельности. Таким образом, можно считать, что когда моделируется предметная область, то объектом моделирования является бесконечное множество ситуаций.

В процессе построения модели предметной области необходимо ответить на следующие три вопроса:

1. какие свойства предметной области должны быть представлены в этой модели;
2. как должна быть устроена эта модель и каким образом в ней представлены свойства предметной области;
3. в каком смысле эта модель должна быть адекватна этой предметной области.

18.7. Какие существенные свойства предметной области представляются в ее модели

Ответ на первый вопрос может быть дан в форме предположений о свойствах предметной области, которые должны быть представлены во всех ее моделях. В то же время эти предположения могут рассматриваться и как методологическая основа для анализа предметных областей, необходимого для построения их моделей. Ответы на другие вопросы будут рассматриваться в следующих разделах этой темы.

Основными компонентами предметной области являются объекты, действительность (рассматриваемая как бесконечное множество не зависящих друг от друга ситуаций), в которой протекает профессиональная деятельность, система понятий, в терминах которых представляются ситуации и знания о них, и знания о действительности, лежащие в основе этой профессиональной деятельности.

Каждый объект предметной области представляется значением, принадлежащим некоторой величине. Информация, относящаяся к отдельной ситуации, всегда включает информацию, относящуюся к конечной совокупности объектов. Каждая ситуация имеет вербальное представление. Онтология предметной области представляет ее систему понятий. Система знаний предметной области представляет законы этой предметной области.

18.8. Компоненты модели предметной области

Ответ на второй вопрос предыдущего раздела может быть дан в форме утверждений об общих свойствах всех моделей рассматриваемого класса и соответствии между свойствами предметных областей и свойствами моделей. Каждая модель предметной области состоит из моделей величин, модели онтологии, модели системы знаний и модели действительности, связанных между собой определенным образом. Модель действительности определяет множество моделей ситуаций.

Обычно в профессиональной деятельности используются неформализованные или полуформализованные модели предметной области. В тех предметных областях, где не применяется математический аппарат, используются неформализованные модели. В них модель знаний описывается на естественном языке. Элементы формализации в таких моделях могут присутствовать лишь в виде использования в них тех или иных размерных значений. В тех предметных областях, где применение математического аппарата является традиционным, используются полуформализованные модели. Модель знаний представляет собой конечную совокупность утверждений, записанных частично на естественном языке, а частично – в виде формализованных утверждений с использованием системы математических символов. Таким способом описываются законы и закономерности предметной области в виде соотношений между переменными и константами. Смысл каждой переменной, используемой в этих соотношениях, а также ее область значений описываются неформально, на естественном языке.

Спецификация модели предметной области на языке прикладной логики является полностью формальной моделью этой предметной области. Единственный неформальный элемент такой модели – это тексты комментариев, предназначенные для понимания этой модели людьми.

18.9. Критерий адекватности модели предметной области

Ответом на третий вопрос, поставленный выше, является критерий адекватности, т.е. необходимые и достаточные условия того, что модель адекватна предметной области, для которой она построена. Модель адекватна предметной области тогда и только тогда, когда модель действительности адекватно представляет действительность предметной области. Иными словами, модель адекватна предметной области тогда и только тогда, когда для любой ситуации действительности существует адекватная ей модель ситуации, входящая в модель действительности, а любая модель ситуации, входящая в модель действительности, адекватна некоторой ситуации действительности этой предметной области. Модель ситуации адекватна некоторой ситуации действительности тогда и только тогда, когда вербальное представление этой ситуации совпадает с моделью прикладной логической теории, моделирующей эту предметную область.

18.10. Анализ предметной области и синтез ее модели

Целью онтолого-ориентированного анализа предметной области является поиск адекватной концептуализации этой предметной области, построение ее адекватной онтологии, а также построение системы знаний в терминах этой концептуализации, явно и наиболее точно описывающей действительность этой предметной области. При этом онтология и система знаний предметной области являются ключом к пониманию и однозначному толкованию результатов онтолого-ориентированного анализа предметной области.

Первым этапом онтолого-ориентированного анализа предметной области является поиск адекватной концептуализации действительности для нее. Аналитик должен попросить эксперта, участвующего в анализе, сформировать возможно более полный список терминов, используемых для представления действительности, а также представительный список описаний ситуаций действительности в этих терминах. Далее аналитик с помощью эксперта пытается представить ситуации из этого списка в виде частичных отображений уже выделенных терминов в множество некоторых значений. Составляется список уже использованных значений. В процессе этой работы списки терминов, значений и ситуаций могут пополняться. Аналитик фиксирует смысл используемых терминов и значений, а также принципы представления с их помощью ситуаций. Самостоятельную часть этого этапа составляет анализ списка значений. Каждое значение должно быть отнесено к некоторой величине, стандартной или нестандартной. Составляется список всех использованных величин, отдельно определяются все нестандартные величины. Если все ситуации из списка уже представлены как элементы концептуализации, величины выделены, а смысл всех терминов и принципы представления с их помощью ситуаций понятны аналитику, то можно считать, что этот этап успешно завершен и адекватная концептуализация предметной области найдена.

Вторым этапом онтолого-ориентированного анализа предметной области является построение для найденной концептуализации адекватной онтологии действительности. Для этого аналитик с помощью эксперта должен построить определения всех терминов концептуализации, используя в этих определениях термины, связанные с величинами, и термины концептуализации, уже получившие определения в онтологии действительности. Особой аккуратности требуют рекурсивные определения. Поиск ошибок в определениях онтологии действительности может быть выполнен с помощью полученного на предыдущем этапе списка ситуаций, представленных как элементы этой концептуализации. Если значение некоторого названия понятия в представлении некоторой ситуации знаний выходит за пределы объема этого понятия, определенного в онтологии действительности, то определение этого понятия неправильно. Наконец, особую трудность представляет формулировка онтологических соглашений. Некоторые онтологические соглашения могут быть предложены экспертом, однако нет никаких надежд на то, что таким образом будут выделены все соглашения. Более систематическим способом их выделения является составление с помощью эксперта списка бессмысленных ситуаций, представимых в форме отображений множества названий понятий в множество всех значений, но не входящих в концептуализацию действительности. Попытка составления такого списка и обсуждение с экспертом причин, по которым эти ситуации не входят в концептуализацию действительности, могут привести к формулировке новых онтологических соглашений. Если же оказывается, что такая ситуация согласуется с онтологией действительности, то либо некоторые онтологические соглашения должны быть уточнены, либо должны быть сформулированы новые онтологические соглашения.

Третьим этапом онтолого-ориентированного анализа предметной области является построение системы знаний, возможно более точно определяющей действительность. Система знаний строится в той же форме, что и система онтологических соглашений (в терминах для описания ситуаций). Заключительная часть этого этапа анализа состоит в поиске ошибок в системе знаний (и онтологии) предметной области. Проверяется, что каждая ситуация, представленная как элемент концептуализации, согласуется с системой знаний предметной области. Затем с помощью эксперта составляется список ситуаций, которые входят в концептуализацию (имеют смысл и, тем самым, согласуются с онтологией), но не входят в действительность. Если оказывается, что такие ситуации согласуются с системой знаний предметной области, то эта система знаний должна быть уточнена.

Система величин, онтология и система знаний предметной области являются основными результатами онтолого-ориентированного анализа предметной области.

В качестве примера рассмотрим следующую задачу. Найти самый короткий план переезда через реку с левого берега на правый на лодке некоторой вместимости некоторого числа миссионеров и некоторого числа людоедов при условии, что во время этого переезда ни в лодке, ни на одном из берегов миссионеры не окажутся в опасности. Миссионеры оказываются в опасности в лодке или на одном из берегов, если их оказывается меньше, чем людоедов в этом месте.

Анализируя предметную область, связанную с этой задачей, определим понятия, с помощью которых описываются исходные данные и результаты решения задачи.

***сорт*** *число миссионеров: I[1, ∞)*

***сорт*** *число людоедов: I[1, число миссионеров]*

***сорт*** *вместимость лодки: I[2, ∞)*

***сорт*** *решение: планы(inf(варианты))*

Здесь основное понятие «решение» определено через вспомогательные понятия «планы» и «варианты». Определим понятие «варианты» как множество длин осуществимых планов.

*варианты ≡ {(i: I[1, ∞)) планы(i) ≠ ∅}*

Понятие «планы» определим как функцию, аргументом которой является длина плана, а значением – множество всех планов такой длины. Если значением этой функции является пустое множество, то это означает, что планы такой длины не осуществимы. Каждый план, в свою очередь, есть функция, которая по номеру шага определяет, какой переезд на этом шаге выполняется. На нечетных шагах плана выполняется переезд слева направо, а на четных – справа налево. На всех шагах плана, начиная со второго, обстановка нам обоих берегах до переезда совпадает с обстановкой на обоих берегах после переезда на предыдущем шаге. На последнем шаге после переезда имеет место обстановка, которая должна быть после успешного выполнения плана. На первом шаге до переезда имеет место начальная обстановка.

*планы ≡ (λ (i: I[1, ∞)) {(v: I[1, 2 \* i - 1] → переезды) (& (k: I[1, i]) v(2 \* k - 1) ∈ переезды слева направо) & (& (k: I[1, i - 1]) v(2 \* k) ∈ переезды справа налево) & (i > 1 ⇒ (& (k: I[2, 2 \* i - 1]) до переезда(v(k)) = после переезда(v(k - 1)))) & миссионеров(на левом берегу(после переезда(v(2 \* i - 1)))) = 0 & людоедов(на левом берегу(после переезда(v(2 \* i - 1)))) = 0 & миссионеров(на левом берегу(до переезда(v(1)))) = число миссионеров & людоедов(на левом берегу(до переезда(v(1)))) = число людоедов})*

Понятие «планы» определено с помощью вспомогательных понятий «переезды», «переезды слева направо», «переезды справа налево», «до переезда», «после переезда», «миссионеров», «людоедов», «на левом берегу». Определим эти понятия. Понятие «переезды» есть объединение понятий «переезды слева направо» и «переезды справа налево».

*переезды ≡ переезды слева направо ∪ переезды справа налево*

Понятия «переезды слева направо» и «переезды справа налево» определены как множества таких поездок на лодке, что эти поездки возможны и в результате них обстановка на берегах меняется соответствующим образом.

*переезды слева направо ≡ {(v: поездка на лодке) миссионеров в лодке(в пути(v)) ≤ миссионеров(на левом берегу(до переезда(v))) & людоедов в лодке(в пути(v)) ≤ людоедов(на левом берегу(до переезда(v))) & миссионеров(на левом берегу(после переезда(v))) = миссионеров(на левом берегу(до переезда(v))) - миссионеров в лодке(в пути(v)) & людоедов(на левом берегу(после переезда(v))) = людоедов(на левом берегу(до переезда(v))) - людоедов в лодке(в пути(v))}*

*переезды справа налево ≡ {(v: поездка на лодке) миссионеров в лодке(в пути(v)) ≤ миссионеров(на правом берегу(до переезда(v))) & людоедов в лодке(в пути(v)) ≤ людоедов(на правом берегу(до переезда(v))) & миссионеров(на правом берегу(после переезда(v))) = миссионеров(на правом берегу(до переезда(v))) - миссионеров в лодке(в пути(v)) & людоедов(на правом берегу(после переезда(v))) = людоедов(на правом берегу(до переезда(v))) - людоедов в лодке(в пути(v))}*

Понятие «поездка на лодке» определена как множество структурных значений с атрибутами «до переезда», «после переезда» (областями значений которых являются значения понятия «обстановка на обоих берегах»), а также – «в пути», областью значений которого является значение понятия «обстановка в лодке».

*поездка на лодке ≡ (до переезда → обстановка на обоих берегах, после переезда → обстановка на обоих берегах, в пути → обстановка в лодке)*

Понятие «обстановка в лодке» определено как множество структурных значений с атрибутами «миссионеров в лодке» и «людоедов в лодке», значениями которых являются неотрицательные целые числа (количество миссионеров и людоедов в лодке соответственно), сумма которых положительна, не превышает вместимости лодки и для которых выполнен закон безопасности миссионеров во время поездки на лодке.

*обстановка в лодке ≡ {(v: (миссионеров в лодке → I[0, вместимость лодки], людоедов в лодке → I[0, вместимость лодки])) миссионеров в лодке(v) + людоедов в лодке(v) ∈ I[1, вместимость лодки] & (миссионеров в лодке(v) ≥ людоедов в лодке(v) ∨ миссионеров в лодке(v) = 0)}*

Понятие «обстановка на обоих берегах» определена как множество структурных значений с атрибутами «на левом берегу» и «на правом берегу», значениями которых являются обстановки на каждом из берегов, для которых выполнены законы сохранения миссионеров и людоедов.

*обстановка на обоих берегах ≡ {(v: (на левом берегу → обстановка на берегу, на правом берегу → обстановка на берегу)) миссионеров(на левом берегу(v)) + миссионеров(на правом берегу(v)) = число миссионеров & людоедов(на левом берегу(v)) + людоедов(на правом берегу(v)) = число людоедов}*

Понятие «обстановка на берегу» определена как множество структурных значений с атрибутами «миссионеров», значением которого является целое число – количество миссионеров на этом берегу, и «людоедов», значением которого является целое число – количество людоедов на нем, и для которых выполнен закон безопасности миссионеров.

*обстановка на берегу ≡ {(v: (миссионеров → I[0, число миссионеров], людоедов → I[0, число людоедов])) миссионеров(v) ≥ людоедов(v) ∨ миссионеров(v) = 0}*

Задание № 9 (по теме "Спецификации предметных областей")

Придумать пример предметной области, связанной с профессиональной деятельностью, в которой решается не менее двух задач, и построить ее модель.

План ответа

1. Название и характеристика предметной области (характеристика ее объектов, действительности, системы понятий и знаний).

2. Характеристика профессиональной деятельности в этой предметной области (характеристика исполнителей и задач, которые они решают).

3. Постановка всех классов задач, решаемых информационной системой при автоматизации этой профессиональной деятельности (для каждой задачи что дано, что надо найти, и как первое связано со вторым).

4. Система величин предметной области и её обоснование (связь с системой понятий).

5. Примеры (не менее двух) вербального представления ситуаций с комментариями.

6. Онтология предметной области (определения всех основных и вспомогательных понятий, связь их объемов и значений с величинами, онтологические соглашения в терминах этих понятий) и ее обоснование (связь с постановками задач п3).

7. Система знаний предметной области в терминах ее онтологии.

8. Прикладная логическая теория, моделирующая онтологию и систему знаний предметной области (с комментариями, в которых показано, как система понятий и система знаний представлена в модели).

9. Аргументы в пользу адекватности этой модели предметной области.

10. Программы (на любом языке программирования, но с комментариями!) решения всех классов задач, указанных в п.3.

11. Как в этой программе моделируются величины, понятия, онтологические соглашения и система знаний предметной области.

12. Примеры конечных состояний памяти при выполнении этих программ с исходными данными, соответствующими примерам п. 5.

19. СПЕЦИФИКАЦИЯ ЗАДАЧ

Существенными свойствами задачи являются ее исходные данные, результаты ее решения и отношения между исходными данными и результатами решения задачи (модель предметной области).

19.1. Что такое спецификация задачи

Будем называть **задачей** вербальное представление информации, содержащее значения двух терминов – «входные данные» и «выходные данные», каждое из которых, в свою очередь, есть конечной набор терминов, а также значения всех терминов из этих наборов. Термины из этих наборов будем называть **терминами для представления входных и выходных данных задач**, соответственно.

Рассмотрим некоторое множество задач, у которых значениями терминов «входные данные» каждой задачи и «выходные данные» каждой задачи являются одни и те же наборы терминов, соответственно, (значения терминов в этих наборах у разных задач различны). Если это множество задач можно рассматривать как нетривиальную совокупность вербальной информации, то такую совокупность будем называть **классом задач**, ее концептуализацию – **концептуализацией класса задач**, а онтологию, представляющую эту концептуализацию, – **онтологией класса задач**. Из этого определения следует, что смысл терминов для представления входных и выходных данных, один и тот же для всего класса задач, определяется онтологией класса задач, а сам класс задает некоторое отношение между значениями входных и выходных данных. Если в онтологии класса задач термины этой онтологии определяются через термины онтологии некоторой предметной области, то задачи этого класса будем называть **прикладными**. Если в онтологии класса задач термины этой онтологии определяются только через математические термины, то задачи этого класса будем называть **математическими**. Таким образом, ссылка на онтологию предметной области является частью онтологии класса прикладных задач, а ссылка на систему знаний предметной области – частью **постановки** класса прикладных задач. Сами термины для представления входных и выходных данных класса прикладных задач могут быть либо названиями понятий предметной области, либо определяться через термины предметной области в онтологии класса задач. Онтологические соглашения и система знаний предметной области индуцируют некоторое отношение между значениями терминов для представления входных и выходных данных класса прикладных задач. Если отношение, определяемое классом прикладных задач, является собственным подмножеством отношения между значениями входных и выходных данных задач этого класса, индуцируемого онтологией и системой знаний предметной области, то постановка класса прикладных задач должна содержать **условия** – дополнительные утверждения об отношении между значениями входных и выходных данных класса прикладных задач. Эти условия играют роль системы знаний о классе прикладных задач (поскольку они не определяют смысл терминов для представления входных и выходных данных класса задач, но выделяют этот класс из концептуализации).

Итак, **онтология класса прикладных задач** содержит ссылку на онтологию предметной области и определения двух терминов – «*входные данные*» и «*выходные данные*», каждый из которых является обозначением вспомогательного понятия онтологии этого класса задач; значение каждого из этих двух терминов есть некоторая конечная совокупность терминов - названий понятий – либо понятий онтологии предметной области, либо понятий, определяемых в онтологии этого класса задач. **Постановка класса прикладных задач** содержит ссылку на систему знаний предметной области и, возможно, некоторое множество условий. Кроме того, постановка класса задач содержит указание на **тип класса задач**. Каждая задача характеризуется своим типом. Если в задаче известны значения входных данных (значения всех терминов, являющихся элементами множества – значения термина «входные данные») и на основе отношения между входными и выходными данными требуется найти значения выходных данных (значения всех терминов, являющихся элементами множества – значения термина «выходные данные»), то тип такой задачи - **задача на вычисление**. Если в задаче заданы значения как входных, так и выходных данных и требуется показать, что значения выходных данных находятся в отношении, определенном в постановке задачи, с входными данными, то тип такой задачи - **задача на доказательство**. Будем считать, что все задачи одного и того же класса имеют один и тот же тип.

Рассуждение, которое показывает, как по значениям входных данных были получены значения выходных данных в задаче на вычисление, является **объяснением** решения задачи **с помощью процесса вычисления**. Рассуждение, которое показывает, как по значениям входных данных было установлено, что значения выходных данных находятся в заданном отношении со значениями входных данных в задаче на доказательство, является **объяснением** решения задачи **с помощью процесса доказательства**.

Прикладная задача некоторого класса (представленная значениями входных и выходных данных) согласуется с постановкой этого класса прикладных задач, если она согласуется с онтологией этого класса задач и системой знаний предметной области, а все условия в постановке этого класса задач не являются тождественно ложными, если в них заменить все термины для представления входных и выходных данных их значениями в данной задаче. Если эта задача – за ходных данных их значениями в данной задаче. Если эта задача – задача на вычисление, то будем говорить, что она **имеет решение**. Если в классе прикладных задач задача на вычисление с такими значениями входных данных единственна, то будем говорить, чтоэта **задача имеет единственное решение**.

Каждой прикладной задаче из некоторого класса, входные и выходные данные которой получены в результате наблюдений или вычислены на основе результатов наблюдений, соответствует некоторый элемент концептуализации этого класса задач, содержащий ссылку на некоторую ситуацию действительности предметной области. Представление этой ситуации как элемента концептуализации предметной области является **декларативной формой объяснения** решения задачи любого типа. Действительно, используя эту ситуацию, онтологию и систему знаний предметной области, а также постановку класса прикладных задач, можно показать, что эта ситуация соответствует онтологии и системе знаний предметной области, а также всем условиям в постановке класса задач.

**Спецификация задачи** представляет собой прикладную логическую теорию, моделирующую постановку класса задач.

В качестве примера постановки класса математических задач рассмотрим задачу поиска всех простых чисел, не превосходящих заданного натурального числа. Понятие «простые числа» определены в математике. Значением понятия «входные данные» является набор из одного термина «верхняя граница». Значением понятия «выходные данные» является набор из одного термина «результат». В онтологии класса задач «верхняя граница» определяется как целое, не меньшее двух. В онтологии класса задач «результат» определяется как подмножество простых чисел. Условие состоит в том, что значением понятия «результат» является множество всех простых чисел, не превосходящих значения понятия «верхняя граница».

В качестве примера постановки класса прикладных задач будет рассматриваться следующая задача: найти наименее трудоемкий для миссионеров план переезда через реку с левого берега на правый на лодке некоторой вместимости некоторого числа миссионеров и некоторого числа людоедов при условии, что во время этого переезда ни в лодке, ни на одном из берегов миссионеров не окажется меньше, чем людоедов.

19.2. Зачем нужна спецификация задачи

Спецификации задач при разработке программ необходимы для отладки и тестирования этих программ. Программу, для которой нет спецификации задачи, для решения которой предназначена эта программа, невозможно отлаживать и тестировать.

Кроме того, для разработчика прикладной программы важна следующая информация о прикладной задаче: имеет ли задача решение; если да, то единственно ли это решение; существуют ли методы решения задачи; если да, то какова минимальная вычислительная сложность этих методов; известен ли хотя бы один метод решения задачи; какова вычислительная сложность всех известных методов решения задачи.

Спецификация задачи позволяет свести иcходную прикладную задачу либо к извеcтной математичеcкой задаче, либо к новой (еще не изученной) математичеcкой задаче. Это сведение обычно состоит в замене терминов предметной области абстрактными обозначениями.

В первом cлучае эта извеcтная математичеcкая задача может:

- иметь извеcтные методы решения приемлемой cложноcти;

- быть *NP*-полной (иметь методы решения только неприемлемой, неполиномиальной cложноcти);

- не иметь извеcтных методов решения.

Еcли иcходная прикладная задача cведена к извеcтной математичеcкой задаче, имеющей методы решения приемлемой cложноcти, то разработчик оcвобождаетcя от необходимоcти иccледования поcтановки математичеcкой задачи и может воcпользоватьcя извеcтными и корректными методами ее решения вмеcто разработки новых. Еcли при этом извеcтны cложноcтные характериcтики этих методов, то он может выбрать метод, наилучший c этой точки зрения, из извеcтных.

Еcли иcходная прикладная задача cведена к *NP*-полной задаче (имеющей методы решения только неприемлемой cложноcти), то на доcтаточно ранней cтадии разработчик обнаруживает невозможноcть решить прикладную задачу в иcходной поcтановке. В этом cлучае он имеет возможноcть переcмотреть поcтановку прикладной задачи c целью поcтроения новой, более приемлемой математичеcкой модели задачи.

Еcли иcходная прикладная задача cведена к извеcтной математичеcкой задаче, не имеющей извеcтных методов решения, то тем не менее экономятcя уcилия разработчика на иccледовании поcтановки задачи (cущеcтвования и единственности решения задачи, существование метода решения задачи и его минимальная вычислительная сложность). Кроме того, появляетcя возможноcть разрабатывать метод решения математичеcкой задачи, а не иcходной прикладной.

Наконец, еcли иcходная прикладная задача cведена к новой, еще неизвеcтной математичеcкой задаче, то разработчик и в этом cлучае получает возможноcть иccледовать поcтановку и разрабатывать метод решения математичеcкой задачи, а не иcходной прикладной.

В нашем примере и математическая задача о поиске простых чисел, и прикладная задача о миссионерах и людоедах сводятся к известным математическим задачам, имеющим известные методы решения.

19.3. Основы онтологического анализа задач

Целью онтологического анализа профессиональной деятельности является идентификация всех классов прикладных задач, решаемых в ходе этой профессиональной деятельности, поиск общей адекватной концептуализации всех этих классов задач, построение адекватной онтологии для этой концептуализации, а также формирование постановок и спецификаций всех этих классов задач в терминах этой концептуализации. При этом общая онтология и постановки всех классов прикладных задач являются ключом к пониманию и однозначному толкованию результатов онтологического анализа профессиональной деятельности.

Первым этапом онтологического анализа профессиональной деятельности является идентификация всех классов задач этой профессиональной деятельности. Аналитик может получить предварительный список классов задач у эксперта, определить тип задач каждого класса, а затем он должен некоторое время наблюдать профессиональную деятельность и получать от эксперта пояснения относительно того, что делает каждый участник этой деятельности в каждый момент. Такое наблюдение может пополнить список классов задач профессиональной деятельности. Другим способом достижения той же цели может быть анализ протоколов выполнения этой деятельности за некоторый период (если такие протоколы имеются в распоряжении эксперта).

Вторым этапом онтологического анализа профессиональной деятельности является поиск общей адекватной концептуализации для всех классов задач, решаемых в ходе этой деятельности. Аналитик должен попросить эксперта, участвующего в анализе, сформировать возможно более полный список терминов, используемых для представления входных и выходных данных всех классов задач, а также представительный список примеров прикладных задач (значений входных и выходных данных) всех классов в этих терминах. Составляется список уже использованных значений. В процессе этой работы списки терминов, значений и прикладных задач могут пополняться. Аналитик фиксирует смысл терминов и значений, а также принципы представления задач с их помощью. Самостоятельную часть этого этапа составляет анализ списка значений. Каждое значение должно быть отнесено к некоторой величине, стандартной или нестандартной. Составляется список всех использованных величин, отдельно определяются все нестандартные величины. Если все задачи из списка уже представлены, величины выделены, а смысл всех терминов и принципы представления с их помощью задач понятны аналитику, то можно считать, что этот этап успешно завершен и общая адекватная концептуализация для всех классов задач найдена.

Третьим этапом онтологического анализа профессиональной деятельности является построение для найденной концептуализации общей адекватной онтологии для всех классов задач. Для этого аналитик с помощью эксперта должен решить, какие из уже выделенных терминов являются терминами, обозначающими вспомогательные понятия общей концептуализации всех классов задач, а какие – терминами предметной области. Для терминов, обозначающих вспомогательные понятия общей концептуализации всех классов задач, аналитик с помощью эксперта должен построить их определения, используя в этих определениях термины предметной области и термины концептуализации, уже получившие определения в общей онтологии всех классов задач. Чтобы сделать это, аналитик должен выполнить онтологический анализ предметной области с помощью методов, описанных выше, и построить онтологию предметной области. Поиск ошибок в определениях общей онтологии всех классов задач может быть выполнен с помощью полученного на предыдущем этапе списка задач.

Четвертым этапом онтологического анализа профессиональной деятельности является формирование постановок всех классов задач. Этот этап начинается с построения системы знаний предметной области, за которым следует формулировка условий для каждого класса задач. Заключительная часть этого этапа анализа состоит в поиске ошибок в условиях каждой задачи. Проверяется, что каждая задача из списка примеров согласуется с постановкой соответствующего класса задач. Затем с помощью эксперта составляется список задач каждого класса, у которых значения выходных данных не согласуются со значениями входных данных. Если оказывается, что такие задачи согласуются с постановками соответствующих классов задач, то эти постановки должны быть уточнены.

Система величин, общая онтология всех классов задач, онтология предметной области, система знаний предметной области и постановки всех классов задач являются основными результатами онтологического анализа профессиональной деятельности.

Спецификация математической задачи о простых числах имеет следующий вид.

*входные данные* *≡ {верхняя граница}*

*выходные данные ≡ {результат}*

***сорт*** *верхняя граница: I[2, ∞)*

*результат ≡ {(v: простые числа) v ≤ верхняя граница}*

*простые числа ≡ I[2, ∞) \ {(v1: I[2, ∞))(v2: I[2, ∞)) v1 \* v2}*

Спецификация задачи о миссионерах и людоедах может иметь следующий вид.

*входные данные* *≡ {число миссионеров, число людоедов, вместимость лодки}*

*выходные данные ≡ {гениальная догадка}*

***сорт*** *гениальная догадка: {(v: планы(inf(варианты))) затраты миссионеров(v) = inf(стоимости планов)}*

Гениальная догадка есть решение, наименее трудоемкое для миссионеров.

*затраты миссионеров ≡ (λ (v: планы(inf(варианты))) (Σ (i: I[1, 2 \* inf(варианты) - 1]) /(людоедов в лодке(в пути(v(i))) = 0 ⇒ 1), (людоедов в лодке(в пути(v(i))) > 0 ⇒ 0)/))*

Затраты миссионеров – это количество поездок на лодке, в которых грести приходится миссионерам (в лодке нет людоедов).

*стоимости планов ≡ {(v: планы(inf(варианты))) затраты миссионеров(v)}*

Стоимости планов – это трудоемкости выполнения решений для миссионеров.

Условия в спецификации задачи имеют вид:

*число миссионеров ≥ 2*

Задание № 10 (по теме "Спецификация задачи")

Придумать содержательный пример прикладной задачи, построить ее спецификацию и предложить алгоритм для ее решения.

План ответа

1. Содержательный пример прикладной задачи.
2. Название и характеристика предметной области, в которой может решаться эта задача.
3. Характеристика профессиональной деятельности в этой предметной области.
4. Онтология этой предметной области.
5. Система знаний этой предметной области
6. Спецификация этой предметной области.
7. Спецификация этой задачи: включающая модель предметной области, входные и выходные данные, а также условия задачи.
8. Алгоритм решения задачи.
9. Аргументы в пользу того, что этот алгоритм является методом решения этой задачи.

20. ОБЩЕЕ ПОНЯТИЕ АЛГОРИТМА

20.1. Спецификация алгоритма

Рассмотрим некоторую произвольную концептуализацию, называемую, обычно, **рабочей средой**. Назовем **процессом рассуждения** последовательность элементов рабочей среды, удовлетворяющую следующим ограничениям: любые два соседних элемента в последовательности различны; для любого элемента последовательности, кроме первого, значение любого термина рабочей среды для этого элемента может быть вычислено по значениям терминов рабочей среды для предыдущего элемента с помощью конечной суперпозиции операций и функций соответствующих величин. Элементы последовательности – процесса рассуждения называются **состояниями** этого процесса рассуждения. Первое в последовательности состояние называется **начальным**, а последнее (если оно существует) – **конечным**.

Назовем **методом рассуждения** множество процессов рассуждения с одной и той же рабочей средой. **Алгоритмом** называется такой метод рассуждения, что любые два процесса рассуждения, входящие в этот метод, имеют разные начальные состояния. Процессы рассуждения, принадлежащие алгоритмам, называются вычислительными процессами.

Если алгоритм является нетривиальной совокупностью вербальной информации, то концептуализацию, адекватную для этой совокупности информации, будем называть **концептуализацией** этого **алгоритма**, онтологию, явно представляющую эту концептуализацию, – **онтологией** этого **алгоритма**, а систему знаний, явно представляющую этот алгоритм, - **заданием** этого **алгоритма**. Модель задания алгоритма, представленная на языке спецификации, называется **спецификацией алгоритма**. Модель задания алгоритма, представленная на формальном процедурном языке, называется **программой**. Задание алгоритма, представленное на математическом диалекте, называется **описанием алгоритма**.

Теперь рассмотрим некоторый класс задач и некоторый алгоритм. Если существует взаимно-однозначное соответствие между задачами этого класса и конечными вычислительными процессами этого алгоритма, то будем говорить, что этот алгоритм **может использоваться для решения** задач этого класса. Явное представление отношения между значениями входных данных задач этого класса и начальными состояниями соответствующих вычислительных процессов этого алгоритма будем называть **спецификацией входной процедуры** для этого класса задач и этого алгоритма. Явное представление отношения между конечными состояниями соответствующих вычислительных процессов этого алгоритма и значениями выходных данных задач этого класса будем называть **спецификацией выходной процедуры** для этого алгоритма и этого класса задач.

Рассмотрим пример спецификации алгоритма решения задачи поиска множества простых чисел, не превосходящих заданного числа *N* (Решета Эратосфена). Рабочая среда этого алгоритма есть объединение трех множеств.

*Рабочая среда ≡ Рабочая среда1 ∪ Рабочая среда2 ∪ Рабочая среда3*

*Рабочая среда1 ≡ (решето → {}I[2, N], i → I[2, N])*

*Рабочая среда2 ≡ (решето → {}I[2, N], простые числа → {}I[2, N])*

*Рабочая среда3 ≡ (решето → {}I[2, N], простые числа → {}I[2, N], минимум → I[2, N], j → I[1, N]).*

Спецификация алгоритма состоит из следующих предложений.

***Сорт*** *Решето Эратосфена: {}seq Рабочая среда*

***Свойства начального состояния:***

*(v: Решето Эратосфена) first(v) ∈ Рабочая среда1 & решето(first(v)) = ∅ & i(first(v)) = 2*

***Свойства конечного состояния:***

*(v: Решето Эратосфена) решето(last(v)) = ∅*

***Связи между соседними состояниями:***

*(v: Решето Эратосфена) (v1: I[1, length(v)-1]) π(v1, v) ∈ Рабочая среда1 & i(π(v1, v)) < N ⇒ π(v1+1, v) ∈ Рабочая среда1 & решето(π(v1+1, v)) = решето(π(v1, v)) ∪ {i(π(v1, v))} & i(π(v1+1, v)) = i(π(v1, v)) + 1*

*(v: Решето Эратосфена) (v1: I[1, length(v)-1]) π(v1, v) ∈ Рабочая среда1 & i(π(v1, v)) = N ⇒ π(v1+1, v) ∈ Рабочая среда2 & решето(π(v1+1, v)) = решето(π(v1, v)) & простые числа(π(v1+1, v)) = ∅*

*(v: Решето Эратосфена) (v1: I[1, length(v)-1]) π(v1, v) ∈Рабочая среда2 & решето(π(v1, v))≠∅ ⇒ π(v1+1, v) ∈ Рабочая среда3 & минимум(π(v1+1,v))=inf(решето(π(v1, v)))& решето(π(v1+1, v)) = решето(π(v1, v)) & простые числа(π(v1+1, v)) = простые числа(π(v1, v)) & j(π(v1+1, v)) = 1*

*(v: Решето Эратосфена) (v1: I[1, length(v)-1])π(v1, v) ∈ Рабочая среда3 & минимум(π(v1, v)) \* j(π(v1, v)) ≤ N ⇒ π(v1+1, v) ∈ Рабочая среда3 & решето(π(v1+1, v)) = решето(π(v1, v)) \ {минимум(π(v1, v)) \* j(π(v1, v))}&*

*простые числа(π(v1+1, v)) = простые числа(π(v1, v)) & минимум(π(v1+1, v)) = минимум(π(v1, v)) & j(π(v1+1, v)) = j(π(v1, v)) + 1*

*(v: Решето Эратосфена) (v1: I[1, length(v)-1])π(v1, v) ∈ Рабочая среда3 & минимум(π(v1, v)) \* j(π(v1, v)) > N ⇒ π(v1+1, v) ∈ Рабочая среда2 & решето(π(v1+1, v)) = решето(π(v1, v)) & простые числа(π(v1+1, v)) = простые числа(π(v1, v)) \ {минимум(π(v1, v))}*

20.2. Описание алгоритмов

Существенным свойством алгоритма является то, что он всегда представлен в виде текста на некотором языке, причем один и тот же алгоритм может быть представлен на разных языках и даже в виде разных текстов на одном и том же языке.

Языки, используемые для описания алгоритмов, относятся к одному семантическому классу - языков с повелительной семантикой (командных языков). Текст на таком языке есть последовательность предложений, каждое из которых является повелительным предложением (командой), предписывающим исполнителю алгоритма (человеку или компьютеру) выполнить некоторое действие (обработку данных). Заметим, что языки спецификаций относятся к другому семантическому классу - языков с декларативной семантикой, в котором каждое предложение есть декларация о свойствах всех объектов моделирования, а семантика языка позволяет для любого объекта моделирования определить, обладает ли он декларируемыми свойствами.

20.3. Алгоритмы, вычисляющие функцию

Алгоритм, который для любого допустимого входа всегда даёт некоторый выход, называется алгоритмом вычисления функции, отображающей допустимые входы на эти выходы. Различные тексты на языках с повелительной семантикой, исполнение которых (исполнение содержащихся в этих текстах команд) вычисляет одну и ту же функцию, являются представлениями одного и того же алгоритма, вычисляющего функцию. Заметим, что существуют и другие алгоритмы (не вычисляющие функцию). Примером могут служить алгоритмы взаимодействия со средой, результатом исполнения которых являются изменения в окружающем мире. Далее будут рассматриваться только алгоритмы, вычисляющие функцию.

20.4. Алгоритм

Описание алгоритма - это его запись на некотором языке представления алгоритмов, т.е. некоторое сообщение. Рецепт - это правила интерпретации этого сообщения и входа алгоритма. Результатом интерпретации задания алгоритма и его входа является вычислительный процесс.

Процесс интерпретации задания алгоритма(выполнение рецепта) состоит из шагов.

На первом шагеинтерпретируется первая команда задания алгоритма и первое состояние вычислительного процесса. Результатами интерпретацииявляются команда в задании алгоритма и состояние вычислительного процесса, определенные для следующего шага.

На очередном шагеинтерпретируются команда и состояние, определённые для этого шага на предыдущем шаге. Процесс получения нового состояния вычислительного процесса из его предыдущего состояния при интерпретации команды на очередном шаге называется непосредственной переработкой*.*

Процесс интерпретации (выполнение рецепта) завершается, если выполнено условие остановки (нормальное завершение) или очередной шаг не может быть выполнен (аварийное завершение).

Заметим, что вычислительные процессы, получаемые при интерпретации задания алгоритмов и некоторых входов, могут быть и бесконечными.

21. Вычислительные модели

21.1. Программа

Вычислительная модель отличается от любого другого задания алгоритмов тем, что язык описания алгоритмов является формальным. Описание алгоритма на языке некоторой вычислительной модели называется программой. Программа является лишь одним из его возможных представлений.

Вычислительные модели могут быть материальными (компьютеры), компьютерными (процессоры языков программирования) и математическими.

21.2. Устройство вычислительной модели

Вычислительная модель состоит из языка этой модели, рецепта, входного ансамбля, выходного ансамбля, ансамбля состояний, входной и выходной процедур. На языке модели описываются все алгоритмы, которые могут быть заданы в этой модели. Рецепт есть способ интерпретации программы. При этом ансамбль входов каждого такого алгоритма совпадает с входным ансамблем модели, ансамбль выходов - с выходным ансамблем модели, а все состояния любого вычислительного процесса при выполнении рецепта принадлежат ансамблю состояний этой модели. Входная процедура модели отображает входной ансамбль в ансамбль состояний, а выходная процедура - ансамбль состояний в выходной ансамбль.

21.3. Нормальные алгоритмы Маркова

Нормальные алгоритмы Маркова являются примером математической вычислительной модели. Ее **входным ансамблем** является множество строк конечной длины, построенных из элементов некоторого конечного набора знаков. **Выходной ансамбль** и **ансамбль состояний** совпадают с входным ансамблем. **Входная и выходная процедуры** реализуют тождественные отображения. **Программа** есть конечная последовательность продукций. Каждая **продукция** состоит из антецедента и консеквента. И **антецедент**, и **консеквент** – это строки конечной длины, состоящие из знаков того же набора и имен переменных. Имя любой переменной, входящее в консеквент некоторой продукции, должно входить и в антецедент этой же продукции. **Выполнение рецепта** состоит в следующем. Первое состояние вычислительного процесса совпадает с его входом. На очередном шаге (в том числе и на первом) ищется первая в последовательности (программе) применимая к текущему состоянию продукция. Продукция является **применимой** к состоянию, если существует такая подстановка строк вместо имен переменных антецедента этой продукции (разные вхождения имени одной и той же переменной заменяются одной и той же строкой), что этот антецедент после выполнения этой подстановки является подстрокой этого состояния. Для первой в последовательности применимой продукции ищется самое раннее ее применение. Одно применение продукции к состоянию (т.е. подстановка вместо переменных ее антецедента) является более ранним, чем другое, если результат применения первой подстановки к антецеденту этой продукции является подстрокой состояния, начинающейся с более раннего элемента состояния, чем результат применения второй подстановки. После того, как найдена первая в последовательности применимая к текущему состоянию продукция и ее самое раннее применение, эта продукция применяется. При этом из текущего состояния удаляется подстрока, совпадающая с результатом применения подстановки к антецеденту продукции, и на это место в состояние вставляется результат применения той же подстановки к консеквенту этой продукции. В результате получается следующее состояние вычислительного процесса. Вычислительный процесс завершается нормально, если на очередном шаге применяется продукция, помеченная специальным символом "!". Вычислительный процесс завершается аварийно, если на очередном шаге ни одна продукция не является применимой.

Рассмотрим несколько примеров программ обработки целочисленной информации на языке нормальных алгоритмов Маркова. В них целое число n будет представляться строкой, состоящей из n палочек, а знак "⇒" разделяет антецедент и консеквент.

Сложение натуральных чисел. Вход – представление двух слагаемых, первое из которых заканчивается знаком "#". Формальное задание алгоритма

⏐#⏐ ⇒ ⏐⏐!

Пример вычислительного процесса

⏐⏐⏐#⏐⏐ → ⏐⏐⏐⏐⏐

Вычитание из большего натурального числа меньшего. Вход – представление уменьшаемого и вычитаемого, разделенных знаком "#". Формальное задание алгоритма

⏐#⏐ ⇒ #

# ⇒ !

Пример вычислительного процесса

⏐⏐⏐#⏐⏐ → ⏐⏐#⏐ → ⏐# → ⏐

Суммирование массива натуральных чисел. Вход – представление элементов массива, каждый из которых заканчивается знаком "#", а весь массив - знаком "⊥". Формальное задание алгоритма

⏐#⏐⇒ ⏐⏐

⊥⇒ ⏐!

Пример вычислительного процесса

⏐⏐#⏐⏐⏐⏐#⏐⊥ → ⏐⏐⏐⏐⏐⏐#⏐⊥ → ⏐⏐⏐⏐⏐⏐⏐⊥ → ⏐⏐⏐⏐⏐⏐⏐

21.4. Тезис Черча

Вычислительная модель называется **представительной**, если в ней может быть описан любой алгоритм (т.е. любое множество вычислительных процессов). **Тезис Чёрча в широком смысле** утверждает, что существуют представительные вычислительные модели. **Тезис Чёрча в узком смысле** - утверждение, что конкретная вычислительная модель является представительной (например, что нормальные алгоритмы Маркова являются представительной вычислительной моделью). Доказано, что все вычислительные модели, относительно которых выдвинут тезис Чёрча в узком смысле, эквивалентны между собою. В частности, показано, что компьютеры и "универсальные" языки программирования и их процессоры эквивалентны представительным вычислительным моделям.

Одним из возможных смыслов всякого понятия является его объём, т.е. множество объектов, которые соответствуют этому понятию. С этой точки зрения объёмом понятия "алгоритм" является множество всех возможных алгоритмов. В силу тезиса Чёрча на языке любой представительной вычислительной модели могут быть представлены все возможные алгоритмы. Любая представительная вычислительная модель является моделью понятия "алгоритм", поскольку в ней представлено указанное выше его существенное свойство.

Рецепт представительной вычислительной модели называется **универсальным алгоритмом** или **универсальным рецептом**. Он представляет собой алгоритм "выполнения любого алгоритма". Поэтому универсальный рецепт также может быть представлен на языке представительной вычислительной модели. Это свойство представительных вычислительных моделей называется их самоприменимостью.

Задание № 11 (по теме "Вычислительные модели")

Придумать пример задачи и написать программу её решения на языке нормальных алгоритмов Маркова.

План ответа

1. Пример задачи.
2. Представление входных данных задачи в вычислительной модели "нормальные алгоритмы Маркова".
3. Представление выходных данных задачи в вычислительной модели "нормальные алгоритмы Маркова".
4. Программа на языке нормальных алгоритмов Маркова (с комментариями!), реализующая алгоритм решения задачи.
5. Пример вычислительного процесса по этой программе.

22. СПЕЦИФИКАЦИЯ ПРОГРАММ

22.1. Что такое "спецификация программы"

Программа - это запись алгоритма на процедурном языке; **спецификация программы** - запись той же программы на декларативном языке (языке спецификаций). Другими словами, спецификация - это описание множества процессов исполнения программы при всех возможных входах.

22.2. Зачем нужна спецификация программ

Спецификация программы используется для достижения двух целей:

1. для восстановления спецификации задачи, для которой алгоритм, реализуемый программой, является методом решения этой задачи;
2. как система аксиом, которая содержит информацию о программе и которая может быть использована для доказательства правильности этой программы.

22.3. Различные формы представления программы при построении ее спецификации

Исходным представлением программы является текст на языке программирования. Другой эквивалентной формой представления такой программы может быть ее блок-схема. Введем еще одно эквивалентное представление программ, называемое граф-программой. **Граф-программа** - это орграф, в котором имеется одна и только одна входная вершина (в которую не входит ни одна дуга), одна и только одна выходная вершина (из которой не выходит ни одна дуга), а на дугах имеются метки двух типов: контрольные предикаты (логические выражения над идентификаторами программы) и множества операторов присваивания, причем контрольные предикаты на дугах, выходящих из одной вершины, образуют полную систему альтернатив (т.е. при любых значениях переменных истинным является один и только один контрольный предикат на таких дугах), а в каждом множестве операторов присваивания все их левые части различны. Отсутствие на дуге контрольного предиката означает его тождественную истинность.

Исполнение граф-программы определяется следующим образом:

1. исполнение начинается с активизации входной вершины, при этом входным переменным программы присваиваются начальные значения (исходные данные);
2. если активизирована некоторая вершина, не являющаяся выходной, то вычисляются значения контрольных предикатов на дугах, выходящих из нее; выполняются все операторы присваивания на той дуге, на которой контрольный предикат истинен; активизируется вершина, в которую входит эта дуга;
3. если активизирована выходная вершина, то исполнение граф-программы завершается.

Любую программу можно представить в форме граф-программы.

В качестве примера рассмотрим программу на условном языке программирования, вычисляющую все простые числа, не превосходящие данного числа, методом решета Эратосфена.

**целые** *N, минимум;* **множество целых** *простые числа, решето;*

**ввод** *N;*

*простые числа := ∅;*

*решето := ∅;*

**для** *i* **от** *2* **до** *N* **цк**

*решето := решето ∪ {i}* **кц**;

**пока** *решето ≠ ∅* **цк**

*минимум := min(решето);*

**для** *j* **от** *1* **шаг** *1* **пока** *минимум \* j ≤ N* **цк**

*решето := решето \ {минимум \* j}* **кц**;

*простые числа := простые числа ∪ {минимум}* **кц**;

Обозначим *S* – входную вершину граф-программы, *F* – ее выходную вершину, а *R1, R2, R3* – ее остальные вершины. В этих обозначениях граф-программа состоит из следующих дуг:

*S → R1*, контрольный предикат отсутствует, операторы присваивания – *{простые числа := ∅; решето := ∅; i := 2}*

*R1 → R1*, контрольный предикат – *i ≤ N*, операторы присваивания – *{решето := решето ∪ {i}; i := i +1}*

*R1 → R2*, контрольный предикат – *i > N*, операторы присваивания отсутствуют

*R2 → R3*, контрольный предикат – *решето ≠ ∅*, операторы присваивания - *{минимум := min(решето); j := 1}*

*R2 → F*, контрольный предикат – *решето = ∅*, операторы присваивания отсутствуют

*R3 → R3*, контрольный предикат – *минимум \* j ≤ N*, операторы присваивания - *{решето := решето \ {минимум \* j}; j := j + 1}*

*R3 → R2*, контрольный предикат – *минимум \* j > N*, операторы присваивания - *{простые числа := простые числа ∪ {минимум}}*

22.4. Спецификация программы

Сопоставим входной вершине граф-программы предикатное имя *S*, аргументами которого являются входные переменные (идентификаторы) программы. Сопоставим выходной вершине граф-программы предикатное имя *F*, аргументами которого являются входные переменные и выходные переменные (идентификаторы) программы. Всем остальным вершинам граф-программы взаимно однозначно сопоставим предикатные имена, обозначаемые буквой *R* с индексами, аргументами которых являются входные переменные и промежуточные переменные (идентификаторы) программы. Каждой дуге граф-программы сопоставим предложение языка прикладной логики, префикс которого содержит описания всех переменных этого предложения, а тело является формулой, имеющей вид: *P1(****v****) & f(****v****) ⇒ P2(****t****)*, где *P1* - предикатное имя, сопоставленное вершине, из которой выходит дуга, ***v*** – вектор переменных - аргументов этого предикатного имени, *f(****v****)* - контрольный предикат дуги, *P2* - предикатное имя, сопоставленное вершине, в которую входит дуга, а ***t*** - вектор термов - аргументов этого предикатного имени, в котором вместо тех переменных, которые являются левыми частями операторов присваивания на дуге, помещены правые части этих операторов присваивания.

**Спецификация программы** состоит из предложений, описывающих сорта всех предикатных имен, сопоставленных вершинам граф-программы, и предложений, сопоставленных дугам граф-программы.

В нашем примере спецификацией программы является следующая прикладная логическая теория.

***сорт*** *S: I[2, ∞) → L*

***сорт*** *R1: (× I[2, ∞), {}I[2, ∞), I[2, ∞)) → L*

***сорт*** *R2: (× I[2, ∞), {}I[2, ∞), {}I[2, ∞)) → L*

***сорт*** *R3: (× I[2, ∞), {}I[2, ∞), {}I[2, ∞), I[2, ∞), I[1, ∞)) → L*

***сорт*** *F: (× I[2, ∞), {}I[2, ∞)) → L*

*(N: I[2, ∞)) S(N) ⇒ R1(N, ∅, 2)*

*(N: I[2, ∞))(решето: {}I[2, ∞))(i: I[2, ∞)) R1(N, решето, i) & i ≤ N ⇒ R1(N, решето ∪ {i}, i + 1)*

*(N: I[2, ∞))(решето: {}I[2, ∞))(i: I[2, ∞)) R1(N, решето, i) & i > N ⇒ R2(N, решето, ∅)*

*(N: I[2, ∞))(решето: {}I[2, ∞))(простые числа: {}I[2, ∞)) R2(N, решето, простые числа) & решето ≠ ∅ ⇒ R3(N, решето, простые числа, inf(решето), 1)*

*(N: I[2, ∞))(решето: {}I[2, ∞))(простые числа: {}I[2, ∞))(минимум: I[2, ∞))(j: I[1, ∞)) R3(N, решето, простые числа, минимум, j) & минимум \* j ≤ N ⇒ R3(N, решето \ {минимум \* j}, простые числа, минимум, j + 1)*

*(N: I[2, ∞))(решето: {}I[2, ∞))(простые числа: {}I[2, ∞))(минимум: I[2, ∞))(j: I[1, ∞)) R3(N, решето, простые числа, минимум, j) & минимум \* j > N ⇒ R2(N, решето, простые числа ∪ {минимум})*

*(N: I[2, ∞))(решето: {}I[2, ∞))(простые числа: {}I[2, ∞)) R2(N, решето, простые числа) & решето = ∅ ⇒ F(N, простые числа)*

22.5. Модели спецификации программы

Логическую модель спецификации программы можно считать моделью процесса выполнения этой программы при некотором входе. Спецификация любой программы имеет модель, так как моделью этой спецификации является в том числе и логическая модель с тождественно ложной интерпретацией всех предикатных имен граф-программы.

Задание № 12 (по теме "Спецификация программы")

Придумать содержательный пример прикладной задачи, написать программу её решения и построить спецификацию этой программы.

План ответа

1. Содержательный пример прикладной задачи.
2. Программа, реализующая алгоритм решения задачи.
3. Представление программы в форме граф-программы.
4. Спецификация программы.
5. Модель этой спецификации программы для некоторых входных данных.

23. Сложность вычислений

23.1. Время и ёмкость

Реальные вычислительные процессы протекают в физическом времени и пространстве. Под пространством здесь понимается размер памяти исполнителя (компьютера, бумажного носителя или человека), необходимый этому вычислительному процессу. В теории сложности вычислений эти свойства вычислительных процессов представляются в соответствующих моделях и называются, соответственно, **временем** и **ёмкостью**. Время и ёмкость вычислительных процессов, соответствующих некоторому алгоритму, зависят от вычислительной модели.

23.2. Время вычислительного процесса

Время вычислительного процесса определяется как количество его шагов. Если шаги вычислительного процесса не являются равнозначными, то каждому шагу может быть приписан его вес, а время определяется как сумма этих весов. Для любого вычислительного процесса может быть определено время его выполнения, которое называется его временной сложностью.

23.3. Ёмкость вычислительного процесса

Для любой вычислительной модели ёмкостью вычислительного процесса называется максимальный размер состояния этого вычислительного процесса. Размер состояния вычислительного процесса - это количество элементарных объектов в этом состоянии.

Для любого вычислительного процесса при фиксированных элементарных объектах и способе кодирования состояний может быть определена его ёмкость, которая называется его ёмкостной сложностью.

23.4. Эффективные алгоритмы

**Сложностью** алгоритма называется функция, которая размеру входа каждого вычислительного процесса, являющегося результатом интерпретации задания этого алгоритма, ставит в соответствие временную или ёмкостную сложность этого процесса. Если размер входа стремится к бесконечности, а эту функцию можно представить в виде конечной суммы бесконечно больших величин и константы, то **асимптотической сложностью** алгоритма называется то слагаемое (бесконечно большая величина или константа), которое имеет наивысший порядок; в случае, если в сумме бесконечно большие величины отсутствуют, асимптотическая сложность есть константа.

Например, программа вычисления суммы элементов массива m длины *n* имеет следующий вид:

s := 0;

для i = 1 шаг 1 до n цк

s := s + m[i] кц;

Можно считать, что временная сложность первой строки этой программы есть *1*, второй строки – *1 + 2 \* n* (на каждом шаге увеличение параметра цикла и сравнение его с *n*), третьей строки – *3 \* n* (на каждом шаге цикла вычисление индексного выражения, сложение и запись). Таким образом, временная сложность этой программы – *2 + 5 \* n*, а ее асимптотическая сложность – *5 \* n*, если *n* стремится к бесконечности.

Алгоритм, вычисляющий некоторую функцию, называется **абсолютно эффективным**, если среди всех алгоритмов, вычисляющих эту функцию, он имеет минимальную асимптотическую сложность. Алгоритм, вычисляющий некоторую функцию, называется **относительно эффективным**, если среди всех известных алгоритмов, вычисляющих эту функцию, он имеет минимальную асимптотическую сложность.

Задание № 13 (по теме "Сложность вычислений")

Придумать пример прикладной задачи, алгоритм её решения и оценить сложность этого алгоритма.

План ответа

1. Придумать содержательный пример прикладной задачи.
2. Написать программу её решения.
3. Для выбранных входных данных вычислить временную сложность вычислительного процесса, определяемого этой программой и этими данными.
4. Оценить асимптотическую сложность алгоритма, реализуемого этой программой.

24. Общее понятие исчисления

24.1. Спецификация исчисления

**Исчислением** называется такой метод рассуждения, что существуют по меньшей мере два разных процесса рассуждения, входящих в этот метод, у которых их начальные состояния совпадают. Процессы рассуждения, входящие в исчисление называют **порождающими процессами**. Если начальные состояния всех порождающих процессов, входящих в исчисление, совпадают, то такое **исчисление** называется **порождающим множество**. Элементами этого множества являются конечные состояния порождающих процессов (возможно преобразованные с помощью выходной процедуры). Исчисление называется **исчислением со входом**, если существует по меньшей мере два процесса рассуждения, входящих в это исчисление, у которых начальные состояния различны. В этом смысле можно говорить, что исчисление со входом вычисляет функцию, которая каждому допустимому входу ставит в соответствие множество, порождаемое исчислением для этого входа. **Конфлюентным** называется такое **исчисление со входом**, что любые два порождающих процесса, входящие в это исчисление и имеющие одинаковые начальные состояния, имеют и одинаковые конечные состояния.

Теперь рассмотрим некоторый класс задач и некоторое конфлюентное исчисление со входом. Будем говорить, что два порождающих процесса, входящих в это конфлюентное исчисление со входом, эквивалентны, если их начальные состояния совпадают. Из определения конфлюентных исчислений следует, что у эквивалентных порождающих процессов и конечные состояния совпадают. Если существует взаимно-однозначное соответствие между задачами этого класса и классами эквивалентности порождающих процессов этого конфлюентного исчисления со входом, то будем говорить, что это конфлюентное исчисление со входом **может использоваться для решения** задач этого класса. Явное представление отношения между значениями входных данных задач этого класса и начальными состояниями порождающих процессов из соответствующих классов эквивалентности этого конфлюентного исчисления со входом будем называть **спецификацией входной процедуры** для этого класса задач и этого конфлюентного исчисления со входом. Явное представление отношения между конечными состояниями порождающих процессов из соответствующих классов эквивалентности этого конфлюентного исчисления со входом и значениями выходных данных задач этого класса будем называть **спецификацией выходной процедуры** для этого конфлюентного исчисления со входом и этого класса задач.

Если исчисление является нетривиальной совокупностью вербальной информации, то концептуализацию, адекватную для этой совокупности информации, будем называть **концептуализацией** этого **исчисления**, онтологию, явно представляющую эту концептуализацию, – **онтологией** этого **исчисления**, а систему знаний, явно представляющую это исчисление, - **заданием** этого **исчисления**. Модель задания исчисления, представленная на языке спецификации, называется **спецификацией исчисления**. Модель задания исчисления, представленная на формальном процедурном языке, называется **логической программой**. Задание исчисления, представленное на математическом диалекте, называется **описанием исчисления**.

24.2. Задание исчисления

Существенным свойством исчисления, как и алгоритма, является то, что оно всегда представлено в виде текста на некотором языке, причем одно и то же исчисление может быть представлено на разных языках и даже в виде разных текстов на одном и том же языке. Таким образом, исчисление - это нечто общее для всех этих представлений.

Языки, используемые для представления исчислений, относятся к одному семантическому классу - **языков с разрешительной семантикой** (языков правил). В отличие от случая алгоритмов, текст на таком языке есть неупорядоченная совокупность предложений, каждое из которых является разрешительным предложением (правилом), разрешающим исполнителю исчисления (человеку или компьютеру) при выполнении определённых условий выполнить некоторое действие (обработку данных). Каждое **правило** в представлении исчисления состоит из двух частей: антецедента и консеквента. **Антецедент** описывает условие, при выполнении которого разрешено выполнять действие, описываемое **консеквентом**.

В антецедент любого правила могут входить переменные и константы. Они называются **посылками** правила. Правила, не содержащие посылок, называются 0-посылочными. Антецеденты таких правил "пусты " (отсутствуют).

24.3. Интерпретация исчисления

Задание исчисления - это описание этого исчисления на некотором языке представления исчислений, т.е. некоторое сообщение. **Рецепт** - это правила интерпретации этого сообщения (и входа - для исчисления со входом). Результатом интерпретации задания исчисления (и его входа) является порождающий процесс.

Для исчисления, порождающего множество, первое состояние порождающего процесса - одно и то же для всех порождающих процессов, являющихся результатом интерпретации задания этого исчисления. Оно определяется 0-посылочными правилами в задании этого исчисления, а именно: выполняются консеквенты всех этих правил, а результаты их выполнения образуют это первое состояние всех порождающих процессов. Для исчисления со входом первое состояние порождающего процесса получается с помощью входной процедуры, которая преобразует вход исчисления в это первое состояние. Процесс интерпретации задания исчисления (выполнение рецепта) состоит из шагов. На первом шаге интерпретируется первое состояние порождающего процесса. На очередном шаге интерпретируется состояние, полученное на предыдущем шаге. На каждом шаге определяется **применимое правило**. Для этого проверяются условия, определяемые антецедентами всех правил, относительно состояния порождающего процесса на этом шаге. Условие, определяемое антецедентом, выполнено, если для каждой посылки правила, являющейся константой, в состоянии процесса на данном шаге существует объект, совпадающий с этой константой, и для каждой посылки, являющейся переменной, в состоянии существует объект, такой, что при замене всех таких посылок соответствующими им объектами антецедент (являющийся логической формулой) истинен. Объекты состояния процесса на данном шаге, соответствующие посылкам (константам и переменным), называются значениями этих посылок. Те правила, для которых условия, определяемые их антецедентами, выполнены, называются применимыми на этом шаге. Для каждого применимого правила множество допустимых наборов значений его посылок (при которых выполнено условие, определяемое антецедентом) не пусто. Среди применимых правил произвольным образом выбирается одно правило и один допустимый набор значений посылок; это правило применяется на этом шаге при этом допустимом наборе значений посылок. Его применение состоит в интерпретации консеквента этого правила, в котором все вхождения посылок-переменных заменены соответствующими значениями из допустимого набора значений, и состояния процесса на этом шаге. Результатом интерпретации является состояние порождающего процесса, определенное для следующего шага. Процесс получения нового состояния порождающего процесса из его предыдущего состояния при интерпретации консеквента применяемого правила на очередном шаге называется непосредственной переработкой. Процесс интерпретации (выполнение рецепта) завершается, если выполнено условие остановки (нормальное завершение) или если на очередном шаге множество применимых правил пусто (аварийное завершение).

Порождение, т.е. способ построения порождающих процессов, отличается от вычисления, т.е. способа построения вычислительных процессов, тем, что при порождении выбор правила и набора значений посылок, применяемых на очередном шаге, происходит произвольным образом из множества применимых правил и допустимых наборов значений посылок (соответственно), в то время как при вычислении команда, применяемая на очередном шаге, и активная часть состояния этого шага определены однозначно.

**Выводом** в исчислении (соответствующим порождающему процессу) называется такая последовательность пар, что каждая пара соответствует одному шагу порождающего процесса и состоит из правила, применённого на этом шаге, и набора значений посылок, при котором это правило было применено.

**Выходом** исчисления, порождающего множество (исчисления со входом для заданного входа), является множество объектов, элементы которого получаются как результат применения выходной процедуры к последним состояниям всех порождающих процессов, являющихся интерпретацией задания этого исчисления (и этого входа).

Заметим, что порождающие процессы, получаемые при интерпретации задания исчислений (и некоторых входов), могут быть и бесконечными.

24.4. Некоторые понятия, связанные с исчислениями

Множество называется **породимым**, если существует такое исчисление, которое порождает это множество. Исчисление называется **полным** относительно некоторого множества, если оно порождает это множество. Множество называется **разрешимым**, если существует алгоритм, позволяющий для любого элемента супермножества, в которое входит это множество, определить, принадлежит этот элемент этому множеству или нет.

25. Порождающие модели

25.1. Устройство порождающей модели

**Порождающая модель** состоит из формального языка этой модели, рецепта, входного ансамбля (для порождающих моделей, моделирующих исчисление со входом), выходного ансамбля, ансамбля состояний, входной (для порождающих моделей, моделирующих исчисление со входом) и выходной процедур. На языке модели представляются все исчисления, которые могут быть заданы в этой модели. Представление исчисления на этом языке называется **формальным заданием** этого исчисления. **Рецепт** есть способ интерпретации формального задания этих исчислений. При этом ансамбль выходов каждого такого исчисления совпадает с выходным ансамблем модели, а все состояния любого порождающего процесса при выполнении рецепта принадлежат ансамблю состояний этой модели. Если порождающая модель моделирует исчисление со входом, то ансамбль входов каждого такого исчисления совпадает со входным ансамблем модели. Входная процедура модели (для порождающих моделей, моделирующих исчисление со входом) отображает входной ансамбль в ансамбль состояний, а выходная процедура - ансамбль состояний в выходной ансамбль.

25.2. Порождающая модель Поста

**Порождающая модель Поста** моделирует исчисления со входом. Ее входной ансамбль, выходной ансамбль, ансамбль состояний, входная и выходная процедуры совпадают с таковыми для нормальных алгоритмов Маркова. Задание исчисления в модели Поста есть множество (а не последовательность) продукций, причем вид продукций совпадает с их видом для нормальных алгоритмов Маркова. Выполнение рецепта состоит в следующем. Первое состояние порождающего процесса формируется так же, как и в случае нормальных алгоритмов Маркова. На очередном шаге определяется множество применимых продукций и для каждой применимой продукции множество всех допустимых наборов значений ее посылок. Определение применимой продукции совпадает с таковым для нормальных алгоритмов Маркова. Затем из множества применимых на этом шаге продукций произвольным образом выбирается одна, а из множества допустимых на этом шаге наборов значений ее посылок произвольным образом выбирается один набор и эта продукция применяется при этом наборе значений посылок так же, как и в случае нормальных алгоритмов Маркова.

25.3. Тезис Поста

Порождающая модель называется **представительной**, если в ней может быть задано любое исчисление (исчисление со входом). **Тезис Поста** в широком смысле утверждает, что существуют представительные порождающие модели. Тезис Поста в узком смысле - утверждение, что конкретная порождающая модель является представительной (например, что модель Поста является представительной порождающей моделью). Доказано, что все порождающие модели, относительно которых выдвинут тезис Поста в узком смысле, эквивалентны между собою.

Объёмом понятия "исчисление" (т.е. существенным свойством этого понятия) является множество всех возможных исчислений. В силу тезиса Поста на языке любой представительной порождающей модели могут быть представлены все возможные исчисления. Любая представительная порождающая модель является моделью понятия "исчисление", поскольку в ней представлено указанное выше его существенное свойство.

Рецепт представительной порождающей модели называется универсальным. Он представляет собой недетерминированный алгоритм "выполнения любого исчисления".

25.4 Задача поиска вывода в исчислении

**Прямая задача** поиска вывода в исчислении, порождающем множество, состоит в поиске некоторого вывода по формальному заданию этого исчисления, а в исчислении со входом - в поиске некоторого вывода по формальному заданию этого исчисления и его входу. Вообще говоря, эта задача имеет не единственное решение. **Обратная задача** поиска вывода в исчислении, порождающем множество, состоит в поиске некоторого вывода по формальному заданию этого исчисления и элементу множества, порождаемого этим исчислением, а в исчислении со входом - в поиске некоторого вывода по формальному заданию этого исчисления, его входу и элементу множества, порождаемого этим исчислением при этом входе. Обратная задача также может иметь не единственное решение.

26. РЕЛЯЦИОННЫЕ КОНФЛЮЕНТНЫЕ ПРОДУКЦИИ

26.1. Системы продукций

В программировании и искусственном интеллекте **системами продукций** называются языки программирования (языки представления знаний), основанные на представительных порождающих моделях. Заметим, что язык программирования ПРОЛОГ не относится к системам продукций, поскольку он основан на конкретном исчислении (предикатов первого порядка), а не на порождающей модели. **Система реляционных продукций** - это такая система продукций, формальные задания исчислений в которой представляются на ограниченном языке исчисления предикатов, а формальный рецепт согласован с правилами вывода в исчислении предикатов. **Система реляционных конфлюентных продукций** – это такая система реляционных продукций, которая обладает свойством конфлюентности.

26.2. Формальное задание исчисления

Формальное задание исчисления в системах реляционных конфлюентных продукций есть множество формул вида

*p1(****v1****) & ... & pm(****vm****) & f├****v****┤ ⇒ q1(****t1****) & ... & qn(****tn****),*

где *p1,...,pm, q1,..., qn* - предикатные символы; ***v1****,...,****vm****,* ***v*** - векторы переменных, причём каждая переменная, входящая в вектор ***v***, входит хотя бы в один из векторов ***v1****,...,****vm***; *f├****v****┤* - формула языка исчисления предикатов первого порядка, в которую входят только переменные из вектора ***v***; ***t1****,...,****tn*** - векторы термов языка исчисления предикатов первого порядка, в которые входят только переменные из векторов ***v1****,...,****vm***.

26.3. Состояния порождающего процесса

Любое состояние порождающего процесса есть конечное множество атомарных формул вида *r(c1,...,ck)*, где *r* - предикатный символ, а *c1,...,ck* - предметные символы. Ансамбли входов и выходов совпадают с ансамблем состояний порождающего процесса, а входная и выходная процедуры являются тождественными преобразованиями.

26.4. Рецепт

Правило исчисления является применимым, если существует подстановка *λ* предметных символов вместо всех переменных, входящих в векторы ***v1****,...,* ***vm***, такая, что *[pi(****vi****)⏐λ]* для всех *i = 1,...,m* входит в текущее состояние порождающего процесса, *[f├****v****┤⏐λ]* истинна и существует такое *j (1≤ j ≤ n )*, что *[qj(****tj****)⏐λ]* не входит в текущее состояние порождающего процесса.

Если на очередном шаге применяется некоторое правило, значения посылок которого определяются подстановкой *λ*, то состояние порождающего процесса на следующем шаге получается из состояния на текущем шаге добавлением тех *[qj(****tj****)⏐λ]*, получаемых из консеквента применяемого правила, которые не входят в текущее состояние порождающего процесса. Порождающий процесс завершается, если на очередном шаге множество применимых правил пусто.

26.5. Свойства систем реляционных конфлюентных продукций

Системы реляционных конфлюентных продукций являются представительной порождающей моделью.

Исчисления, задаваемые в рамках систем реляционных конфлюентных продукций, являются монотонными, т.е. для каждого порождающего процесса его состояние на очередном шаге является собственным подмножеством его состояния на следующем шаге.

Все исчисления, задаваемые в рамках систем реляционных конфлюентных продукций, являются конфлюентыми. Для конфлюентного исчисления, порождающего множество, если хотя бы один его порождающий процесс при данном входе завершается, то и все его порождающие процессы при этом входе завершаются, а конечные состояния всех этих порождающих процессов совпадают. Реляционные конфлюентные продукции могут использоваться как средство описания методов решения задач. При этом реализация этой порождающей модели решает прямую задачу поиска вывода.

Рассмотрим в качестве примера описание метода построения таблицы возможных обстановок в лодке в задаче о миссионерах и людоедах (см. раздел 16). Входное состояние представляется одноместными предикатами "*чисмис*", аргументом которого является число миссионеров, "*числюд*", аргументом которого является число людоедов, и "*вместлодки*", аргументом которого является вместимость лодки. Выходное состояние представляется двухместным предикатом "*обствлодке*", первым аргументом которого является возможное число миссионеров в лодке, а вторым – возможное число людоедов в лодке при условии, что выполняется закон безопасности миссионеров в лодке. Входное состояние есть множество, состоящее из трех атомных формул: *чисмис(n1), числюд(n2), вместлодки(n3)*. Метод построения таблицы возможных обстановок в лодке в задаче о миссионерах и людоедах описывается следующей системой продукций:

*чисмис(v1) & вместлодки(v2) ⇒ обствлодке(min(v1, v2), 0)*

*числюд(v1) & вместлодки(v2) ⇒ обствлодке(0, min(v1, v2))*

*обствлодке(v, 0) & v > 1 ⇒ обствлодке(v - 1, 0)*

*обствлодке(0, v) & v > 1 ⇒ обствлодке(0, v - 1)*

*обствлодке(v1, v2) & вместлодки(v3) & числюд(v4) & v1 – v2 ≥ 1 & v1 + v2 + 1 ≤ v3 & v2 < v4 ⇒ обствлодке(v1, v2 + 1)*

Задание № 14 (по теме "Системы реляционных конфлюентных продукций")

План ответа

1. Придумать содержательный пример прикладной задачи.
2. Придумать представление входных данных этой задачи.
3. Придумать представление выходных данных этой задачи.
4. Написать систему реляционных конфлюентных продукций для решения этой задачи.
5. Для конкретных входных данных построить порождающий процесс.

27. Прикладное исчисление предикатов первого порядка как пример исчисления СО ВХОДОМ

27.1. Форма определения прикладного исчисления предикатов

первого порядка

Основными конструкциями языка исчисления предикатов первого порядка являются термы, формулы и теории.

**Термы** могут быть следующих видов:

- предметный символ;

- переменная;

- *f(t1,...,tn)* (функциональный терм), где *f* - *n*-местный функциональный символ, а *t1,...,tn* - термы (*n ≥ 1*).

**Формулы** могут быть следующих видов:

- *p(t1,...,tm)* (атомарная формула), где *p* - *m*-местный предикатный символ, а *t1,...,tm*- термы (*m ≥ 1*);

- *¬ϕ,ϕ∧ψ,ϕ&ψ, ϕ⇒ψ*, где *ϕ* и *ψ* - формулы;

- *(∀v)ϕ*, *(∃v)ϕ* (квантифицированная формула), где *v* - переменная, а *ϕ* - формула.

Если в квантифицированной формуле формула *ϕ* не содержит квантифицированных формул, то любое вхождение переменной *v* в формулу *ϕ* называется **связанным**, а сама формула *ϕ* - **областью действия квантора**. Если формула *ϕ* содержит квантифицированные формулы с переменной *v*, то **областью действия квантора** является та часть формулы *ϕ*, которая остается после удаления из *ϕ* всех таких квантифицированных формул. Все вхождения переменной *v* в область действия квантора называются **связанными**. Вхождение переменной в формулу называется **свободным**, если оно не является связанным. Формула, не содержащая свободных вхождений переменных, называется **предложением** языка исчисления предикатов первого порядка.

Прикладное исчисление предикатов первого порядка есть исчисление со входом. Входами этого исчисления являются конечные множества предложений языка исчисления предикатов первого порядка (нелогических аксиом). Состояниями порождающих процессов этого исчисления являются конечные множества формул специального вида языка исчисления предикатов первого порядка. Эти формулы имеют вид: *◊p1 ∨ ◊p2 ∨...∨ ◊pn*, где *◊* - знак отрицания или его отсутствия, а *pi* - атомная формула для всех *i = 1, ..., n*.

Само исчисление будет определено в такой форме, в которой будут отсутствовать *0*-посылочные правила (логические аксиомы), а множество правил исчисления будет состоять только из одного правила - "принципа резолюции". Входная процедура преобразует вход в состояние.

27.2. Входная процедура

Входная процедура определяется как последовательность эквивалентных преобразований множества предложений в множество формул вида, определённого в предыдущем разделе:

1. исключение импликации (замена *p ⇒ q* на *¬p ∨ q*);
2. раскрытие отрицаний перед скобками и кванторами таким образом, чтобы знак отрицания стоял только перед атомными формулами;
3. стандартизация переменных, т.е. переобозначение их таким образом, чтобы переменные, связанные разными кванторами, имели разные обозначения;
4. преобразование каждого предложения в пренексную форму, т.е. такую форму, где все кванторы предложения собраны в начале предложения в том порядке, в котором они входят в исходное предложение (эта часть пренексной формы называется префиксом), а затем следует бескванторная формула, называемая матрицей;
5. исключение кванторов существования путём замены их функциями Сколема;
6. исключение кванторов всеобщности, т.е. отбрасывание всех префиксов (поскольку других кванторов к этому шагу уже нет, все переменные связаны только кванторами всеобщности, а порядок кванторов всеобщности в префиксе не имеет значения);
7. преобразование каждой формулы в конъюнктивную нормальную форму;
8. исключение конъюнкции, т.е. замена каждой формулы множеством входящих в неё конъюнктов.

В результате выполнения этой последовательности преобразований исходное множество предложений (вход) заменяется эквивалентным множеством формул, каждая из которых имеет форму, указанную в предыдущем разделе (т.е. первым состоянием порождающего процесса).

27.3. Принцип резолюции

**Подстановкой** называется последовательность *λ = <(v1/t1), ... , (vm/tm)>*, где *v1,..., vm* - переменные, а *t1,..., tm* - термы. **Применение подстановки** *λ* к формуле *f* состоит в последовательной замене всех вхождений переменной *v1* в формулу *f* термом *t1*, ..., всех вхождений переменной *vm* в формулу *f* термом *tm*. Результат применения подстановки *λ* к формуле *f* обозначается *[f ⎢λ]*.

**Унификатором** формул *f1* и *f2* называется такая подстановка *λ*, что *[f1⎜λ] ⇔ [f2⎜λ]*.

**Принцип резолюции** есть правило вывода *f1 ∨ L1, ¬f2 ∨ L2, [f1⎜λ] ⇔ [f2 ⎜λ] ⊥ [L1⎜λ] ∨ [L2⎜λ]*, где *f1, f2* - атомные формулы, *L1, L2* - формулы вида, определённого в п. 24.1, *λ* - унификатор *f1* и *f2*. Принцип резолюции означает, что если на очередном шаге порождающего процесса в состояние входят формулы *f1 ∨ L1* и *¬f2 ∨ L2* такие, что для *f1* и *f2* существует унификатор *λ*, то состояние следующего шага порождающего процесса получается из состояния текущего шага добавлением к нему формулы *[L1⎜λ] ∨ [L2⎜λ]*.

27.4. Автоматическое доказательство теорем

Формула языка исчисления предикатов первого порядка называется **выводимой** при данном входе, если для этого входа существует порождающий процесс такой, что эта формула принадлежит состоянию этого процесса на некотором шаге. Таким образом, установление выводимости формулы есть решение обратной задачи поиска вывода в исчислении предикатов первого порядка. Исчисление предикатов первого порядка обладает свойством полуразрешимости, т.е. существует алгоритм, позволяющий для любого входа и любой формулы, выводимой при этом входе, определить факт её выводимости, но не существует алгоритма, позволяющего для любого входа и любой формулы установить факт её выводимости или невыводимости для этого входа.

Принцип резолюции позволяет установить противоречивость множества предложений языка исчисления предикатов: это множество является противоречивым, если для него выводима "пустая" (тождественно ложная) формула.

Принцип резолюции позволяет доказывать теоремы, т.е. определять выводимость некоторого предложения (теоремы) из множества других предложений (аксиом), методом от противного. Теорема со знаком отрицания добавляется к множеству нелогических аксиом. Определяется противоречивость полученного множества предложений. Если это множество оказывается противоречивым, теорема доказана.

В качестве примера рассмотрим доказательство теоремы о том, что треугольник не может иметь двух тупых углов. Запишем формулировку этой теоремы на языке исчисления предикатов первого порядка, сигнатура которого состоит из трехместного предиката "*углы*" и знаков арифметических операций и отношений.

*(∀α) (∀β) (∀γ) углы(α, β, γ) ⇒ ¬ (α > π / 2 & β > π / 2)*

Нелогические аксиомы.

Углы в треугольнике считаются положительными.

*(∀α) (∀β) (∀γ) углы(α, β, γ) ⇒ γ > 0*

Теорема о сумме углов треугольника.

*(∀α) (∀β) (∀γ) углы(α, β, γ) ⇒ α + β + γ = π*

Свойства неравенств.

*(∀v1) (∀v2) (∀v3) (∀v4) v1 > v3 & v2 > v4 ⇒ v1 + v2 > v3 + v4*

*(∀v1) (∀v2) v1 > v2 / 2 + v2 / 2 ⇒ v1 > v2*

*(∀v1) (∀v2) (∀v3) v1 + v2 = v3 & v2 > 0 ⇒ ¬ (v1 > v3)*

Исключение знака импликации из формулировки теоремы дает следующее предложение

*(∀α) (∀β) (∀γ)¬ углы(α, β, γ) ∨ ¬ (α > π / 2 & β > π / 2)*

В результате преобразований отрицания теоремы получаем три конъюнкта

1. *углы(α\*, β\*, γ\*)*, где *α\*, β\* и γ\** - константы Сколема,
2. *α\* > π / 2*
3. *β\* > π / 2*

Преобразование аксиом дает следующие конъюнкты

1. *¬ углы(α, β, γ) ∨ γ > 0*
2. *¬ углы(α, β, γ) ∨ α + β + γ = π*
3. *¬ (v1 > v3) ∨ ¬ (v2 > v4) ∨ v1 + v2 > v3 + v4*
4. *¬ (v1 > v2 / 2 + v2 / 2) ∨ v1 > v2*
5. *¬ (v1 + v2 = v3) ∨ ¬ (v2 > 0) ∨ ¬ (v1 > v3)*

Вывод пустого дизъюнкта

1. *γ\* > 0*, (1 и 4) *λ = (α|α\*, β|β\*, γ|γ\*)*,
2. *α\* + β\* + γ\* = π*, (1 и 5) *λ = (α|α\*, β|β\*, γ|γ\*)*,
3. *¬ (v2 > v4) ∨ α\* + v2 > π / 2 + v4*, (2 и 6) *λ = (v1|α\*, v2|v2, v3|π / 2, v4|v4)*,
4. *α\* + β\* > π / 2 + π / 2*, (3 и 11) *λ = (v2|β\*, v4|π / 2*),
5. *α\* + β\* > π*, (12 и 7) *λ = (v1|α\* + β\*, v2|π)*,
6. *¬ (v1 + γ\* = v3) ∨ ¬ (v1 > v3)*, (9 и 8) *λ = (v1|v1, v2|γ\*, v3|v3)*,
7. *¬ (α\* + β\* > π)*, (14 и 10) *λ = (v1|α\* + β\*, v3|π)*,
8. , (13 и 15), *λ = ()*.

Задание № 15 (по теме "Доказательство теорем")

Придумать пример содержательной теоремы, множества нелогических аксиом и доказать эту теорему с использованием принципа резолюции на основе этих аксиом.

План ответа

1. Формулировка теоремы (содержательная и на языке исчисления предикатов).
2. Множество формулировок нелогических аксиом (содержательных и на языке исчисления предикатов).
3. Вход и его преобразования к первому состоянию порождающего процесса.
4. Порождающий процесс и вывод.

28. Взаимосвязь между алгоритмами и исчислениями

28.1. Основной вопрос

Алгоритм и исчисление являются двумя способами описания множества процессов (напомним, что вычислительные и порождающие процессы ничем не отличаются друг от друга, кроме способа их построения). Возникает вопрос, как соотносятся между собой эти понятия, более точно: существует ли алгоритм, который реализует такую функцию, которая не может быть реализована никаким исчислением со входом? Обратно, существует ли исчисление со входом, которое реализует такую функцию, которая не может быть реализована никаким алгоритмом?

28.2. Алгоритмы - частный случай исчислений

Справедливо утверждение: для любого алгоритма существует исчисление со входом, описывающее множество процессов, которое описывается этим алгоритмом. Покажем это.

В разд. 14 были введены граф-программы, которые являются представительной вычислительной моделью В силу тезиса Чёрча это означает, что любой алгоритм может быть представлен некоторой граф-программой. Граф-программе однозначно соответствует спецификация этой программы. По форме предложения, образующие спецификацию программы, являются правилами некоторого исчисления, заданного в порождающей модели систем реляционных конфлюентных продукций (ср. разд.14 и разд. 22). Легко видеть, что между множеством процессов, описываемых граф-программой, и множеством процессов, описываемых соответствующим ей исчислением, существует взаимно-однозначное соответствие. Отсюда следует, что для любого алгоритма, реализующего некоторую функцию, существует исчисление со входом, реализующее ту же функцию.

Обратное неверно, а именно: не любое правило исчисления, заданное в порождающей модели систем реляционных конфлюентных продукций, имеет форму предложения, которое может входить в спецификацию программы (ср. также разд.14 и разд. 22).

28.3. Реализация исчислений с помощью алгоритмов

Покажем, что для любого исчисления со входом, реализующего некоторую функцию, существует алгоритм, который реализует ту же функцию.

Возьмём представительную порождающую модель, все исчисления которой являются конфлюентными (например, системы реляционных конфлюентных продукции). Построим алгоритм, который реализует рецепт этой порождающей модели, т.е. по входу и формальному заданию исчисления в этой модели строит некоторый порождающий процесс этого исчисления. При выборе применимого правила этот алгоритм выбирает первое правило в списке применимых правил, при выборе значений посылок этого правила этот алгоритм выбирает первый в списке набор значений его посылок. В силу конфлюентности исчисления этот алгоритм реализует ту же функцию, что и это исчисление.

ЛИТЕРАТУРА

1. Бауэр Ф.Л., Гооз Г. Информатика. Вводный курс: В 2-х частях. М:Мир, 1990.
2. Мальцев А.И. Алгебраические системы. М.: Наука, 1970. 302 с.
3. Успенский В.А., Семёнов А.Л. Теория алгоритмов: основные открытия и приложения. М.: Наука, 1987. 288 с.
4. Хоар К. О структурной организации данных // Структурное программирование. М.: Мир, 1975. С. 98-197.
5. Чень Ч., Ли Р.. Математическая логика и автоматическое доказательство теорем. М.: Наука, 1983. 359 с.
6. Клещев А.С., Артемьева И.Л. Необогащенные системы логических соотношений. Ч.1. // Научно-техническая информация, сер.2.-2000.-№ 7.-С.18-28.