Введение в формальные системы

Аксиоматический метод – это способ построения научной теории, при котором в основу теории кладутся некоторые исходные положения называемые аксиомами, а все остальные положения теории (вспомогательные – леммы и ключевые теоремы) получаются как логические следствия из аксиом.

Внутренняя логическая связность некоторой математической теории и её адекватность для описания того или иного круга физических явлений – это две совершенно разные вещи. Данный факт был осознан математиками, когда им пришлось объяснять существование евклидовой геометрии и различных неевклидовых геометрий (например, геометрии Лобачевского-Бойаи).

Аксиомой называется исходное положение научной теории, принимаемое без доказательства.

Формальная система описывается с помощью символического (формального) языка, для которого точно определяется синтаксис (посредством «правил образования») и логика (посредством «правил вывода» или «правил преобразования»). Обсуждение свойств формальной системы производится в некоторой другой теории (средствами другого языка), которую назовём метатеорией (а этот другой язык – метаязыком). Саму же формальную систему называют предметной (или объектной) теорией (и соответственно употребляют термины «предметный язык» или «язык-объект»).

Изучение свойств формальной системы, проводимое содержательными математическими методами в рамках метаязыка, будем называть математикой, или теорией доказательств.

Пример 1.

Пусть алфавит формальной системы есть A={0, 1, 2, 3, 4, 5, 6, 7, 8, 9, +, –}.

Мы хотим определить два множества, а именно множество целых чисел без знака и множество целых чисел. Чтобы сформулировать правила образования этих множеств, мы введём метаязыковую переменную C, которая принимает любое значение из множества {0, 1, 2, 3, 4, 5, 6, 7, 8, 9}, метаязыковую переменную , принимающую значение из множества целых чисел без знака, и метаязыковую переменную E, принимающую значение из множества целых чисел.

Скобки «⟨» и «⟩» будут использоваться для разграничения вхождения переменных; вертикальная черта «|» обозначает союз «или»; символ «::=» означает разрешение заменить то, что стоит слева от него, каким-либо из выражений, стоящих справа от него и разделённых вертикальными чертами. Для обозначения конкатенации (соединения двух цепочек) используется «классический» типографский прием – простое соположение (без специального знака).

Правила порождения интересующих нас множеств:

Правило 1: $\langle W\rangle ::= \langle C\rangle \mid \langle W\rangle \langle C\rangle$;

Правило 2: $\langle E\rangle ::= \langle W\rangle \mid +\langle W\rangle \mid -\langle W\rangle$.

Интерпретация формальной системы и теории

Поскольку формальная система получается (как правило) в результате формализации некоторых разделов обычной неформальной или полуформальной математики, то символы, формулы и прочие элементы такой формальной системы истолковываются (интерпретируются) в терминах соответствующей неформальной или полуформальной математики.

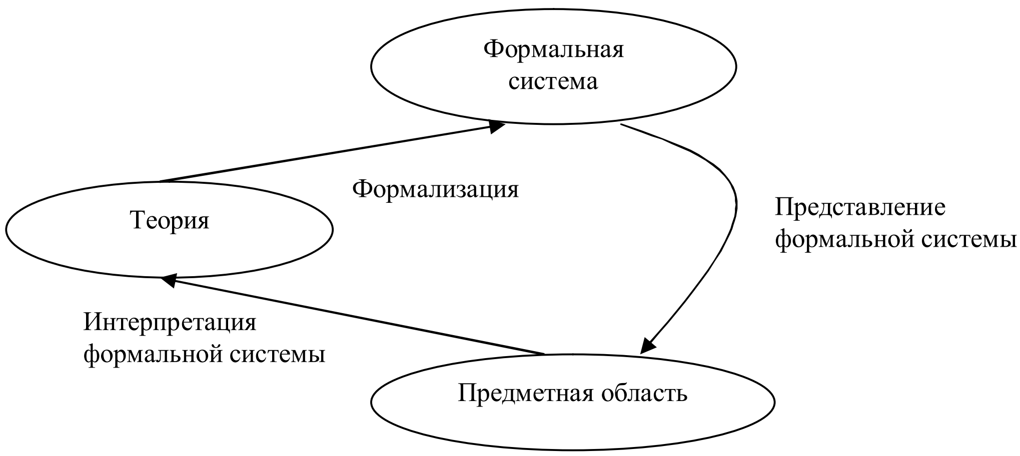
Интерпретацией теории называется установление соответствия между элементарными высказываниями формальной теории и содержательными высказываниями некоторой предметной области.

Другими словами, интерпретация – это совокупность значений, приписанных таким образом символам, формулам и прочим элементам формальной системы. Если мы не знаем этой интерпретации, данная формальная система не представляет для нас интереса.

В математики, в соответствии с её задачами, формальная система должна изучаться как таковая, т.е. просто как система символов, лишённых всякого смысла, а её интерпретация не должна приниматься во внимание. Когда говорят об интерпретации, то речь идёт не о математике.

Обычно формальная система создаётся как идеализированный образ чего-то другого (это «другое» можно было бы назвать, например, «теорией»), по отношению к чему эта формальная система и приобретает смысл. Тогда формальная система представляет собой формализацию некоторой теории, образуя синтаксис последней; теория же представляет собой интерпретацию данной формальной системы. Правильно построенные слова (правильные фразы) формальной системы получают смысл в рамках формализуемой теории; обратно, любому высказыванию, имеющему смысл в данной теории, соответствует некоторый класс правильно построенных слов (правильных фраз) в формальной системе.

Обратное движение от формальной системы к некоторым объектам опыта приводит к понятию представления системы.



Интерпретация и представление формальной системы.

Представлением системы называется любой способ рассмотрения объектов формальных систем как конкретных объектов (полученных из опыта) при условии, что содержательные объекты сохраняют структуру формальных.

Всякая формальная система произвольного типа имеет синтаксическое представление.

Структура языка и выражения. Функторы

Объекты при описании их средствами математической логики должны иметь имена. За определением имён следует описание соотношений между объектами и выражение свойств этих отношений. Построение решения задачи производится на основе логического вывода, манипуляцией предложениями, описывающими задачу. В математической логике константы, переменные и функции объединены общим названием – терм. Терм – это переменные, константы и функции вида $f(t_1,\dots, t_n)$, где каждое t_i – терм, а f – n-арный функциональный символ или функтор (арность – это число аргументов).

Особенность функции состоит в том что она принимает значение элемента предметной области D_n или, иными словами, представляет собой некоторое отображение совокупности из n элементов из предметной области в элемент предметной области.

Отношения определяемые над объектами отличаются от функций. Отношение определяет совокупность элементов из предметной области и представляет собой отображение из D_n в множество {ИСТИНА, ЛОЖЬ}. Например отношение ***мама(x, y)*** определят совокупность пар ***(x, y)*** таких, что элементы множества людей x и y находятся в отношении родства мама. В математической логике отношениям даются имена, называемые предикатными символами, а сами отношения называются предикатами.

В алгебре логики есть набор связок с помощью которых можно объединять предикаты и другие формулы (“и, “или”, “не”, “если…то…”) и кванторы общности ($\forall$) и существования ($\exists$). Кванторы определяют пределы изменения переменных. Формула, стоящая за квантором называется областью действия квантора.

Формула – это либо предикат, либо выражение, составленное из формул с помощью логических связок и кванторов.

Предложение – это формула , в которой каждая переменная находится в области действия квантора общности.

Предложения, построенные в соответствии с введёнными выше правилами, образуют язык логики первого порядка. В этом языке терма представляют собой объекты, а предикаты – отношения между ними. С помощью этого языка можно описать все задачи, решаемые на ЭВМ. На основе языка логики первого порядка можно построить различные языки логического программирования, различающиеся по правилам формирования предложения.

Грамматикой называются правила, определяющие предложения языка.

Фразами называются комбинации символов, образующие грамматические единицы.

Функтором называется средство соединения фраз для образования других фраз.

Существуют три основных класса фраз: имена, предложения, функторы. Имена и предложения называют замкнутыми фразами, чтобы отличить их от функторов. Фразы, соединяемые функтором, называются его аргументами, а результат соединения – его значением.

Конструкционная последовательность – это последовательность, в которой каждый её член либо является начальным элементом, либо строится из некоторых впереди стоящих элементов посредством некоторой операции.

Порождающая грамматика – это формальная грамматика, позволяющая построить любую правильную цепочку символов.

Алгоритмы как формальные системы

Алгоритмы обладают двумя характерными особенностями:

- Наличие чёткого описания. Алгоритм всегда описывается настолько чётко, что на основе описания можно действовать механически не вникая в смысл.

- Детерминированность. Это значит, что если процедура является алгоритмом, то каждый последующий шаг точно определяется исходными данными и результатами уже выполненных шагов.

По Хартли Роджерсу характеристика понятия «алгоритма» должна включать в себя следующие аспекты:

1. конечность,
2. наличие исполнителя,
3. результативность,
4. дискретность,
5. детерминированность.

Неформально и схематически алгоритм представляется в виде чёрного ящика. Алгоритм есть процедура (или способ вычисления), осуществляемая чёрным ящиком для получения выхода из входа.

Это интуитивное, а не строгое математическое определение алгоритма или «вычислительной процедуры». Раньше считалось, что все задачи «вычислительного» характера имеют алгоритм решения, но со временем возникли некоторые задачи несомненно «вычислительного» характера, все попытки нахождения алгоритмов для которых заводили в тупик, и появилась гипотеза, что по крайней мере для части этих задач вообще никаких алгоритмов не существует (решением этих проблем занимались такие учёные как К.Гёдель, Э.Пост, А.Тьюринг и на основе их разработок доказано, что для многих «вычислительных» задач алгоритмы невозможны).

Теория алгоритмов возникла в математической логике как вспомогательная дисциплина, предназначенная для исследования проблемы неразрешимости и обоснования математики.

Формализация и обобщение понятия алгоритма

В настоящее время существует много вариантов определения понятия алгоритма. Обычно они трактуются как самостоятельные математические понятия, наибольшее распространение получили две из этих конструкций – машина А.Тьюринга и нормальный алгоритм А.А.Маркова.

Основные типы алгоритмических моделей различаются исходными трактовками, что такое алгоритм.

Первый тип трактует алгоритм как некоторое детерминированное устройство, способное выполнять в каждый момент лишь строго фиксированное множество операций. Основной теоретической моделью такого типа является машина Тьюринга, предложенная им в 30-х годах и оказавшая существенное влияние на понимание логической природы разрабатываемых ЭВМ. Другой теоретической моделью данного типа является машина произвольного доступа (МПД) – введенная достаточно недавно (в 70-х годах) с целью моделирования реальных вычислительных машин и получения оценок сложности вычислений.

Второй тип связывает понятие алгоритма с традиционным представлением – процедурами вычисления значений числовых функций. Основной теоретической моделью этого типа являются рекурсивные функции – исторически первая формализация понятия алгоритма.

Третий тип алгоритмических моделей – это преобразования слов в произвольных алфавитах, в которых операциями являются замены кусков слов другим словом. Основной теоретической моделью этого типа являются нормальные алгоритмы Маркова.

Теория алгоритмов оказала существенное влияние на развитие ЭВМ и практику программирования. В теории алгоритмов были предугаданы основные концепции, заложенные в аппаратуру и языки программирования ЭВМ. Упоминаемые выше главные алгоритмические модели математически эквивалентны; но на практике они существенно различаются сложностными эффектами, возникающими при реализации алгоритмов, и породили разные направления в программировании. Так, микропрограммирование строится на идеях машин Тьюринга, структурное программирование заимствовало свои конструкции из теории рекурсивных функций, языки символьной обработки информации (РЕФАЛ, ПРОЛОГ) берут начало от нормальных алгоритмов Маркова и систем Поста.

Для того, чтобы построить строгое формальное определение алгоритма необходимо формально определить:

- объекты, над которыми выполняются действия, указанные в описании алгоритма;
- понятие действия;
- порядок выполнения действий.

Начнём с формализации понятия объекта.

В алгоритме Евклида и других вычислительных алгоритмах объектами являются числа. В алгоритме игры в шашки объекты - это позиции на доске и отдельные шашки (анализируя текущую позицию, алгоритм вырабатывает наилучший ход). В алгоритме управления летательными аппаратами объекты - это сигналы датчиков и управляющие сигналы (алгоритм анализирует сигналы датчиков и вырабатывает необходимое управляющее воздействие). Таким образом, объекты, с которыми оперируют алгоритмы весьма разнообразны.

При формализации понятия объекта мы будем считать, что алгоритм оперирует не самими объектами, а их изображениями.

Например, числа являются абстрактными объектами и доступны только в виде изображений. 256, 458 - это изображения натуральных чисел в десятичной системе счисления.

Алгоритм сложения двух чисел: последовательность символов "256 + 458" переводит в последовательность символов "714".

Алгоритм игры в шашки: последовательность символов "Белые: a1, c1, d4", "Чёрные: a7, b6, b8", ход может быть записан "a1-b2".

Таким образом, изображением можно считать последовательность символов (слов) в некотором алфавите.

Алфавит - это набор допустимых символов, т.е. любое непустое конечное множество символов. Элемент этого множества будем называть **буквой**. Конечное множество букв называется **цепочкой** или **словом**.

Таким образом, объекты, к которым применяются алгоритмы, можно формально описывать как слова в некотором алфавите.

Пусть A – алфавит $A=\{a_1, a_2, \ldots, a_n\}$, состоящий из букв a_i , i=1,n, тогда $A^* = \cup A_i = A^0 \cup A^1 \cup \ldots \cup A^n$ - множество слов над алфавитом A, где

$A^0 = \{\varepsilon\}$ пустая цепочка

$A^1 = A$ – цепочки, состоящие из одного элемента

$A^2 = A*A$ – слова, состоящие из двух букв и т.д.

Если $\varphi \in A^*$, то φ называется цепочкой или словом над алфавитом A.

Пусть β - несобственная буква, т.е. $\beta \notin A$. Тогда расширением алфавита A называется алфавит $\bar{A} = A \cup \{\beta\}$.

Если $\psi \in \bar{A}^*$, то ψ называется несобственным словом над алфавитом Ā.

Например:

Алфавит {0, 1}

$A^*=\{\varepsilon, 0, 1, 00, 01, 10, 11\}$ – множество всех слов над алфавитом A.

$\bar{A}=\{\beta, 0, 1\}$– расширение алфавита A.

$\bar{A}^*=\{\beta, \varepsilon, 0, 1, \beta 0, \beta 1, 00, 01, \beta\beta, \ldots, \beta 11\}$– множество всех символов над алфавитом Ā.

Алгоритм (алгорифм) – способ (программа) решения вычислительных и других задач, точно предписывающий, как и в какой последовательности получить результат, однозначно определяемый исходными данными.

Алгоритм задаётся как предложение формального языка.

Алгоритмический язык – это формальный язык для однозначной записи алгоритмов.

Алгоритм считают **правильным (correct)**, если на любом допустимом (для данной задачи) входе он заканчивает работу и выдаёт результат, удовлетворяющий требованиям задачи. В этом случае говорят, что алгоритм **решает (solves)** данную задачу. Неправильный алгоритм может (для некорректного входа) вовсе не остановиться или дать неправильный результат.

Вычислимая функция – функция f(x), для которой существует алгоритм оценки для любого элемента x в области определения f(x).

Последовательность преобразований для заданного элемента, позволяющая достичь заданной цели преобразования за конечное число шагов, называется **эффективным процессом**.

Описание эффективного процесса – это описание, содержащее следующие условия: допустимость заданного элемента для данного процесса, описание преобразования и его результата, условие достижения цели.

Эффективная процедура – алгоритм, удовлетворяющий следующим требованиям:

1. он должен состоять из конечного множества простых (элементарных) команд, порядок исполнения которых должен быть однозначно задан;
2. при задании на вход алгоритма набора значений x из области определения f(x) вычисления должны закончиться после конечного числа шагов и выдать соответствующее значение f(x) , а если набор значений x не принадлежит области определения f(x), то вычисления не приводят ни к какому результату.