Введение в математическую логику и теорию алгоритмов

Лекция 12

Алексей Львович Семенов

План

- Ансамбли (напоминание)
- Действия, проверки
- Алгоритмы
- Вычислимые функции
- Универсальный алгоритм, универсальная функция
- Диагональ (напоминание)
- Диагональ невычислимости
- Исчисления (напоминание)
- Связь между породимостью и вычислимостью
- Разрешимость
- Непородимость
- Грамматики (напоминание)
- Алгоритмы Маркова
- Тезис об исчислениях (напоминание)
- Тезис Чёрча

Ансамбли (напоминание)

- Цепочка = конечная последовательность, которая может быть и пустой – Λ. Длина цепочки – число элементов в ней.
- Алфавит = цепочка различных символов
- Слово (в данном алфавите) – цепочка символов этого алфавита.
- Ансамбль слов в данном алфавите – все слова. Часто алфавит: 0 1
- Ансамбль цепочек слов в данном алфавите – все цепочки слов.
- Ансамбль списков в данном алфавите – все слова и все цепочки списков (индуктивное определение):
  - 011; (01,0); (0, ((01,0), 0))… - для задания цепочки используются запятые и скобки.

Ансамбль двоичных слов

- Двоичные слова – слова в алфавите 0, 1
- Линейный порядок:

Λ, 0, 1, 00, 01, 10, 11, 000, 001,…

Следующе слово может быть получено действием пересчета (счисления)

Слова получают номера: 1, 2, 3… (их тоже можно писать как двоичные числа)

- Кодирование слов в произвольном алфавите двоичными словами
- Цепочки слов и списки могут кодироваться словами (в произвольном алфавите)
- Кодирование – действие

Действия и проверки. Описания

Действие – исходное понятие.

Действие: Слово на каком-то языке,

- человек понимает, что значит применить это слово к любому исходному данному из фиксированного ансамбля, при этом ясно, что всегда получается результат (применения) – элемент (возможно, другого) фиксированного ансамбля;
- применить слово может и подходящее устройство.

- Действие задает всюду (на данном ансамбле) определенную функцию
- Проверка – действие с результатом 0 или 1

Проверка задает характеристическую функцию множества (где она дает 1), она допускает его элементы (а другие – не допускает)

Алгоритмы

Алгоритм – цепочка вида

А

Пока Б

В

Г

где А, В, Г – действия, Б – проверка.

- А - начало
- Б - продолжение
- В - переработка
- Г - извлечение результата.
- Пишем элементы цепочки по вертикали для наглядности.

Применение алгоритма

к исходному данному x – это (конечная или бесконечная) последовательность: x_0, x_1, …, x_n, …, получаемая так:

- x_0 – это x
- x_1 – это результат применения **Начала** А к x_0
- при i>0 к уже полученному x_i нужно применить **Продолжение** Б.

Затем:

  - если проверка Б дает 1, то к x_i применить **Переработку** В; то, что получится, – это x_{i+1} .
  - если проверка Б дает 0, то к x_i применить **Извлечение результат**а Г, результат этого применения является последним в последовательности.

- Результат применения алгоритма к x – последний элемент последовательности.

Действия (в том числе – проверки) всегда дают результат

Задача. Когда алгоритм не дает результат?

Вычислимые функции

- Алгоритм задает (вычисляет) функцию, возможно, не всюду определенную
- Функция, вычисляемая каким-то алгоритмом, называется вычислимой. Говорят также, что алгоритм является ее описанием.

Одно из основных понятий данного курса и математики

Универсальный алгоритм и универсальная функция

- Фиксирован язык (для записи) алгоритмов.
- Можно считать, что алфавит = **B** = {0, 1}. Алгоритм использует символ ∞, алгоритмы можно записывать в **B**
- Универсальный алгоритм:

  Начало: выясняет, является ли исходное данное кодом пары, где первый элемент – это код алгоритма,

  если да, выделяем в первом элементе – алгоритме, назовем его P, его составные части и применяем **Начало** P

  если нет, то результат ∞.

  Продолжение: Если на входе ∞, то 1, иначе **Продолжение** P

  Переработка: Если на входе ∞, то ∞, иначе **Переработка** P

  Извлечение результата : **Извлечение результата** P

Универсальная функция: $У(\langle \textbf{код}\ P, x \rangle) = P(x)$.

Существуют ли невычислимые функции?

- Сколько есть вычислимых функций?
- Не больше, чем алгоритмов.
- Не больше, чем слов.
- Счетное количество
- Всех функций – несчетное.

Диагонали (повторение)

В квадратной таблице выписано (по горизонтали) конечное число цепочек.

Как построить цепочку, которой нет в таблице?

1	2	8	2	3
0	3	7	3	5
7	6	0	3	9
8	2	7	5	8
3	0	6	5	0

Диагональ (конечная)

№чл.посл. / № посл.	1	2	3	4	5
1	1	2	8	2	3
2	0	3	7	3	5
3	7	6	0	3	9
4	8	2	7	5	8
5	3	0	6	5	0

Взять диагональ и испортить её в каждом члене.

Например, прибавляем 1 и получаем 24161.

t(i,j) – это элемент, стоящий в i–ой строке и j–ом столбце.

Цепочка a(i) = t(i,i) +1.

Пусть она в строке c. Тогда t(c,c) = t(c,c) +1. Противоречие

Диагональ несчетности (Кантора)

Аргумент № характ. функции	1	2	3	4	5	…
1	1	0	0	1	0	
2	1	1	0	1	0	
3	1	0	0	1	1	
4	0	0	0	1	0	
5	0	1	0	1	0	
…						

Функция, которой нет в таблице – это (1 – t(i,i)), **не** t(i,i), нули заменили на единицы, единицы – на нули.

Таблица вычислимых функций

Алфавит описаний функций (кодов алгоритмов) – **B**. Имена строк и столбцов – (непустые) слова.

В клетке (i,j) – значение функции с описанием i на аргументе j. Если значения нет (например, если i – не описание), оставим клетку пустой (здесь 1 оказалась кодом какой-то функции):

Аргумент Функция	Λ	0	1	…	1111
Λ				…	
0				…	
1	11	1	0111	…	01
……	…	…	…	…	…
1111	0	11	111	…	(диаг) 10

Диагональ невычислимости

Аргумент Функция	Λ	0	1	…	1111
Λ				…	
0				…	
1	11	1	0111	…	01
……	…	…	…	…	…
1111	0	11	111	…	10

- «Диагональная» функция t(i,i) вычислима.
- Вычислима и «испорченная диагональ» d(i), которая равна нулю, если t(i,i) > 0, и равна 1, если t(i,i) = 0.
- **Задача**. Есть ли d в таблице?

Диагональ невычислимости

- **Задача**. Может ли функция d(i,i) быть всюду определенной?
- **Задача**. Можно ли продолжить функцию d(i,i) до всюду определенной?
- **Задача**. Можно ли продолжить функцию d(i,i) до вычислимой всюду определенной?

Исчисления (напоминание)

Исчисление в данном ансамбле – это пара из двух проверок:

•<проверка возможности (создания), проверка завершения>.

•проверка возможности применяется к цепочке объектов, проверка окончания – к объекту.

•создаваемый исчислением объект определяется так:

Если проверка возможности допускает цепочку объектов a_1,… a_n и все элементы этой цепочки, кроме последнего – создаваемы,

то и последний элемент создаваем.

•Если проверка возможности допускает цепочку из одного элемента, то его называют начальным объектом.

Исчисления. Породимые множества

- Фиксирован какой-то ансамбль
- Объект порождаем данным исчислением, если он создаваем и его допускает проверка окончания.
- Исчисление порождает множество из всех порождаемых им объектов – множество, порождаемое этим исчислением.
- Порождение
  - «недетерминированный процесс»
  - возможность чего-то: если цепочка допускается проверкой возможности, то ее последний элемент *возможно* создать «из предыдущих» (если предыдущие созданы)

Выводы

- Фиксируем исчисление.
- Если $a_1, \ldots, a_n$ – допускается проверкой возможности, то говорим. что a_n создается из $a_1, \ldots, a_{n-1}$ или выводится из них (в данном исчислении).
- Вывод объекта a – цепочка объектов S, каждый из которых создается из какой-то цепочки объектов, встретившихся в S раньше него.
- **Задача**. Объект создаваем тогда и только тогда, когда имеется его вывод.
- **Задача**. Пусть дано исчисление. Как организовать процесс выписывания всех выводов?
- **Задача**. Пусть дано исчисление. Как организовать процесс выписывания всех порождаемых (в нем) объектов (и только их)?

Порождение

- Объект порождаем, если существует его вывод и для него выполнена проверка окончания
- Можно перейти к кодам
- Существует алгоритм, который по любому объекту проверяет, является ли он кодом вывода, и:
  - если не является, то не дает результата
  - если является, то дает порождаемое выводом слово

  Если порождаемое множество непусто, то можно для не-кодов давать всегда один и тот же фиксированный порождаемый объект.

Породимые множества

Породимое множество – множество, порождаемое каким-то исчислением.

Теоремы замкнутости для исчислений

Т. Объединение и пересечение породимых множеств породимы.

Перечислимые множества

О. Перечислимое множество – это множеством значений вычислимой функции.

Задача. Следующие свойства множества эквивалентны:

1. Оно – перечислимо
2. Оно – область определения вычислимой функции
3. Оно – породимо

Можно указать общие способы (алгоритмы) построения по любому из описаний 1 – 3 любого другого.

- **Замечание**. Есть действие S (счисление), для которого: множество {Λ, S(Λ), S(S(Λ)),…} содержит все элементы ансамбля.
- **Задача**. Перечислимое множество пусто или является множеством значений всюду определенной вычислимой функции.
- **Задача**. Нет алгоритма, который по описанию множества в каком-то варианте из 1 – 3 строит описание такой всюду определенной функции.
- **Итог**. Мы можем определять породимость через вычислимость

Определение вычислимости через породимость

Задача. Функция вычислима $\leftrightarrow$ ее график породим (перечислим)

- Вычислимость можно определить через породимость.

Разрешимые множества

О. Множество разрешимо, если его характеристическая функция вычислима

Задача. Множество разрешимо $\leftrightarrow$ оно и его дополнение перечислимы

Задача. Бывает ли перечислимое множество с не перечислимым дополнением?

Вариант ответа: номера функций, у которых на диагонали – не пустая клетка?

Грамматики (напоминание)(Хомский)

Определение.

Грамматика Г – это цепочка <Σ,Ω,П,S>

- Σ – основной алфавит Г
- Ω – вспомогательный алфавит Г
- S – начальный символ Г
- Σ∩Ω=Ø, объединение Σ и Ω – это алфавит Г, обозначим его Δ.
- П – это конечное множество пар слов в алфавите Δ. Эти пары называются заменами.

Грамматика

определяет исчисление Г*

Проверка возможности Г* допускает:

- S
  - Всякий вывод в исчислении начинается с S.
- Для каждой замены <u,v> из П, все пары вида <tup,tvp>, где t,p – произвольные слова в алфавите Δ
  - Один шаг вывода состоит в замене в слове некоторого входящего в него u на v.

Проверка окончания для грамматики Г допускает

- все слова в алфавите Σ.
  - Порождаемые слова не могут содержать букв из вспомогательного алфавита.

Код грамматики - слово в конечном алфавите (можно считать 0 1).

Алгоритмы Маркова

- **Определение.** Описание алгоритма Маркова – это цепочка $\Phi = \langle \Sigma, \Delta, \Pi \rangle$, где
- Δ – алфавит грамматики
- Σ – *основной алфавит* Φ, у нас $\{0,1\}$, $\Sigma \subseteq \Delta$,
- Π – цепочка слов вида $u \to v$ или $u \to\bullet v$ – *замен* Φ; u, v – в алфавите Δ; $\to, \bullet \notin \Delta$,
- u называется *левой частью* замены, v – *правой*.
- Замены, содержащие $\to\bullet$, называются заключительными.

Алгоритмы Маркова

- Какой алгоритм определяется этим заданием?
  - Дан объект – слово
- **Начало**:
  - не требуется делать ничего – тождественное преобразование объекта.
- **Продолжение**:
  - 0, если нет ни одной замены, левая часть которой входит в объект, или в объекте есть •; иначе 1.
- **Переработка**:
  - Найти первую замену, левая часть которой входит в объект, найти первое ее вхождение в объект и заменить его на правую часть этой замены.
- **Извлечение результата**:
  - Стереть в слове все символы •.

Пример:

Алфавит из трех символов: 0, 1 и дополнительного символа | (палочки). Цепочка замен:

|0 → 0||

1 → 0|

0 →

Исходный объект: 101. Работа алгоритма:

0|01

00||1

00||0|

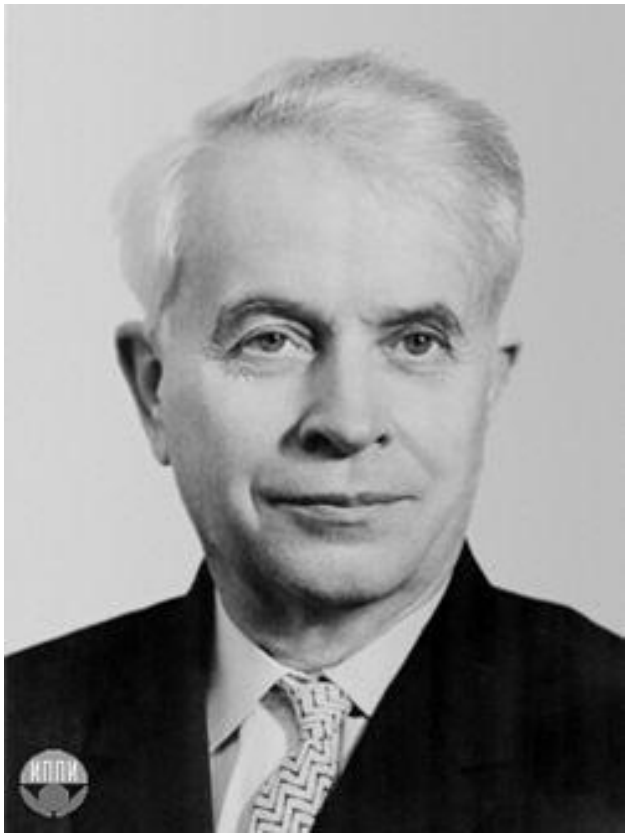
00|0|||

000|||||

00|||||

0|||||

|||||



Андрей Марков мл.

(9 [22].09.1903 – 11.10.1979)

Тезис об исчислениях (напоминание).

Всякое породимое множество порождается некоторой грамматикой.

Тезис Черча (вариант).

Всякая вычислимая функция вычислима некоторым алгоритмом Маркова.

Алонзо Чёрч (14.06.1903 — 11.08. 1995)

Алгоритмические проблемы

- Проблемы построения алгоритмов
- Проблема построения алгоритма разрешения – вычислимость характеристической функции

10-ая Проблема Гильберта

- Построить алгоритм, который по всякому алгебраическому уравнению от нескольких неизвестных с целыми коэффициентами выясняет, есть ли у него решение в целых числах.
- Отрицательное решение: Юрий Матиясевич (02.03.1947 -)