Содержание

[1. Математика и логика – История развития и взаимосвязь 4](#_Toc468912116)

[2. Логика высказываний 5](#_Toc468912117)

[3. Две теоремы о тавтологиях 6](#_Toc468912118)

[3.1. Логическое следование и логическая эквивалентности 6](#_Toc468912119)

[3.2. Подстановка и замена 6](#_Toc468912120)

[4. Формальные Теории 8](#_Toc468912121)

[4.1. Понятия выводимости 8](#_Toc468912122)

[4.2. Непротиворечивость формальной теории 8](#_Toc468912123)

[4.3. Другие аксиоматизации 9](#_Toc468912124)

[5. Логика предикатов 10](#_Toc468912125)

[5.1. Теоремы первого порядка. Неопределённая форма. 10](#_Toc468912126)

[5.2. Свободное и связанное вхождение переменных 11](#_Toc468912127)

[6. Формальная аксиоматическая теория К исчисления предикатов 12](#_Toc468912128)

[6.1. Теория К 12](#_Toc468912129)

[*6.2.* Общие замечания по теории К 12](#_Toc468912130)

[6.3. Непротиворечивость и полнота чистых ИППП 13](#_Toc468912131)

[6.4. Мета теория: 13](#_Toc468912132)

[6.5. Логическое следование, логическая эквивалентность в чистых ИППП 13](#_Toc468912133)

[7. Правила переименования переменных 14](#_Toc468912134)

[7.1. Правило переименования свободных переменных 14](#_Toc468912135)

[7.2. Правило переименования свободных переменных 14](#_Toc468912136)

[8. Предваренные нормальные формы (ПНФ) (канонические) 15](#_Toc468912137)

[9. Сколемизация (Skolem) 18](#_Toc468912138)

[9.1. Правило Скольма 18](#_Toc468912139)

[9.2. Клаузальная форма (Конъюнтивная Скольмовская форма) 18](#_Toc468912140)

[10. Правила резолюции 20](#_Toc468912141)

[10.1. – правило в логике высказывание 20](#_Toc468912142)

[10.2. Метод резолюций 20](#_Toc468912143)

[10.3. Примечания к методу резолюций 21](#_Toc468912144)

[10.4. Стратегии метода резолюций 21](#_Toc468912145)

[10.5. Правило резолющий и метод резолюций в логике предикатов 22](#_Toc468912146)

[Понятия универсума и универсификации 22](#_Toc468912147)

[Пример 1: 22](#_Toc468912148)

[Пример 2: 22](#_Toc468912149)

[Пример на метод резолюции 24](#_Toc468912150)

[11. Прикладное ИППП 25](#_Toc468912151)

[11.1. Теория равенства 25](#_Toc468912152)

[11.2. Формальная арифметика Джузеппе Пеано 26](#_Toc468912153)

[11.3. Теория групп (абелевы группы, неаксмоматизируемые) 26](#_Toc468912154)

[Теорема Гёдела о неполноте 27](#_Toc468912155)

[11.4. Первая теорема Гёдела о неполноте 27](#_Toc468912156)

[Интерпретация авторской формулировки 1-ой теоремы Гёдела: 28](#_Toc468912157)

[Стандартная интерпретация формулы А: 28](#_Toc468912158)

[1-ая теорема Гёдела в форме Россера: 28](#_Toc468912159)

[Замечания к 1-ой теоремы 28](#_Toc468912160)

[11.5. 2-ая теорема Гёдела 28](#_Toc468912161)

[Замечание по поводу 2-ой теоремы Гёделя 29](#_Toc468912162)

[12. Неклассические логики 29](#_Toc468912163)

[12.1. Нечеткая логика 29](#_Toc468912164)

[12.2. Обобщение операций в нечеткой логике 31](#_Toc468912165)

[12.3. Нечеткая лингвистическая логика 32](#_Toc468912166)

[13. Модальные логики 33](#_Toc468912167)

[13.1. Алетическая модальная логика 34](#_Toc468912168)

[13.2. Виды модальных логик 34](#_Toc468912169)

[13.3. Отношения модальностей 34](#_Toc468912170)

[1. Отношение подчинений 34](#_Toc468912171)

[2. Отношений противоречий 34](#_Toc468912172)

[3. Отношения контрарности 35](#_Toc468912173)

[4. Субконтрарность 35](#_Toc468912174)

[13.4. Пример семантической интерпретации модальной логики - модель Крипки 36](#_Toc468912175)

[14. Темпоральные логики 37](#_Toc468912176)

[14.1. Виды темпоральных логик 38](#_Toc468912177)

[CTL\* - computation tree logic "star". 38](#_Toc468912178)

[LTL - linear time logic 39](#_Toc468912179)

[CTL - computation tree logic 39](#_Toc468912180)

[15. \*\*\*ТЕОРИЯ АЛГОРИТМОВ\*\*\* 39](#_Toc468912181)

[16. Понятие алгоритма 39](#_Toc468912182)

[16.1. Требования к алгоритмам 40](#_Toc468912183)

[16.2. Формализация понятия алгоритма 40](#_Toc468912184)

[17. Машина Тьюринга (Turing Machine) 41](#_Toc468912185)

[17.1. Численная машина Тьюринга 42](#_Toc468912186)

[17.2. Применимость и конфигурация машины Тьюринга 43](#_Toc468912187)

Глобальное TODO:

Более литературный стиль (раскрывать сокращ.)

Вопросы в конце разделов

# Математика и логика – История развития и взаимосвязь

* Логика – цепь взаимосвязанных суждений (орудие доказательства)
* Интуиция – наитие (орудие изобретательства)

Этапы развития математики

1. Накопление первичных знаний (до 5/6 век ДНЭ) – понятие числа
2. (5В ДНЭ – 17 В НЭ) Греки/философы – зарождение логики
   1. Непоставимые числа (Пифагор)
   2. Планометрия
   3. Аристотель
      1. Категории, об истолкований (классификация), первая аналитика (логика, теорема доказательства), 2-ая аналитика, Топика (?)
3. (18 В – (начало) 20 В) Классическая математика
   1. Интегралы, дифференциальные уравнения, бесконечно большие/малые
   2. Теория множеств
   3. Вул – ввел математику и формулы в логику.
4. Развитие логики
   1. Логицизм (Тарасов, Фрёге)
   2. Конструктивизм
   3. Формализм

# Логика высказываний

Определяется утверждениями, которые истинны или ложны, но не то и другое вместе

О Высказывания: повествовательное предложение, которое истинно или ложно. Они могут быть простые и составные.

* Простые – (подлежащее +) сказуемое. Пример – Вечереет.

Составные – формируются из составных с помощью логических связей:

О Формула, которая истина во всех возможных интерпретациях называется обще-значная или тавтология.

О Формула, которая ложна во всех возможных интерпретациях называется не выполнимой или противоречивой.

О Формула, которая истина в некоторых интерпретациях называется выполнимой

* Логика высказываний занимается исследованиями зависимостей между истинности или ложности, простых или сложных высказываний.

# Две теоремы о тавтологиях

Т1 Если F – тавтология, то – противоречива, и наоборот.

Если F – противоречива, то – тавтология.

Если F – тавтология, то неверно, что – противоречива.

Если F – выполнимая формула, то не противоречива и не тавтология.

Т2 Если тавтологии, то Q – тавтология.

Если – тавтология ( ), то

## Логическое следование и логическая эквивалентности

О логический следует из , если во всех интерпретациях в которых истинно также истинно.

О Формулы называются логически эквивалентными, если они логический следуют друг из друга

T3 Теорема о логической эквивалентности

Если – логически эквивалентны, то – тавтология

(Доказательство следует из определения логической эквивалентности)

Т4 Формула логический эквивалентна

(Доказательство через таблицу истинностей)

Т5 (Позволяет выполнять логический преобразования)

Если в равенствах Теорий множеств, выражающая свойства операций над множествами, считать обозначение множеств простыми высказываниями, а операции над ними логическими связками, то имеет место следующие логические эквивалентности.

Теория множеств Логика

Метод таблицы истинности применяемы к всему выше сказанному позволяет выявить

1. Является ли произвольная формула тавтологии, противоречием, или выполнимой
2. Имеет ли место логическое следование
3. Является ли логически эквивалентны

## Подстановка и замена

* Пусть F – формула в которую входит переменная (простое высказывание)

Пусть P – формула, тогда обозначение будем считать формулу, полученную подстановкой в F, P вместо всех вхождений . А – полученная из F, заменой некоторого вхождений

- Свойство подстановки: если F – тавтология и формула, то подстановка является тавтологией.

Т6 - Правило подстановки: подстановка сохраняет тавтологию формулы.

* Правило замены. Данное правило ослабляет строгость и не сохраняет тавтологичность.
* Частным случаем формулы: формула B, полученная из А подстановкой (унификатор) вместо называется частным случаем формулы A
  + Унификатор – на формулы B и А унифицируется (почти эквивалентны), на основе правила подстановки.

# Формальные Теории

Формальная Теория считается заданной, если выполняются 4 условия:

1. Задана – множество символов в теории , то есть её алфавит.
2. Задана – множество слов в алфавите **,** то естьмножество всех возможных формул теории .
3. – множество аксиом теории . Если , то – называется аксиоматической теории
4. Задано множество (англ. relation - отношение), которое называется множеством отношений между формул правилом вывода

* может быть конечно или бесконечно
* Формулы порождаются индуктивно
* - сигнатура (язык) теории
* . Используется коечное множество схем аксиом, из которых рождается бесконечное множество собственных аксиом.
* Все аксиомы делятся на логические и собственные (нелогические).

## Понятия выводимости

* Пусть – формулы теории (). Если существует правило вывода , такое, что находится в отношении , то говорят, что G непосредственно выводимо из по правилу R
  + В этом случае – посылки R, a G – вывод
  + G непосредственно выводимо из по R
* Пусть (логическим) выводом G из в формальной теории называют последовательность формул такую, что:

а)

б) верно:

-либо (т.е. аксиома)

-либо (посылка вывода)

-либо получено путём вывода

-Если множество посылок – пустое, то говорят, что G выводимо непосредственно из аксиом ( является теорией теории )

-Когда непустое, называют гипотезами (-выводиться из)

* Свойства выводимости

1. расширяет множество гипотез не нарушая выводимость
2. Если и для всякой формулы

## Непротиворечивость формальной теории

О Формальная теория называется семантически непротиворечивой, если не одна теорема не является противоречием.

Формула теории является теоремой, тогда и только тогда, когда в данной теории существует логический вывод этой формулы и эта формула тавтология.

О Формальная теория назывется формально непротиворечивой, если в ней невозможно выводимости формулы одновременно в процессе одного логического вывода.

О Теория сематический непротиворечива тогда и только тогда, когда она формально непротиворечива

- В непротиворечивой теореме невозможно построить одновременно вывод тогда и только тогда, когда не одна теория не противоречива

* Разрешимость формальной теории: формальная теория может быть разрешима или полу-разрешима
  + называется разрешимой, если существует алгоритм, позволяющий для любой формулы данной теории определить, является ли она теоремой теории
  + Теория полу-разрешима, если существует алгоритм, который для любой формулы дает «да», если формула является теоремой , и не дает никакого ответа, если не является формулой

## Другие аксиоматизации

1. Д. Гильберт, В. Аккерман
   1. Логические связки
   2. Форма:
   3. MP
2. Беркли Россер
   1. МР
3. Ст. Клини
   1. 10 аксиом, например:
   2. МР

# Логика предикатов

## Теоремы первого порядка. Неопределённая форма.

- Praedicatum – высказывание, сказанное

О (1) выражения имеющие вид повествовательного предложения и содержащие неопределённые термины называются предикатами. Стоит отметить, что предикатам называется не сам предмет, а его свойства. Предикат – функция высказывания.

«х является человеком» – Одноместный (местность указывается сверху)

«у является отцом х» – Двухместный

«y,z являются родителями х» - трехместный

при подстановке x,y,z – она становиться высказыванием

О n – местным предикатом на множестве М называется функция n аргументов область определения которой является М, а область значений – истинно или лож

- их можно соединять связками, как высказывания

- логические связки действуют на предикат, а не на переменные

неверно, что…

неверно выражение, что неверно, что…

О Два предиката , определённые на множестве М называются эквивалентными () на М, если принимают одинаковое значение «истинно» на одинаковых значениях аргументов.

О Предикат называется тождественно истинным (ложным) на множестве М, если для всех наборов ( (во всех интерпретациях) принимает значение истинно (ложно).

О называется выполнимым на М, если множество его значение включает значения истинно и ложно ()

О Пусть – одноместный предикат на М. – функция, которая принимает значение истинно, если истинный предикат для любого . называется квантором всеобщности.

О Пусть – одноместный предикат на М. – функция, которая принимает значение истинно, в том случае, когда в М есть хотя-бы один (истинно). называется квантором существования.

* Пример: Все реки текут -

– x - река

– х течет, где х может быть ведро, крыша

* Формула, полученная из символов, записанных предикатов, соединённых логическими связками с введением кванторов называется формулой логики предикатов.
* Введение квантора называется операции квантификации (навешиванием кванторов)
* Если – одноместный предикат на тогда:
* Формула замены кванторов

## Свободное и связанное вхождение переменных

- используемые обозначения

О Вхождение переменной в атом называется свободным.

Если переменная входит свободно в предикатной формулой A и B, то она остается свободной в

О Если в формулу А, х входит свободно, то вхождение переменной х в формулы называется связанным.

При этом, если в формулу А помимо х входят другие переменные, то они остаются свободными и в указанных квантифицированных формулах

- одна и та же переменная может входить как связано, таи и свободно.

- если все переменные, входящие в формулу, связаны, то формула называется замкнутой.

* Пример:

Для любого х, если х – человек, то существует такой у, что у является отцом х.

(Данная формулировка является замкнутой)

О – термом называют свободным для переменной х в формуле А, если никакое свободное вхождение х в А не лежит в области действия квантора по переменной, входящий в терм

- В частности, если никакая переменная не является свободной переменной формулой А, то – свободен для любой переменной формулы А.

-Пример:



– свободен в А (т.к. нет квантора с у, и из-за второго условия (?) )



-Пример:



- x для А в этом не свободен

# Формальная аксиоматическая теория К исчисления предикатов

## Теория К

* 1. Алфавит ()
     + переменные
     + предикатные символы
     + функциональные символы (имена термов)
     + знаки логических связок
       - Основные
       - Дополнительные
     + Скобки
     + Кванторы
  2. Множество формул (Ф)
  3. Множество аксиом ()
     + Аксиома теории L + предикатные аксиомы
     + t - терм
     + терм t свободен для переменной х в формуле
     + (TODO)
  4. Правила выводов (R)
     + () MP
     + () правило связывания квантором всеобщности (Gen - generalization)
     + () правило связывания квантором существования
       - Ограничения , A(x) содержит свободное вхождение x, а В их не содержит
       - (нарушение приводит к ложным выводам из истинных посылок)

Пример “x делится на 6”; “х делится на 3”

() (Формула не всегда истина)

()

(, но получилась ерунда)

ложна из-за некорректного использования

## Общие замечания по теории К

* 1. Аксиомы – Логические (а не собственные)

О Исчисление предикатов (ИП) в теории К без собственных аксиом называется чистым.

- В чистом ИП из теории предикатов часто удаляют константы и формальные символы

О ИП в теории с собственными аксиомами и/или предметные константы и/или формальными символами называется прикладное

О ИП в которых кванторы могут связывать только переменные, но не связывает предикаты называется ИП первого порядка (сокращенно - ИППП).

- Теории которые реализуют ИППП называются теориями первого порядка.

- Теория К – чистое ИППП

- Теории высших порядков допускают квантификации по предикатам (и по функциональным символам). (n штук)

- На практике достаточно бывает чистого и прикладное ИППП

* 1. Иногда удаляется , как несамостоятельный.

## Непротиворечивость и полнота чистых ИППП

Т. Формальная теория К непротиворечива семантически и формально. То есть ей свойственно не противоречивость, и невозможно одновременно.

(Доказательство через h-преобразования)

– следующее преобразования F в формулу логики высказывания (пример – в формулу теории L)

* 1. Удалены все кванторы и термы
  2. Остаются предикатные символы и логические связки.

- Имена предикатов рассматриваются, как простые высказывания

=> h преобразования от формулы ИП оставляют только её структуру

## Мета теория:

1. Всякая теорема чистого ИППП общезначима. Это является аналогом тавтологии в логике высказываний.
2. Всякая общезначимая формула является теорией в чистом ИППП

## Логическое следование, логическая эквивалентность в чистых ИППП

О аналогичны соответствующим в теории L

О формула В теории K является логическим следствием формулы А теории К, если В выполняется на любом наборе и в любых интерпретациях, в котором выполняется А

О формулы А и В теории К логический эквивалентны, если являются логическим следствиями друг друга

1. Все формулы замены квантора и т.д.

# Правила переименования переменных

## Правило переименования свободных переменных

Если формула выводима в вычисление ППП, и содержит свободное вхождение х, не одной из которых не лежит в области действия квантора по у, то выводима .

1. дано по условию
2. выводимая формула без свободных вхождений х. – замкнута.
4. 1,3,MP
5. 4,
6. 2,5,MP
8. 6,7,MP

## Правило переименования свободных переменных

Если выводимы формулы для и , то из выводимости для , следует , а из выводимости , следует выводимость , при условии, что не содержит свободных вхождений и содержит свободное вхождение , не одна из которых не лежит в области действия квантора по у.

1. дано по условию
2. применяем правило к второй формуле возможно т.к. в правой части у – свободен, а в левой части, свободных вхождений у по условию нет.
3. 2, по у
4. 1,3,MP

# Предваренные нормальные формы (ПНФ) (канонические)

Понятия логического следования и логической эквивалентности в логике предикатов позволяет получить некоторые тождеств, доказуемые, как леммы теории К.

( не входит свободно не в С, не в D в )

-----------------------------------------------------------------

Система эквивалентностей 1-6 дает возможность для всякой формулы теории К:

1. Выносить кванторы вперед
2. Заменять один квантор другим
3. «Спускать» отрицание внутрь области действия квантора.

О формула вида где – кванторы, – переменные (различные), а F – формула без кванторов, называется ПНФ

ПНФ состоит из префикса, представляющий из себя кванторы с квантифицируемыми переменными, и матрицы F - без кванторная форма.

Пример:



– ПНФ

– ~~ПНФ~~

– ~~ПНФ~~

Пример приведения к ПНФ

1. (4), w вместо z
2. (3), u вместо y
3. (4), v вместо w

!!! После получения ПНФ, необходимо узнать о возможности возвращения имен переменных. На основании правила переименования связанных переменных, утверждаем: Если переименованная переменная входила в исходную форму связано, то её первоначальное имя может бить возвращено. То есть исходная формула должна быть замкнута по данной переменной.

можно вернуть.

* В процессе развития математической логики как прикладной дисциплины, в частности при разработки декларативных языков и логического программирования, стало удобно получать ПНФ без импликации и эквивалентности. Сначала надо избавится от импликации, потом получит ПНФ, которое не импликативное или прикладное.

Шаги

1. Исключить логические связки импликации на основе эквивалентности
2. Если необходимо, переименовываем связанные переменные так, чтобы никакая переменная одновременно не входила в формулу и связанно и свободно. Данная процедура должна выполнятся во всей формуле, включая её части.
3. Если необходимо, надо разделить связанные переменные. Достигается исключение повторений, случайные совпадения связанных переменных



в формулу В входит , в под-формулу входит Р, а в Р

входит , причем связан (два и более раз) разными кванторами по всей формуле.

1. Удаление квантификации, в области действия которых нет квантифицированных переменных. Элиминация квантора.
2. Протаскивания отрицания – сужение из области действия

В прикладных ПНФ можно избавляться от двойных отрицаний

Формула замены кванторов

Правила двух отрицаний

Правило Деморгана

1. Смещение кванторов влево
   * 1. ()
     2. (в «В» не должно быть свободного x)

1…6 получили прикладную ПНФ

Пример:

1. (исключение импликации)
2. (нет переменных, которые входили бы связано и свободно, следовательно шаг 2 не нужен)



1. (разделение связанных переменных)
2. ( – в нет свободных вхождений )
3. ( – в нет свободных вхождений )

Префикс Матрица

Каждая формула может допускать множество различных ПНФ (Зависит от переменных и порядка использования Лемм)

# Сколемизация (Skolem)

* Подход Сколема к упрощению не импликативных ПНФ:
  1. – пусть она выполнима в некоторых импликациях, тогда в области определения предиката P(M) найдется постоянная c :
  2. Пусть предикативная формула включает в себя две переменные. Тогда выполнима тогда и только тогда, когда выполнима формула , где – терм, предикат.

Убрав , в самой формуле, вместо введен функциональный символ , то есть предикат.

Причем его аргументом является переменная, квантифицированная по

## Правило Скольма

О Формой Скольма называется ПНФ, не содержащий и получена из исходной ПНФ, применением правила Скольма.

1. Если стоит в начале префикса, то он элиминируется, а вместо подкванторной переменной вводится предметная константа
2. Если стоит в середине или конце, то элиминируется, а вместо подкванторной переменной вводится функциональный символ, местность которого равно числу предшествующих кванторов , а аргументами являются подкванторные переменные этих .

Пример:



Пример:



Пример:



В логике первого порядка доказаны следующие теоремы

1. Если замкнутая формула и её Скольмовская формула, то и одновременно выполнимые или невыполнимые.

## Клаузальная форма (Конъюнктивная Скольмовская форма)

О Клаузальной формой называется Скольмовская форма, матрица которой является формулой в КНФ

В основе закона дистрибутивности, правила Деморгана, закон ассоциативности

Дальнейшие преобразования формулы полученные на всех предыдущих этапах включает следующие шаги:

1. Элиминация
2. Разделение формулы на дизъюнкты (элементарная конъюнкция)

Доказано, что клаузальная формула, и формула, полученная после элиминации одновременно выполнима или невыполнима.

В итоге распадаются на множество дизъюнктов (предложений)

Т. Если Г – множество дизъюнктов, полученных из исходной формулы логики предикатов F, то F является противоречием тогда и только тогда, когда множество Г невыполнимо. Другими словами - не существует интерпретация для всех предметных переменных (аргументов предикатов, входящих в формулы, составляющие Г в которой все формулы Г имели бы значение истинно).

- Невыполнимость множества Г значит, что множество Г не имеет модели.

Формула F и F\* получены после элиминации из F, одновременно выполнимы или невыполнимы в любой интерпретации входящих в них переменных. Поскольку F\* - КНФ, она является противоречием тогда и только тогда, когда нет интерпретации в которой все входящие в F\* дизъюнкты имели бы значение истинно одновременно. Следовательно F\*, а значит и F, являются противоречиями тогда и только тогда, когда Г – невыполнима.

Таким образом выполняется последующее преобразование исходной формулы логики предикатов:

1. Получение ПНФ
2. Скольмизация
3. Клаузальной формы
4. Элиминация
5. Элиминация

Приводит к множеству дизъюнктов Г. К этому множеству применяется правило резолюции (самая мощная), лежащая в основе метода резолюций, как основного способа автоматического доказательства теорем. Данный способ еще называется машинным или программным логическим выводом.

Через Res обозначается правило резолюции

# Правила резолюции

Res

## – правило в логике высказывание

Пусть высказ, где

Где А, В – произвольные высказывание (составное, простое, или пустое)

– родительские, резольверуемые, резольверуемые дизъюнкты

– резольвента, резольвирует , Р и – контрарные литералы

– частный случай . Если отсутствует или – ложно, то правило резолюции:

Если отсутствует и , и , то говорят, что правило порождает пустой дизъюнкт .

Т. Резольвента является логическим следствием резольвируемых дизъюнктов.

Правило дает основание для метода резолюций, которое позволяет проверить формулы на логическое следование.

Пусть даны формулы . Можно показать, что ( тогда и только тогда, когда формула – является противоречием.

Из определение логического следования вытекает, что возможно в том, и только в том случае, когда ( – тождественно истинно. Формула должна быть тавтологией. Отрицание тавтологии является противоречием. = false

В свою очередь, можно показать, что из формул следует любая формула – т.е. пустой дизъюнкт.

## Метод резолюций

Последовательность этапов:

* 1. Формируем множество , каждый элемент которого является формулой, преобразованной к дизъюнкту. Иначе - ничего не получится.
  2. К дизъюнктам множества Г попарно применяем правило резолюции, и каждый раз резольвент добавляем в множество Г. Резольвента вправе участвовать в последующих резольвированиях наравне с прочими формулами множества Г.
     1. ; ; ; а после,
  3. В ходе циклического выполнения пункта 2 возможны 3 варианта
     1. В множестве Г, нет резольвируемых дизъюнктов. Нельзя в Гамма найти такую пару формул, в которых был бы контрарный литерал. Другими словами -резольвировать нечего. Следовательно, множество Г выполнимо, т.е. имеет модель. А значит существует интерпретация, в которой все формулы из Г тождественно истинны, а значит в том числе истинно и формула . А раз то – ложно, и не является тождественно истинной формулой, когда все – истинны.
        1. – теорема опровергнута
     2. Результатом очередного применения стал пустой дизъюнкт, т.е. в Г были два литерала, которые поглотили друг друга () – говорят, что теорема доказана, т.к. Г оказывается невыполнимо, то есть не имеет модели. Не существует интерпретации, в которой все формулы из множества Г тождественно истины, а значит формула – тождественно ложно в любой интерпретации, а значит имеет место логическое следование ().
     3. Не наблюдается ни первого варианта, ни второго, т.е. процесс зацикливается, а это означает, что Г на каждом шаге пополняется новыми резольвентами, пустого дизъюнкта среди них не появляется, и процесс резольвирования не заканчивается и является бесконечным процессом. Алгоритм не дает ответа на вопрос, имеет ли место:

## Примечания к методу резолюций

* 1. Метод рез. - вид доказательства от противного. Он легко программируется, благодаря формализации с помощью формулы Res.
  2. Теория, одним из правил вывода в которой является правило резолюций, и которая реализует в себе алгоритм метода резолюций, является полу-разрешимой (в 3-м пункте, когда не дает ответа, не да, не нет). Алгоритм метода резолюций ограничен по области применения.

Пример:

* 1. Резольвируем.

----------------------------------- Резольвенты:

* 1. 2,4
  2. 6,1
  3. 7,3
  4. 8,5

Итак,

## Стратегии метода резолюций

Процесс резольвирования (циклический) - есть экспоненциальный алгоритм (перебор). Для этого, их оптимизируют

* 1. Стратегия насыщения уровня: Если Г, множество дизъюнктов, то ,

- множество дизюнктов, порожденных объедининием со всеми возможными резольвентами по . Стратегия не упрощает вычислительныю сложность, но упорядочивает процесс наращивания множества дизъюнктов, т.е. добавление резольвентов. То есть данная стратегия является аналогом поиска в ширину.

* 1. Линейная - достигается когда на каждом шаге резольвирования, одним из родительских дизъюнктов является дизъюнкт, полученный на шаге. (Смотрите пример выше)
  2. Предпочтение одночленам - с начала пытаемся резольвировать литералы. Если получаем пустой дизъюнкт, всё хорошо. Если нет, то резольвируем одночлены (литералы) с парами. Если получили пустой дизъюнкт, всё хорошо. Если нет, то пара с парой, и т.д. Суммы длин родительских дизъюнктов образуют не убывающую последовательность.

## Правило резолющий и метод резолюций в логике предикатов

### Понятия универсума и универсификации

Мы пытаемся построить модель множества Г, то есть найти хотя бы одну интерпретацию для n+1 формул, в которой они все были бы истинны. Однако - каждая формула дизъюнкт с произвольным количеством аргументов, а значит надо вести поиск по очень большому количеству интерпретаций.

Из-за этого, был введен термин универсификации. Универсум Эрборна H(Г)

Понятие универсума Эрборна

1. Множество всех предметных констант из Г принадлежит H(Г). Если не одной константы в Г нет, она вводиться принудительно, производно
2. Если термы принадлежат множеству Г, то Н(Г) содержит и терм , где - любой к-менстный символ из Г.
3. Никаких других термов в Н(Г) нет.

Н(Г) является наиболее общим множеством для поиска интерпретации (для модели). И если модель существует то она существует именно среди Эрборновских интерпретациях. А если модели нет, то её именно нет в Н(Г)

- да или нет - достаточно рассмотреть только Эрборновские интерпретации.

Н(Г) - бесконечное, но счетное множество.

### Пример 1:

Пусть

- множество констант, входящие в Г

*-* функциональные символы (термы)

### Пример 2:

В Г входят формулы, которые содержат константы , переменные , и два предиката

- так как нет термов.

Подбор конкретных значений из универсума Эрборна составляет основу операции унификации, а сам набор значений называют унификатором.

Над формулами, представляет собой дизъюнкты выполняется операция унификации предметных переменных. Эта завершающая операция перед применением правила резолюции в логике предикатов. Это является добавочной, но завершающей операцией.

Суть унификации в замене некоторых переменных на константы с тем, чтобы было возможно применить правило резолюции, если к исходным формула (до унификации) правило резолюции не применимо.

- не контрарные литералы, так как а - константа

*-* операция унификации

Если в общем случае, этих подстановок несколько: {//, //, //}, то должны выполнятся два условия.

1. Если - переменная, то ,
2. ~~{/}~~ - замена не допустима!

О Унификатором предикатных формул первого порядка называется множество подстановок {//, //, //} в формулы , на котором (мнгожестве) становятся тождественными, то есть унифицируются.

Примечание:

Унификатор возможен и для одного предиката, имеющего семантический различный аргумент

Пример:

;

{t(x,y)//u} - во вторую формулу

{f(z)//y} - во первую

{p(u)//z} - в первую

- пропали. Вместо них введены термы

О пусть предикатные формулы, представлены в виде дизъюнктов. Тогда правило резолюции в логике предикатов имеет следующий вид. при условии, что в существуют контрарные литералы , то есть представимо в виде:

А литералы - являются формулами унифицируемы (вообще говоря) наиболее общим оператором . В общем случае, для двух формул может существовать бесконечное число унификаторов, но среди них всегда можно найти тот, из которого выводятся все остальные. Этот и называется наиболее общим, . (Все )

Резольвента является дизъюнктом, полученным из дизъюнкта введение унификатора

Литералы при унификации с наиболее общим унификатором дают унифицированную формулу

//x}

дают , точно так же, как и в логике высказываний.

Метод резолюций в логике предикатов, тот же, что и в логике высказываний

1. Имеем . Если получен при резулвировании с учетом операции унификац. получен пустой дизъюнкт, то Г противоречим, то есть не имеет модели, не выполнима, а значит G

1. Если при резольвировании пустой дизъюнкт не получается и резольвировать нечего, то Г выполнима, имеет модель, и G значит теорием опровергнута, и ответ на вопрос отрицательный

1. Когда процесс резольвировании в множестве Г приведение всех возможных унификаторов из Н(Г) не заканчивается (уходит в бесконечность), алгоритм не дает ответа на вопрос

Третий пункт усложняется тем, что теоретический Н(Г) может быть бесконечным, при бесконечным процессе резольвирования, однако на практике, наблюдается цикличность появлений новых дизъюнктов в Г, что и говорит о зацикливании процесса.

Пример на метод резолюции (хрестоматийный, классический пример)

Даны предикаты

"x является отцом у"

: "x является мужчиной"

"x - единокровен с у"

"x является братом y"

Запишем формулы о родственных связях:

а) Все отцы являются мужчинами

б) Если у детей один отец, то они единокровные

в) брат является единокровным мужчиной.

Даны факты, которые подтверждают истинность данной формулы

1. - данная формула является ПНФ, скольмизация не требуется, является формулой в клаузальной форме, элиминируя получаем:

1. Приводим к дизъюнктам все возможные формулы
   1. - сколемизация не требуется, форма - клаузальная, элиминируем кванторы:

- дизъюнкт.

c.   
¬S(x,y)∨¬M(x)∨B(x,y)

1. Сколемизация не требуется, так как нет квантора
2. Все кванторы в клаузальной форме
3. Элиминируем кванторы

3.

1. Эти формулы окончательные дизъюнкты.

5.

6.F(петр, зоя)

1. Для применения правил резолюции, для каждой пары необходимо подобрать унификатор из универсума Эрборна:
2. Лучшие взять стратегию линейную с предпочтением одночленам.

(2) //x, перт//y}

8. (2,5)

9.{анна//z}

(8,4)

10. (1){петр//x, зоя//y}

(1,6)

11. (3) {петр//x}

(10,3)

12. (11) ={анна//y}

(11,9)

13. (7) ={петр//w}

(12,7)

Вывод - получение пустого дизъюнкта говорить о том, что теорема доказана, вопрос был использован, и из исходных формул и фактов, следует G (- неверно).

# Прикладное ИППП

## Теория равенства

Теория равенства - это прикладное исчисления предикатов, в котором имеются:

1. Собственный двухместный предикат = (инфиксная запись )
2. Собственные аксиомы (схемы аксиом):

В теории выводимы следующие теоремы:

1. для любого терма t
   1. - своего рода A(x)
   2. (1),(2),MP
   3. (1){x=x/A(x)}
   4. MP
   5. (выведено выше) аксиома 1

1. по теории дедукции
   1. z/A(x), y/x, x/y
   2. На основании предыдущего равенства
   3. (2),(1), правило транзитивности
   4. по дедукции

## Формальная арифметика Джузеппе Пеано

Формальная арифметика - это прикладное исчисление предикатов, в котором имеются:

1. Предметная константа о
2. Двухместные функциональные символы + и , одноместный функциональный символ '.
3. Двухместный предикат =
4. Собственные аксиомы (схемы аксиом)

: схема аксиом мат. индукции

:

A\_4:

где P - любая формула; а - любые термы теории

## Теория групп (абелевы группы, неаксмоматизируемые)

Теория групп - это прикладное исчисление предикатов с равенством, в котором имеются:

1. Предметная константа о
2. Двухместный функциональный символ +
3. Собственные аксиомы (схемы аксиом)

Группа называется абелевой, если имеет место собственная аксиома :

Говорят, что в абелевой группе все элементы имеют конечный порядок n, если выполнена собственная аксиома - это сокращение для - k раз.

Эта формула не является формулой , поскольку содержит "посторонние" предметные предикаты и переменные. Однако для любого конкретного n собственная аксиома может быть записана в виде допустимой формулы теории

Абелева группа называется делимой, если выполнена собственная аксиома

Бесконечное множество допустимых формул теории

Можно показать, что любое конечное множество формул, истинное во всех делимых абелевых группах, истинно и в некоторой неделимой абелевой группе, то есть теория делимых абелевых групп не является конечно аксиоматизируемой.

Абелева группа называется периодической, если выполнима собственная аксиома

Данная "бесконечная" формула не является допустимой для нанесения предикатов и тем более . Таким образом, теория периодических абелевых групп не является аксиоматизируемой, если не включать в теорию групп всю формальную арифметику.

# Теорема Гёдела о неполноте

(Курт Гёдель)

Теоремы Гёдела (в 31 году) сведены к двум, которые показывают неразрешимость второй задачи Гильберта (о самодостаточности мат. аппарата)

ω- непротиворечивости

Теория T называется -непротиворечивой, если для любой формулы невозможен одновременный вывод формул . Из выводимости формул следует выводимость . k - числа Гёдела (нумерация формул).

Обычное противоречивость - частный случай - непротиворечивости

## Первая теорема Гёдела о неполноте

Теорема утверждает, что в произвольной, непротиворечивой теории с доказуемыми арифметическими высказываниями, можно построить истинное высказывание, истинность которого не доказуемо средствами (или аппаратом) самой теории.

Другими словами - всякая эффективная, аксиоматизируемая теория достаточная для представления арифметическими операций, не может одновременно обладать свойствами непротиворечивости и полноты.

### Интерпретация авторской формулировки 1-ой теоремы Гёдела:

Если формальная теория Т непротиворечива, то формула А не выводима в ней; если формальная теория Т ω-непротиворечива, то не выводима в Т. Таким образом, если Т ω-непротиворечива, то она не полна и А служит примером неразрешимой формулы.

Стандартная интерпретация формулы А: не существует вывода А

Если Т не \_\_\_\_ и истинно в стандартной интерпретации

По Гёделю - истинно на множестве , но в Т не выводима.

### 1-ая теорема Гёдела в форме Россера:

Пусть имеется формальная теория S (system) и её формула . если S непротиворечива, то в ней не выводимы обе формулы B и . То есть если S непротиворечива, то S неполна и В служит примером неразрешимой формулы.

Интерпретация формулы В: Если существует вывод формулы В, то существует вывод формулы

Варианты

1. Всякая, достаточно сильная, рекурсивно эффективно аксиоматизируемая теория неполна.
2. Любая непротиворечивая формальная аксиоматическая система содержит неразрешимые формулы.

### Замечания к 1-ой теоремы

1. предположение непротиворечивости формальной теории в посылки условия теоремы принципиальна.
2. Истинное высказывание недоказуемой в формальной теории называют последовательностью Гёделя: Можно указать на бесконечное число высказываний, обладающих свойствами истинности, недоказуемой в рамках данной теории (аппарат теории принципиально ограниченный).

## 2-ая теорема Гёдела

Любая рекурсивно аксиоматизируемая формальная теория включающая формальную арифметику и некоторые высказывания о формальной доказуемости (имеющая механизм построения логического вывода), содержит утверждение о свой непротиворечивости, тогда и только тогда, когда эта теория противоречива.

Пояснения: Если теория Т противоречива, то в ней выводима любая формула, в том числе и утверждение о непротиворечивости Т. Если Т непротиворечива, то Т не содержит в себе утверждение о свой непротиворечивости, что следует из первой теореме Гёделя.

Можно построить такую формул в Т, которая в стандартной интерпретации будет истинна тогда и только тогда, когда Т противоречива. В этом случай, возможна формулировка:

Если формальная теория Т непротиворечива, то в ней не выводима формула, содержательно утверждающая непротиворечивость Т.

То есть она утверждает, что непротиворечивость любой \_\_\_\_ системы, не может быть доказано средствами самой этой системы.

### Замечание по поводу 2-ой теоремы Гёделя

1. Запись в виде формул утверждение о том, что Т непротиворечива, может быть различно, причем эти формулы могут быть не эквивалентны друг другу, и даже некоторые, доказуемые в самой этой теории
2. Теорема утверждает, что в противоречивой теории, доказуемо всё, что угодно, вплоть до её непротиворечивости. Следовательно, построение доказательства о её непротиворечивости средствами самой теории не дает никакого результата, и только усиление теории (выход за её пределы) позволяет доказать непротиворечивость
3. Если теория Т2 доказывает непротиворечивость Т1, то аппаратом теории Т1 невозможно доказать непротиворечивость Т2 (основное следствие 2-ой теоремы Гёделя).

# Неклассические логики

## Нечеткая логика

В основе лежит теория нечетких множеств

Нечетким множеством на универсуме U называется множество упорядоченных пар. - функция принадлежности

Функция количественно определяет принадлежность х по базовому множеству U

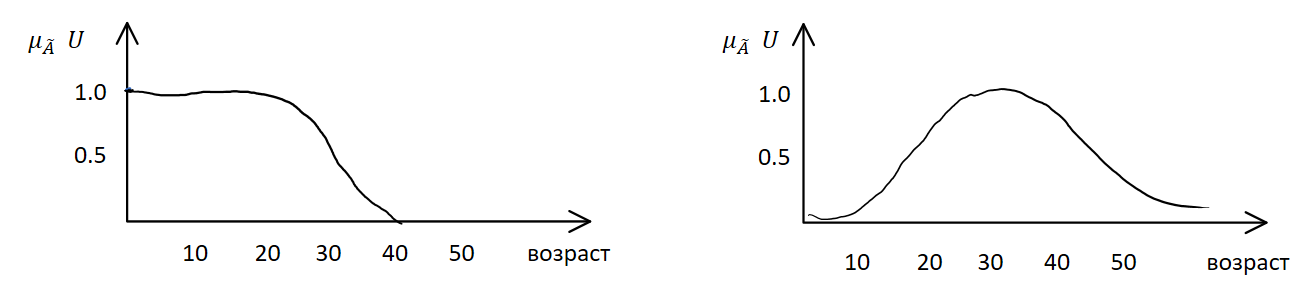
Если , то х точно не принадлежит; если равно В, то точно принадлежит, а если , то такие элементы назывются нечёткими.

{0.2/5, 0.4/7, 1.0/9, 0.1/12}

Практические примеры нечеткой логики

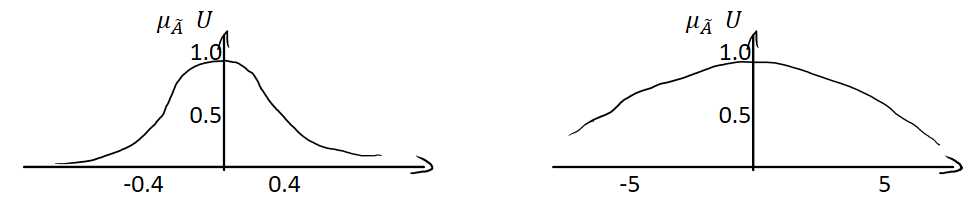
1. U - множество людей различного возраста.

- множество молодых людей



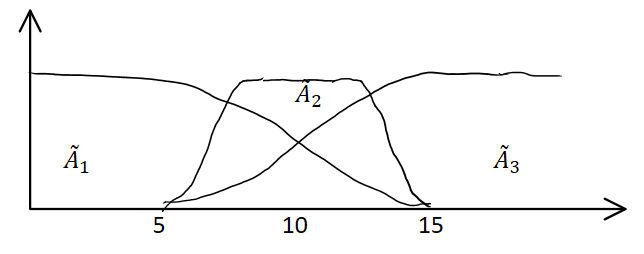
1. U - множество действительных чисел.

- числа близких с 0



1. U -

- холод, - тепло, - жарко



На нечётких множествах используют операции классической логики, соответственно работают и свойства этих операций

На нечётких множествах работают операции преобразования классической логики

Нечеткая логика оперирует предложениями в отношении которых можно сделать заключение о степени истинности или ложности

Нечёткая логика количественно выражает значения истинности в диапазоне от ложно до истинно

Для каждого предложения нечёткой логики необходимо задать универсум и нечёткое множество на нём. При этом функция принадлежности будет отображать предложение в нечёткое множество пар: значение элемента, значение функции принадлежности.

Предложением, высказыванием или предикатом нечёткой логики называют словосочетание вида "" где , а - нечеткое множество на универсуме U

Аналогично операции над нечёткими множествами вводятся логические операции нечёткой логики

1. (?)

Логику с этими операциями называют логикой с макси-минными операциями

Обобщение логических операций - позволяют вводить разное из содержание в нечёткой логике рассуждения.

Отрицание можно задать как отображение not:

L - множество значений истинных предложений нечёткой логики

Условия отображения

1. not(0)=1
2. not(1)=0

## Обобщение операций в нечеткой логике

t-норма - обобщение конъюнкции ()

k-норма - обобщение дизъюнкции ()

Оба являются бинарными операциями на множестве L=[0,1]

Обе операции обладают монотонности, ассоциативности, коммутативность

При этом вводятся граничные условия:

Для t-нормы: 1:

0:

t и k нормы обладают свойством двойственности.

формулы для определения истинности обобщенных конъюнкции и дизъюнкций могут быть разными

t- норма Гёделя (как пример):

k-норма Гёделя:

Варианты формул для оценки истинности:

1. - классическая формула
2. - алгебраическая произведение
3. - граничное произведение

Пример:

К вечеру будет довольно пасмурно, и не исключено, что пойдет дождь.

Не четкое предложение Не четкое предложение

Пусть соответствующие степени истинности заданы дискретное множествами

{0.2/19, 0.5/20, 0.9/21}

{0.5/19, 0.7/20, 0.8/21} , где 19,20,21 - время в часах

1. {0.2/19, 0.5/20, 0.8/21}
2. {0.1/19, 0.35/20, 0.72/21}
3. {0/19, 0.2/20, 0.7/21}

1. - классическая формула
2. - алгебраическая сумма ст. ист.
3. - гр. сумма ст. ист.

Пример:

К вечеру будет довольно пасмурно, или даже пойдет дождь.

Не четкое предложение Не четкое предложение

1. {0.5/19, 0.7/20, 0.9/21}
2. {0.6/19, 0.85/20, 0.98/21}
3. {0.7/19, 1/20, 1/21}

Разброс значений полученный по разным формулам позволяет выбирать тот или иной расчёт нечеткости в зависимости от условий задачи.

Для нечетной импликации:

1. - кл. формула (импликация Заде, или нечеткая импликация Гёделя), для
2. (собственная нечеткая импликация Заде), для любых сравнений
3. - формула Мамдани
4. - формула Лукасевича
5. Дж. Гоген (при )

Пример:

Если к вечеру немного похолодает, то может быть прояснение.

Не четкое предложение Не четкое предложение

2. {0.8/19, 0.5/20, 0.8/21}

4. {1/19, 1/20, 0.9/21}

5. {1/19, 1/20, /21}

## Нечеткая лингвистическая логика

Автор - Лод Физаде (?)

В основе - понятие нечеткого множества, и лингвистической переменной

Цель - формализовать рассуждение максимально приближенной к реальном с помощью нечетких множеств.

Виды нечеткости

1. Нечеткий количественные понятия (много, мало, несколько, около, почти)
2. Нечеткий истинностные значения (существенно истинный, более или менее истинный, более ложный, менее ложный)
3. Нечеткие понятия категорий (молодой/пожилой, холодный/прохладный/теплый, строгий/снисходительный)

О Лингвистическая переменная - переменная, которая принимает значения слов и выражений естественного языка.

Пример - Лингвистическая переменная стоимости - {дорогой, бюджетный, дешёвый, премиум классы}

Каждому значению лингвистической переменной должен быть поставлен в соответствии смысл (семантическое содержание)

Линвистическая переменная - 5-и местный картеж.

X - имя

T(X) - множество значений

U – универсум, или базовое множество

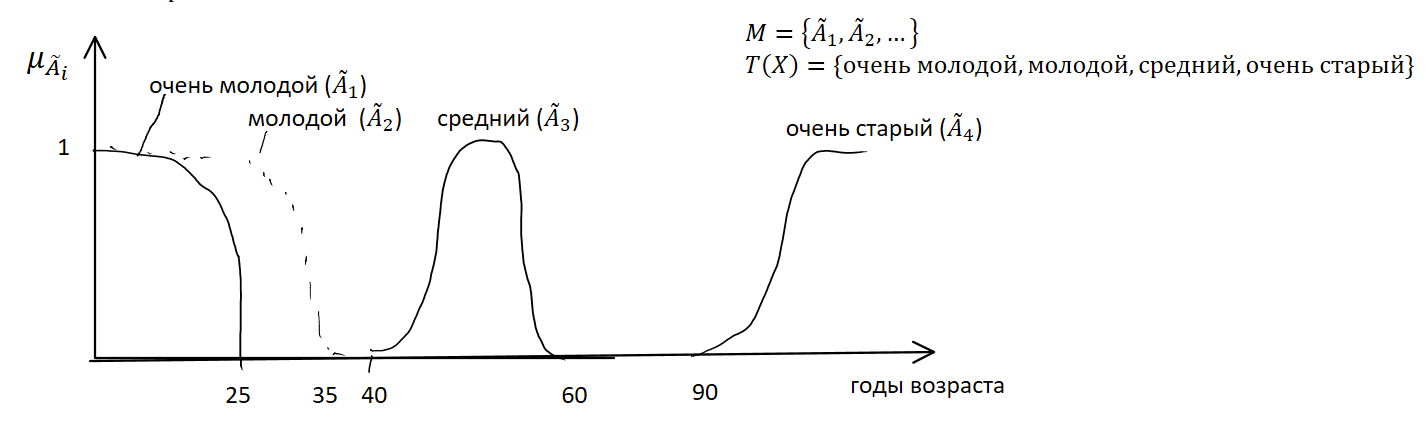
G - синтаксические правила, порождающие множество переменных.

M - семантические правила, которые каждому значению переменной ставят в соответствии смыл, то есть характеристическую функцию.

Множество T,U - четкие, а значения нечеткие.

Пример:

X – возраст



Вывод: Лингвистическая переменная содержит в себе множество нечетких множеств, каждая из которых со своей функции принадлежности отвечает за одно значение лингвистической переменной.

Значение истинный для каждого значение лингвистической переменной определяется по функции принадлежности нечеткому множеству.

"очень молодой" с

"молодой" с

Нечеткая логика построенная на лингвистических переменных называется нечеткой в широком смысле, так как приближает формальное рассуждение к рассуждению на естественному языке.

# Модальные логики

Основаны на включения в логику рассуждений так называемых модальностей, то есть истинности тех или иных высказываний в большей или меньшей степени

Традиционно рассматривают две модальности - необходимость и возможность.

Еще есть доказано, разрешено, достаточно, требуется, приказано.

## Алетическая модальная логика

(от гр. )

В ней присутствуют все обычные логические связки и помимо них, ведены два унарных оператора: оператор необходимо , и оператор возможно

Модальности выражаются друг через друга

□

Отношения модальности могут быть выражены с помощью модального шестиугольника (Я.А. Слинин), который позволяет строить связи между различными модальностями.

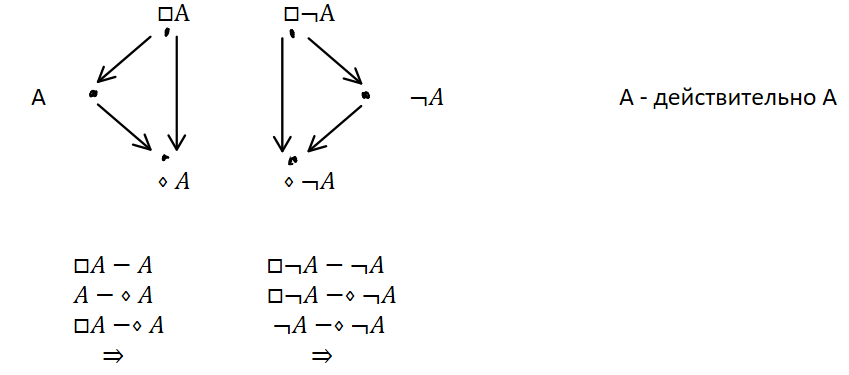
## Виды модальных логик

* Представительные модальности
  + Рекомендуется, обязательно, возможность, имеет право.
* Запретительные
* Обязывающие

Отношения модальностей можно определить с помощью шестиугольника Слинина Я.А.

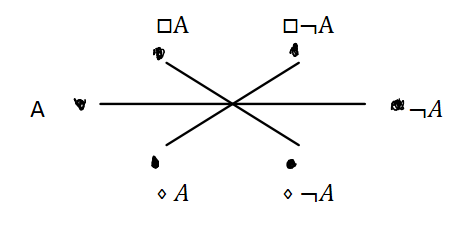
## Отношения модальностей

### Отношение подчинений



Необходимость подчиняет себе действительность и возможность

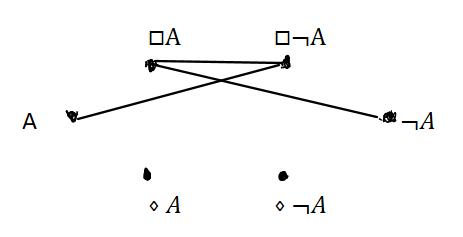
### Отношений противоречий



A: Необходим свет

Для фотосинтеза, необходим свет

### Отношения контрарности

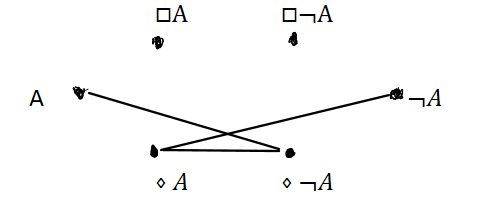


Общее правило контрарности - связанные контрарностью предложения модальной логики могут быть одновременно ложными, но не могут быть одновременно истинными.

Пример: Необходимо всем студентом изучать Французский язык - Необходимо всем студентом не изучать Французский язык (пример двух одновременно ложных)

Действительно, для жизни необходима вода - необходимо, чтобы для жизни не была необходима вода. (не могут быть одновременно истинными)

### Субконтрарность



Из ложности первого суждения логический следует истинность второго. Однако, из истинности первого суждения, ничего нельзя сказать об истинности второго.

Пример A(x) - х является птицей

B(x) - x летает

пусть (множество птиц)

- вообще говоря ложное на множестве Р (вершина А)(False)

(вершина )(True)

Пример А – Возможно, что свидетель говорит правду. Предположим, что это истинное высказывание, тогда ничего нельзя сказать об истинности высказывания «свидетель не говорит правды», «возможно, что свидетель не говорит правду.

Возможно, что свидетель говорит правду (). Свидетель говорит не правду ().

Пример Дело братьев скитских

* Есть разновидность алетической модальной логики, где необходимость и возможность трактуется как обязательно и разрешено (Деонтическая логика ). Используется для формализации инструкций, распоряжений и т.д.

* Пример - дежурный оператор обязан находиться на рабочем месте.

Разрешается отставлять рабочее место только при замене другим оператором.

* Пример - каждая переменная в программе должна быть объявлена до первого обращения к ней

Разрешается объявлять переменные не только в начале, но и в теле программы.

* Эта логики характеризует действия, которые присутствуют в рассуждениях, с точки зрения определенной системы норм.
* Модальные логики не имеют количественное выражение степени истинности суждения.
* При введении количественной меры - модальная логика переводиться в нечеткую

## Пример семантической интерпретации модальной логики - модель Крипки

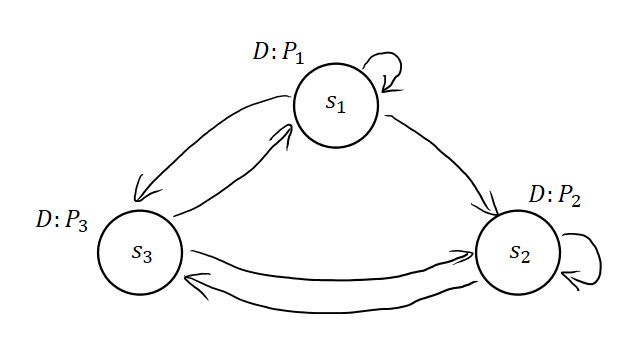
(Saul Aaron Kripke)

* Возможность в одной области не может значить возможность в другой области, а может значит.
* Ввел понятие достижимости из одной области в другой.
* Модель К. - недетерминированный конечный автомат, предназначенный для верификации модели представленных ориентированным графом, вершины которого соответствуют достижимым состояниям модели, а дуги - переходом между ними.
* Каждой вершине сопоставляется характеристическая функция L, которая для каждого состояния задает множество высказываний истинных в нем (в данном состоянии). Другими словами, это свойство, которое выполняется в данной вершины модели.
* Задается 4-х местным картежом

1. S- множество миров (областей, вариантов развития события).
2. R- бинарные отношения на множестве S (нестрогий порядок, предпорядок, отношение эквивалентности). Достижимость одного состояния из другого из
3. D - функция на множестве S,которая каждому , ставит в соответствии с , множество суждений (предложений), истинных в данной вершине
4. W - оценка параметров модели.

* Возможность в модели Крипки означает достижимость одной вершины из другой. При этом истинность возможна хотя-бы в одной вершины (мире).
* Необходимость в модели Крипки означает истинность предложения во всех вершинах (мирах).
* На практике верификации - модели Крипки разворачивается в бесконечное дерево.

* Пример



Основная проблема - экспоненциальный рост дерева развертки.

Фэйс (Face) - лучшая книга по нечёткой логики "модальная логики"

* Крипки - "семантический анализ модальной логики"
* Шютте - "полная система модальных и интуиционистских логик"

Тейз, Грибомон - "логический подход к искусственному интеллекту"

* От модальной логики, к логики баз данных

Кофман - "введение в теорию нечетких множеств"

# Темпоральные логики

Temporal logics - анл.

Вводят фактор времени в неклассические логики. Фактор времени выражается через "было, есть, будет, в то же время," и истинность предложений (суждений) может быть разным в разные моменты времени.

Пример - трамвай движется (только между остановками и если есть напряжение в сети).

Если трамвай ходит, то студент успевает, а если встал, то студент опоздает к первой паре.

Предложения имеют степень истинности

Первые модели были представленные в 54-м году Прайером.

А современные были представленные А. Пнуели.

Есть два вида темпоральных логических моделей - отсюда и две логики: линейные, и древовидные.

Линейные

* Для каждого момента времени существует единственный непосредственно следующий момент времени

Древовидные модели

* Для каждого момента времени, может быть несколько следующих моментов времени, которые понимаются, как несколько альтернатив (вариантов) возможного развития событий

Темпоральные логики используются для верификации параллельных вычислений.

Вводятся дополнительные операторы в темпоральную логику, которые его расширяют.

* **U** (until) - бинарный оператор A **U** B - до тех пор, пока. (А до тех пор, пока не наступит B. A =T, B=F B=T, A=F)
* **R** (release) - бинарный оператор A **R** B - А высвобождает В (B=T до тех пор, пока T не становиться A. А может и не стать T)
* **Х** - (next) унарный оператор **X**A - в следующий момент времени наступит событие А
* **F** (future) **F**A - А должно стать истинным к какой-то неопределенный момент в будущем.
* **A** (all) **A**B - событие В должно быть истинным на всех путях древовидной модели. **А** - квантор всеобщности пути.
* **E** (exists) **E**A - квантор существования пути. На хотя бы одном пути древовидной модели наступит событие В.
* **G** (globally) - **G**A событие А должно быть истинно во все будущие моменты времени.

Пример:

* - В какой-то момент времени в будущем наступит событие "()", когда А будет выполнятся до тех пор, пока не наступит В, или В будет истинным до тех пор, пока не наступит С.
* - на всех путях древовидной модели, то есть во всех возможных вариантах развития событий, если в следующий момент времени наступит момент С, то А будет выполняться до тех пор, пока не наступит В.
* На всех ветвях нашей программы, если в следующий момента будет ошибка, то будет запушен обработчик ошибок, до тех пор, пока не удастся передать управление программе более высокого уровня

## Виды темпоральных логик

### CTL\* - computation tree logic "star".

Является улучшением обычного CTL

* 1. Синтаксис: кванторы **A**,**E**, и все остальные операторы.
  2. Виды (типы) формул:
     1. Формулы состояний - описывают свойство некоторого состояния модели
        1. Синтаксис - Если , где р - это формула состояния, а Р - это множество предложений. **А** - квантор всеобщности пути. Если выполняется, то р - формула состояния
        2. Если формулы состояний, то - это тоже формулы состояний
        3. *Пример*: Питание подано на системный блок, и следующий момент времени запуститься процедура начальной загрузки: А. **X**B или
        4. *Пример:* А - поступаем в магистратуру. В- учимся на 4-ом курсе бакалавриата.

A **R** B

1. Формулы пути - описывает свойства некоторого пути модели (последовательность событий)
   1. Если - это формула пути, то квантификация темпоральными кванторами - это формулы пути.
   2. - формулы пути. Тогда , а так же применение всех темполарьный оператовор за исключением кванторов () также оставляет в множестве формул пути.
   3. *Пример*: A - малый свет выключен. В - яркий свет включен EXC - найдется момент, в который в следующий момент сработает сигнализация. XC - формула состояния, а EXC - формула пути.
2. Темпоральные операторы

- похоже на формулу замены квантора, хотя F,G не являются кванторами.

- правило Деморгана.

- обычная замена кванторов

Данная логика очень выразительна. Однако, высказывания в CTL\* плохо поддаются исчислениями формальными методами.

### LTL - linear time logic

* + Только выражения вида , где формула пути. Все входящие в формулы состояния являются атомарными высказываниями, связанными логическими связками.
  + *Пример* - на всех ветвях алгоритма, если встречается цикл и он четный, то количество повторений тела цикла - известно.

### CTL - computation tree logic

* + Допускает только следующие конструкции:

- a,b - без темпоральных операторах. Упрощены. А нет, выражаем через E.

* Все указанные темп. логики чаще всего применяют к верификации ПО, а также программ МК, когда их отлаживают на стадии проектировании, и для описание параллельных вычислений (древовидной модели).
* Основная проблема та же, что и модели Крипки - экспоненциальный рост числа состояний модели при её разворачивании в дерево исчислений.

# \*\*\*ТЕОРИЯ АЛГОРИТМОВ\*\*\*

# Понятие алгоритма

Мухамед Бен Муса Аль Харезм - 9 век, Перс.

О Интуитивное понятие алгоритма: алгоритм - это точное предписание или эффективная процедура, определяющая вычислительный процесс для произвольных исходных данных, который направлен на получение однозначного результата, соответствующее исходным данным.

* Определение математический не строгое. Строгость появилось только с появлением не вычислительных областей математики, а именно неевклидовая геометрия и формальных теорий. Появилась теория (мета математика). В её недрах возникла теория алгоритмов.
* Она (теория алгоритмов) дала четкое доказательство алгоритмической неразрешимости ряда проблем.
* Она дала требования, которым должен отвечать алгоритм.

## Требования к алгоритмам

1. Алгоритм должен применяться к исходным данным и выдавать результат. Исходные данные должным быть при этом корректны.
   * Данные представляют собой конечный набор объектов записанных в исходном алфавите и должен присутствовать конечный набор правил построение сложных объектов из простых.
2. Данные должны размещаться в памяти. Модель, которая имитирует алгоритм обладает памятью.
   * Память однородная, дискретная и бесконечная. Понятие ячейка памяти - в каждой ячейкой может храниться только один символ исходного алфавита, или никакой символ - то есть храниться пустой символ
3. Алгоритм должен состоять из последовательности элементарных действий или шагов
   * Последовательность этих элементарных шагов - конечная.
4. Порядок шагов должно быть детерминированным
   * За каждым элементарным шагом указывается следующий, либо дается команда завершения алгоритма
5. Алгоритм должен отвечать требованиям результативности.
   * Остановка алгоритма должна происходить после конечного числа шагов, с однозначным указанием, что является результатом.
   * Результат должен обладать свойством сходимости. Хотя метода проверки сходимости произвольного алгоритма на произвольных исходных данных не существует.

Применительно к алгоритмам, определяют следующие понятия

* Механизм реализации алгоритма - применение инструментальных средств для написание кода
* Описание алгоритма - процесс отладки программного кода. Также кодирование, но с процессом отладки и тестирования.
* Процесс реализации алгоритма - выполнение программного кода.

## Формализация понятия алгоритма

Пусть А - алгоритм. Совокупность объектов, к котором применим А, называется областью применимости алгоритма.

О Говорят ,что алгоритм А вычисляет функцию , если его область применимости совпадает с областью определения функции , и всякий элемент х из области применимости алгоритма перерабатывается им в .

О назывется вычислимой, если существует вычисляющий её алгоритм.

Под алгоритмом можно понимать процесс на области применимости которого определена вычислимая этим процессом функция.

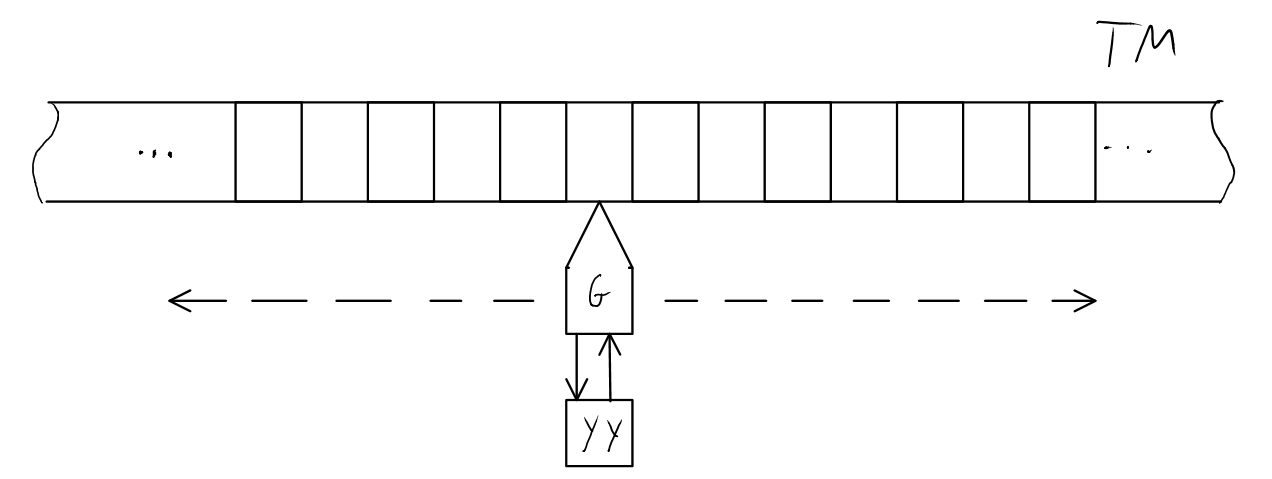
И это определение не вполне математически строгая, так как не определено вычислимость(?)

Четкое определение появилось только при разработки рекурсивной функции.

Обобщение этой модели было достигнуто на уровне частично-рекурсивных функций.

1. Рекурсивные функции
   * Первая попытка формализовать понятие алгоритма.
   * Сводит понятия алгоритма к вычислениям числовых функций, как наиболее традиционным мат. понятием
2. машины Тьюринга (машина Поста)
   * Представляют алгоритм как абстрактное детерминированное устройство, которое в каждый дискретный момент времени выполняет какую-то примитивную операцию, что обеспечивает однозначность алгоритма, детерминированность последовательности его шагов, и автоматический обеспечивает требования к результативности
3. Преобразование слов в конечных алфавитах (два вида).
   * Канонические системы Поста
   * НАМ - Нормальные Алгоритмы Маркова.
   * Подстановки и замены под-слов другими словами. Максимально абстрактная модель, применима к произвольной предметной области. Еще и является очень наглядной.

# Машина Тьюринга (Turing Machine)



Состав:

* Память - бесконечная лента, разделенная на ячейки. В каждой ячейки может быть записан один символ конечного алфавита , либо пустой символ В (blank).
* Исходные данные - слова в конечно алфавите, которые на ленте.
* Универсальная головка (G) в каждый момент дискретной времени обозревает только одну ячейку, считывая записанное у неё символ.
* По команде управляющего устройства (УУ), G может записать в ячейку любой символ А, смещаться пошагово влево или вправо по ленте, или оставаться на месте.

Машина Тьюринга может находиться в одном и только в одном состоянии

УУ - характеризуется конечным множество состояний в котором еще и вариативно выделяют . В каждый момент времени машина Тьюринга находиться в одном из своих состояний из Q. Из них выделяют

1. - начальное состояние
2. - заключительное состояние. В q\_z, ТМ останавливается.

Действия машины Тьюринга зависит от состояние устройства и обозреваемой ячейки.

Возможны следующие действия:

1. запись в ячейку
2. перемещение на одну ячейку влево или вправо.
3. установка внутреннего состояния (переход в другое состояние из множества Q)

Детерминированность машины Тьюринга определяется следующим образом:

* + Однозначно заданы следующее состояние - состояние, в которое перейдет ТМ в следующий момент времени.
  + - символ, который над записать в ту же ячейку памяти вместо , либо В.
  + Направление сдвига (двигает голоку относительно ленты)

Работа машины Тьюринга осуществляется по программе, состоящей из команд следующего формата:

Если для какой-либо пары нет команды, то машина Тьюринга останавливается. Машина Тьюринга останавливается как алгоритм (завершается) только тогда, когда выходит на состояние , либо для которого нет команды в программе ТМ.

Программа машины Тьюринга записывается как последовательности команд, либо в табличном виде, где строки - состояние, а столбы - символы алфавита.

Таблица:

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Так как нет в таблице, остановка возможна только при попадании на пустую клетку таблицы.

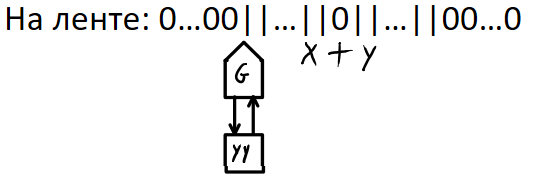
Программа:

И таблица и программа оба являются способами построения машины Тьюринга.

## Численная машина Тьюринга

ТМ, алфавитом которой является множество называется численным. При этом 0 - это пустой символ, а ("палочка") . Предназначена для работы с натуральными числами и с нулем. , , , (n+1 раз)

Построим машину Тьюринга для сложение двух чисел. Она будет вычислять алгоритм



Если бы мы складывали числа 2 и 3:

2 = |||, 3 = ||||, |||0||||. Если на место 0 записать |, то получим число 7 (||||||||), а не 5(||||||). Поэтому, при сложении надо стереть две «палочки» для получения правильного ответа. Данный алгоритм можно реализовать на следующий машине Тьюринга:

(1=| для простоты записи)

## Применимость и конфигурация машины Тьюринга

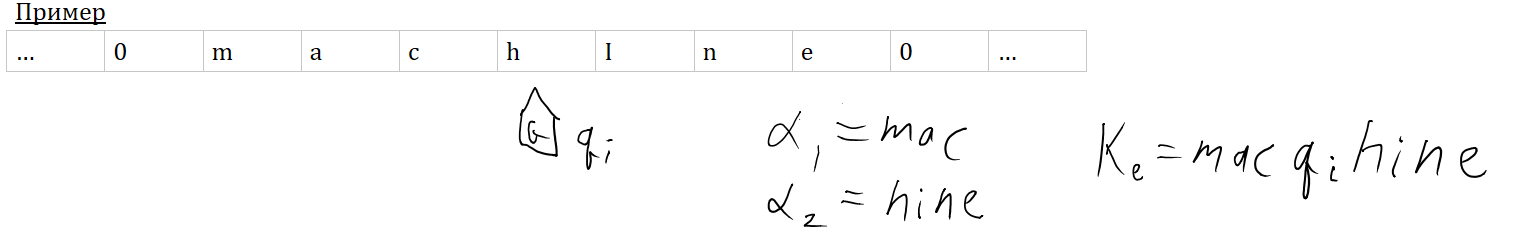
Пусть - исходное слово на ленте

Машина Тьюринга, начав работать на слове либо:

1. Останавливается через конечное число шагов
   1. Результатом работы машины на основе будет некоторое слово () и говорят, что машина Тьюринга Т применима к слову
2. Не останавливается никогда
   1. Конечного слова на ленте не формируется, машина Тьюринга не останавливается, и говорят, что машина Тьюринга не применима к слову .

## Полное состояние машины

Полное состояние машины - внутреннее состояние устройства управления, текущее слово на ленте и положение универсальной головки G. Также называется машинным словом, или конфигурацией машины Тьюринга. Обозначают конфигурацию - - слово на ленте слева от G, - слово начинаю с текущей ячейки, и до конца крайнего правого или пустого символа. Слева от и справа от нет непустых символов



  - начальная конфигурация

Если машина применима к

, где

Если недостижимо или отсутствует, но машина останавливается, то в заключительную конфигурацию входит текущее состояние и слово на ленте в момент останова.

В всякой не заключительное состояние соответствует ровно одна команда машины Т, которая (команда) переводит машину в следующую конфигурацию .

Пусть существует начальная и заключительная конфигурация, тогда для них можно построить цепочку =

эта цепочка однозначно определяется исходной конфигурацией и полностью описывает работу машины начиная с

Конечную такую последовательность называют Тьюринговым вычислением, и говорят, что заключительно выводимо из .

Пример

Т:

данная машина не применима не к какому слову из А, так как никогда не остановиться не на каком в данном алфавите.

Т:

Такая машина применима к любому слову алфавита А, и существует Тьюринговое вычисление.

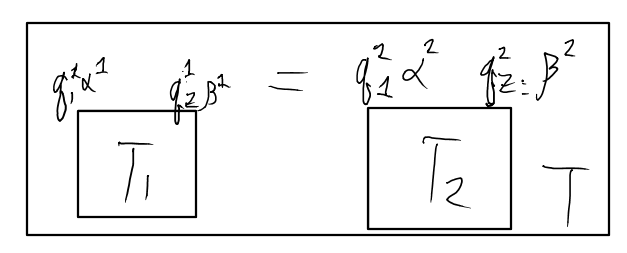
## Понятия функции, вычислимой по Тьюрингу.

Пусть дана некоторая машина Т, которая работает в алфавите . Пусть - функция, которая отображает множество слова в исходном алфавите в множество слов в результирующем алфавите. Тогда, если это слово в исходном алфавите, а слово в результирующем алфавите, то можно записать, что . Тогда для любых таких, что - сущестует Тьюринговое вычисление от конфигурации начальной, до заключительной. Всякая машина Тьюринга, которая начинает от исходной конфигурации и останавливается в заключительной конфигурации можно поставить в соответствии вычисляемую этой машиной функции . Эту функцию и называют вычислимой по Тьюрингу. Если две машины работающие в одинаковом исходном алфавите вычисляют одну и ту же функцию , то эти машины называют эквивалентными.

## Композиция машины Тьюринга

По аналогии с композиции функции можно ввести понятия композиции машин Тьюринга, при том, что , функции, вычислимые по Тьюрингу.

Т. Если вычислимы по Тьюрингу, то их композиция также вычислима по Тьюрингу.



## Универсальная машина Тюринга

Анализ работы любой машины Тюринга представляет собой также алгоритм воспроизводящий функционирования конкретной машины.

Словесно, можно записать этот алгоритм следующим образом: Для текущей конфигурации найти в системе команд, команду с левой частью . Если правая часть этой команды имеет вид , то в текущей конфигурации заменить . Получится конфигурация . Обозреваем теперь первый символ . Если правая часть команды имеет вид то заменить в текущей конф. Получится конфиг .

Словесное описание может быть заменено более строгим - формальным. Можно поставить задачу о построении машины Тюринга, реализующий описанный выше алгоритм. Такая машина называется универсальной машиной Тюринга.

На пример: для произвольной машины Тюринга, вычисляющей функции одного аргумента. Постановка задачи будет следующее: построить машину Тьюринга U, вычисляющую функцию двух аргументов, такую, что для любой машины Т с системой команд : где - исходное слово, к которому применима машина Т. То есть - машина остановиться, начав работать на слове . Или не останавливается, если не останавливается Т, работающее на слове . То есть когда Т не применима к

Любая машина U обладающая этим свойством и есть универсальная машина Тюринга. Указанную формулировку можно применить к любому количеству параметров

## Проблема при построении универсальной машины Тюринга

Как и любая ТМ, она должна работать в , она может не воспринимать команды из системы и слово записанное в алфавите машине Т. Выход: кодировать алфавит машины Т и состояние машины Т в состояния и алфавит машины U, и следовательно система команд тоже кодируется

## Тезис Тюринга

Для всякой вычислимой функции, можно построить машину Тюринга.

или же:

Всякий алгоритм может быть реализован машиной Тюринга.

Машина Тюринга нужная таким образом для доказательства существования алгоритма, то есть возможности алгоритмического решения задач.

1. Доказать Тезис Тюринга невозможно в силу того, что понятие алгоритма ставим на уровень понятия машины Тюринга.

При этом, тезис имеет фундаментальный характер.

1. Тезис Тюринга относится к реальным алгоритмам применимо с той же адекватность, с которой относятся математические модели соотносимы с реальными физическими объектами.
2. Подтверждение Тезиса Тюринга
   1. Реальная математическая практика.
   2. Описание алгоритма в терминах любой другой известной алгоритм. модели может быть сведено к описанию машиной Тюринга.
3. Тезис Тюринга позволяет преодолеть неточное определения вида: "существует эффективный подход…", заменив их точным утверждением о существовании машины Тюринга. С другой стороны, утверждение о не существовании машины Тюринга можно истолковывать, как утверждение о не существовании алгоритма решение данной задачи вообще.
4. Нельзя утверждать, что всякая теория алгоритмов может быть сведена к теорией машин Тюринга. Результаты верные в одной модели могут быть неверные в другой. К примеру - теория вычислительной сложности.

## Проблема остановки машины Тюринга

Одним из требований - результативность (если есть произвольный алгоритм а, и произвольное исходное слово , то приведет ли работа а над к конечному результату за конечное время). Над построить агоритм (если за конечное время, конечный результат не достижим.). То есть построить ТМ , которая, для любой ТМ Т и дасть значение истино, если Т останавливается, либо дает занчение ложно, если не останавливается.

Задача в указанной формулировки и называется проблемой остановки ТМ. Она очень похоже на формулировку задачи построения универсальной машины Тюринга.

Т. Не существует ТМ , решающей проблему остановки произвольной ТМ Т.

Так как алгоритмической неразрешимой оказывается проблема остановки ТМ, то проблема определения результативности алгоритмов также является алгоритмически неразрешимой. Невозможно построение универсального алгоритма.

Существуют и другие виды ТМ, помимо классической, которую мы рассмотрели. К примеру - трех-ленточная, много-ленточные (параллельные процессы), RAM.

# Рекурсивные функции

О вычислимая функция, определенная на множестве целых и неотрицательных чисел, и содержащая обращение к самой себе.

Элементарные функции:

1. функция константы
2. функция следования
3. семейство функций тождества

Правила

А. Операция суперпозиции - подстановка одних рекурсивных функций в других в качестве аргумента.

\ /

Б. Операция примитивной рекурсии

\* - схемы примитивной рекурсии, где - переменные рекурсии

Рекурсия введется фактический по переменной у, и в момент применения рекурсии все остальные параметры зафиксированы.

Частный случай, когда у функции аргументов нет вообще

## Примитивно-рекурсивные функции

О Функция называется примитивно-рекурсивной, если она может быть получена из константы, функции следования и функции тождества () при помощи конечного числа применения операций суперпозиции и примитивной рекурсии.

Вывод: вычислимая, примитивно рекурсивная.

ДЗ:

## Частично-рекурсивные функции

Среди рекурсивных функций есть не полностью определенные, или так называемые частично-рекурсивные функции. Если к не полностью определенным функциям применять выше описанный алгоритм, то будут порождаться вновь, не полностью определенные функции. Характер неопределенности может быть сложным.

Понятие частичной рекурсии обобщает понятие примитивной рекурсии и является более сложной по отношении к ней. В частности, помимо суперпозиции и примитивной рекурсивности, вводится третья операция - -оператор, который применяют к предикатом. Истинность некоторого предиката связан с выполнением некоторого равенства, и наоборот.

В ряде случаев, невозможно ничего сказать об истинности данного предиката.

Понятие частичной-рекурсии оказалось исчерпывающе в формализации (теории) вычислимой функции.

В частном случае: Если частично рекурсивная функция всюду определена, то это просто рекурсивная, или общерекурсивная функция.

Общими рекурсивными функциями можно покрыт все множество рекурсивных функций, для которых существует алгоритм их вычисляющий.

Механизм проявления неопределённости в частичной рекурсивности, такой же, как и в машинах Тьюринга, то есть в случае неопределенности, вычисления не останавливаются.

## Тезис Чёрча - связь рекурсивных функций с машиной Тьюринга

Тезис Чёрча в теории рекурсивных функций является аналогом тезиса Тьюринга в машине Тьюринга

1. Всякая функция вычислимая некоторым алгоритмом является частично рекурсивной. Данный тезис является не доказуемым.

Из сопоставление двух тезисов, следует две доказуемые теоремы:

1. Всякая частично-рекурсивная функция вычислима на машине Тьюринга.

(Всякая функция вычислимая на машине Тьюринга частично-рекурсивная.)

(Функции вычислимые на машине Тьюринга, являются вычислимой по Тьюрингу )

1. Функция вычислима по Тьюрингу тогда и только тогда, когда она частично-рекурсивна.

Класс функций вычислимых по Тьюрингу, оказывается эквивалентным классом частично-рекурсивных функций, что косвенно подтверждает недоказуемость тезисов Тьюринга и Чёрча.

Всякий алгоритм, описанный в терминах частично-рекурсивных функций, может быть реализован машиной Тьюринга. И наоборот: всякая машина Тьюринга вычисляет частично рекурсивную функцию, а следовательно, любые утверждения о (не) существований алгоритмов, сделанные в одной или в другой модели взаимно верны, и подобные утверждения можно формулировать об алгоритмов вообще, не привязываясь к контексту той или иной модели.

Тогда возможно изложение теории алгоритмов:

а) инвариантное, как способ определения понятия алгоритма (исключая качественные свойства, как сложность)

б) инвариантное к определению вычислимой функции (например - общерекурсивную функцию можно считать всюду определенной вычислимой функцией)

# Вычислимость и разрешимость

## Нумерация алгоритмов

Множество всех алгоритмов счетное - факт очевиден. Так как любой алгоритм можно задать конечным описанием в фиксированным алфавите, а множество слов в любом фиксированном алфавите - счетное. Следовательно, существует соответствие между множеством всех алгоритмов и ряда натуральных чисел.

функция назывется нумерации алгоритмов, где n-номер алгоритма А в нумерации

Из взаимной-соответствия следует и существования обратной функции .

Алгоритмическая неразрешимость

Понятие нумерации позволяет работать с алгоритмами как с числами. Например задачу о построении универсальной машины Тьюринга, можно сформулировать в инвариантном виде как задачу о построения алгоритма .

Т. Существует алгоритм , такой, что для любого алгоритма А, с номером верно, что то есть для исходных данных у и заданной нумерации ,универальный алгоритм U становиться алгоритмом A. Аналогично, в инвариантном виде можно сформулировать теорему об алгоритмической неразрешимости проблемы остановки машины Тьюринга.

Т. Не существует алгоритма , который по номеру х произвольного алгоритма А и исходным данным у, давал бы ответ на вопрос: остановиться ли алгоритм А на данных у, или нет.

В частном случае, когда на ленте машины Тьюринга записана её собственная система команд, проблема остановки равна проблеме само-применимости машины Тьюринга и в инвариантном виде её можно сформулировать в виде следующей теореме: проблема само-применимости алгоритмов алгоритмический неразрешима.

Т.

О Задача, для которой не может быть построен алгоритм называется алгоритмический неразрешимым. Это означает, что для этой задачи нельзя получить решение путем вычисление частично рекурсивной функции или построения машины Тьюринга.

Примеры: проблема само-применимости алгоритмов, остановка машины Тьюринга, проблема распознавание выводимости в математической логики - то есть в заданной формальной теории выводима ли некоторая произвольная формула, записанная в её терминах, проблема разрешимости в радикалах, в степени выше 4-ой.

Вывод: неразрешимость обычно является следствием чрезмерно-общей постановки задачи. То есть говорит о невозможности решения всех задач для данного класса одним универсальным методом.

## Разрешимые множества

Множество М называется разрешимым, если существует алгоритм который для любого объекта дает ответ о его (не) принадлежности М.

Множество всех тавтологий логики высказываний разрешимо.

Метод таблиц истинности является разрешающим алгоритмом.

Алгоритмическая неразрешимость может быть сведена к проблемам само-разрешимости

## Теорема Райса

Т. Райса: Никакое нетривиальное свойство вычислимой функции не является алгоритмически разрешимой.

Пусть С- любой класс вычислимых функций одной переменной. Класс С нетривиален в том смысле, что существует функции как принадлежащие, так и не принадлежащие С

Т. Райса: не существует алгоритма, который по номеру х, функции определял бы: принадлежит классу С, или нет. То есть множество номеров х неразрешимо. Неразрешимо и множество нумераций

Из Т. Райса следует, что из номера вычислимой функции нельзя узнать, какого она характера. Тогда смысл теоремы Райса в том, что по описанию алгоритма, то есть его нумерации, невозможно установить, какими свойствами обладает вычислимая этим алгоритмом функция.

По Т. Райса алгоритмически не разрешима задача: вычислима ли каким-либо алгоритмом функция, обладающая заданным нетривиальным свойством. Нетривиальность - есть те у которых есть это свойство, а есть те, у которых нет.

По Т. Райса алгоритмически неразрешима проблема эквивалентности двух алгоритмов. По описанию двух алгоритмов, невозможно определить одну и ту же функцию они вычисляют, или нет.

Нельзя сказать одно и то же будут ли делать две программы, написанные разными программистами с одинаковым заданием, пока мы их не запустим

Выводы:

1. Неразрешимость это не недостаток, а научный факт.
2. Отсутствие общего алгоритма не означает его отсутствия в конкретных частных случаях.