Коллоквиум по дискретной математике 2

Ми (@technothecow)

Содержание

1 Логика и машины Тьюринга

1.1 Структуры и сигнатуры. Нормальные структуры. Изоморфизм структур.

Структура – кортеж множеств $(M, \mathcal{F}, \mathcal{R}, \mathcal{C})$, где

1. M – непустое множество, *носитель структуры*
2. $\mathcal{F}$ – множество функций вида $f\colon M^n \to M$
3. $\mathcal{R}$ – множество кортежей из M
4. $\mathcal{C}$ – подмножество M

Сигнатура – кортеж попарно непересекающихся множеств $(Fnc, Prd, Cnst)$, где Fnc – множество функциональных символов, Prd – непустое множество предикатных символов и $Cnst$ – множество константных символов. (просто набор символов)

*σ-структура (или интерпретация сигнатуры σ) – это формально кортеж $\mathcal{M} = (M, \mathcal{F}, \mathcal{R}, \mathcal{C}, \mathcal{I})$, где $\mathcal{I}(Fnc) = \mathcal{F},\ \mathcal{I}(Prd) = \mathcal{R}$ и $\mathcal{I}(Cnst) = \mathcal{C}$. Вводим обозначения: $\mathcal{I}(Fnc) = f^{\mathcal{M}}$, $\mathcal{I}(Prd) = R^{\mathcal{M}}$ и $\mathcal{I}(Cnst) = c^{\mathcal{M}}$. Для задания σ-структуры достаточно только M и $\mathcal{I}$.

Нормальная структура – содержащая двувалентный предикатный символ "=" $:= \{(a, a) \in M^2 \mid a \in M\}$, где M – носитель структуры.

Изоморфизм структур: интепретации $\mathcal{M}$ и $\mathcal{N}$ сигнатуры σ с носителями M и N соответственно изоморфны если существует биекция $\eta\colon M \to N$ для которой выполняются следующие свойства:

1. $\eta(f^{\mathcal{M}}(a_1, \ldots, a_n)) = f^{\mathcal{N}}(\eta(a_1), \ldots, \eta(a_n))$
2. $(a_1, \ldots, a_n) \in R^{\mathcal{M}} \iff (\eta(a_1), \ldots, \eta(a_n)) \in R^{\mathcal{N}}$
3. $\eta(c^{\mathcal{M}}) = c^{\mathcal{N}}$, где c – один символ

1.2 Формулы первого порядка данной сигнатуры. Параметры (свободные переменные) формулы. Предложения.

Формулы первого порядка – это выражения в логике первого порядка (предикатной логике), построенные по правилам синтаксиса, установленным для данной сигнатуры.

Формулы первого порядка строятся из термов и предикатов, используя логические связки и кванторы. Основные элементы синтаксиса формул первого порядка:

1. Термы: переменные, константы и функции, примененные к термам.
2. Атомарные формулы: предикаты, примененные к термам.
3. Сложные формулы: атомарные формулы, соединенные логическими операциями $(\neg, \wedge, \vee, \to, \leftrightarrow)$ и кванторами $(\forall, \exists)$.

Свободные переменные формулы – это переменные, которые не находятся под действием кванторов ($\forall$ или $\exists$) внутри этой формулы. То есть, они не "связаны" кванторами и могут принимать любые значения из области определения. Множество свободных переменных в формуле φ обозначается как $FV(\varphi)$.

Предложения в логике первого порядка – это формулы, которые не содержат свободных переменных, то есть все переменные в них связаны кванторами. Такие формулы имеют логическое значение (истинность или ложность) в интерпретации.

1.3 Оценка переменных. Значение терма и формулы в данной структуре при данной оценке. Независимость значения формулы от значений переменных, не являющихся ее параметрами.

Оценка переменных – способ присвоения конкретных значений переменным в формуле. По сути это функция μ, которая ставит в соответствие *каждой* переменной какое-то значение.

Значение терма t и формулы φ в данной структуре $\mathcal{M}$ при данной оценке μ:

1. если t – переменная, то t принимает значение $\mu(t)$
2. если t – константный символ c, то t принимает значение интерпретации c в $\mathcal{M}$: $c^{\mathcal{M}}$
3. если t – функция f, применяемая к термам $t_1, \ldots, t_n$, то значение t – это $f^{\mathcal{M}}(v_1, \ldots, v_n)$, где $v_1, \ldots, v_n$ – это значения термов при данной оценке

4. если φ – атомарная формула $P(t_1, \ldots, t_n)$, то она истинна, если $(v_1, \ldots, v_n) \in R^{\mathcal{M}}$, где $v_1, \ldots, v_n$ – это значения термов при данной оценке

5. для сложных формул φ используются стандартные логические правила

Независимость значения формулы от значений переменных, не являющихся ее параметрами означает, что если мы изменим значения переменных, которые не являются свободными в данной формуле, то значение формулы останется неизменным. Другими словами, переменные, не являющиеся *свободными* в формуле, не влияют на ее истинностное значение.

1.4 Значение терма и формулы на наборе элементов структуры. Выразимые в структуре множества (отношения, функции, элементы). Примеры выразимых множеств.

Значение терма или формулы $\alpha(x_1, \ldots, x_n)$ на наборе элементов $y = (y_1, \ldots, y_n)$ структуры $\mathcal{M}$ определяется значением функции $\alpha^{\mathcal{M}}(y) = [\alpha](\pi + (x_1 \to y_1) + \ldots + (x_n \to y_n))$, где π – любая оценка.

Выразимые в структуре $\mathcal{M}$ множества – это множества $D \subseteq \mathcal{M}$, которые можно описать с помощью формул логики первого порядка

Примеры:

1. пустое множество: $\varphi(x) = (x \neq x)$

2. носитель структуры $\mathcal{M}$: $\varphi(y) = (y = y)$

3. четные числа: $\varphi(z) = \exists a(a \in \mathbb{N} \wedge a + a = z)$

Выразимые в структуре предикаты – это предикаты, для которых существуют эквивалентные формулы логики первого порядка

1.5 Значение формулы при изоморфизме структур. Элементарная эквивалентность структур. Изоморфные структуры элементарно эквивалентны.

*Если σ-предложение φ истинно в $\mathcal{M}$, то это обозначается так: $\mathcal{M} \models \varphi$

*Теория в языке сигнатуры σ – это какое-то множество σ-предложений.

*Модель теории T в языке сигнатуры σ – это такая σ-структура $\mathcal{M}$, что все предложения в ней истинны.

*Модель предложения φ в языке сигнатуры σ – это модель теории $\{\varphi\}$.

*Теория σ-структуры $\mathcal{M}$ – это все σ-предложения, истинные в $\mathcal{M}$. Обозначение: $Th(\mathcal{M})$.

Элементарная эквивалентность структур: σ-структуры $\mathcal{M}$ и $\mathcal{N}$ эквивалентны если $Th(\mathcal{M}) = Th(\mathcal{N})$. Обозначение: $\mathcal{M} \equiv \mathcal{N}$

Значение формулы φ при изоморфизме η структур $\mathcal{M}$ и $\mathcal{N}$: для любого $a \in M^n$ равносильны $\mathcal{M} \models \varphi(a)$ и $\mathcal{N} \models \varphi(\eta(a))$.

Элементарная эквивалентность изоморфных структур: изоморфные структуры элементарно эквивалентны.

TODO: дополнить доказательствами два последних утверждения

1.6 Значение формулы при изоморфизме структур. Сохранение выразимых множеств автоморфизмами структуры. Примеры невыразимых множеств.

Значение формулы φ при изоморфизме η структур $\mathcal{M}$ и $\mathcal{N}$: для любого $a \in M^n$ равносильны $\mathcal{M} \models \varphi(a)$ и $\mathcal{N} \models \varphi(\eta(a))$.

Сохранение выразимых множеств автоморфизмами структуры: семейство выразимых множеств сохраняется между автоморфизмами

Примеры невыразимых множеств: множество всех простых чисел (для этого необходимо проверять все возможные делители); множество натуральных чисел, являющихся степенью двойки (для этого требуется, например, рекурсия, которой нет).

TODO: дополнить доказательствами

1.7 Эквивалентность формул первого порядка. Лемма о фиктивном кванторе. Общезначимые и выполнимые формулы. Квантор всеобщности и общезначимость.

Эквивалентность формул первого порядка: формулы φ и ψ являются эквивалентными, если их значения совпадают в любой интерпретации при любой оценке. Обозначение $\varphi \equiv \psi$.

Лемма о фиктивном кванторе: пусть x не лежит в множестве свободных переменных формулы φ, тогда $\varphi \equiv \forall x \varphi$

Общезначимая формула – формула, истинная при любой интерпретации и оценке.

Выполнимая формула – формула, для которой существует интерпретация и оценка, в которой она истинна.

Квантор всеобщности и общезначимость: формула φ общезначима $\iff$ формула $\forall y \varphi$ общезначима

1.8 Основные эквивалентности логики первого порядка. Замена подформулы на эквивалентную.

Основные эквивалентности логики первого порядка для произвольных φ и ψ:

1. Пусть x не является параметром ψ, тогда $\forall\{\exists\}x(\varphi \wedge \{\vee\}\psi) \equiv \forall\{\exists\}x\varphi \wedge \{\vee\}\psi$ (итого 4 равенства)
2. $\forall x(\varphi \wedge \psi) = \forall x\varphi \wedge \forall x\psi$
3. $\forall x(\varphi \vee \psi) = \forall x\varphi \vee \forall x\psi$
4. $\neg\forall x\varphi \equiv \exists x\neg\varphi$
5. $\neg\exists x\varphi \equiv \forall x\neg\varphi$

Пусть φ – какая-то формула, $\varphi \equiv \varphi'$, тогда замена φ на φ' эквивалентна в случаях использования логического и, или, не, импликации, "тогда и только тогда квантора всеобщности и существования.

Замена подформулы на эквивалентную: пусть $\varphi \equiv \varphi'$ и ψ' была получена путем замены вхождений φ в ψ на φ', тогда $\psi \equiv \psi'$.

1.9 Пропозициональные формулы и задаваемые ими булевы функции. Тавтологии первого порядка.

Пропозициональная формула – формула, построенная из пропозициональных переменных (простых букв) с помощью булевых связок.

Каждая пропозициональная формула задаёт булеву функцию, так как для каждого набора значений переменных (истина или ложь) формула принимает одно определённое значение (истина или ложь). То есть, если у вас есть пропозициональная формула A с переменными p и q, можно построить таблицу истинности, которая покажет значение формулы для всех возможных значений p и q.

Тавтология – это формула, истинная при любых значениях ее переменных. Любая тавтология общезначима.

1.10 Лемма о корректной подстановке.

*Терм t свободен для переменной x в формуле φ, если при подстановке терма t вместо переменной x в формуле φ не происходит никаких изменений значений других свободных переменных. Иными словами, терм t можно подставить на место x в φ без появления новой привязки переменных, которая может изменить интерпретацию формулы. Обозначение: $t - x - \varphi$.

*Замена y на x в формуле φ обозначается как $\varphi(y/x)$

Лемма о корректной подстановке: в любой интерпретации при любой оценке π для всех φ - формул, t, s - термов, и x - переменной, если $t - x - \varphi$, то выполняется:

$$[s(t/x)](\pi) = [s](\pi + (x \to [t](\pi))) \text{ и } [\varphi(t/x)](\pi) = [\varphi](\pi + (x \to [t](\pi)))$$

TODO: доказательство

1.11 Понятие корректной подстановки («терм свободен для переменной в формуле»). Пример некорректной подстановки. Лемма о корректной подстановке (без доказательства). Переименование связанной переменной. Общезначимость формул вида $\forall x\varphi \to \varphi(t/x)$ и $\varphi(t/x) \to \exists x\varphi$ в случае корректной подстановки.

см. билет 1.10

Пример некорректной подстановки: возьмем формулу $\varphi(x, y) = \forall y(P(x, y))$ и терм $t = y$. Подставляем: $\varphi(x/t, y) = \forall y(P(y, y))$. Смысл формулы изменен т.к. терм не свободен для переменной в формуле.

Переименование связанной переменной:

Лемма 1. Пусть $y \notin V(\varphi)$ (т.е. y нет в φ), тогда $\forall x\varphi \equiv \forall y\varphi(y/x)$.

Лемма 2. Для любого терма t и любой формулы φ, если $y \notin V(\varphi)$, то для любой оценки π верно: $[t(y/x)](\pi) = [t](\pi + (x \to \pi(y)))$ и $[\varphi(y/x)](\pi) = [\varphi](\pi + (x \to \pi(y)))$

1. $\forall x\varphi \to \varphi(t/x)$, если t свободен для x в φ
2. $\varphi(t/x) \to \exists x\varphi(x)$, если t свободен для x в φ

TODO: дописать доказательства

1.12 Переименование связанной переменной (без доказательства). Теорема о предваренной нормальной форме.]

Переименование связанной переменной:

Лемма 1. Пусть $y \notin V(\varphi)$ (т.е. y нет в φ), тогда $\forall x\varphi \equiv \forall y\varphi(y/x)$.

Лемма 2. Для любого терма t и любой формулы φ, если $y \notin V(\varphi)$, то для любой оценки π верно: $[t(y/x)](\pi) = [t](\pi + (x \to \pi(y)))$ и $[\varphi(y/x)](\pi) = [\varphi](\pi + (x \to \pi(y)))$

*Предваренная формула – такая, что имеет кванторы только в кванторном префиксе в начале формулы.

Теорема о предваренной нормальной форме: для любой формулы найдется эквивалентная ей предваренная.

Доказательство: индукция по построению. Разберем все случаи:

1. Если формула атомарная, то она уже предваренная.
2. Если формула начинается с квантора, то по предположению индукции заменяем формулу под этим квантором на эквивалентную предваренную.
3. Если формула начинается с отрицания, то по предположению индукции заменяем формулу под отрицанием на эквивалентную предваренную и проносим отрицание вовнутрь, переменяя кванторы.
4. Если в формуле главная связка бинарная, то по предположению индукции заменяем формулы под связкой на эквивалентные предваренные и переименовываем связанные переменные так, чтобы все кванторы можно было вынести наружу и выносим их.

1.13 Понятие теории первого порядка. Примеры содержательных теорий. Модель теории. Логическое (семантическое) следование (для теорий и предложений).

Теория первого порядка – логическая система, включающая в себя сигнатуру (набор символов, включающий константы, функции и предикаты), аксиомы (набор утверждений или формул, принимаемых без доказательств) и правила вывода (правила, по которым из аксиом и других утверждений можно выводить новые утверждения)

Примеры содержательных теорий:

1. Теория групп:
   (a) Сигнатура: бинарная операция $*$ и константа e

   (b) Аксиомы: ассоциативность, существование нейтрального элемента, существование обратного элемента

2. Теория колец:
   (a) Сигнатура: две бинарные операции: $+$ и $*$ и константы 0 и 1.
   (b) Аксиомы: дистрибутивность, ассоциативность, коммутативность, существование обратного элемента по сложению

Модель теории – это интерпретация сигнатуры, в которой все аксиомы теории истинны. Например, для теории групп это множество целых чисел с операцией сложения и нулем.

Логическое следование – отношение между формулами и теориями, которое говорит, что если истинны определенные формулы, то и другие формулы истинны.

Для теорий: Теория T логически следует из множества аксиом A, если любая модель A также является моделью T.

Для предложений: Предложение φ логически следует из теории T ($T \models \varphi$), если φ истинно в каждой модели T.

1.14 Исчисление предикатов с равенством (в гильбертовской форме). Теорема о полноте и корректности исчисления предикатов (без доказательства). Теорема о компактности в двух формах: про выполнимость теории и про логическое следование из теории.

Исчисление предикатов с равенством – это система логики первого порядка, включающая равенство как основной предикат. В гильбертовской форме исчисления предикатов используются аксиомы и правила вывода.

Аксиомы для равенства:

1. Рефлексивность: $\forall x(x = x)$
2. Симметричность: $\forall x \forall y(x = y \to y = x)$
3. Транзитивность: $\forall x \forall y \forall z(x = y \land y = z \to x = z)$
4. Замена в формулах: если t – терм, а P – предикат, то $\forall x \forall y(x = y \to (P(x) \leftrightarrow P(y)))$

Общие аксиомы и правила вывода:

1. Аксиомы логики первого порядка
2. Правило Modus Ponens: из φ и $\varphi \to \psi$ следует ψ
3. Правило обобщения: из φ следует $\forall x\varphi$, если x не свободная в φ

Теорема о полноте и корректности исчисления предикатов: если φ логически следует из A, тогда и только тогда φ выводима из A в исчислении предикатов.

Теорема о компактности: если любая конечная подсистема множества предложений имеет модель, то и все множество имеет модель.

Теорема о компактности в форме про выполнимость теории: если каждое конечное подмножество множества формул T выполнимо, то и все множество T выполнимо.

Теорема о компактности в форме про логическое следование из теории: формула φ логически следует из теории T тогда и только тогда, когда φ логически следует из некоторого конечного подмножества теории T.

TODO: дополнить доказательствами

1.15 Теорема компактности (без доказательства). Любой пример применения.

см. билет 1.14

Пример: хотим показать, что существует бесконечное множество.

Пусть T – это теория, содержащая набор формул $F = \{\varphi_n : n \in \mathbb{N}\}$, где φ_n утверждает, что в нашем множестве существует как минимум n различных элементов. Любое конечное подмножество F выполнимо в модели потому что можно найти конечное число элементов, принадлежащих множеству. Применяем теорему компактности: раз каждое подмножество F имеет модель, то и все множество F имеет модель, значит существует модель, содержащая бесконечно много элементов.

1.16 Одноленточная машина Тьюринга (допустимо неформальное определение с лентой и головкой). Сложение натуральных чисел (при унарном и бинарном кодировании).

Одноленточная машина Тьюринга — это теоретическая модель вычислений, состоящая из следующих частей: лента (бесконечная в обе стороны, разделенная на ячейки, каждая из которых может хранить один символ из конечного алфавита, который обычно содержит спец.символ "пусто": #), головка для чтения/записи (устройство, которое может перемещаться влево или вправо по ленте, считывать символы с ленты и записывать символы на ленту), множество состояний (конечное множество состояний, одно из которых является начальным, а одно или несколько могут быть конечными) и таблица переходов (определяет правила, по которым машина переходит из одного состояния в другое, в зависимости от символа под головкой)

Сложение натуральных чисел в унарном виде: очевидно

Сложение натуральных чисел в бинарном виде: пусть длина одинаковая, числа записаны в виде "[0,1]*+[0,1]* тогда сначала идем вправо до конца, ставим знак равенства, идем влево до конца, и если там 1/0, тогда помечаем символ "решеткой идем вправо до конца и после знака равно ставим 1/0, потом идем до знака плюса, берем 1/0, помечаем символ "плюсом идем вправо до конца и к последнему числу добавляем 1/0. таким образом получим запись в сломанной троичной системе счисления. осталось только перевести в бинарную

TODO: переписать с каким-нибудь нормальным алгоритмом

1.17 Многоленточная машина Тьюринга (допустимо неформальное определение с лентами и головками). Удвоение входного слова за линейное время.

Многоленточная машина Тьюринга — это расширение классической машины Тьюринга, у которой есть несколько лент и несколько головок для чтения/записи. Каждая лента бесконечна в обе стороны и содержит свой собственный алфавит символов.

Удвоение входного слова за линейное время: копируем символы пока не дойдем до решетки. Как дошли до решетки, идем на верхней ленте влево в начало слова и повторяем процедуру.

1.18 Конфигурации одноленточной и многоленточной машин Тьюринга. Меры сложности «время» и «зона» и их соотношение в обоих случаях.

Конфигурация машины Тьюринга – это описание текущего состояния машины, которое включает состояние машины, содержимое ленты (лент), позиция головки (головок).

Время выполнения (или временная сложность) алгоритма на машине Тьюринга – это количество шагов, которые машина делает для выполнения задачи. Временная сложность оценивается в зависимости от размера входных данных n.

Зона выполнения (или пространственная сложность) алгоритма на машине Тьюринга – это количество ячеек ленты, которые машина использует для выполнения задачи.

Существуют работы, которые показывают, что алгоритм, выполненный на МТ из k лент эмулируется за $T \log T$ на двуленточной МТ.

Многоленточные машины Тьюринга более эффективны по времени (например, задача удвоения входного слова) по сравнению с одноленточными машинами, так как позволяют параллельно обрабатывать несколько лент и перемещаться быстрее по необходимым данным. Однако, пространственная сложность остаётся асимптотически такой же, как и для одноленточных машин.

1.19 Сокращение ленточного алфавита и его цена.

См. страницы 21-24 в "Введении в сложность вычислений" Крупского

1.20 Сокращение числа лент и его цена.

См. страницы 24-27 в "Введении в сложность вычислений" Крупского

2 Вычислимость

2.1 Вычислимые функции (при интуитивном понимании алгоритма). Разрешимые и перечислимые множества. Связь конечности, разрешимости и перечислимости. Разрешимые множества под действием операций алгебры множеств и декартова произведения.

Вычислимая функция – это такая частичная функция $f\colon \mathbb{N} \to \mathbb{N}$, что для нее существует программа (алгоритм), которая на любом входе $x \in \operatorname{dom} f$ выписывает $f(x)$, а иначе зацикливается.

Разрешимое множество – такое множество, чья характеристическая функция (функция, которая ест элемент и выплевывает единицу если элемент в множестве и ноль иначе) вычислима.

Перечислимое множество – такое множество, для которого есть программа, которая последовательно выписывает все элементы множества и только их. Для каждого элемента множества должно существовать $k \in \mathbb{N}$, что после k-ого шага элемент будет выписан.

Связь конечности, разрешимости и перечислимости: 1) конечно, значит разрешимо; 2) разрешимо, значит перечислимо.

Доказательство: 1) конечно, значит можно пронумеровать элементы $\{a_1, ..., a_n\}$. Искомая характеристическая функция равна дизъюнкции (логическому или) булевских значений $x = a_1 \vee x = a_2 \vee \ldots \vee x = a_n$. Для пустой функции всегда возвращаем ноль, что также вычислимо.

2) перебираем все натуральные числа и выводим текущее если характеристическая функция вернула единицу

Разрешимые множества под действием операций алгебры множеств и декартова произведения: A, B – разрешимы $\implies$ разрешимы: $A \cup B, A \cap B, A \times B, \overline{A}, \overline{B}$

Доказательство: выразим характеристические функции: $\chi_{A \cup B}(x) = \max(\chi_A(x), \chi_B(x))$, и т.д.

2.2 Перечислимые множества под действием операций алгебры множеств, декартова произведения и проекции. Теорема Поста.

Перечислимые множества под действием операций алгебры множеств, декартова произведения и проекции: A, B – перечислимы $\implies$ перечислимы: $A \cup B, A \cap B, A \times B, \mathrm{pr}^i A, \mathrm{pr}^i B$.

Доказательство: перечислимость $A \cup B$: просто выводим числа по очереди; перечислимость $A \cap B$: по очереди выполняем по шагу алгоритмов A и B и когда получаем очередной элемент a_i выводим его только если нам уже попадался равный ему b_j. Аналогично поступаем с новыми элементами из B; перечислимость $A \times B$: по очереди выполняем по шагу алгоритмов для A и B и когда получаем очередной элемент a_i выписываем пары со всеми до этого полученными $b_1, \ldots, b_k$. Аналогично поступаем и для B; перечислимость проекции: просто для каждого нового $a = (a_1, \ldots, a_n)$ выводим a_i.

Теорема Поста: множество разрешимо $\iff$ его дополнение и оно само перечислимо.

Доказательство: 1) слева направо следует из леммы о связи конечности, разрешимости и перечислимости (билет 2.1)

2) справа налево доказывается с помощью следующего вычислимого алгоритма: будем выполнять по очереди по одному шагу алгоритма для множества и его дополнения. Рано или поздно в первом или втором появится наш проверяемый элемент