Билет 1. Понятие высказывания. Логические связки. Формулы логики высказываний.

Понятие высказывания.

Всякое имеющее смысл повествовательное предложение, относительно которого можно утверждать истинно оно или ложно.

Логические связки.

Простые высказывания обозначаются переменными, которые наз. высказывательными или пропозициональными, принимающими значение И или Л. Из нескольких простых высказываний с помощью логических операций (логических связок) можно составить сложное или составное высказывание.

¬,&,V, →,~ (отрицание, и, или, импликация, эквивалентность) - пропозициональные связки

P	Q	P&Q	PVQ	P→Q	P~Q	P⊕Q
И	И	И	И	И	И	Л
И	Л	Л	И	Л	Л	И
Л	И	Л	И	И	Л	И
Л	Л	Л	Л	И	И	Л

Формулы логики высказываний.

Алфавит логики высказываний содержит следующие символы:

1. Высказывательные или пропозициональные переменные: x_1, $x_{2,...}$, $x_{i,...}$
2. Логические символы: ¬, V,&,->,~, (,) (отрицание, и, или, импликация, эквивалентность, скобки)

Слово в алфавите логики высказываний – это формула, если оно удовлетворяет следующему определению:

1. Любая высказывательная переменная – формула.
2. Если А и В – формулы, то $\bar{A}$, A&B, AVB,A→B, A~B – тоже формулы. (отрицание, и, или, импликация, эквивалентность)
3. Формулами являются лишь те слова, для которых это следует из двух предыдущих условий.

Подформулой формулы А называется любое подслово А, само являющееся формулой.

Упорядоченный набор.

Упорядоченный набор высказывательных переменных $<x_1,\dots x_n>$ называется списком переменных формулы А, если он содержит все переменные этой формулы (некоторые переменные могут быть фиктивными).

Конкретный набор.

Конкретный набор истинностных значений, приписанных набору $<x_1,\dots x_n>$, назовём оценкой списка переменных или интерпретацией формулы А.

Билет 2. Равносильность формул логики высказываний. Основные равносильности.

Равносильность формул логики высказываний.

Пусть А и В - формулы зависящие от одного и того же списка переменных $<x_1,\dots x_n>$. Будем называть их равносильными, если на любой оценке этого списка они принимают одинаковые значения.

Основные равносильности.

1.Закон сохранения: A&A≡A; A→A≡И; AVA≡A; A~A≡ И

2.Закон коммутативности: A&B≡ $B\&A$; A~B≡B~A; AVB≡BVA; A→B≠B→A

3.Закон «лжи и истины»: A&И≡A; AVИ≡И; A&Л≡Л; AVЛ≡A;

4.Закон противоречия: A&$\bar{A}$ ≡Л

5.Закон исключения третьего: AV$\bar{A}$ ≡И

6.Закон двойного отрицания: $\bar{\bar{A}} \equiv A$

7.Законы ассоциативности: A&(B&C) ≡ (A&B)&C; AV(BVC) ≡ (AVB)VC; A~(B~C) ≡ (A~B) ~C; A→(B→C) ≠(A→B)→C

8.Закон дистрибутивности: A&(BVC)≡(A&B)V(A&C); AV(B&C)≡(AVB)&(AVC)

9.Закон де Моргана: $\overline{A\&B} \equiv \bar{A}V\bar{B}$; $\overline{AVB} \equiv \bar{A}\&\bar{B}$

10.Закон поглощения: A&(AVB) ≡A; AV(A&B) ≡A

11.Формулы расщепления: A≡ (A&B)V(A&$\bar{B}$); A≡(AVB)&(AV$\bar{B}$)

Конъюнкция, логическое умножение	&, ·, ∧
Дизъюнкция, логическое сложение	∨, +
Отрицание, инверсия	¬, −
Разделительная дизъюнкция, исключающее *или*, сложение по модулю 2	⊕, Δ
Импликация, следование	→, ⇒
Равносильность, равнозначность, эквиваленция	↔, ⇔, ≡, ~

12. Выражения импликации: A→B≡ $\bar{A}$VB≡ $\overline{A\&\bar{B}}$; AVB≡ $\bar{A}$ →B≡ $\overline{\bar{A}\&\bar{B}}$; A&B≡ $\overline{A \to \bar{B}} \equiv \bar{A}V\bar{B}$(отрицание над правой частью); A~B≡(A→B)&(B→A)≡(A&B)V($\bar{A}$&$\bar{B}$)≡(BV$\bar{A}$)&(AV$\bar{B}$)

Билет 3 Тождественно-истинные формулы логики высказываний. Важнейшие тавтологии. Правильные рассуждения. Утверждение о правильности рассуждения по схеме $(P_1 \dots P_n) \Rightarrow Q$

Тождественно-истинные формулы логики высказываний

Формула А- тавтология (или тождественно истинная формула или общезначимая) если на всех оценках списка своих переменных она принимает значение «И».
Формула А-выполнимая, если она истина хотя бы в одной своей интерпретации.
Формула А – опровержимая, если она принимает значение «Л» хотя бы в одной интерпретации.
Формула А-тождественно ложная или противоречивая, если она ложна на всех оценках списка своих переменных.
С точки зрения мат.логики тавтологии по сути являются логическими законами, поскольку при любой подстановке вместо переменных тождественно-истинных высказываний получаются истинные высказывания.

Наиболее важные тавтологии:

1. $A \vee \overline{A} = И$
2. $A \to A = И$
3. $(A \& B) \to A \qquad (A \& B) \to B$
4. $A \to (A \vee B) \qquad B \to (A \vee B)$
5. $A \to (B \to A)$
6. $A \to (B \to (A \& B))$
7. $(A \to B) \to ((B \to C) \to (A \to C))$ цепное рассуждение
8. $(A \to (B \to C)) \to ((A \to B) \to (A \to C))$
9. $(\overline{A} \to \overline{B}) \to ((\overline{A} \to B) \to A$
10. $((A \to B) \to A) \to A$ закон Пирса

Правильные рассуждения

При доказательстве утверждений математических рассуждений используют рассуждения, которые на языке логики можно выразить формулами. Рассуждение называется правильным, если из конъюнкции посылок следует заключение. Таким образом, всякий раз, когда посылки истины, заключение тоже истинно. Таким образом, правильность рассуждений можно установить, составив соответствующую ему формулу логики высказываний и доказать, что она является тавтологией.

Утверждение о правильности рассуждения по схеме $(P_1 \dots P_n) \Rightarrow Q$

Пусть $P_1 \dots P_n$ - посылки, Q- заключение. Тогда для определения правильности рассуждения $(P_1 \dots P_n) \Rightarrow Q$ следует установить тождественную истинность формулы $(P_1 \& \dots \& P_n) \to \text{Q}$.

Утверждение: Рассуждение по схеме $P_1 \dots P_n \Rightarrow Q$ правильное тогда и только тогда, когда $(P_1 \& \dots \& P_n)$ - тавтология.

Доказательство: 1. $\to$ необходимость

Пусть рассуждение по схеме $P_1 \dots P_n \Rightarrow Q$ - правильное. Если в нём все посылки $P_1 \dots P_n$ - истины, то Q тоже истинно, иначе исходное рассуждение не было бы правильным.

В этом случае $I((P_1 \& \dots \& P_n) \to Q) = И$ (И англ от Р) по определению импликаций. Если среди посылок $P_1 \dots P_n$ есть хотя бы одна ложная, то $I(P_1 \& \dots \& P_n) = Л$ и $I(P_1 \& \dots \& P_n \to \text{Q}) = И$ при любых Q. Следовательно, $(P_1 \& \dots \& P_n) \to Q$ - тавтология.

2. $\leftarrow$ достаточность

Пусть $(P_1 \& \dots \& P_n) \to \text{Q}$ - тавтология. Следовательно $I(Q) = И$, иначе $P_1 \& \dots \& P_n \to Q$ - не тавтология. Таким образом формула Q есть логическое следствие формулы $P_1 \& \dots \& P_n$, т.е рассуждение по схеме $(P_1 \dots P_n) \Rightarrow Q$ -правильное.

Билет 4. Проблема разрешимости в логике высказываний и методы ее решения.

Вопрос к какому классу формул (тавтология, выполнимая или тождественно-ложная формула) относится текущая формула А называют проблемой разрешимости, которая элементарно решается с помощью таблицы истинности. Однако для формул таблицы громоздкие и их использование не рационально. Другой способ основан на приведение формулы А к СКНФ или СДНФ с использованием специального алгоритма (алгоритм ниже), который позволяет определить являются ли данная формула тавтологией или нет. Одновременно решается проблема разрешимости

$f(x_1,\dots,x_n) = \cup x_1^{б1} \& \dots \& x_n^{б2}$
$f(б_1,\dots,б_n) = И$
} совершенная ДНФ

$f(x_1,\dots,x_n) = \&(\overline{x}_1^{б1} \vee \dots \vee \overline{x}_n^{бn})$
$f(б_1,\dots,б_n) = Л$
} совершенная КНФ

Правила перехода от СКНФ к СДНФ

$$\text{СКНФ } f = \overline{\text{СДНФ } \bar{f}}$$

Вышеназванный алгоритм состоит в следующем: сначала рассматривается формула А
Если $A \equiv И \Rightarrow$ тогда задача решена. Если нет, то рассматривается $\bar{A}$. Если $\bar{A} \equiv И \Rightarrow A \equiv Л \Rightarrow$ решена.
Если $\bar{A} \neq И \Rightarrow$ А-выполнима (опровержима).
Установление тождественной истинности формулы А основано на следующих теоремах:

1. Для того чтобы элементарная дизъюнкция была тождественно истинной необходимо и достаточно, чтобы в ней содержались переменная и ее отрицание: $x_i \vee \overline{x_i} \equiv И$
2. Для того чтобы элементарная конъюнкция была тождественно ложной необходимо и достаточно чтобы в ней содержались переменная и ее отрицание: $x_i \& \overline{x_i} \equiv Л$
3. Для того чтобы формула высказывания А была истинной необходимо и достаточно чтобы любая элементарная дизъюнкция, входящая в КНФ А, содержала переменную и ее отрицание.
4. Для того чтобы формула высказывания А была тождественно ложной необходимо и достаточно чтобы любая элементарная конъюнкция, входящая в ДНФ А содержала переменную и ее отрицание.

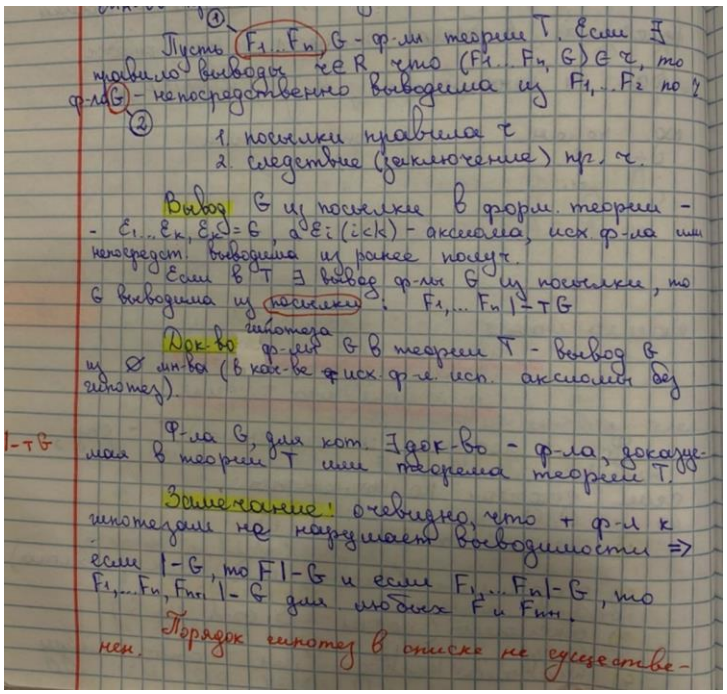
Билет 5. Определение и виды формальных теорий.

Обозначения : ≤ - «входит или равно», < - входит (про множества)\\

Формальная теория (исчисление) Т считается определённой, если заданы её компоненты:

1. Мн-во символов А – алфавит
2. F≤A* - (мн-во слов и букв алфавита А) – формулы теории
3. B<F – аксиомы теории
4. Множество отношений между формулами - R – наз. правилами вывода теории.

Алфавит А может быть конечным и бесконечным. Чаще всего под алфавитом подразумевают конечное множество букв, к которым приписываются, если нужно, натуральные числа в качестве индексов.
Мн-во формул обычно задается индуктивным опред-ем, как правило оно бесконечно.
Множества А и F образуют вместе язык (сигнатуру – формальную теорию).
Мн-во аксиом может быть конечным или бесконечным. Если мн-во бесконечно, то оно задается, как правило, с помощью конечного мн-ва схем аксиом и правил порождения конкретных аксиом из схем аксиом. Мн-во правил вывода обычно конечно.
Выводимость (фотография):



|---$_T$ G – G теорема теории Т, G доказуема в теории Т

Виды формальных теорий.

Форм. теория наз. семантически непротиворечивой, если ни одна ее теорема не является противоречием.
Форм. теория наз. формально непротиворечивой, если в ней не явл-ся выводимыми одновременно ф-лы F и $\bar{F}$.
Эти два определения эквивалентны и формальная теория семантически непротиворечива тогда и только тогда, когда она формально непротиворечива.
Исчисление наз. полным в узком смысле, если добавление к списку его аксиом любой недоказуемой в исчислении формулы в качестве новой аксиомы приводит к противоречивому исчислению.
Исчисление наз. полным в широком смысле, если любая тождественно-истинная формула в нём доказуема.
Очевидно, что полнота в широком и узком смыслах различается.
Формальная теорема наз. разрешимой, если существует алгоритм, который для любой формулы теории определяет, явл-ся ли эта формула теоремой теории.
Замечание: при изучении формальной теории имеем дело с двумя типами высказываний:

1. Высказываний самой теории (теоремы), кот. рассматриваются как формальные объекты, определенные ранее.
2. Выск-я о теории (о св-вах ее теорем, док-в и т.д), кот. формулируются на метаязыке, и наз. поэтому метатеоремами.

Различия между теоремами и метатеоремами не всегда будет проводиться явно, но его обязательно надо иметь в виду.

Например, если удалось построить вывод формулы G из F1…Fn, то высказывания F1…Fn|--G – метатеорема (|-- значит выводима). Её можно рассматривать как дополнительные правила вывода, которые можно присоединить к исходным и использовать в дальнейшем.

Ясно, что общезначимые высказывания типа AV Ā, имеющие силу общих логических законов, должны содержаться в любой теории, претендующей на логический смысл.

Билет 6. Язык, системы аксиом и основные правила вывода исчисления высказываний.

Исчисление высказываний – аксиоматическая логическая система, кот. описывает тождественно истинные логические схемы, а ее интерпретация это алгебра высказываний.

Исчисление высказываний определяется следующим образом:

1.Алфавит ИВ состоит из пропозициональных переменных (буквы латинского алфавита с индексами или без); знаков логических связок (&,¬, ∨,→); символов (,); операция следования |--- .

2. Формула ИВ – наз. слово алфавита ИВ, удов. условиям :

-пропозициональная переменная явл ф-лой, наз. элементарной или атомарной

-если A и B – формулы, то Ā, A&B, A∨B, A→B тоже ф-лы

- других формул нет!

Подформулой A формулы B ИВ наз. подслово A слова B, само являющееся формулой

3.Аксиомы.

Системы аксиом I:

I.1 A→(B→A)

I.2 (A→(B→C))→((A→B)→(A→C))

I.3 (A&B)→A

I.4 (A&B)→B

I.5 A→(B→(A&B))

I.6 A→(A∨B)

I.7 B→(A∨B)

I.8 (A→C) →((B→C)→((A∨B)→C))

I.9 (A→B)→((A→$\bar{B}$)→Ā)

I.10 (из двойного отрицания A следует A)

$\bar{\bar{A}}$ ->A

Другая система использует только две логич связки : отрицание и импликацию, при этом сокращается алфавит. Операции & и ∨ рассматриваются как сокращения, употребление кот. удобно, но не обязательно (одновременно & и ∨ исключаются из алфавита высказываний)

В этом случае A∨B≡Ā→B; A&B≡отрицание над(A→B)

В рез-те система аксиом становится компактнее

Система II.

II.1≡I.1

II.2≡I.2

II.3≡(Ā→$\bar{B}$)→((Ā→B)→A)

Приведенные системы аксиом равносильны в том смысле, что порождают одно и то же множество формул. Д-во такого утв-я состоит из д-ва выводимости всех аксиом системы II из системы I и наоборот (с учетом замечания для & и ∨).

С одной стороны, система II компактнее, и, соответственно, и более компактны док-ва ее св-в. Но в сист I короче вывод различных формул, содержащих И и ИЛИ.

Замечание : возможны и другие сист аксиом, равносильные первым двум.

4.Правила вывода

1)Правило подстановки : если α(A) - формула, содержащая A, то выводима и формула α(β), получающаяся из α заменой всех вхождений A на произвольную формулу β : α(A)/α(β)

2)Правило заключения(Modus Ponens – MP):

Если α и α→β – выводимые формулы, то β тоже выводимая : (α,α→β)/β

Замечание :

В этом описании ИВ аксиомы явл-ся формулами исчисления. А формулы α и α→β, используемые в правилах вывода – метаформулы (схемы формул).

Схема фор-л α→β обозначает мн-во всех тех формул ИВ, кот. получаются если ее метапеременные заменить формулами метаисчисления : если α зам на A, β на A&B, то из α→β получим A→(A&B).

Использование схем формул можно распространить и на аксиомы.

Билет 7. Производные правила вывода в ИВ: выводимость A→ A, правило введение импликации, транзитивность выводимости.

Система II.

II.1≡I.1 II.3≡(Ā→$\bar{B}$)→((Ā→B)→A)

II.2≡I.2

1.Теорема 1 **|---L A→ A (А импликация А доказуема из теории Л), при ∀ А. Док-во:**

1) подставим в(II.2): B | A→A, C | A : {(A→((A→A)→A))}→[((A→(A→A))→(A→A))].

2)Подставим в(II.1): B | A→A: {A→((A→A)→A)}

3) Из (1) и (2) по МодусПоненс получим выводимость [(A→(A→A))→(A→A)]

4)В (II.1) подставим B | A: A→(A→A)

5) Из (4) и (3) по правилу МодусПоненс: A→A доказуема, ч.т.д.

2.Теорема 2: **A|---L B→A (В импликация А выводится из А). Док-во:**

1) A – гипотеза по условию.

2) A→(B→A) выводима – аксиома (II.1)

3) из (1) и (2) по правилу MP получим выводимость B→A при гипотезе A.

Как отмечалось ранее, всякую доказанную в ИВ выводимость вида Г|-- α, где Г – список формул, α – отдельная формула, можно рассматривать как правило вывода: Г/ α, которое можно присоединить к уже имеющимся.

Доказанную в теореме 2 выводимость вместе с правилом подстановки можно рассматривать как новое производное правило вывода:

1. Правило введения импликации

Полученную выводимость можно вместе с правилом подстановки рассматривать как новое произв.

Правило вывода – правило введения импликации**:*α/(β→α):*** если ф-ла α выводима, то выводима и формула β→α, где β ∀ формула.

2. Транзитивность выводимости

Если Г|-- α(если из Г выводится α), α|-- β, то Г|-- β.

Это правило непосредственно вытекает из определения выводимости.

Билет 8. Производные правила вывода в ИВ: теорема дедукции (без доказательства), правило силлогизма, правило введения отрицания.

В математических рассуждениях часто доказывают рассуждение B, основываясь на верности A: «Поскольку верно A, то справедливо и B». В ИВ этот приём обосновывается теоремой дедукции: (Г, α |--L β) · (Г, |-L α→β)

Следствие из т. дедукции – правило силлогизма**:** A→B, B→C |-L A→C.

Док-во:

1) A→B- гипотеза по условию,
2) B→C – гипотеза по условию
3) A- гипотеза
4) Из (3) и (1) по правилу MP получим: A, A→B |-B, B-доказуемо
5) Применим MP для (4) и (2): B, B→C |-C, C-доказуемо
6) Из (1) - (5) имеем :A,A→B, B→C |-C
7) Из (6) по теореме дедукции: A→B, B→C |-A→C.

Докажем, что в ИВ справедлив метод доказательства от противного: Если Г,A |-B и Г,A |-$\bar{B}$, то Г|-$\bar{A}$.

Получили правило введения отрицания.

Док-во**:** 1) по теореме дедукции, если Г,A |-B и Г,A |-$\bar{B}$, то Г |-A→B и Г |→A→$\bar{B}$

2)из (1) и (I.9) ((A→B)→((A→$\bar{B}$)→Ā)) двойным применением MP получим Г |-$\bar{A}$.

Билет 9 Лемма для теоремы об общезначимых формулах исчисления высказываний.

A(a_1,…, a_n), I(A) – некая интерпретация A.

Обозначим: A_i' = { a_i, если a_i=И; $\bar{a}_i$, если a_i=Л } A' = { A, если I(A)=И; $\bar{A}$, если I(A)=Л } в данной интерпретации.

Лемма: Из гипотез $A_1', \ldots, A_n' \vdash_L A'$

Док-во:

1. Переменная формула. Пусть A=a, тогда $a \mid -_L a, \overline{a} \mid -_L \overline{a}$

2. Отрицание: пусть $A = \overline{B}$. Возможны 2 случая: а). пусть I(B)=И, тогда I(A)=Л и A'= $\overline{A} = \overline{\overline{B}}$. По индукционному предположению $A_1^{'},...,A_n^{'} \mid -_L B$. Но в исчислении высказываний есть $\mid -_L B \to \overline{\overline{B}}$ по теореме о выводимости => $A_1^{'},...,A_n^{'} \mid -_L \overline{\overline{B}} = A^{'}$ б) Пусть I(B)=Л, тогда I(A)=И и A'= $A = \overline{B}$ по индукционному предположению $A_1^{'},...,A_n^{'} \mid -\overline{B} = A^{'}$

3. Импликация. Пусть A=(B→C). По индукционному предположению из гипотез $A_1^{'},...,A_n^{'} \mid -B^{'}$ и $A_1^{'},...,A_n^{'} \mid -C^{'}$

а) Пусть I(B)=Л, тогда для ∀ C I(A)=И и $B^{'} = \overline{B}$, а A'=A, но $A_1^{'},...,A_n^{'} \mid -_L \overline{B}$ (по индукционному предположению) и по теореме о выводимости $\mid -_L \overline{B} \to (B \to C)$. => из гипотез $A_1^{'},...,A_n^{'} \mid -_L (B \to C) = A^{'}$ по правилу MP.

б) Пусть I(B)=И = I(C), тогда I(A)=И, C'=C, A'=A=(B→C). Имеем: $A_1^{'},...,A_n^{'} \mid -_L C$ (по индукционному допущению) и $\mid -_L C \to (B \to C)$ по аксиоме II.1 с подстановкой C. => по правилу MP $A_1^{'},...,A_n^{'} \mid -_L (B \to C) = A^{'}$

в) Пусть I(B)=И, I(C)=Л. Тогда $A^{'} = \overline{A} = \overline{B \to C}$, B'=B, C'= $\overline{C}$. Имеем из гипотез: $A_1^{'},...,A_n^{'} \mid -_L B$ и $A_1^{'},...,A_n^{'} \mid -_L \overline{C}$ (по индукционному предположению) и $\mid -_L B \to (\overline{C} \to \overline{B \to C})$ (по теореме о выводимости) отсюда 2 раза применяя правило заключения(MP) получим $A_1^{'},...,A_n^{'} \mid -_L (\overline{B \to C}) = A^{'}$.

Билет 10 Теорема об общезначимых формулах в исчислении высказываний

Теоремами исчисления высказываний являются тождественно истинные формулы и только они: ($\mid -_L A$) ⇔ (A-тавтология).

Док-во

1.Необходимость Аксиомы любой системы являются тавтологией. Правило заключения сохраняет тождественную истинность => теоремы исчисления высказываний это тавтология.

2.Достаточность Пусть формула A-тавтология, тогда $A_1^{'},...,A_n^{'} \mid -_L A$ в любой интерпретации. Таким образом имеется 2^n различных выводимостей: $A_1^{'},...,A_n^{'} \mid -_L A$ среди них есть 2 различающиеся по значению A'n принадлежит либо a_n либо $\overline{a_n}$: $A_1^{'},...,A_{n-1}^{'},a_n \mid -_L A$ и $A_1^{'},...,A_{n-1}^{'},\overline{a_n} \mid -_L A$ по теореме дедукции отсюда имеем $A_1^{'},...,A_{n-1}^{'} \mid -_L a_n \to A$ и $A_1^{'},...,A_{n-1}^{'} \mid -_L \overline{a_n} \to A$ по теореме о выводимости выполняется следующее $\mid -_L (a_n \to A) \to ((\overline{a_n} \to A) \to A)$ применяем 2 раза правило MP, получаем $A_1^{'},...,A_{n-1}^{'} \mid -_L A$. Повторить весь процесс ещё n-1 раз докажем выводимость $\mid -_L A$ => A выводимо из ненулевого множества гипотез => A теорема.

Билет 11. Метод резолюций в исчислении высказываний.

Метод резолюций проверки выводимости $\{A_1, ..., A_n\} \vdash A$ основан на проверке тождественной истинности $A_1 \& ... \& A_n \to A$ путём рассмотрения резольвент.

Опр. Пусть $A_1 = A_1' \vee B$, A2 = A'2V $\overline{B}$, $A_1' \vee A_2'$ - резольвента дизъюнктов A_1 и A_2 по переменной B и обозначается $res_B(A_1, A_2)$

Если переменная, по которой берется резольвента не важна, то пишут просто $res(A_1, A_2)$. Если дизъюнкты A_1 и A_2 не содержат противоположных элементов, то резольвент у них не существует res(B, $\overline{B}$) = |т.к resB(A'1VB, A'2V $\overline{B}$) = $A_1' \vee A_2'$, то при A'1 ≡ A'2 ≡ 0, очевидно, что|=0.

Если res(A_1,A_2) существует, то {A_1,A_2}|-- res(A_1,A_2).

Опр. Пусть Г={$A_1, \dots, A_n$}-множество дизъюнктов. $B_1, B_2, \dots, B_n$ - резолютивный вывод из Г , если ∀ B_i(i=1,..n) выполняется одно из двух условий:

1. $B_i \in Г$ 2. Существует $j,k < i$: $B_i = res(B_j,B_k)$

Теорема о полноте метода резолюций.

Множество дизъюнктов Г противоречиво в том и только том случае, когда существует резолютивный вывод из Г, заканчивающийся 0.

Применим эту теорему для проверки выводимости формулы A из данного множества формул Г={A_1, A_2,..., A_n}. Докажем, что условие {A_1, A_2,..., A_n} |— A равносильно условию {A_1, A_2,..., A_n, $\overline{A}$} |— (система {A_1, A_2,..., A_n, $\overline{A}$} противоречива). Действительно, если {A_1, A_2,..., A_n} |— A, то $A_1 \& A_2 \& \dots \& A_n \to A \equiv 1$, тогда $\overline{A_1 \& A_2 \& \dots \& A_n} \vee A \equiv 1$, $A_1 \& A_2 \& \dots \& A_n \& \overline{A} \equiv 0$, т.е. $Г_1$={A_1, A_2,..., A_n, $\overline{A}$} |—. Это равносильно условию, что B |— , где $B = A_1 \& A_2 \& \dots \& A_n \& \overline{A}$. Приведем формулу B к КНФ, т.е. к виду $B = C_1 \& C_2 \& \dots \& C_m$. Тогда если B |— , то $C_1 \& C_2 \& \dots \& C_m$ |—. Таким образом, задача проверки выводимости {A_1, A_2,..., A_n} |— A сводится к проверке противоречивости множества дизъюнктов $Г_1$={C_1, C_2,..., C_m}. А последнее равносильно существованию резолютивного вывода нуля из $Г_1$.

На практике в общем случае этот метод неэффективен, т.к. число переборов резольвент может быть очень большим. Оно экспоненциально зависит от числа дизъюнктов и содержащихся в них переменных. Однако метод успешно применяется к хорновским дизъюнктам.

Дизъюнкт называют хорновским, если он содержит не более одной переменной степени 1(без отрицания) В общем случае хорновский дизъюнкт имеет вид: $\overline{A_1} \vee, \dots, \vee \overline{A_n}$ - негативный дизъюнкт.

$\overline{A_1} \vee, \dots, \vee \overline{A_n} \vee$B – точный дизъюнкт.

Дизъюнкт А- унитарный позитивный, $\overline{A}$ - унитарный негативный.

Если Г- множество хорновских дизъюнктов, то его проверка на непротиворечивость происходит следующим образом: в Г выбирается унитарный дизъюнкт В(или $\overline{B}$) и дизъюнкт С из Г, который содержит $\overline{B}$ (или В). Далее Г заменяется на $Г \setminus \{C\} \cup \{res_B(C,B)\}$ этот процесс идёт до тех пор, пока либо Г не будет содержать 0, либо не найдётся дизъюнктов В и С указанного вида.

В первом случае указанное множество противоречиво, а во втором непротиворечиво.

Билет 12. Проблемы аксиоматического исчисления высказываний.

Исчисление высказываний для своего обоснования требует решения 4-х проблем: разрешимости, непротиворечивости, независимости, полноты.

1. Проблема разрешимости исчисления высказываний.

Состоит в поиске алгоритма, который позволил бы для любой заданной формулы исчисления высказываний определить, является ли она доказуемой или не является. В исчислении высказываний такой алгоритм существует. (см. вопрос 4)

Действительно, любая формула исчисления высказываний может рассматриваться как формула алгебры высказываний, и, следовательно, можно рассматривать ее логические значения на различных наборах значений, входящих в нее переменных.

2. Проблема непротиворечивости исчисления высказываний.

Т.е. не существования в нём формулы А такой, которая доказуема вместе с $\overline{A}$.

Если в исчислении обнаруживаются доказуемые формулы вида А и $\overline{A}$, то такое исчисление называется противоречивым.

Теорема: ИВ непротиворечиво.

Док-во: По теореме об общезначимых ф-х всякая выводимая формула тождественно истинна. Отрицание формулы не является тождественно истинной формулой, следовательно, ни для какой формулы А невозможно, чтобы одновременно выводились А и $\overline{A}$.

3. Проблема независимости аксиом исчисления высказываний.

Заключается в невыводимости каждой аксиомы из остальных аксиом по правилам вывода данной системы. Это означает, что в системе нет лишних аксиом.

Теорема: Система аксиом ИВ независима.

Док-во: основано на некоторых интерпретациях переменных и логических операций исчислений. В простейшем случае допускается, что переменные в аксиомах могут принимать только 2 значения: 1 и 0. Все логические операции кроме одной определяются так же, как в Алгебре высказываний $\overline{\ }, \vee, \&, \rightarrow$. Одну из логический операций определяют специально так, чтобы та аксиома, в которую эта операция входит и независимость которой доказывается, не являлась тождественно равная 1.

При такой интерпретации все аксиомы, кроме исследуемой принимают значение 1. Ясно, что если такая интерпретация возможна, то исследуемая аксиома не зависит от остальных, так как если бы она была выводима из остальных, то она, как и все формулы выводима из совокупности аксиом, кроме исследуемой, приняла бы единственное значение 1.

4. Проблема полноты исчисление высказываний – имеет два аспекта полноты.

Определение 1.

Аксиоматическое исчисление высказываний называется полным в узком смысле, если добавление к списку его аксиом любой недоказуемой в исчислении формулы в качестве новой аксиомы приводит к противоречивому исчислению.

Теорема: ИВ полно в узком смысле.

Доказательство. Пусть A — произвольная невыводимая формула, $<x_1,\ldots, x_n>$ — список ее переменных. По теореме об общезначимых формулах существует оценка $<\alpha_1,\ldots, \alpha_n>$ списка переменных $<x_1,\ldots, x_n>$ такая, что $A(x_1,\ldots,x_n)\big|_{<\alpha_1,\ldots,\alpha_n>} = Л$.

Пусть $B_1, B_2,\ldots,B_n$ — любые тождественно истинные формулы, зависящие от переменных $x_1,\ldots, x_n$. Рассмотрим следующий набор формул: $B_1^{\alpha_1}, B_2^{\alpha_2},\ldots, B_n^{\alpha_n}$, где $B_i^{\alpha_i} = \begin{cases} B_i, \text{если } \alpha_i = И, \\ \bar{B}_i, \text{если } \alpha_i = Л. \end{cases}$ Подставим их в формулу A и докажем, что $A(B_1^{\alpha_1}, B_2^{\alpha_2},\ldots, B_n^{\alpha_n}) \equiv Л$. Для этого заметим, что для любого набора $\delta_1,\ldots, \delta_n$ $B_i(x_1,\ldots,x_n)\big|_{<\delta_1,\ldots,\delta_n>} \equiv И$, т.к. B_i — тождественно истинная формула. Значит, $B_i^{\alpha_i}(x_1,\ldots,x_n)\big|_{<\delta_1,\ldots,\delta_n>} \equiv \alpha_i$ и $A(B_1^{\alpha_1}, B_2^{\alpha_2},\ldots, B_n^{\alpha_n}) = A(\alpha_1, \alpha_2,\ldots,\alpha_n) = Л$.

Таким образом, если $A(B_1^{\alpha_1}, B_2^{\alpha_2},\ldots, B_n^{\alpha_n}) \equiv Л$, то $\overline{A(B_1^{\alpha_1}, B_2^{\alpha_2},\ldots, B_n^{\alpha_n})} \equiv И$, т.е. получили тождественно истинную формулу, которая по теореме об общезначимых формулах выводима в исчислении высказываний.

С другой стороны, если формулу $A(x_1,x_2,\ldots,x_n)$ добавить к списку аксиом исчисления, то она станет выводимой в новом исчислении как аксиома. В новом исчислении формула $A(B_1^{\alpha_1}, B_2^{\alpha_2},\ldots, B_n^{\alpha_n})$ получается из $A(x_1,x_2,\ldots,x_n)$ в результате одновременной подстановки, т.е. также будет выводимой. Следовательно, новое исчисление высказываний будет противоречивым, т. к. в нем одновременно $\vdash A$ и $\vdash \bar{A}$. □

Определение 2.

Исчисление высказываний называется полным в широком смысле, если любая тождественно истинная формула в нем доказуема.

На основании теорем об общезначимых формулах ИВ проблема полноты в широком смысле решается положительно.

Билет 13 Определение предиката. Область определения, множество истинности предиката. Операции над предикатами, кванторы существования и всеобщности.

Определение. Предикат – это функция $P(x_1,x_2,\ldots,x_n)$, которая может принимать значения (И) или (Л) , а её переменные x_i могут принимать любые значения из некоторой области М. n – местный предикат, Р – это функция P: M→{0,1}, где M=M1 x …x Mn – декартовое произведение.

Из этого определения видно, что высказывание – 0 – местный предикат.

Область определения. Множество М, на котором определен предикат - называется областью определения предиката $P(x_1,x_2,\ldots,x_n)$.

Множество всех x1∈M1, x2∈M2, …, xn∈Mn , при которых $P(x_1,x_2,\ldots,x_n) \equiv И$, наз. множеством истинности предиката $P(x_1,x_2,\ldots,x_n)$ и обозначается Ip: Ip={x|x∈M, P(x)=И}.

Если I_p=M, то предикат $P(x_1,x_2,\ldots,x_n)$ называется тождественно истинным,

Если I_p=ø, то предикат тождественно ложный.

Так как предикаты принимают лишь два значения И,Л, то к ним применимы все операции ЛВ.

Операции над предикатами

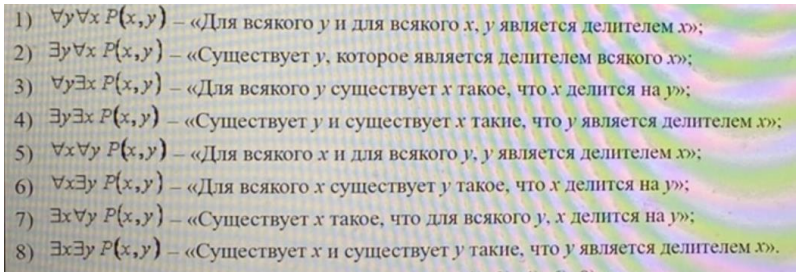
$P(x)\&Q(x)$	$P(x)VQ(x)$	$P(x)\sim Q(x)$	$P(x) \to Q(x)$	$P(x)$
$I_P \cap I_Q$	$I_P \cup I_Q$	$I_P = I_Q$	$I_P \subset I_Q$	$I_P = M \backslash I_P$

Кроме операций логики высказываний к предикатам применяются операции связывания - кванторы.
∃-квантор существования, ∀-квантор всеобщности.

Пусть P(x) задан на M. $\exists xP(x)$ будем понимать высказывание истинным, когда существует элемент $x \epsilon M$: $P(x)$=И, и ложным в противоположном случае. $\forall xP(x)$ будем понимать высказывание истинным, когда для каждого элемента $x \epsilon M$: $P(x)$=И, и ложным в противоположном случае.

В предикате $P(x)$ переменная x называется свободной, а в высказываниях $\exists xP(x), \forall xP(x)$ x называется связанной.

Кванторные операции применяются к многоместным предикатам. В частности, если применить кванторные операции к предикату по обеим переменным, получим 8 высказываний: ∀x∀yP(x,y);



∀x∃yP(x,y) и т.д.

Применение кванторной операции к предикату P(x,y) по переменной x ставят в соответствие двухместному предикату P(x,y) одноместный предикат: ∀xP(x,y) или ∃xP(x,y) – 1-местный предикат, зависящий от y и не зависящий от x.

Билет 14 Формулы логики предикатов, свободные и связанные переменные.

Алфавит логики предикатов содержит следующие символы:

1) Символы переменных высказываний P_1, Q_2, r_3(лат. Буквы с индексом и без, заглавные и маленькие)

2) Символы предметных переменных и предметных констант $x_1, x_2, \dots, x_n$

3) Символы предикатов $A_1^{(t)} \dots A_n^{(t)}$, где t=0,1,2… (для t – местного предиката)

4) Функциональные символы $f_j^{m_j}$, где $m_j \in N$, $f_j^{m_j}$- m_j - местный функциональный символ

5) Логические символы &, V, →, ¬, ~ 6) Кванторы ∃,∀ 7) Скобки (и)

Опр: Множество предикатных букв вместе с множеством функциональных букв и предметных констант называется сигнатурой языка данной теории.

Опр: Термами языка ЛП являются 1)предикатные переменные и предметные константы 2) если f^n – n местный функциональный символ и $t_1, \dots, t_n$-термы, то $f^n(t_1, \dots t_n)$–тоже терм.

Опр: Атомной или атомарной формулой сигнатуры называют выражение $P^n(t_1, \dots t_n)$, где P^n –n местный предикатный символ, $t_1, \dots, t_n$-термы. Все предметные переменные атомарной формулы свободные, связных нет.

Литеральными наз. формулы вида A и $\overline{A}$, если A – атомарная формула.

Слово в алфавите ЛП называется формулой, если оно удовлетворяет след. индуктивному определению:

1)Атомарная формула – это формула

2)Если A и B – формулы, то $A \sim B, A\&B, A \to B, AVB$ тоже формулы, при условии что одна и та же переменная не является в A – свободной, а в B – связанной или наоборот.

3)Если A – формула, то $\overline{A}$ тоже формула, свободные и связанные переменные совпадают у A и $\overline{A}$.

4)Если A(x) – формула, то $\exists A(x), \forall A(x)$ тоже формулы, причем если в A(x) x была свободной, то в $\exists xA(x), \forall xA(x)$ x-связанная

*из этого определения видно, что всякая формула ЛВ является формулой ЛП.

Опр: Подслово формулы A, которой само является формулой, называется подформулой формулы A.

Если ф-ла A содержит свободную переменную x, то в формулах для ∀xA и ∃xA A наз. областью действия квантора.

Пр. $\exists xP(x,y) \to \forall xP(x,y)$ y - свободная переменная, x-связанная

Билет 15 Равносильность формул в логике предикатов и в различных интерпретациях. Основные равносильности: перестановка кванторов и переименование связанных переменных.

Равносильность формул в логике предикатов и в различных интерпретациях.

Пусть формулы F и G имеют одно и то же множество свободных переменных (быть может пустое).

Опр: Формулы F и G равносильны в данной интерпретации, если на любом наборе значений свободных переменных они принимают одинаковые значения. (если в одной и той же интерпретации эти формулы выражают один и тот же предикат)

Опр: Формулы F и G равносильны на множестве M, если они равносильны во всех интерпретациях, заданных на мн-ве M.

Опр: Формулы F и G равносильны (в Логике Предикатов) если они равносильны на всех множествах: F≡G.

Для формулы Логики Предикатов (ЛП) сохраняются все равносильности и правила равносильных преобразований логики высказываний, но имеются равносильности самой ЛП, связанные с кванторами.

Основные равносильности ЛП:

1. Перестановка одноименных кванторов

$\forall$ – квантор всеобщности $\exists$ – квантор существования

$\forall x \forall y P(x,y) \equiv \forall y \forall x P(x,y)$ $\exists x \exists y P(x,y) \equiv \exists y \exists x P(x,y)$

Однако разноименные кванторы переставлять нельзя, пример:

1) $\forall y \exists x P(x,y)$ 2) $\exists x \forall y P(x,y)$

Пусть $P(x,y) = (x > y)$ на мн-ве M=N*N.

Тогда 1е высказывание означает, что какое бы натуральное число мы не взяли, всегда найдется натуральное число еще больше этого. Это высказывание истинно.

2е высказывание означает, что существует самое большое натуральное число. Оно ложно на M.

2. Переименование связанных переменных

Заменяя связанную переменную формулы A другой переменной, не входящей в эту формулу в кванторе и всюду в области действия квантора, получим формулу равнозначную A. Например $\forall xP(x,y) \equiv \forall tP(t,y)$

Билет 16. Правила переноса квантора через отрицание в формулах логики предикатов.

1) $\overline{\forall x A(x)} \equiv \exists x \overline{A}(x)$

2) $\overline{\exists x A(x)} \equiv \forall x \overline{A}(x)$

Докажем первый закон: пусть $x_1, x_2, \dots, x_n$ – множество (может пустое) всех свободных переменных формулы A, отличных от *x*. Пусть пара { M , f }-произвольная интерпретация. Рассмотрим произвольный набор значений свободных переменных: $<a_1, a_2, \dots, a_n>$

$a_i \in M$.

Исследуем, какие логические значения на этом наборе примут формулы: $\overline{\forall x A(x)}$ и $\exists x \overline{A}(x)$. Возможны два случая:

1) Для $\forall a \in M$: $A(x)|_{<a_1,a_2,\dots,a_n>} = $ И

2) Для $\forall a_0 \in M$: $A(x)|_{<a_0,a_1,a_2,\dots,a_n>} = $ Л

1. Для $\forall a \in M$: $\overline{A}(x)|_{<a_0,a_1,a_2,\dots,a_n>} = $ Л, отсюда по определению:

   $\exists x \overline{A}(x)|_{<a_0,a_1,a_2,\dots,a_n>} = $ Л, с другой стороны

   $\forall x A(x)|_{<a_1,a_2,\dots,a_n>} = $ И, т.к. в этом случае $\forall a \in M : A(x)|_{<a,a_1,\dots,a_n>} = $ И, отсюда

   $\overline{\forall x A(x)}|_{<a_1,a_2,\dots,a_n>}$=Л, значит $\overline{\forall x A(x)} \equiv \exists x \overline{A}(x)$ выполняется в случае 1.

2. Для $\forall a_0 \in M$ $\overline{A}(x)|_{<a_0,a_1,a_2,\dots,a_n>} = $ И, отсюда $\exists x \overline{A}(x)|_{<a_1,a_2,\dots,a_n>} = $ И,

   с другой стороны в этом случае $\forall x A(x)|_{<a_1,a_2,\dots,a_n>}$ =Л, следовательно

   $\overline{\forall x A(x)}|_{<a_1,a_2,\dots,a_n>}$ =И, следовательно и во втором случае искомая равносильность доказана, значит она полностью доказана.

Билет 17. Правила выноса квантора за скобки в формулах логики предикатов.

Вынос квантора за скобки:
Постоянный предикат можно вносить под знак квантора и выносить из-под него в конъюнкции и дизъюнкции.
C&∃ x B(x) ≡ ∃ x (C&B(x)) $\quad C \vee \forall x\, B(x) \equiv \forall x\big(C \vee B(x)\big)$
$C \vee \exists x\, B(x) \equiv \exists x\big(C \vee B(x)\big)$
$C\&\forall x\, B(x) \equiv \forall x\big(C\&B(x)\big)$
Док-во: *(1ой равносильности)*
Пусть x1…xn – все свободные переменные формулы $C\&\exists x\, B(x)$ и для ∃ x (C&B(x)). Рассмотрим произвольную интерпретацию с множеством M. Пусть <a1,...,an>, ai∈M – произвольный набор свободных переменных <x1,…,xn>. Так как формула C не содержит переменные x, то можно определить ее значение на наборе <a1,…,an> (точнее на его части, относящейся к свободной переменной формулы C).
Если $C|_{<a_1,\dots,a_n>} = Л$, то $\big(C\&\exists B(x)\big)|_{<a_1,\dots,a_n>} = Л$ и для $\forall a \in M$ на наборе значений <a, a1,…,an> свободных переменных <x, x1,…,xn> формула C&B(x) принимает значение Л. Отсюда $\exists x\big(C\&B(x)\big)|_{<a_1,\dots,a_n>} = Л$.
Если же $C|_{<a_1,\dots,a_n>} = И$, то для $\forall a \in M$ на наборе <a, a1,…,an> формулы C&B(x) и B(x) принимают одинаковые истинностные значения. Следовательно,
$\big(C\&\exists x B(x)\big)|_{<a_1,\dots,a_n>} = \exists x B(x)|_{<a_1,\dots,a_n>} = \exists x\big(C\&B(x)\big)|_{<a_1,\dots,a_n>}$
Таким образом, исходная равносильность полностью доказана.
Замечание: если оба предиката содержат переменную x, то выполняется лишь 2 равносильности из четырех: $\forall x\big(A(x)\&B(x)\big) \equiv \forall x A(x)\&\forall x B(x)$ и $\exists x(A(x) \vee B(x)) \equiv \exists x A(x) \vee \exists x B(x)$.

Билет 18 Нормальные формы логики предикатов. Теорема о ПНФ.
Формулы, в которых из логических символов имеются только символы &,∨, →, причем → встречается только над символами предикатов, называются нормальными формами (НФ).
Используя равносильности алгебры высказываний и логики предикатов (ЛП) каждую ф-лу ЛП можно привести к НФ.
Теорема.Для ∀ ф-лы ЛП ∃ равносильная ей НФ, причем множество свободных и связных переменных этих ф-л совпадают.
Среди НФ особое значение имеют предваренные нормальные формы (ПНФ).
ПНФ наз. такая нормальная форма, в которой либо полностью отсутствуют кванторные операции, либо в последовательности символов, образующих ф-лу, кванторы предшествуют всем остальным символам.
Эта форма имеет вид: $(\partial x_1)(\partial x_2)\dots(\partial x_n)A(x_1, x_2, \dots, x_m)$, где $\partial x_i \equiv \forall x_i$ или $\exists x_i$. В ф-ле А кванторов нет.
Смысл записи ф-лы в ПНФ состоит в том, чтобы разделить ф-лу на 2 части: кванторную и бескванторную. В таком виде ее проще анализировать.
Примеры: 1)$\forall x \exists y(\overline{A}(x) \vee B(x, y))$ – ПНФ
2)$\forall x \overline{A}(x)\&\exists y B(y)$ - НФ, не являющаяся предваренной
Если все ∂x_i - кванторы всеобщности(∀), то ф-ла называется универсальной - $\partial x_i \equiv \forall x_i$
Если же все $\partial x_i \equiv \exists x_i$, то существующая ф-ла.
Если ∃ i (0≤i≤n) такое, что $\partial_1, \partial_2, \dots, \partial_i$ - кванторы существования(∃), а $\partial_{i+1}, \partial_{i+2}, \dots, \partial_n \equiv \forall$, то ф-ла называется существующей универсальной ф-лой или скулемовской.
Теорема о ПНФ
Всякая форма ЛП может быть приведена к равносильной ей ПНФ.
Идея док-ва:
Если ф-ла содержит отрицания над кванторами, то с помощью равносильностей не(∀xA(x))= ∃xне(A(x)) и не(∃xA(x))=∀x не(A(x)) отрицание вводится под знак квантора. Если ф-ла имеет вид $A_1 \vee A_2$ или $A_1 \& A_2$ то при необходимости сначала переименовать в A_2 связные переменные так, чтобы все они в A_1 и A_2 были различны, а затем по второму закону логики предикатов вынести все кванторы за скобку.
Пример. Привести к ПНФ
$\overline{P \to \exists x R(x)} \equiv \overline{\overline{P} \vee \exists(x) R(x)} \equiv P\&\overline{\exists x R(x)} \equiv P\&\forall x \overline{R}(x) \equiv \forall x(P\&\overline{R}(x))$

Билет 19. Выполнимость и общезначимость для предикатов. Основные общезначимые формулы в логике предикатов.
А выполнима в данной интерпретации если существует набор $< a_1 \dots a_n >$ где $a_i \in M$ значений свободных переменных $x_1 \dots x_n$ формулы А такой, что $A|_{<a_1 \dots a_n>} = UИ$.

Формула А истинна в заданной интерпретации, если она принимает значение И на ∀ наборе $< a_1 \dots a_n >$ своих переменных $x_1 \dots x_n$.
Формула А общезначима (тождественно истинна в лп) если она истинна в каждой интерпретации.
Формула А выполнима в лп, если ∃ интерп, в которой она выполнима.
Утв1(об общезн-ти) Ф-ла $\forall x A(x) \to A(y)$ (1) где *y* не входит в ф-лу А(х), общезначимая.
Док-во: пусть $x_1 \dots x_n$ все своб перем-е в-лы А(х), тогда $y, x_1 \dots x_n$ все свободные переменные формулы (1). Рассмотрим интерпретацию с множеством М.
Пусть$b, a_1 \dots a_n$ где $b \in M, a_i \in M$, i=1,…,n - произв набор значений всех свободных переменных (1).
Покажем: $\forall x A(x) \to A(y)|_{<b,a_1 \dots a_n>} = И$. Для $A(x)$ либо $\exists a_0 \in M$: $A(x)|_{<a_0,a_1 \dots a_n>} = И$ либо $\forall a \in M$(в тч для a=b): $A(x)|_{<a_0,a_1 \dots a_n>}$=И. Если $A(x)|_{<a_0,a_1 \dots a_n>}$=Л, то $\forall x A(x)|_{<a_1 \dots a_n>} = Л$ тогда
$\forall x A(x) \to A(y)|_{<b,a_1 \dots a_n>} = И$
Во втором случае $\forall x A(x)|_{<a_1 \dots a_n>} = И$, $A(y)|_{<b,a_1 \dots a_n>} = И$. Значит, $\forall x A(x) \to A(y) \equiv И$, т.е
$И \to И \equiv И$=> (1) общезначимая
Утв2(об общезначимости) $A(y) \to \exists x A(x)$ (2), где *y* не входит в А(х), общезначимая
Док-во: в силу утв1$\forall x A(x) \to A(y)$ - общезначимая, тогда заменим А на не(А):
$\forall x \bar{A}(x) \to \bar{A}(y) \equiv \overline{\forall x (\bar{A})(x)} \vee \bar{A}(y) \equiv \exists x A(x) \vee \bar{A}(y) \equiv \bar{A}(y) \vee \exists x A(x) \equiv A(y) \to \exists x a(x)$, следственно, (2) общезначимая.

Замечание:
Тк одноименные кванторы можно переставлять, то общезначимыми являются след формулы:
$\exists x \exists y A(xy) \sim \exists y \exists x A(xy)$ $\forall x \forall y A(xy) \sim \forall y \forall x A(xy)$
Док: Равносильности ф-л ЛП требует либо детального опред значений ф-л, либо использования известных равносильностей.
Для определения общезн-ти ф-лы лп иногда не достаточно использ-я равнос-й. Облегчить решение этой задачи помогут т. Об общезначимости

Билет 20. Теоремы об общезначимости и выполнимости в логике предикатов. Проблема разрешимости в общем случае (теорема пол) и для формул, содержащих только одноместные предикатные символы.
Теорема 1.
Формула А общезначима тогда и только тогда, когда (¬А) невыполнима, т.е всюду тождественно ложна.
Доказательство: Необходимо. Пусть А –общезначима, тогда (¬А) тождественно ложная на любой области, т.е. нигде не выполнима. Достаточно. Пусть ф-ла (¬А) не выполнима на любой области М. Тогда она тождественно-ложна по опр. Сл-но А – тожд. Истинная формула в любой области М, т.е она общезначима.
Теорема 2
Формула А выполнима тогда и только тогда, когда формула (¬А) не общезначима.
Доказательство вытекает из того, что если формула А выполнима, то для нее существуют 2 области М1 и М2 в одной из которых она истина, а в другой ложна.
Необходимо. Если взять ту область, где А-истинна, то (¬А) там будет ложна.
Достаточно. Тоже самое, в другую сторону.
Теорема 3
Функции А и В равносильны в ЛП, тогда и только тогда, когда формула А~В – общезначима.
Задача распознавания общезначимости ФЛП называется, как и в ЛВ, проблема разрешимости: указать эффективный алгоритм распознания общезначимости формул.
Метод перебора всех вариантов для такой задачи в общем случае не применим, т.к. вариантов может быть бесконечно много. Поэтому Черч доказал:
Теорема Черча.
Не существует алгоритма, который для любой формулы ЛП устанавливает, общезначима она или нет.
Однако в некоторых частных случаях проблема разрешимости решается. Например, соответствующий алгоритм существует для формул ЛП, содержащих только одноместные предикатные символы. Логика, в которой используются только одноместные предикаты практически соответствует логике, описанной ещё Аристотелем. Здесь кванторные операции можно заменить операциями V или & и тем самым свести формулу ЛП к формулам ЛВ, для которой проблема разрешимости имеет алгоритмическое решение.

Алгоритм проверки таких формул осуществляется с помощью теоремы:
Теорема.Критерии выполнимости ФЛП (формулы логики предикатов).
Пусть F формула, содержащая ровно n 1-местных предикатных символов. Для того, чтобы формула F была выполнима тогда и только тогда, когда F выполнима во всех интерпретациях множества M, где |M|<=2^n – кол-во символов. Кроме того, можно проверить общезначимость ФЛП, у которой в ПНФ содержатся кванторы одного типа.

Билет 21. Язык, система аксиом и основные правила вывода исчисления предикатов.
В ЛП добавим знак выводимости и придём к формальной теории ИП, которая обозначается К-исчисление предикатов и определяется следующим образом:

1. Алфавит ИП = алфавит ЛП + знак следования «|-»
2. Формула ИП определяется так же, как и в логике предикат.

Внимание надо обратить на след условия:
1) В формуле свободные и связанные переменные обозначаются разными буквами.
2) Если квантор находится в области действия другого квантора, то переменные, связанные этими кванторами, обозначаются разными буквами.
Нарушение любого из этих условий в ИП наз. коллизией переменных.
ИП, не содержащие функциональных букв и предметных констант, наз. чистыми ИП.

3. Аксиомы ИП (делятся на 2 группы):

1) Одна из систем аксиом исчисления высказываний (т.к. ИП это расширение ИВ)
2) Непосредственно касается предикатов и состоит из 2ух аксиом:
(P1) $\forall$ x F(x)→F(y) - аксиома общности
(P2) F(y)→ $\exists$x F(x) – аксиома введения квантора существования.

4. Правила вывода

1) Modus Ponens – правило заключения, как в ИВ
2) Правило обобщения: то $\rightarrow \frac{F \rightarrow C(x)}{F \rightarrow \forall x C(x)}$
3) Правило введения квантора существования: $\left.\frac{C(x) \rightarrow F}{\exists x C(x) \rightarrow F}\right\}$ C(x) – содержит свободные вхождения x, F- не содержит.

Замечания:

1. Возможны и другие системы аксиом и правила вывода
2. Правило подстановки здесь не фиксируется. Тем самым из 2-ух возможных истолкований систем аксиом выбрано второе, при котором правило подстановки формально отсутствует, а вместо аксиом рассматриваются схемы аксиом (т.е. подстановкой можно пользоваться по умолчанию).

Построение ИП с правилом подстановки существенно более громоздко из-за необходимости различать свободные и связанные вхождения предметных переменных.

Билет 22. Производные правила вывода в исчислении предикатов: правила переименования связанных переменных, правило связывания квантором.
Правило переименования связанных переменных

1) $\forall x F(x) | - \forall y F(y)$
2) $\exists x F(x) | - \exists y F(y)$

F(x) не содержит свободных вхождений y, но содержит свободные вхождения x, ни одно из которых не входит в область действия квантора по y.
Доказательство:

1) $\forall x F(x)$ – гипотеза по условию
2) По аксиоме (P1) |--$\forall x F(x) \rightarrow F(y)$
3) К шагу 2 применим правило обобщения $| - \forall x F(x) \rightarrow \forall y F(y)$
4) MP для (1) и (3): $\forall x F(x) | - \forall y F(y)$

Для квантора существования используем P.2 и правило введения квантора существования.

Все теоремы, выводимые в ИВ, включая все производные правила вывода ИВ(теорема дедукции. Правило силлогизма и т.д) являются выводимыми и в ИП. Совершая подстановки в выводимые формулы ИВ, можно получить новые выводимые формулы ИП. Единственное дополнительное условие при таком обобщении – отсутствие коллизии переменных.

Правило связывания кванторов.

$F(x)| -_K \forall x F(x)$

1) F(x) – гипотеза по условию
2) По т.2 ИВ $A| -_L B \to A$, то есть формула $B \to A$ доказуема для любого B, в том числе и для доказуемой. Значит $F(x)| -_K B \to F(x)$
3) По правилу обобщения $\frac{B \to F(x)}{B \to \forall x F(x)}$, B не содержит свободных вхождений x.
4) Из (3) по правилу заключения MP (для B и $B \to \forall x F(x)$) следует доказуемость $\forall x F(x)$.
5) Из (1)-(4) получаем $F(x)| -_K \forall x F(x)$.

Билет 23. Теоремы об общезначимых формулах (доказать необх-ть) и о замене эквивалентных подформул в исчислении предикатов (без док-ва).

Теорема об общезначимых формулах

Теоремы ИП – общезначимые формулы: (|-k F) ⇔ (F – общезначимая).

Док-во:

1) → Аксиомы ∀ системы – по сути тавтологии. Все правила вывода сохраняют общезначимость, т.е. их применение к общезначимым формулам снова даёт общезначимые формулы, значит теоремы ИП – общезначимые формулы.

2) ← Доказал впервые Фридрих Гёдель.

Опр. Две формулы F и G эквивалентны, если ⊢F~G

Из теоремы об общезначимых формулах ИП следует, что всякая равносильность F≡G в соответствует доказуемая эквивалентность в ИП |- F~G соответствия. (F~G≡(F→G)&(G→F)).

Доказательства общезначимости в ЛП существенно сложнее, чем в ЛВ, поэтому эквивалентные формулы целесообразнее получать не производя формальный вывод, а с помощью законов ЛП. Такой процедуре соответствует следующая теорема:

Теорема о замене эквивалентных подформул (идеи док-ва смотри в лекциях параграф 8, глава 2).

Пусть F(A) – формула, в которой выделено вхождение формулы A. F(B) – формула, полученная из F(A) заменой всех вхождений A формулой B. Тогда, если $\vdash A \sim B$, то $\vdash F(A) \sim F(B)$. Благодаря этому правилу можно получить доказуемые эквивалентности, не строя их непосредственного вывода.

Билет 24. Наиболее важные эквивалентности исчисления предикатов и их применение для построения предваренной нормальной формы.

A и B формулы, не содержащие свободных вхождений x.

1) ∀ x(F(x)&G(x))~(∀ x(F(x)& ∀ xG(x))
2) ∃x(F(x)VG(x))~ (∃x(F(x)V∃xG(x))
3) A& ∀ xF(x)~ ∀ x(A&F(x))
4) A& ∃ xF(x)~ ∃ x(A&F(x))
5) AV ∀ xF(x)~ ∀ x(AVF(x))
6) AV ∃ xF(x)~ ∃ x(AVF(x))
7) A → ∀ xF(x)~ ∀ x(A→F(x))
8) A→ ∃ xF(x)~ ∃ x(A→F(x))
9) ∀ xF(x) → B ~ ∃x(F(x) → B)
10) ∃xF(x) → B ~ ∀ x(F(x) → B)

Из 10 данных эквивалентностей первые 6 аналогичны закону (2) ЛП, остальные 4 доказываются с помощью закона (1) ЛП и равносильности A→B≡ $\overline{A}$ vB. Используя данные 10 эквивалентностей в формулах ИП можно выносить кванторы за скобки. С помощью правил переноса квантора через отрицание, а также правила переименования переменных, кванторы можно выносить за скобки в ∀ формуле. При этом получится предварённая нормальная форма в ИП.

Билет 25 Проблемы аксиоматического исчисления предикатов.

Исчисление предикатов для своего обоснования требуют решения 4ёх проблем: разрешимости, непротиворечивости, независимости и полноты.

1ая проблема: (разрешимости) заключается в поиске алгоритма, который бы для любой заданной формулы исчисления определял бы – является ли она доказуемой в этом исчислении или нет. В исчислении предикатов разрешающийся алгоритм построить невозможно из-за бесконечности

предметной области, который приводит в общем случае к бесконечным таблицам истинности. Доказательство этого факта связано с серьёзными трудностями, зависящими от наличия точного формального определения алгоритма. Лишь после появления такого определения стало возможным доказательство неразрешимости проблем разрешимости ИП (Теорема Чёрча вопрос 20)

2ая проблема: (не противоречивость исчисления), то есть не существует в этом исчислении формулы А такой, которая была бы доказуема с не А.

Теорема: Исчисление предикатов не противоречиво.

Док-во (от противного): Допустим ИП противоречиво. Тогда в нём выводима всякая формула, в частности формула А, состоящая из 1ой буквы, но тогда А была бы выводима и в ИВ , что неверно по теореме об общезначимых формулах ИВ.

3я проблема: (независимости системы аксиом исчисления предикатов). Она заключается в невыводимости каждой аксиомы из остальных аксиом по правилам вывода данной системы. Это означает, что в этой системе нет лишних аксиом.

Теорема. Система аксиом ИП независима. Эту независимость можно установить сведением к вопросу о непротиворечивости данной системы, либо явным заданием множества, из которого принимают значения переменные, содержащиеся в аксиомах.

4я проблема: (Проблема полноты имеет два аспекта полноты).

Аксиоматическое исчисление называется полным в узком смысле, если добавление к списку его аксиом любой, не доказуемой в этом исчислении формулы, в качестве новой аксиомы, приводит к противоречивому исчислению. Исчисление предикат оказывается не полным в узком смысле. К его аксиомам можно присоединить без противоречия недоказуемую в нём формулу $\exists xF(x) \rightarrow \forall xF(x)$ (все предметы тождественные). Действительно, каждая выводимая формула ИВ имеет аналог в ИП и может быть присоединена к аксиомам этого исчисления.

Аксиоматическое исчисление называется полным в широком смысле слова, если любая тождественно истинная формула в нём доказуема . Исчисление предикат полно в широком смысле по теореме об общезначимых формулах.

Билет 26 Формализация понятия алгоритма.

Алгоритм – общий, единообразный, точно установленный способ решения любой задачи из данной массовой проблемы (бесконечного множества однотипных задач).

Характерные черты алгоритма:

1) Дискретность (алгоритм состоит из действий, выполняемых по шагам).

2) Детерминированность (между всеми величинами, получаемыми алгоритмом, существует жёсткая причинная связь, каждая последующая величина получается из значения предыдущей по определенному, как правило простому, закону).

3)Массовость (начальная система величин выбирается из некоторого множества, начальные условия могут варьироваться в бесконечных пределах).

4)Результативность или сходимость (обязательно остановка алгоритмического процесса, после конечного числа шагов, с указанием достигнутого конструктивного объекта в качестве искомого результата).

Примеры конструктивных объектов: полиномы, матрицы, графы, слова, тексты. Процесс формирования конструктивных объектов называется алгоритмическим процессом.

Если у данной массовой проблемы существует алгоритм её решения, то вполне можно пользоваться интуитивным определением алгоритма. Однако доказать отсутствие алгоритма решения массовой проблемы без строгого формального определения алгоритма невозможно. Такого определения не было до середины 30-х годов 20 века. Затем почти одновременно ряд математиков предположили несколько определений алгоритмов (иными словами была предложена формализация понятия алгоритма).

Алан Медисон Тьюринг – машина Тьюринга.

Эмиль Леон Пост – система продукций.

Алонзо Чёрч, С.К.Кишни, опираясь на работы Ж. Эрбрана и К.Гёделя – рекурсивные функции.

А.А.Марков – нормальные алгоритмы.

А.Н.Колмогоров – общее определение алгоритма.

Типы алгоритмических моделей:

1) Рекурсивные функции. Этот тип связывает понятие алгоритма с числовыми функциями, заданными на множестве натуральных чисел и принимающие значения на том же множестве.
2) Машины Тьюринга. Связывает понятия алгоритма с механическим устройством. Способна выполнять дискретно элементарные действия над элементарными объектами.
3) Нормальный алгоритм Маркова. Связывает понятия алгоритма с классом словарных преобразований, в результате замены части слов или всего слова другим словом.

Билет 27. Понятие рекурсивных функций. Примитивно рекурсивные функции: базовые функции и элементарные операции.

Рекурсивным заданием функции называется способ определения значения в некоторой точке, через известные значения этой функции в предшествующих точках.

Очевидно, что в каждой точке должен быть задан набор значений аргументов данной функции, а в некоторой исходной точке значение самой функции.

Заданные рекурсивно числовые функции $f: M \to N_0$, где $M \subseteq N_0^n$ будем называть рекурсивными.

Последовательность функциональных равенств, описывающих по шагам дискретный процесс вычисления числ. функции от её исходного значения при заданных значениях независимых переменных, называют рекурсивным описанием (протоколом).

Примитивно рекурсивные функции.

Базовые фукции.

Простые одношаговые рекурсивные функции называются базовыми.

1)Нуль функция 0(x)=0

2)Функция тождества(проектирующая функция, функция введения фиктивных переменных)

$I_m^n(x_1, \dots, x_n) = x_m$, где $1 \le m \le n$, если $n = 1$, то $I(x) = x$

3)Функция следования(прибавление единицы) $\lambda(x) = x + 1,\ x' = x + 1$

*эта функция позволяет по числу построить сколь угодно большой начальный отрезок множества натуральных чисел

Все приведённые функции вычислимы в интуитивном смысле.

Элементарные операции.

Простейшие операции, с помощью которых можно получить из базовых функций рекурсивные называются элементарными.

1) Операция суперпозиции

Говорят, что n-местная функция ψ(x1…xn) (кси) получена с помощью операторов суперпозиции из m-местной функции $\varphi(x_1 \dots x_m)$ и n-местных функций $f_1(x_1 \dots x_n)$, …,$f_m(x_1 \dots x_n)$, если

ψ(x1...xn)= $\varphi(f_1(x_1 \dots x_n), \dots, f_m(x_1 \dots x_n))$

Если заданы функции тождества I_m^n и оператор суперпозиции, то можно считать заданными всевозможные подстановки, перестановки и переименования любых функций.

2) Операция примитивной рекурсии

Говорят, что n+1-местная функция $f(x_1, \dots, x_n, y)$ получена из n-местной $\varphi(x_1, \dots, x_n)$ и n+2- местной функции $\psi(x_1, \dots, x_n, y, z)$ с помощью операции примитивной рекурсии, если ее значения можно вычислить по схеме примитивной рекурсии:

$$\begin{cases} f(x_1 \dots x_n, 0) = \varphi(x_1, \dots, x_n) \\ f(x_1, \dots, x_n, y+1) = \psi(x_1, \dots, x_n, y, f(x_1, \dots, x_n, y)) \end{cases}$$

Схема примитивной рекурсии позволяет определить значение функции f не только через значения функций φ и ψ, но и через значение самой функции f во всех предшествующих точках.

При n=0 схема примитивной рекурсии имеет вид

$$\begin{cases} f(0) = a \quad a = const \\ f(y+1) = \psi(y, f(y)) \end{cases}$$

Если при этом a=0, т.е $\begin{cases} f(0) = 0 \\ f(x+1) = g(f(x)) \end{cases}$,то будем говорить, что f получена из g(x) с помощью итераций f(x)=it (g(x)) .

Замечание:

Если n-местная функция φ и ((n+2)-местная функция ψ определены на всем множестве N0, то для каждой пары таких функций (φ, ψ) существует единственная (n+1)-местная функция f, удовлетворяющая условию примитивной рекурсии.

Функция называется примитивно-рекурсивной, если она базовая или может быть построена исходя из базовых функций конечным числом применения операторов суперпозиции и примитивной рекурсии.

Применяя операции суперпозиции и примитивной рекурсии ко всюду определенным функциям, образуются также всюду определенные функции. Поэтому все примитивно-рекурсивные ф-ии являются всюду определенными. Для каждой из них существует алгоритм вычисления её значений.

Билет 28. Примеры простейших примитивно рекурсивных функций.

1) Базовые функции (см вопрос 27)
2) $0(x_1, x_2, \ldots, x_n) = 0$ – ПРФ, так как $0(x_1, x_2, \ldots, x_n)=0(I_n^1(x_1),\ldots,I_n^n(x_n))$, т.е. получаем из базовых функций 0(x) и I_n с помощью операции суперпозиции.
3) f(x)=x+n явл ПРФ т.к. $f(x)=\lambda(\lambda(\lambda \ldots (x) \ldots))$ (n раз λ).
4) Сложение $f_+(x,y) = x + y$ – ПРФ т.к. ее можно получить из ПРФ тождества и следования с помощью оператора примитивной рекурсии.

   $f_+(x,0) = x + 0 = I_2^1(x,y)$

   $f_+(x,y+1) = x + (y+1) = (x+y) + 1 = f_+(x,y) + 1$

   $\begin{cases} \varphi(x) = x \\ \psi(x,y,z) = z+1 \end{cases}$
5) Умножение $f_*(x,y) = x * y$ – ПРФ т.к. ее можно получить из ПРФ нуля и сложения с помощью оператора примитивной рекурсии.

   $f_*(x,0) = 0 = 0(x)$

   $f_*(x,y+1) = x * y + x = f_*(x,y) + x = f_+(f_*(x,y),x)$

   $\begin{cases} \varphi(x) = 0 \\ \psi(x,y,z) = z + x \end{cases}$
6) Поскольку ПРФ определены на всем мн-ве N, то вместо обычной разности в теории рекурсивной функции вводят арифметическую или усеченную разность:

   Функция $x \dot{-} 1$ определяется следующей схемой примитивной рекурсии:

   $\begin{cases} 0 \dot{-} 1 = 0 = O(x); \\ (y+1) \dot{-} 1 = y = I_2^2(x,y). \end{cases}$

   Для $f(x,y) = x \dot{-} y$ схема примитивной рекурсии имеет следующий вид:

   $x \dot{-} y = \begin{cases} x - y, x \geq y, \\ 0, x < y. \end{cases}$ $\begin{cases} f(x,0) = x \dot{-} 0 = x = I_2^1(x,y); \\ f(x,y+1) = x \dot{-} (y+1) = (x \dot{-} y) \dot{-} 1 = f(x,y) \dot{-} 1. \end{cases}$ $\begin{cases} \varphi(x) = x; \\ \psi(x,y,z) = z \dot{-} 1. \end{cases}$
7) Сигнум

   $\text{sgn}(x)=\begin{cases} 0, & x = 0 \\ 1, & x > 0 \end{cases} \leftrightarrow \begin{cases} \text{sgn}(0) = 0 \\ \text{sgn}(y+1) = 1 \end{cases}$

   $\text{sgn}(x,y)=\begin{cases} 0, & x \leq y \\ 1, & x > y \end{cases}$ => sgn(x,y)=sgn(x-y)

   К этим двум функциям можно ввести дополнительные:

   $\overline{\text{sgn}(x)} = 1 - \text{sgn}(x) = \begin{cases} 1, x = 0 \\ 0, x > 0 \end{cases}$

   $\overline{\text{sgn}(x,y)} = 1 - \text{sgn}(x,y) = \begin{cases} 1, x \leq y \\ 0, x > y \end{cases}$

Билет 29. Теорема о примитивной рекурсивности суммы и произведения примитивно рекурсивной функции. (без доказательства). Примитивная рекурсивность функции "частное от деления *x* на *y*", "остаток от деления *x* на *y*", "признак деления *x* на *y*".

Теорема. Пусть f – n – местная ПРФ, тогда также являются примитивно рекурсивными. :

$S(x_1,x_2\ldots x_n) = \sum_{i=0}^{x_n} f(x_1,\ldots,x_{n-1},i)$

$P(x_1,x_2\ldots x_n) = \prod_{i=0}^{x_n} f(x_1,\ldots,x_{n-1},i)$

Примитивная рекурсивность функции "частное от деления *x* на *y*", "остаток от деления *x* на *y*", "признак деления *x* на *y*".

1) Частное от деления $\left[\frac{x}{y}\right] = \sum_{i=1}^{x} \overline{sgn}(iy - x)$ - ПРФ по предыдущей теореме

   x=7 y=3 $\Rightarrow \left[\frac{x}{y}\right] = 2$ и $\sum_{i=1}^{x} \overline{sgn}(3;-7)$ =1+1+0+0+0+0+0=2

   По теореме о примитивной рекурсивности суммы произведения примитивно рекурсивной функции.

2) Остаток от деления $\frac{x}{y}$. $rest(x,y) = x - \left(y\left[\frac{x}{y}\right]\right)$ - ПРФ, как суперпозиция других ПРФ.

3) Если $x \vdots y$, то $rest(x,y) = 0$

   Введем отдноместную функцию $div(x,y) == \begin{cases} 1, & rest(x,y) = 0 \\ 0, & rest(x,y) > 0 \end{cases}$

   $div(x,y) = \overline{sgn}(rest(x,y))$

Данная ф-ция явл. ПРФ как суперпозиция других ПРФ.

Билет 30. Ограниченный оператор минимизации и его применения. Теорема Робинсона об одноместных примитивно рекурсивных функциях (без доказательства).

Оператор минимизации.

В теории рекурсивных функций важную роль играет действие нахождения данной функции φ(z) значения z, которая является наименьшим корнем уравнения φ(z)=0. Выполняющий это действие оператор является оператором минимизации или μ - оператор. Существует несколько разновидностей.

Ограниченный μ-оператор g(x1,…,xn-1,z)= μy(индекс y<=z)[f(x1,…,xn-1,y)=0] равен наименьшему значению y, удовл. $y \le z$ и $f(x_1 \ldots x_{n-1},y) = 0$(1)если такое y существует и равен $z+1$, если не существует y. (условие 1)

Замечание : 1) Ограниченный μ-оператор g по данной n-местной функции f строит новую функцию, значение которой равно наименьшему *y*, удовлетворяющему условию (1).

2) Для того, чтобы из всюду определенности ф-ции f следовала всюду определенность g, описание оператора необходимо дополнить ещё одним условием на знач. ф-ции g, когда не существует y, удовл нерав-ву y<=z. Это значение можно считать любым, так как переопределение ф-ции в одной точке не нарушает её примитивной рекурсивности. Значение z+1 взято для удобства доказательства теоремы об ограниченном мю-операторе.

3) f=0 – уравнение минимизации

Теорема: Если ф-ция f(x1,…,xn-1,y) – ПРФ, то g(x1,…,xn-1,z) также ПРФ.

Пример 1) $[\sqrt{z}] = \mu y_{y \le z}[\overline{sgn}((y+1)^2 \dot{-} z) = 0]$

$[\sqrt{5}] = \mu y_{y \le z}[\overline{sgn}((y+1)^2 \dot{-} 5) = 0]$; $y = 1: \overline{sgn}(4 \dot{-} 5) = 1 \neq 0$; $y = 2: \overline{sgn}(9 \dot{-} 5) = 0$ - y=2 – корень.

2) $q(x) = x \dot{-} [\sqrt{x}]^2$ - ПРФ как суперпозиция других ПРФ (в ч-ти, $\sqrt{x}$)

Теорема Робинсона об одноместных ПРФ: Все одноместные ПРФ могут быть получены из функций $\lambda(x) = x + 1$ и $q(x) = x \dot{-} \left[\sqrt{x}\right]^2$ - конечным числом применений операции суперпозиции, итерации и сложении двух функций.

Билет 31. Неограниченный оператор минимизации. Частично рекурсивные функции(ЧРФ). Тезис Черча о вычислимых функциях.

Неограниченный оператор минимизации

Переход f к g называется неограниченным μ-оператором минимизации и обозначается:
$g(x_1 \ldots x_n)=\mu_y[f(x_1 \ldots x_n,y)=b]$

Ф-ция $g(x_1 \ldots x_n)$ вычисляется след. образом:

1) Пусть y=0. Найдём $f(x_1 \ldots x_n,0)$. Если $f(x_1 \ldots x_n,0)=b$, то $g(x_1 \ldots x_n)=0$. Если нет, то переходим ко 2-ому шагу.
2) Найдём $f(x_1 \ldots x_n,1)$. Если $f(x_1 \ldots x_n,1)=b$, то $g(x_1 \ldots x_n)=1$. Если нет, то след. шаг и тд.

Ф-ция g считется неопределенной, если существует *y* такой, что $f(x_1 \ldots x_n,y)$ не равен b или для какого-то *y* не определено $f(x_1 \ldots x_n,y)$.

Существует 3 варианта выполнения этих условий:

1. $f(x_1 \ldots x_n,0)$ не определено,
2. $f(x_1 \ldots x_n,y)$ определены в любых y=0,1,…,a-1 и отличны от «b», а значение $f(x_1 \ldots x_n,a)$ не определено,
3. Значение $f(x_1 \ldots x_n,y)$ определены во всех y и отличны от «b».

В остальных случаях данный процесс останавливается и даёт результат, а именно наименьшее значение y=a, являющийся корнем уравнения минимизации.

Частично рекурсивные функции. Определение

Функция называется частично рекурсивной если она базовая или может быть получена исходя из базовой функции, конечным числом применения операторов суперпозиции, примитивной рекурсии и неограниченного оператора минимизации. (оператор прим рекурсии опр следующую функцию через предыдущую)

Тезис Черча

Всякая вычислимая функция является частично-рекурсивной.

В формулировке тезиса входит интуитивное понятие вычислимости, поэтому его нельзя доказать в обще принятом математическом смысле, опровергнуть его можно построив хотя бы один контр пример т.е. интуитивно вычислимую функцию не являющуюся ЧР. До сих пор примера построено не было и скорей всего их не существует!

Замечание: Ограничение y<=z в огранич. мю-операторе даёт гарантию окончания процесса вычисления. Такое ограничение является существенной особенностью ПРФ. А неограниченный мю-оператор таким свойством не обладает, поэтому не является ПФР. Следовательно, ЧРФ в отличие от ПРФ не всегда является всюду определенными. Например 1/(sgn(x)). Кроме того, среди ЧРФ встречаются даже нигде не определенные ф-ции.

Билет 32. Общерекурсивные функции. Функция Аккермана. Теорема Аккермана (без доказательства).

Всюду определённая ЧРФ называется общерекурсивной ОРФ $\subset$ ЧРФ

Из этих двух возникает гипотеза:

ПРФ $\subset$ЧРФ
ОРФ $\subset$ЧРФ } ⇨ ПРФ=ЧРФ-? НЕТ

По аналогии с тем что sgn^{-1} (x) является ЧРФ, но не является ПРФ. Есть обще рекурсивные ф-ии, которые не являются ПРФ-ми.

Функция Аккермана

Первый пример такой ОРФ (всюду определяемая и вычислимая) придумал немецкий математик Аккерман. Идея заключалась в построении последовательности функций B(x, y) каждая из которых растёт существенно быстрее предыдущей и создание с помощью этой последовательности функции A(x), которая растёт быстрее любой ПРФ.

Построение функции Аккермана

φ_0(a,y)=a+y
φ_1(a,y)=a*y
φ_2(a,y)=a^y

Тогда

$\varphi_0(a,0)=a$ $\varphi_0(a,1)=a+1$

$\varphi_1(a,0)=0$ $\varphi_1(a,1)=a$ (2)

$\varphi_2(a,0)=1$, $\varphi_2(a,1)=a$

Эти формулы связаны между собой след. соотношениями:

$\varphi_1(a,y+1)=a+ay=\varphi_0(a, \varphi_1(a,y))$ $\varphi_2(a,y+1)=a*a^y= a^(y+1)=\varphi_1(a, \varphi_2(a,y))$

Продолжим эту последовательность, положив по определению:

$$\begin{cases} \varphi_{n+1}(a,0)=1 \\ \varphi_{n+1}(a,1)=a \\ \varphi_{n+1}(a,y+1)=\varphi_n(a,\varphi_{n+1}(a,y)) \end{cases} \quad (3)$$

Заметим что схема (3) согласуется с равенством (2) для $\forall n \in N$ (случай n=0 исключаем)

Схема (3) имеет вид примитивно рекурсивной, следовательно все функции φ_nпримитивно рекурсивны растут они крайне быстро

Например:

$\varphi_3(a,1)=a$

$\varphi_3(a,2)= \varphi_2(a,a)=a^a$

$\varphi_3(a,3)=a^{a^a}$ и т.д

$\varphi_3(a,2)=\varphi_2(a,a)=a^a$, $\varphi_3(a,3)=a^a$ и т.д. Зафиксируем теперь значение a: a=2. Получим последовательность одноместных функций: $\varphi_0(2,y)$, $\varphi_1(2,y)$ …Определим теперь функцию $B(x,y)$, которая перечисляет эту последовательность:

$$B(x,y)=\varphi_x(2,y), \quad (3)$$

т.е. $B(0,y)=\varphi_0(2,y)=2+y$, $B(1,y)=\varphi_1(2,y)=2y$, $B(2,y)=\varphi_2(2,y)=2^y$, $B(3,y)=\varphi_3(2,y)$ и т.д., а также диагональную функцию $A(x)=B(x,x)$. Из соотношений (2) и (3) следует, что

$$\begin{cases} B(0,y)=2+y, \\ B(x',0)=\operatorname{sgn} x, \\ B(x',y')=B(x,B(x',y)). \end{cases} \quad (4)$$

Соотношения (4) позволяют вычислять значения функций $B(x,y)$ и, следовательно, $A(x)$, причем для вычисления функции в данной точке нужно обратиться к значению функции в предшествующей точке — совсем как в схеме примитивной рекурсии. Однако здесь рекурсия ведется сразу по двум переменным (такая рекурсия называется двойной, двукратной, или рекурсией 2-й ступени), и это существенно усложняет характер упорядочения точек, а следовательно, и понятие предшествования точек также усложняется. Например, $B(3,3)=B(2,B(3,2))=B(2,\varphi_3(2,2))=B(2,2^2)$, т.е. вычислению B в точке (3; 3) должно предшествовать вычисление B в точке (2; 4). Аналогично $B(3,4)=B(2,B(3,3))=B(2,\varphi_3(2,3))=B(2,2^{2^2})=B(2,16)$. Важно отметить, что это упорядочение не предопределено заранее, как в схеме примитивной рекурсии, где n всегда предшествует n+1, а выясняется в ходе вычислений и для каждой схемы вида (4), вообще говоря, различно. Поэтому схему двойной рекурсии не всегда удается свести к схеме примитивной рекурсии.

Теорема Аккермана

Функция Аккермана растёт быстрее, чем любая.

Для любой 1местной ПРФ f(x) существует n из натуральных чисел: $\forall x \geq n$ A(x)>f(x).

Поскольку функция Аккермана всюду определена, то она является частным случаем ЧРФ, а именно ОРФ. То, из тезиса Чёрча следует, что термины «ОРФ» и «всюду определенная вычислимая ф-ция» являются, по сути, синонимами.

Билет 33. Словарные функции. Определение машины Тьюринга.

Основная идея: имитация алгоритмических процессов с помощью абстрактной математической машины(т.е. машины Тьюринга). Эта машина за конечное число шагов из исходных числовых данных в соответствии с заданными правилами может получить искомый числовой результат. Такая модель была предложена А.М. Тьюрингом в 1936 году.

Словарные функции.

Пусть А -набор букв(алфавит),то А={a1,...an}.Всякая конечная последовательность букв этого алфавита называется **словом**. α =a1,…,an, α -слово, a1,…,an-буквы, ai$\in$A, $|\alpha|$-длина слова(кол-во букв), λ -пустое слово, т.е. $|\lambda|=0$.

Свойства слов.

1) $\lambda * \alpha = \alpha * \lambda = \alpha$,

2) Закон ассоциативности: $(\alpha * \beta)* \gamma = \alpha *(\beta * \gamma)$,

3) Отсутствие коммутативности: $\alpha * \beta \neq \beta * \alpha$

Обозначается: A^n – множество всех слов длиной n, A* – множество всех слов, из букв алфавита A :

$A^* = \bigcup_{n=0}^{\infty} A^n$, a^n = a*a*…*a- n букв a, заметим, что $0^0 = 1^0 = \lambda$.

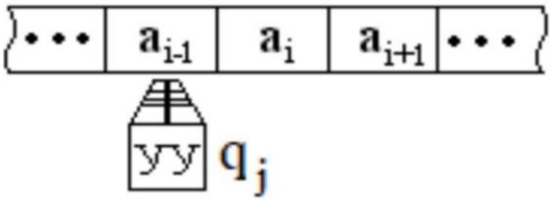
Слово, состоящее из бесконечного числа букв, называется **сверхсловом**.

A^ω – множество всех сверхслов, состоящие из букв алфавита A.

Отображение f: $A^* \to B^*$ – **словарная функция**.

Машина Тьюринга включает в себя:

1. ***Внешний алфавит*** - конечное множество символов $A = \{a_0, a_1, a_2, \ldots, a_n\}$. $a_0 = \lambda$ – пустой символ. Информация вводится в машину в виде символов этого алфавита. Введенную информацию МТ перерабатывает в новую. В качестве символьного алфавита могут быть не только буквы (часто используется разделительный символ |).
2. ***Внутренний алфавит*** - конечное множество символов $Q = \{q_0, q_1, q_2, \ldots, q_m, R(направо), L(налево), C(наместе), P(печать), E(удалить)\}$. Иногда символы C,P,E могут отсутствовать. Симфолы q выражают состояние машины: q_1 - начальное состояние машины, q_0 - заключительное состояние (стоп-состояние).
3. ***Бесконечная лента*** характеризует память машины. Она разбита на клетки. В каждую клеточку может быть записан только один символ. Клетка с символом a0 считается пустой, она всегда появляется при движении вправо или влево, когда заканчивается слово исходной информации. A0,a1,…,an,a0; a0 – пустая клетка. Для вычисления по МТ достаточно, чтобы лента была бесконечной только в одну сторону. Бесконечное вправо – правая полулента, влево – левая полулента.
4. ***Управляющее устройство*** движущееся вдоль ленты, которая может останавливаться напротив какой-либо клетки и считывать записанный там символ.



Работа машины складывается из следующих один за другим тактов, по ходу которых происходит преобразование начальной информации в промежуточные информации (к концу каждого такта совокупность знаков, хранящихся на ленте, образует соответствующую промежуточную информацию). В качестве начальной информации на ленту можно подать любую конечную систему знаков внешнего алфавита (любое слово в этом алфавите), расставленную произвольным образом по ячейкам.

В каждом такте работы (не УГ, а YY) может двигаться только на одну клетку или оставаться на месте. Полное состояние МТ, по которому однозначно модно определить её дальнейшее поведение, определяет её внутр. сост. словом, записанным на ленте. Такое состояние будем называть конфигурацией и обозначать $\alpha_1 q_i \alpha_2$, где qi текущее внутреннее состояние. α_1 - слово слева от YY, α_2 - слово, начиная с буквы, обозреваемой II. При чём все символы слева от α_1 и справа от α_2 - пустые. Конфигурацию вида a1q1 назовём стандартной начальной конфигурацией. А a0α - стандартной заключающей конфигурацией.

В каждом такте работы машины она действует по функциональной схеме, которая имеет вид:

$aiqj \Rightarrow avДqs$

Здесь *ai,av* – буквы внешнего алфавита; *qiqs* – состояния машины; Д$\in$ {R,L,C,P,E}.

Символ стирания E подразумевает запись в данную ячейку пустого символа a0.

Билет 34. Способы задания машин Тьюринга. Реализация на машине Тьюринга программы “перенос нуля”. Композиция машин Тьюринга.

Программу МТ представляют графом, протоколом или таблицей. При графовом представлении вершины графа соответствуют состояниям машины, а дугам – переходы в те состояния, которые предусмотрены командой. При этом на дуге указывают «считываемый сигналом обозреваемой ячейки/записываемый в обозрев. ячейку» и команду по перемещению II.

Граф МТ, реализующий заданный алгоритм часто называют граф-схемой алгоритма (ГСА). ГСА обеспечивает наглядность и позволяет применять различные методы исследования графов для оптимизации структуры алгоритма.

При протокольной записи все команды должны быть заданы упорядоченном списком. Для удобства текущее состояние МТ пишут перед обозреваемой ячейкой.

При табличном представлении строки описывают текущее состояние системы, а столбцы – содержимое обозреваемой ячейки, а клетками этой таблицы являются правые части команд для соответствующей пары текущего состояния машины. Табличные формы описания более компактны и позволяют применить матричные методы анализа для оптимизации структуры алгоритма.

Перенос нуля.

$q_1 001^x 0 \to q_0 01^x 00$

При табличном представлении строки описывают текущее состояние МТ, столбцы - содержимое обозреваемой ячейки, а клетками этой таблицы являются правые части команд для соответствующей пары текущего состояния МТ.

Команда	Результат
$q_1 0 \to q_2 R$	$0\, q_2 01^x 0$
$q_2 0 \to q_3 1$	$0\, q_3 11^x 0$
$q_3 1 \to q_3 R$ x+1 раз	$011^x\, q_3 0$
$q_3 0 \to q_4 L$	$01^x\, q_4 10$
$q_4 1 \to q_5 0$	$01^x\, q_5 00$
$q_5 0 \to q_б L$	$01^{x-1} q_э 100$
$q_б 1 \to q_б L$ x раз	$q_б 01^x 00$
$q_б 0 \to q_0 0$	$q_0 01^x 00$

Композиция МТ.

При вычислении на МТ композиции ф-ций f2(f1(x)) часто бывает целесообразно сначала вычислить с помощью машины T1 (с мн-вом состояний Q1={q10,q11,…,q1n} ф-цию f1, а затем на машине T2 Q2={q20,q21,…,q2n} функцию f2, используя в качестве исходных данных стандартную начальную конфигурацию, полученную из стандартной заключ. конфигурации машины T1, полученную заменой q10 на q21. Можно сказать, что мы таким образом построили композицию T=T2(T1) MT1 и MT2 с множеством внутренних состояний $Q1 \cup Q2$, где q10=q21. В этом случае программа для T1 и T2 фактически является подпрограммами для программы.

Билет 35. Неприменимость машины Тьюринга к исходной информации (привести пример). Вычислимость по Тьюрингу. Тезис Тьюринга. Теорема о связи машин тьюринга и рекурсивных функций (без доказательства).

Формирование правой части функциональной системы происходит по командам, совокупность которых образует программу МТ.

Работа МТ полностью определяется её программой:

1. Работа МТ может заканчиваться так: После конечного числа тактов машина останавливается в состоянии q0. На ленте при этом оказывается переработанная информация. Говорят, что в этом случае МТ применима к начальной информации I1 и перерабатывает её в информацию I2.
2. МТ никогда не остановится, т.е никогда не переходит в состояние q0. В этом случае МТ ***неприменима*** к начальной информации I1.

Пример:

Пример 3. Пусть A={a_0, a_1, a_2} и начальная конфигурация имеет вид $a_0a_1a_2q_1a_2a_0$, а машина Тьюринга управляется функциональной схемой, представленной в табл. 1.

Таблица 1.

	a_0	a_1	a_2
q_1	a_2Lq_3	a_1Rq_2	a_2Lq_1
q_2	a_1Cq_0	a_2Cq_1	a_1Cq_2
q_3	a_0Rq_0	a_1Rq_4	a_2Cq_1
q_4	a_1Cq_3	a_0Rq_4	a_2Rq_4

$a_0a_1a_2q_1a_2a_0$

$a_2Lq_1 \Rightarrow a_0a_1q_3a_2a_0$ - 1-я конфигурация

$a_2Lq_1 \Rightarrow a_0q_1a_1a_2a_0$

$a_1Rq_2 \Rightarrow a_0a_1q_2a_2a_0$

$a_1Cq_2 \Rightarrow a_0a_1q_2a_1a_2a_0$

$a_2Cq_1 \Rightarrow a_0a_1q_1a_1a_2a_0$ - совпадает с 1-й конфигурацией.

Таким образом, мы пришли в начальное состояние. Процесс работы машины начал повторяться, и, следовательно, конечный результат не может быть получен, т.е. данная машина Тьюринга **не применима** к исходной информации.

Пример 4. Пусть теперь для машины из примера 3 начальная конфигурация имеет вид: $a_0q_1a_1a_0$.

Далее: $a_0a_1q_2a_0$,

$a_0a_1q_0a_1a_0$.

Так как машина перешла в конечное состояние, то слово a_1a_1 – результат ее работы, и машина **применима** к исходной информации.

Таблица 1.

	a_0	a_1	a_2
q_1	a_2Lq_3	a_1Rq_2	a_2Lq_1
q_2	a_1Cq_0	a_2Cq_1	a_1Cq_2
q_3	a_0Rq_0	a_1Rq_4	a_2Cq_1
q_4	a_1Cq_3	a_0Rq_4	a_2Rq_4

Тезис Тьюринга. Любая интуитивно вычислимая функция вычислима по Тьюрингу.
Теорема о связи машин тьюринга и рекурсивных функций. Функция вычислима по Тьюрингу тогда и только тогда, когда она частично рекурсивная.
Теорема о вычислимости словарных функций. Все словарные функции, вычислимые по Тьюрингу – ЧРФ.
Опр. Словарная функция вычислима по Тьюрингу, если она вычислима на подходящей МТ.

Билет 36 . Определение нормального алгоритма Маркова и порядок его работы.
Нормальный алгоритм переводит слово «а» в слово «в», если он применим к слову «а», и не преобразует слово «а» ни в какое другое слово, если он не применим к нему.
Определение нормального алгоритма.
Будем рассматривать слова в произвольном алфавите A={a0,a1,…,an}, не содержащий пустой буквы и символов $\rightarrow$ и $\Rightarrow$. Если слово δ - подслово α, то $\delta \leq \alpha$. Часто вхождение слова δ в слово α может быть не словом.

$\alpha|_{\tau}^{\delta} = \beta, \delta \leq \alpha$ и β получается заменой в слове α первого вхождения слова δ на слово τ.

Считается, что пустое слово λ входит в любое слово α рядом с каждой его буквой. a1a2 и $\lambda a1\lambda a2$ - 2 записи одного и того же слова.

Возьмём $\delta, \tau \in A^* \Rightarrow записи \ \ \delta \rightarrow \tau, \ \ \delta \Rightarrow \tau \ (1)$ называются подстановками (простая и заключительная). При этом δ - посылка подстановки. Говорят, что подстановка (1) активна основе α, если $\delta \subseteq \alpha$.

Действие «переработка подстановкой» (1) слова α состоит в том, что слово α заменяется словом $\beta = \alpha|_{\tau}^{\delta}$. Очевидно, что действие переработкой подстановки слова может быть выполнено тогда и только тогда, когда подстановка активно настроена. В частности подстановка $\lambda \rightarrow \tau$ активна на любом слове α и $\beta = \alpha|_{\tau}^{\delta} = \tau\alpha$.

Списком подстановок называют линейно упорядоченную конечную последовательность подстановок вида (1). Обычно подстановки списка записываются столбиком, одну под другой и считают упорядоченные сверху вниз.

Считается, что «список подстановок может переработать слово α», если в нём есть подстановка, активная на основе α, и не может переработать слово α, если такой подстановки в списке нет.

Если список подстановок может переработать слово α, то действие «переработка списком подстановок слова α» состоит в том, что список просматривается по порядку, и первая активная на слове α подстановка осуществляет действие переработки слова α.

Опр: Нормальный алгоритм N задаётся списком подстановок и алфавитом A, в котором он работает. Список подстановок является функциональной схемой алгоритма. Как и МТ, нормальный алгоритм Маркова (НАМ) предназначен для решения задач определенной массовой проблемы.

Билет 37. Пример работы нормального алгоритма Маркова. Тезис Маркова. Теорема об эквивалентности машин Тьюринга и нормальных алгоритмов Маркова (без док-ва). Отличия нормальных алгоритмов Маркова от машин Тьюринга.

Пример: N1 в алф A={0,1}

$$\Phi CA: \begin{cases} 101 \Rightarrow 1 \\ 01 \to 1 \\ 0 \to 00 \end{cases}$$ - функц схема алг

a) Исх слово $\alpha_0 = 11$ - работа алгоритма заканчивается после шага 0, т.к. алгоритм не может переработать α_0. N0(11)=11

b) $\alpha_0 = 001$

Протокол:

001

011 - результат работы $N(001) = 111$ (111 переработать не может)

111

c) $\alpha_0 = 10101$

Протокол работы $\underline{10}101$ - $N(10101) = 101$ (т.к. 101 – заключительная подстановка)

101

d) $\alpha_0 = 10$

Протокол работы $\underline{10}$ - алгоритм не применим, т.к. бесконечен.

$1\underline{00}$

$10\underline{00}$

.....

Замечание: Для выполнения работы алгоритма вводят вспомогательные буквы, отсутствующие в алфавите A. Эти буквы играют главную роль в составлении протокола. При этом говорят, что алгоритм работает над алфавитом A в алфавите B, содержащем не только все элементы A, но и вспомогательные элементы. Все буквы обычно порождаются последними подстановками списка, но использовать эти подстановки часто приходится раньше других.

Тезис Маркова: Любой алгоритм в алфавите A может быть реализован некоторым нормальным алгоритмом над алфавитом A.

Теорема об эквивалентности МТ и НАМ: Какова бы ни была МТ – T (НАМ - N) в алфавите A, существует НАМ N (МТ T) над алфавитом A такой, что для всех слов $\alpha, \beta \in A^*$, справедливо: $\beta = N(\alpha)$ тогда и только тогда, когда $\beta = T(\alpha)$

Отличия НАМ от МТ:

Общая черта - в обеих алгоритмических моделях исходные данные задач представляются одним словом α_0. Размером исходных данных считается длина $|\alpha_0|$ исходного слова α_0.

Отличия этих двух моделей заключаются в следующем:

1. Пустая буква не входит в алфавит A, в котором работает нормальный алгоритм, но входит во внешний алфавит машины Тьюринга.
2. Машина Тьюринга может переработать любое имеющееся на ленте слово. Для нормального алгоритма возможна ситуация, когда он не может переработать имеющееся слово. Это один из двух стандартных признаков окончания работы нормального алгоритма.
3. Машины Тьюринга обладают свойством локальности - на каждом шаге их работы в имеющемся на ленте слове изменяется не более одной буквы. Нормальные алгоритмы свойством локальности не обладают - длина изменяющейся на каждом шаге работы алгоритма части слова может быть как угодно большой.

Билет 38. Сравнительный анализ трех типов алгоритмических моделей. Оценка сложности алгоритма.

Анализ трех типов моделей показал, что основные свойства алгоритмов дискретности, детерминированности, массовости и результативности остаются неизменными для различных способов описания. Все три алгоритмических модели эквивалентны, что позволяет рассматривать вычислительный алгоритм инвариантным к способу описания. Все алгоритмические модели обеспечивают способ получения значений вычислимой функции, причем, согласно тезису Черча, совпадают множества вычислимых и частично рекурсивных функций, а также множества всюду определенных вычислимых и общерекурсивных функций.

Различия наблюдаются в использовании конструктивных объектов.

	Рекурсивные функции	**Машины Тьюринга**	**Нормальные алгоритмы Маркова**
Конструктивные объекты	Числовые функции, определенные на множестве натуральных чисел с нулем.	Символы алфавитов внешней памяти (на информационной ленте) и внутренней памяти (состояний управляющего устройства).	Слова.
Задание процесса вычисления	Операторами суперпозиции, примитивной рекурсии и минимизации.	Протоколом, использующим функции перемещения управляющего устройства, а также функции выхода (записи нового символа на ленте) и перехода (из одного внутреннего состояния в другое).	Правилами подстановки, изменяющими состав и структуру исходного слова до искомого результата.

При реализации алгоритма с привлечением одной из трех моделей возникает задача **оценки сложности алгоритма**. Выделяют ***сложность описания*** алгоритма и ***сложность вычисления*** алгоритма.

Сложность описания алгоритма есть величина, характеризующая длину описания алгоритма.

	Рекурсивные функции	**Машины Тьюринга**	**Нормальные алгоритмы Маркова**
Сложность описания алгоритма	Число букв и символов, используемых в описании операторов.	Число команд.	Число правил подстановки.

Сложность вычисления алгоритма есть функция, дающая числовую оценку трудоемкости применения алгоритма к исходным данным для получения искомого результата. Чаще всего рассматриваются время и объем памяти, необходимые для вычисления.

Время вычисления алгоритма характеризуется произведением числа шагов алгоритма от исходных данных до искомого результата на среднее физическое время реализации одного шага алгоритма. Число шагов алгоритма определяется его описанием в данной алгоритмической модели.

Среднее физическое время зависит от типа компьютера, способов хранения и выборки данных и скорости обработки информации.

Объем памяти, как количественная характеристика алгоритма, определяется количеством единиц памяти, используемых в процессе вычисления алгоритма. Эта величина не может превосходить максимального числа единиц памяти, используемых в данном компьютере на одном шаге алгоритма.

Основной задачей программиста является разработка эффективного алгоритма, который позволял бы достичь требуемого результата при минимальных затратах времени и памяти.

Билет 39. Алгоритмически неразрешимые проблемы: проблема остановки машины Тьюринга, проблема ее самоприменимости, проблема разрешимости в исчислении предикатов. Теорема Райса (без док-ва) и ее смысл.

*Рассмотрим следующую задачу***.** По любому алгоритму A и данным a определить, приведет ли к результату работа алгоритма A при исходных данных a. Иначе говоря, нужно построить алгоритм B такой, что $B(A, a)=И$, если $A(a)$ дает результат, и $B(A, a)=Л$, если $A(a)$ не дает результата. В силу тезиса Тьюринга эту задачу можно сформулировать как задачу о построении машины Тьюринга: построить машину T_0, такую, что для любой машины Тьюринга T и любых исходных данных a для машины T $T_0(S_T, a)$=И, если машина T(a) останавливается, и $T_0(S_T, a)$=Л, если машина T(a) не останавливается. (Здесь S_T – система команд машины T). Эта задача называется проблемой остановки машины Тьюринга.

Не нужно Теорема о неразрешимости проблемы остановки для произвольной машины Тьюринга. Не существует машины Тьюринга T_0, решающей проблему остановки для произвольной машины Тьюринга T.

В силу тезиса Тьюринга невозможность построения машины Тьюринга означает отсутствие алгоритма решения данной проблемы. Поэтому полученная теорема дает первый пример *алгоритмически неразрешимой проблемы* (и потому у нее такое название!).

НЕ нужно Важное замечание. В утверждениях об алгоритмической неразрешимости речь идет об отсутствии единого алгоритма, решающего данную проблему; при этом вовсе не исключается возможность решения этой проблемы в частных случаях, но различными средствами для каждого случая. В частности, данная теорема не исключает того, что для отдельных классов машин Тьюринга проблема остановки может быть решена. Поэтому неразрешимость общей проблемы остановки вовсе не снимает необходимости доказывать сходимость предлагаемых алгоритмов, а лишь показывает, что поиск таких доказательств нельзя полностью автоматизировать. Неразрешимость проблемы остановки можно интерпретировать как несуществование общего алгоритма для отладки программ, точнее, алгоритма, который по тексту любой программы и данным для нее определял бы, зациклится ли программа на этих данных или нет. Если учесть сделанное ранее замечание, такая интерпретация не противоречит тому эмпирическому факту, что большинство программ в конце концов удается отладить, т. е. установить наличие зацикливания, найти его причину и устранить ее. При этом решающую роль играют опыт и искусство программиста.

Частным случаем проблемы остановки является проблема самоприменимости. Суть этой проблемы заключается в следующем. Программу машины Тьюринга можно закодировать каким-либо определенным шифром. На ленте машины можно изобразить ее же собственный шифр, записанный в алфавите машины. Здесь, как и в случае обычной программы возможны два случая:

- машина применима к своему шифру, т.е. она перерабатывает этот шифр и после конечного числа тактов останавливается;
- машина неприменима к своему шифру, т. е. машина никогда не переходит в стоп-состояние.

Таким образом, сами машины (или их шифры) разбиваются на два класса: самоприменимых и несамоприменимых тьюринговых машин. Проблема заключается в следующем: как по любому заданному шифру установить, к какому классу относится машина, зашифрованная им, к классу самоприменимых или несамоприменимых.

Теорема. Проблема распознавания самоприменимости алгоритмически неразрешима.

Теорема. Проблема эквивалентности слов в любом ассоциативном исчислении алгоритмически неразрешима.

Эта проблема решена лишь в некоторых ассоциативных исчислениях специального вида.

Теорема Райса. Никакое нетривиальное свойство вычислимых функций не является алгоритмически разрешимым.

Смысл этой теоремы заключается в том, что по описанию алгоритма ничего нельзя узнать о свойствах функции (является ли она периодической, ограниченной, монотонной, содержит ли среди своих значений заданное число и т.п.), которую он вычисляет. В частности, оказывается неразрешимой проблема эквивалентности алгоритмов: по двум заданным алгоритмам нельзя узнать, вычисляют они одну и ту же функцию или нет.

Напомним, что еще одной, ранее рассмотренной в данном курсе алгоритмически неразрешимой проблемой, является *проблема разрешимости формул логики предикатов*. Эта проблема решается лишь в некоторых частных случаях.

Теорема Черча. Не существует алгоритма, который для любой формулы логики предикатов устанавливает, общезначима она или нет.

Идея доказательства теоремы Черча состоит в том, чтобы в чистом исчислении предикатов описать предикат $Q(i, a, x)$: «машина Тьюринга с номером i, будучи применена к исходным данным a, закончит вычисление в момент x». Предикат $\exists x\, Q(i, a, x)$ — это предикат остановки, а $\exists x\, Q(a, a, x)$ — предикат самоприменимости; о неразрешимости обоих предикатов говорилось в первых теоремах этого параграфа.

Билет 40. Алгоритмически неразрешимые проблемы: проблема эвивалентности слов в ассоциативном исчислении и комбинаторная проблема Поста.

Еще одной алгоритмически неразрешимой проблемой является проблема эквивалентности слов для ассоциативных исчислений.

Рассмотрим некоторый алфавит $A=\{a, b, c,...\}$ и множество слов в этом алфавите. Будем рассматривать преобразования одних слов в другие с помощью некоторых допустимых подстановок $\alpha \to \beta$, где □ и □ — два слова в том же алфавите А. Если слово *y* содержит □ как подслово, например, $\alpha_1\alpha\alpha_2\alpha_3\alpha$, то возможны следующие подстановки: $\alpha_1\beta\alpha_2\alpha_3\alpha$, $\alpha_1\alpha\alpha_2\alpha_3\beta$, $\alpha_1\beta\alpha_2\alpha_3\beta$.

Ассоциативным исчислением называется совокупность всех слов в некотором алфавите вместе с какой-нибудь конечной системой допустимых подстановок. Для задания ассоциативного исчисления достаточно задать соответствующий алфавит и систему подстановок. Очевидно, что одним из примеров ассоциативного исчисления являются нормальные алгоритмы Маркова.

Если слово *R* может быть преобразовано в слово *S* путем однократного применения определенной подстановки, то *R* и *S* называются смежными словами. Последовательность слов $R_1, R_2, \ldots, R_{n-1}, R_n$ таких, что все пары слов $(R_i, R_{i+1}), i=1,2,..., n-1$ являются смежными, называется *дедуктивной цепочкой,* ведущей от слова R_1 к слову R_n. Если существует цепочка, ведущая от слова *R* к слову *S*, то *R* и *S* называются эквивалентными: $R \sim S$.

Для каждого ассоциативного исчисления возникает своя специальная проблема эквивалентности слов: для любых двух слов в данном исчислении требуется узнать, эквивалентны они или нет.

Еще одной алгоритмически неразрешимой проблемой является проблема соответствий Поста.

Определение 1. *Постовской системой соответствия* над алфавитом А называется упорядоченная пара конечных последовательностей $((x_1,..,x_n),(y_1,...,y_n))$, где $x_i \in A^*$ и $y_i \in A^*$ для всех *i*.

Замечание. Систему $((x_1,..,x_n),(y_1,...,y_n))$ иногда изображают в виде $\left[\frac{x_1}{y_1}\right],...,\left[\frac{x_n}{y_n}\right]$.

Определение 2. *Решением* постовской системы соответствия $((x_1,..,x_n),(y_1,...,y_n))$ называется непустая последовательность индексов $(i_1,..,i_k)$, удовлетворяющая условию $x_{i_1}...x_{i_k} = y_{i_1}...y_{i_k}$, где $1 \le i_j \le n$ для каждого *j*.

Не надо Пример. Пусть $A=\{a, b, c\}$. Рассмотрим постовскую систему соответствия $\left[\frac{aab}{a}\right],\left[\frac{a}{aa}\right],\left[\frac{caa}{bc}\right]$.

Последовательность $(2,1,3,2,2)$ является решением этой системы, так как
$a \cdot aab \cdot caa \cdot a \cdot a = aa \cdot a \cdot bc \cdot aa \cdot aa$, $a^3bca^4=a^3bca^4$

Определение 3. *Проблемой соответствий Поста* называется проблема нахождения алгоритма, выясняющего для каждой постовскои системы соответствия, существует ли решение этой системы.

Теорема. Пусть $|A| \square 2$. Тогда не существует алгоритма, позволяющего по произвольной постовской системе соответствия над алфавитом *A* узнать, имеет ли она решение. (Другими словами, проблема соответствий Поста неразрешима.)

Неразрешимость данной проблемы широко используется для доказательства неразрешимости других алгоритмических проблем.

Билет 41. Особенности прикладных исчислений. Аксиомы для равенства. Свойства равенства.

ИП не содержащие функциональных букв и предметных констант называются частными ИП.

Прикладные ИП характеризуются тем, что в них добавляются собственные аксиомы, содержащие индивидуальные функциональные буквы и предметные константы.

Типичные примеры индивидуальных предикатных букв: предикаты =,<,>; функциональные буквы +,-,x; предметные константы:0,1,пустое множество (0зачеркнуть)

Кроме того в системах аксиом (P1) и (P2) участвуют уже не предметные константы а произвольные …

(P1') $\forall x\, F(x) \to F(t)$,

(P2') $F(t) \to \exists x\, F(x)$, где F(t) – результат подстановки терма t в F(x) вместо всех свободных схождений x, причем все переменные t должны быть свободными в F(t).

Большинство прикладных исчислений содержит предикат равенства = и определяющие его аксиомы. Аксиомами для равенства могут служить следующие.

(E1) $\forall x\ x = x$ (конкретная аксиома),

(E2) $(x = y) \to (F(x, x) \to F(x, y))$ (схема аксиом),

где $F(x, y)$ – получается из $F(x, x)$ заменой некоторых (не обязательно всех) вхождений x на y при условии, что y в этих вхождениях также остается свободным. Всякая теория, в которой (E1) и (E2) являются теоремами или аксиомами, называется *теорией* (или исчислением) *с равенством.* Дело в том, что из (E1) и (E2) выводимы **основные свойства равенства** - рефлексивность, симметричность и транзитивность.

Теорема. В любой теории с равенством:

1) □□ $t=t$ для любого терма t;

2) □□ $x = y \to y = x$;

3) □□ $x = y \to (y = z \to x = z)$.

Доказательство.

1) Непосредственно следует из аксиом (E1) и (P1') (где $F(x)$ имеет вид $x=x$) по правилу заключения.

2) Из (E2) имеем $(x = y) \to (x = x \to y = x)$. Отсюда $x=y$, $x=x$ □□ $y=x$ (двойное применение правила заключения). Но так как □□ $y=x$, то $x=y$ □□ $y=x$ и, следовательно, по теореме дедукции $x = y \to y = x$.

3) Поменяем местами в (E2) x и y, в качестве $F(y, y)$ возьмем $y=z$, а в качестве $F(y, x)$ возьмем $x=z$. Получим $y = x \to (y = z \to x = z)$ и по правилу заключения $y=x$ □□ $(y = z \to x = z)$. Но так как в силу п. 2 $x=y$ □□ $y=x$, то по правилу силлогизма получаем $x=y$ □□ $(y = z \to x = z)$ и по теореме дедукции □□ $(x = y) \to (y = z \to x = z)$.

Три свойства равенства, полученные этой теоремой, верны, как известно, для любого отношения (и, следовательно, соответствующего предиката) эквивалентности. Схема аксиом (E2) выражает более сильное свойство, присущее лишь равенству: неотличимость элементов, для которых выполняется равенство.

Билет 42. Формальная арифметика как прикладное исчисление предикатов. Аксиомы формальной арифметики и их смысл.

Наиболее изученной формальной теорией, которая играет фундаментальную роль в основаниях математики, является ***формальная арифметика*** – прикладное исчисление предикатов, в котором имеются:

1. Предметная константа 0.

2. Двухместные факторы + и , одноместный фактор ‘.

3. Двухместный предикат =.

4. Собственные аксиомы (схемы аксиом):

A1. $F(0) \& \forall x (F(x) \to F(x')) \to \forall x F(x)$ – принцип индукции,

A2. $t'_1 = t'_2 \to t_1 = t_2$,

A3. $\overline{t' = 0}$,

A4. $t_1 = t_2 \to (t_1 = t_3 \to t_2 = t_3)$,

A5. $t_1 = t_2 \to t'_1 = t'_2$,

A6. $t + 0 = t$,

A7. $t_1 + t'_2 = (t_1 + t_2)'$,

A8. $t \cdot 0 = 0$,

A9. $t_1 \cdot t'_2 = t_1 \cdot t_2 + t_1$.

В этих аксиомах использованы три функциональных символа +, ·, ', один индивидуальный предикат (предикатная буква) = и одна предметная константа 0. Они придают аксиомам A2–A9 вполне конкретный вид: все предикатные и функциональные буквы в них зафиксированы, и единственный способ их варьировать – это подставлять различные термы вместо метапеременных t, t_1, t_2. В частности, схемы A6–

A9 – это просто предикат равенства, в который подставлены термы определенного вида. Схема A1 имеет обычный вид: $F(x)$ – метаобозначение, употребляемое в том же смысле, в каком оно использовалось в чистом исчислении предикатов. Впрочем, схемы A2–A9 можно заменить конкретными аксиомами (т.е. формулами самой арифметики): A2'. $x'_1 = x'_2 \to x_1 = x_2$ и т.д., из которых любые формулы а A2–A9 можно получить с помощью правила обобщения и схемы аксиом (P1).