Программа-минимум

1. Основные равносильности (законы) логики высказываний.

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&$\overline{A}$ ≡Л

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

6.Закон двойного отрицания: $\overline{\overline{A}}$ ≡ 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: $\overline{A\&B}$ ≡ $\overline{A}$V$\overline{B}$; $\overline{AVB}$ ≡ $\overline{A}\&\overline{B}$

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

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

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

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

2. Метод редукций проверки тождественной истинности формул логики высказываний.

Алгоритм редукции пытается найти значения пропозициональных переменных формулы φ, при которых значение функции f_ϕ равно 0, на основе того, что импликация является ложной в том и только в том случае, когда посылка истинна, а заключение ложно.

Пример Проверить общезначимость формулы

$f(A,B) = (A \to B) \to ((A \to \neg B) \to \neg A)$

$f = 0\ A \to B = 1, ((A \to \neg B) \to \neg A) = 0$

1. Пусть $(A \to \neg B) = 1, \neg A = 0$
2. значит при $A=1$ функция ложна. Найдем B:
3. A ® Ø B = 1, A = 1Þ B = 0
4. проверим, подставив в первую посылку: $A \to B = 1 \to 0 = 0$ ¹ 1, получили противоречие, значит, предположение не верно и формула тождественно истинна.

3. Определение формальной теории.

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

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

4. Аксиомы исчисления высказываний.

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→$\bar{B}$)→$\bar{A}$)
I.10 (из двойного отрицания А следует А) $\bar{\bar{A}}$ ->A

Система II.
II.1≡I.1
II.2≡I.2
II.3≡($\bar{A}$→$\bar{B}$)→(($\bar{A}$→B)→A

5. Основные правила вывода исчисления высказываний.

Правила вывода
1)Правило подстановки : если α(A) - формула, содержащая А, то выводима и формула α(β), получающаяся из α заменой всех вхождений А 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 А, β на A&B, то из α→β получим А→(A&B).

6. Производные правила вывода в исчислении высказываний: правило введения импликации, теорема дедукции, правило силлогизма, правило введения отрицания.

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

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

4. Докажем, что в ИВ справедлив метод доказательства от противного: Если Г,А │-В и Г,А │-$\bar{B}$, то Г|- $\bar{A}$. Получили правило введения отрицания.

7. Определение предиката. Свободные и связанные переменные.

Предикат – это функция $P(x_1, x_2, \dots, x_n)$, 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.

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

8. Основные равносильности (законы) логики предикатов.

Основные равносильности ЛП:

1. Перестановка одноименных кванторов

$\forall -$ квантор всеобщности $\quad \exists -$ квантор существования

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

Однако разноименные кванторы переставлять нельзя, пример:

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

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

9.Аксиомы исчисления предикатов.

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.

10. Основные правила вывода исчисления предикатов.

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).

11. 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.

Правило связывания кванторов.

$F(x) | -_{K} \forall x F(x)$

12. Теорема об общезначимых формулах (в исчислении высказываний и в исчислении предикатов).

Теорема об общезначимых формулах

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

13. Примитивно рекурсивные функции: базовые функции и элементарные операции.

Базовые функции.
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) $\lambda(x) = x + 1,\ x' = x + 1$

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)$\varphi(x_1 \dots x_n) =\ \varphi(f_1(x_1 \dots x_n),\dots, f_m(x_1 \dots x_n))$
Если заданы функции тождества $I_m^n(x_1, \dots, x_n) = x_m$ 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 φи ψ, но и через значение самой функции f во всех предшествующих точках.

14. Определение и примеры примитивно рекурсивных функций.

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 числовые функции от её исходного значения при заданных значениях независимых переменных, называют рекурсивным описанием (протоколом).

Примеры:

1) Базовые функции (см вопрос 27)
2) $0(x_1, x_2, \dots, x_n) = 0$ – ПРФ, так как $0(x_1, x_2, \dots, x_n)=0(I_n^1(x_1),\dots,I_n^n(x_n))$, т.е. получаем из базовых функций 0(x) и I_n с помощью операции суперпозиции.
3) f(x)=x+n явл ПРФ т.к. $f(x)=\lambda(\lambda(\lambda \dots (x) \dots))$ (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:

$$x \dot{-} y = \begin{cases} x - y, x \geq y, \\ 0, x < y. \end{cases} \quad \begin{cases} f(x,0) = x \dot{-} 0 = x = I_1^1(x,y); \\ f(x,y+1) = x \dot{-} (y+1) = (x \dot{-} y) \dot{-} 1 = f(x,y) \dot{-} 1. \end{cases} \quad \begin{cases} \varphi(x) = x; \\ \psi(x,y,z) = z \dot{-} 1. \end{cases}$$

7) Сигнум

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

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

К этим двум функциям можно ввести дополнительные:

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

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

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. $y \leq z$ и $f(x_1 \dots x_{n-1}, y) = 0(1)$если такое y существует и равен $z +$ 1, если не существует y. (условие 1)

Неограниченный оператор минимизации

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

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

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

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

Существует 3 варианта выполнения этих условий:

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

16. Определения общерекурсивных и частично рекурсивных функций.

Всюду определённая ЧРФ называется общерекурсивной ОРФ $\subset$ ЧРФ

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.

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)

17. Определение и способы задания 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.

Способы задания:

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.

18. 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.

Нормальный алгоритм Маркова - один из стандартных способов формального определения понятия алгоритма.

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 $\beta = \alpha|_{\tau}^{\delta} = \tau\alpha$.

Порядок работы НАМ N над словом a0 принадлежащее A* состоит из выполнения одним за другим однотипных шагов. На шаге i=0 зарождается исходное слово, на последующих шагах функциональная схема алгоритма применяется к имеющемуся слову α_{i-1} и перерабатывает его в новое слово α_i

$\alpha_{i-1} \mid \rightarrow \alpha_i$

Таким образом в процессе работы алгоритма получается последовательность слов:

$\alpha_0, \alpha_1, \ldots \alpha i - 1, \alpha_i, \ldots$

Процесс работы НАМ заканчивается после выполнения шага K=0,1,2,... На слове α_k, если на шаге k осуществимо действие заключительной перестановки или на шаге k+1 список подстановок не может переработать слово α_k.
α_k =N(α_0) – результат работы НАМ.

Если процесс работы никогда не окончится, то нормальный алгоритм N не применим к слову α_0.

19. Тезисы Черча, Тьюринга и Маркова о вычислимых функциях.

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!

Тезис Тьюринга. Любая интуитивно вычислимая функция вычислима по Тьюрингу.

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

20. Теорема Райса и ее смысл.

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.