1. Классическая логика предикатов первого порядка

Определение. *Предикат* – форма, при помощи которой задаются отношения между объектами и субъектами
Слово «предикат» происходит от латинского «предсказывать».
Предикаты нулевого порядка – без использования кванторов.
Предикаты первого порядка – кванторы используются только по отношению к предметам
Предикаты высшего порядка – кванторы используются по отношению к функциям.

Определение. *Алфавит (сигнатура) КЛП* – это набор счетных множеств:
1. *предметных переменных,* которое обозначается как $\mathrm{Var} = \{x_1, x_2, ..., x_n, ...\}$;
2. *предметных констант*, которые соответствуют именам предметов и обозначаются как $\mathrm{Const} = \{c_1, c_2, ..., c_n, ...\}$;
3. *функциональных символов* $\mathrm{Func} = \{f_1^{k_1}, f_2^{k_2}, ..., f_m^{k_m}, ...\}$, где k_i – *местность* функционального символа, причем $k_i \geq 1$ (в противном случае функциональный символ является константой);
4. *предикатных символов,* которые используются для обозначения отношений между предметами и обозначаются $\mathrm{Pred} = \{P_1^{l_1}, P_2^{l_2}, ..., P_r^{l_r}, ...\}$, где l_i – *местность* предикатного символа (если $l_i = 0$, то данный предикатный символ обозначает утверждение, не зависящее от каких-либо предметов).

Служебные символы – это:
- логические связки: &, ∨, →, ¬;
- кванторы: ∀ (всеобщности) и ∃ (существования);
- знаки препинания: ( ) ,

Слова в алфавите – это цепочки символов.

Определение. *Термом* называется всякое слово, которое может быть построено по следующим правилам:
1) любая переменная является термом;
2) любая константа является термом;
3) если *f* – k-местный функциональный символ, а $t_1, t_2, ..., t_k$ - термы, то $f(t_1, t_2, ..., t_k)$ также будет являться термом;
4) других термов нет.

Множество всех термов обозначается как Term.

Запись $t(x_1, x_2, ..., x_n)$ используется для обозначения терма, в котором содержатся только переменные из множества $\{x_1, x_2, ..., x_n\}$.

Если *t* – терм, то выражением Var_t будем обозначать множество всех переменных, которые содержатся в терме *t*.

Если Var_t является пустым множеством, то терм *t* называется *остовным термом*.

Определение. *Формулой* называется слово в языке КЛП, которое может быть построено по следующим правилам:

1) если *P* – m-местный предикат, а $t_1, t_2, ..., t_m$ - термы, то запись вида $P(t_1, t_2, ..., t_m)$ будет являться формулой (*атомарной формулой*);
2) если ψ и φ - формулы, то формулой также будет являться любое выражение вида $\neg\varphi, \neg\psi, \varphi \vee \psi, \varphi \& \psi, \varphi \to \psi$;
3) если φ - формула, а *x* – предметная переменная, то формулой также будет являться любое выражение вида $\forall x\varphi$, $\exists x\varphi$;
4) никаких других формул нет.

В формулах наибольший приоритет имеют кванторы и отрицание, затем конъюнкция, дизъюнкция и импликация.

Множество всех формул обозначается как Form.

Определение. *Областью действия кванторов* называется формула φ из выражения $\forall x\varphi$ или $\exists x\varphi$. При этом вхождение переменной *x* в этих выражениях называется *связанным*.

Если вхождение переменной не связанное, то оно называется *свободным*.

Определение. *Связанной (свободной)* переменной называется переменная *x*, если она имеет связанное (свободное) вхождение в некоторую формулу.

Запись Var_φ обозначает множество всех свободных переменных, входящих в формулу φ.

Определение. Если множество Var_φ пусто, то формула φ называется *замкнутой формулой (предложением)*.

Смысловое содержание языка логики предикатов определяется его семантикой. При этом описание выражений естественного языка является гораздо более сложной задачей, нежели описание некоторых математических утверждений.

Определение. *Интерпретация* – математическая конструкция, которая позволяет описывать все многообразие воображаемых миров. Интерпретацией будем называть алгебраическую систему $I = < D, \overline{\mathrm{Const}}, \overline{\mathrm{Func}}, \overline{\mathrm{Pred}} >$, которая состоит из следующих компонент:
- $D_I \neq \varnothing$ *- область интерпретации (предметное множество, универсум)*, которое представляется всеми возможными предметами воображаемого мира;
- $\overline{\mathrm{Const}}$ - отображение $c \in \mathrm{Const} \to \overline{c} \in D_I$ (предмет в мире *I,* носящий имя константы *c*)
- $\overline{\mathrm{Func}}$ - отображение $\overline{\mathrm{Func}} : f^{(n)} \in \mathrm{Func} \to (\overline{f^{(n)}} : D_I \to D_I)$;
- $\overline{\mathrm{Pred}}$ - отображение $\overline{\mathrm{Pred}} : P^{(m)} \in \mathrm{Pred} \to (\overline{P^{(m)}} : D_I \to \{T, F\})$.
Таким образом, все элементы нашего языка приобретают четкий математический смысл.

На основании понятия интерпретации можно оценивать все формулы логики предикатов.

Определение. Пусть *I* – интерпретация, φ - формула от n переменных, $d_1, d_2, ..., d_n$ – набор предметов. Тогда *отношением выполнимости* называется следующее отношение: $I \models \varphi(x_1, x_2, ..., x_n)[d_1, d_2, ..., d_n]$, которое обозначает «формула φ в интерпретации *I* выполняется на значениях $d_1, d_2, ..., d_n$ ее свободных переменных» и определяется следующим образом:
1. Значение терма на данной интерпретации:

Если *t* – терм, $d_1, d_2, ..., d_m$ – предметы, то $t(x_1, x_2, ..., x_m)[d_1, d_2, ..., d_m]$ - это предмет из области интерпретации, который является значением терма:
- если $t = x_i$, то $t(x_1, x_2, ..., x_m)[d_1, d_2, ..., d_m]$ будет равен d_i;
- если *t* – это константа *c*, то $t(x_1, x_2, ..., x_m)[d_1, d_2, ..., d_m] = \overline{c}$;
- если $t = f^{(k)}(t_1, t_2, ..., t_k)$, то
$$t(x_1, x_2, ..., x_m)[d_1, d_2, ..., d_m] = \overline{f^{(k)}}(t_1(x_1, x_2, ..., x_m)[d_1, d_2, ..., d_m], ..., t_k(x_1, x_2, ..., x_m)[d_1, d_2, ..., d_m]).$$
2. Отношение выполнимости формул
Если φ - атомарная формула вида $P^{(k)}(t_1, t_2, ..., t_k)$, то выполнимость этой формулы эквивалентна истинности интерпретации предиката *P*.
Если φ - формула вида $\neg\psi, \psi_1 \vee \psi_2, \psi_1 \& \psi_2, \psi_1 \to \psi_2$, то ее выполнимость эквивалентна
1) в случае $\psi_1 \& \psi_2$: $I \models \varphi(x_1, x_2, ..., x_n)[d_1, d_2, ..., d_n] \Leftrightarrow \begin{cases} I \models \psi_1(x_1, x_2, ..., x_n)[d_1, d_2, ..., d_n] \\ I \models \psi_2(x_1, x_2, ..., x_n)[d_1, d_2, ..., d_n] \end{cases}$
2) в случае $\psi_1 \vee \psi_2$: $I \models \varphi(x_1, x_2, ..., x_n)[d_1, d_2, ..., d_n] \Leftrightarrow \left[\begin{array}{l} I \models \psi_1(x_1, x_2, ..., x_n)[d_1, d_2, ..., d_n] \\ I \models \psi_2(x_1, x_2, ..., x_n)[d_1, d_2, ..., d_n] \end{array}\right.$
3) в случае $\psi_1 \to \psi_2$: $I \models \varphi(x_1, x_2, ..., x_n)[d_1, d_2, ..., d_n] \Leftrightarrow \left[\begin{array}{l} I \not\models \psi_1(x_1, x_2, ..., x_n)[d_1, d_2, ..., d_n] \\ I \models \psi_2(x_1, x_2, ..., x_n)[d_1, d_2, ..., d_n] \end{array}\right.$
4) в случае $\neg\psi$: $I \models \varphi(x_1, x_2, ..., x_n)[d_1, d_2, ..., d_n] \Leftrightarrow I \not\models \psi(x_1, x_2, ..., x_n)[d_1, d_2, ..., d_n]$.
Если φ - формула вида $\forall x_0\psi(x_0, x_1, ..., x_n)$, $\exists x_0\psi(x_0, x_1, ..., x_n)$, то ее выполнимость эквивалентна
1) в случае $\forall x_0\psi(x_0, x_1, ..., x_n)$:
$I \models \varphi(x_1, x_2, ..., x_n)[d_1, d_2, ..., d_n] \Leftrightarrow \forall d_0 \in D_I I \models \psi(x_0, x_1, ..., x_n)[d_0, d_1, d_2, ..., d_n]$,
2) в случае $\exists x_0\psi(x_0, x_1, ..., x_n)$:
$I \models \varphi(x_1, x_2, ..., x_n)[d_1, d_2, ..., d_n] \Leftrightarrow \exists d_0 \in D_I I \models \psi(x_0, x_1, ..., x_n)[d_0, d_1, d_2, ..., d_n]$.

Определение. Формула φ называется *выполнимой* в интерпретации *I*, если для некоторого набора предметов $d_1, d_2, ..., d_n$ $I \models \varphi(x_1, x_2, ..., x_n)[d_1, d_2, ..., d_n]$.

Определение. Формула φ называется *истинной* в интерпретации *I*, если для любого набора предметов $d_1, d_2, ..., d_n$ $I \models \varphi(x_1, x_2, ..., x_n)[d_1, d_2, ..., d_n]$.

Определение. Формула φ называется *невыполнимой (*или *противоречивой)* в интерпретации *I*, если она не является выполнимой (т.е. если эта формула соответствует тождественно ложному утверждению).

Определение. Формула φ называется *общезначимой*, если она является истинной в любой интерпретации. Общезначимость формулы обозначается $\models\varphi$.

Выполнимые, но не общезначимые формулы наиболее интересны для рассмотрения, тогда как общезначимые формулы обычно мало информативны.

Определение. Пусть $\Gamma \subseteq \mathrm{Form}$ - множество замкнутых формул. Тогда интерпретация *I* называется *моделью* для множества Γ, если любая формула из Γ выполнима в данной интерпретации.

Определение. Пусть φ_0 – замкнутая формула, а $\Gamma \subseteq \mathrm{Form}$ - множество замкнутых формул. Тогда φ_0 называется *логическим следствием* Γ (обозначается $\Gamma \models \varphi_0$), если каждая модель для Γ является моделью для φ_0.

Сведем задачу нахождения логического следствия к проблеме общезначимости.

Теорема (о логическом следствии). Если $\varphi_0, \varphi_1, ..., \varphi_n$ - замкнутые формулы, то $\{\varphi_1, ..., \varphi_n\} \models \varphi_0$ эквивалентно $\models (\varphi_1 \& ... \& \varphi_n) \to \varphi_0$ (т.е. для любого логического следствия данного множества формул такая импликация общезначима).

Множество всех логических следствий – это и есть множество всех логических законов.

Замечание. Существуют формулы, выполнимые на конечных интерпретациях и невыполнимые на бесконечных.
Рассмотрим формулу $\varphi_0 = \varphi_1 \& \varphi_2 \to \varphi_3$, где $\varphi_1 = \forall x \neg R(x, x)$,
$\varphi_2 = \forall x \forall y \forall z \left(R(x, y) \& R(y, z) \to R(x, z)\right)$, $\varphi_3 = \exists x \forall y \neg R(x, y)$.
Теорема: На любой интерпретации *I*, такой, что D_I - конечное множество, φ_0 выполнима, однако φ_0 не общезначима.

Определение. Упорядоченная пара множеств формул $< \Gamma, \Delta >$ называется *семантической таблицей*.
Если пересечение Γ и Δ не пусто, то такая семантическая таблица называется *закрытой*.
Если φ - формула, то *таблицей для формулы* φ назовем семантическую таблицу $T_\varphi = < \varnothing, \{\varphi\} >$

Содержательный смысл семантической таблицы таков: Γ – множество формул, которые мы хотим считать истинными, Δ - множество формул, которые мы хотим считать ложными. Тогда закрытая таблица противоречива, а таблица для формулы φ будет являться исходным состоянием для доказательства общезначимости формулы φ от противного.

Определение. Таблица $T = < \Gamma \mid \Delta >$ называется *выполнимой*, если существует интерпретация *I*, такая, что любая формула φ из множества Γ будет выполнима в данной интерпретации, а любая формула ψ из множества Δ выполнимой в данной интерпретации не будет.

Утверждение. Закрытая таблица не является выполнимой.

Для доказательства общезначимости формулы, следует преобразовать ее таблицу. Эта операция называется *табличным выводом*.

Определение. *Правилом табличного вывода* называется фигура вида $\dfrac{T_0}{T_1(T_2)}$.

Введем следующую систему правил:
1. $L\neg$: $\dfrac{< \neg\varphi, \Gamma \mid \Delta >}{< \Gamma \mid \varphi, \Delta >}$. L означает, что мы преобразовываем левую часть таблицы, причем отрицание – главная логическая связка в формуле φ.

2. $R\neg$: $\dfrac{< \Gamma \mid \neg\varphi, \Delta >}{< \varphi, \Gamma \mid \Delta >}$.
3. $L\&$: $\dfrac{< \varphi_1 \& \varphi_2, \Gamma \mid \Delta >}{< \varphi_1, \varphi_2, \Gamma \mid \Delta >}$.
4. $R\&$: $\dfrac{< \Gamma \mid \varphi_1 \& \varphi_2, \Delta >}{< \Gamma \mid \varphi_1, \Delta >< \Gamma \mid \varphi_2, \Delta >}$. В данном случае происходит ветвление процесса табличного вывода, и выполнимость T_0 эквивалентна выполнимости или T_1, или T_2.
5. $L\vee$: $\dfrac{< \varphi_1 \vee \varphi_2, \Gamma \mid \Delta >}{< \varphi_1, \Gamma \mid \Delta >< \varphi_2, \Gamma \mid \Delta >}$.
6. $R\vee$: $\dfrac{< \Gamma \mid \varphi_1 \vee \varphi_2, \Delta >}{< \Gamma \mid \varphi_1, \varphi_2, \Delta >}$.
7. $L\to$: $\dfrac{< \varphi_1 \to \varphi_2, \Gamma \mid \Delta >}{< \varphi_2, \Gamma \mid \varphi_1, \Delta >}$.
8. $R\to$: $\dfrac{< \Gamma \mid \varphi_1 \to \varphi_2, \Delta >}{< \varphi_1, \Gamma \mid \varphi_2, \Delta >}$.

С помощью данных правил можно существенно упростить таблицу. Для упрощения формул с кванторами требуется ввести дополнительные понятия.

Определение. *Подстановкой* называется любое отображение $\theta : \mathrm{Var} \to \mathrm{Term}$. *Областью подстановки* называется множество $\mathrm{Dom}_\theta = \{x : \theta(x) \neq x\}$. Если область подстановки конечна, то такая подстановка называется *конечной*. Множество всех конечных подстановок будем обозначать как Subst. Если θ - конечная подстановка, то она представима в виде множества *связок*: $\theta = \{{}^{x_1}/_{\theta(x_1)}, {}^{x_2}/_{\theta(x_2)}, ..., {}^{x_n}/_{\theta(x_n)}\}$.

Определение. Пусть *E* – логическое выражение θ - постановка $\theta = \{{}^{x}/_{t}\}$. Тогда запись вида $E\theta$ будет называться *результатом применения подстановки* θ *к E*. Он определяется следующим образом:
1) *E* – переменная ($E = y \in \mathrm{Var}$). Тогда $E\theta = \begin{cases} t, y \equiv x \\ y, y \neq x \end{cases} = \theta(y)$.
2) *E* – константа (*E = c*). Тогда $E\theta = E$.
3) *E* – составной терм ($E = f(t_1, t_2, ..., t_k)$). Тогда $E\theta = f(t_1\theta, t_2\theta, ..., t_k\theta)$.
4) *E* – атомарная формула ($E = P(t_1, t_2, ..., t_k)$). Тогда $E\theta = P(t_1\theta, t_2\theta, ..., t_k\theta)$.
5) *E* – формула вида $\varphi_1 \bullet \varphi_2$, где $\bullet$ означает конъюнкцию, дизъюнкцию или импликацию. Тогда $E\theta = \varphi_1\theta \bullet \varphi_2\theta$, а если *E* – формула вида $\neg\varphi$, то $E\theta = \neg\varphi\theta$.
6) *E* – формула с квантором $\forall y\varphi$ или $\exists y\varphi$. Тогда $E\theta = \begin{cases} E, y \equiv x \\ \substack{\forall \\ \exists} y\varphi\theta, y \neq x \end{cases}$.

Определение. Переменная *x* является *свободной для терма t в формуле* φ, если в φ нет свободных вхождений переменной *x* в области действия кванторов, связывающих переменные из множества Var_t (множества переменных терма *t*).

На основе понятия подстановки введем 4 новых правила табличного вывода:

9. $L\forall$: $\dfrac{< \forall x\varphi(x), \Gamma \mid \Delta >}{< \forall x\varphi(x), \varphi(x)\{{}^{x}/_{t}\}, \Gamma \mid \Delta >}$ при условии, что переменная *x* в формуле $\varphi(x)$ свободна для терма *t*.
10. $R\forall$: $\dfrac{< \Gamma \mid \forall x\varphi(x), \Delta >}{< \Gamma \mid \varphi(x)\{{}^{x}/_{c}\}, \Delta >}$, где *c* – это константа, которая не содержится в Γ, Δ или $\varphi(x)$.
11. $L\exists$: $\dfrac{< \exists x\varphi(x), \Gamma \mid \Delta >}{< \varphi(x)\{{}^{x}/_{c}\}, \Gamma \mid \Delta >}$.
12. $R\exists$: $\dfrac{< \Gamma \mid \exists x\varphi(x), \Delta >}{< \Gamma \mid \exists x\varphi(x), \varphi(x)\{{}^{x}/_{t}\}, \Delta >}$
Таким образом, имеем 12 правил, располагая которыми, можно построить табличный вывод.

Определение. *Табличным выводом семантической таблицы T_0* называется корневое дерево, вершины которого помечены таблицами, причем это дерево удовлетворяет следующим требованиям:
1. Корневая вершина помечена таблицей T_0.
2. Каждая висячая вершина должна быть помечена либо закрытой таблицей, либо такой таблицей, что в ней содержатся только атомарные формулы (т.е. без связок и кванторов).
3. Не висячая вершина, помеченная таблицей T_i, обязательно имеет одного или двух наследников, которые получаются из T_i по одному из 12 правил табличного вывода.
Дерево (или вывод) считается *успешным* (успешный вывод называется *доказательством*), если каждая ветвь дерева завершается закрытой таблицей.

Для табличного вывода как для доказательства общезначимости сразу возникают следующие вопросы:
- корректности;
- полноты (является ли табличный вывод универсальным способом доказательства);
- эффективности (каким образом и в каком порядке следует применять правила табличного вывода).

Теорема (корректности): Если таблица T_0 имеет успешный табличный вывод, то T_0 невыполнима.
Следствие 1. Если таблица $T_\varphi = < \varnothing, \{\varphi\} >$ имеет успешный табличный вывод, то формула φ общезначима.
Следствие 2. Если таблица $T_\varphi' = < \{\varphi\}, \varnothing >$ имеет успешный табличный вывод, то формула φ является противоречивой.

Теорема (полноты табличного вывода): Если таблица T_0 невыполнима, то T_0 имеет успешный табличный вывод.
Следствие 1 (теорема Гёделя о полноте): Если формула φ общезначима, то таблица $T_\varphi = < \varnothing, \{\varphi\} >$ имеет успешный табличный вывод.
Следствие 2 (теорема Ливенгейма-Сколема о счетной модели): Если формула φ выполнима, то φ имеет модель с не более, чем счетной бесконечной областью интерпретации.
Следствие 3 (теорема Мальцева о компактности): Пусть $\Gamma \subseteq \mathrm{Form}$ - произвольное множество формул. Тогда Γ имеет модель тогда и только тогда, когда каждое ее подмножество имеет модель.
Доказательство утверждения предлагается провести самостоятельно.

2. Метод резолюций

Метод резолюций – один из наиболее эффективных способов автоматического доказательства теорем. Общие принципы доказательства утверждений с помощью метода резолюций таковы:
1. Если мы хотим доказать общезначимость формулы φ, то для этого достаточно показать, что формула $\varphi_1 = \neg\varphi$ противоречива (т.е. провести доказательство от противного).
2. Формула φ_1 приводится к предваренной нормальной форме φ_2, в которой все кванторы стоят в начале формулы.
3. Формула φ_2 приводится к сколемовской стандартной форме, в которой отсутствуют кванторы существования: $\varphi_3 = \forall x_1 \forall x_2 ... \forall x_n (D_1 \& D_2 \& ... \& D_n)$, где $D_i = L_1 \vee L_2 \vee ... \vee L_m$, а $L_j = \begin{Bmatrix} A \\ \neg A \end{Bmatrix}$. Следует убедиться, что содержимое φ не было потеряно.
4. Доказательство общезначимости φ сводится к доказательству противоречивости системы дизъюнктов $S = \{D_1, D_2, ..., D_n\}$. Для этого применяется *правило резолюций Робенсона*: $\{D' \vee L, D'' \vee \neg L\} \Rightarrow \{D' \vee D'' = \varnothing\}$. Чтобы применение этого правила стало возможным, Робенсон ввел *алгоритм унификации* для системы $\{D' \vee L', D'' \vee \neg L''\}$. Если L' и L'' унифицированы, то такой метод построения вывода для общезначимой формулы станет полным.

Введем новую связку $\varphi \equiv \psi$, которая эквивалентна формуле $(\varphi \to \psi) \& (\psi \to \varphi)$.
Определение. Формулы φ и ψ называются *равносильными*, если общезначима формула $\varphi \equiv \psi$.

Утверждение. Отношение равносильности является отношением эквивалентности.

Утверждение. Если формулы φ и ψ равносильны, то:
1. Из общезначимости φ следует общезначимость ψ;
2. Из выполнимости φ следует выполнимость ψ;
3. Если *I* является моделью для φ, то она является также моделью для ψ.

Теорема: Следующие пары формул являются равносильными:
1) $\models (\varphi \to \psi) \equiv (\neg\varphi \vee \psi)$
2) $\models (\varphi \mathbin{\substack{\& \\ \vee}} \psi) \equiv (\psi \mathbin{\substack{\& \\ \vee}} \varphi)$
3) $\models (\varphi \mathbin{\substack{\& \\ \vee}} (\psi \mathbin{\substack{\& \\ \vee}} \chi)) \equiv ((\varphi \mathbin{\substack{\& \\ \vee}} \psi) \mathbin{\substack{\& \\ \vee}} \chi)$
4) $\models (\varphi \mathbin{\substack{\& \\ \vee}} \varphi) \equiv \varphi$
5) $\models (\varphi \mathbin{\substack{\vee \\ \&}} (\psi \mathbin{\substack{\& \\ \vee}} \chi)) \equiv (\varphi \mathbin{\substack{\vee \\ \&}} \psi) \mathbin{\substack{\& \\ \vee}} (\varphi \mathbin{\substack{\vee \\ \&}} \chi)$
6) $\models \neg\neg\varphi \equiv \varphi$
7) $\models \neg(\varphi \mathbin{\substack{\& \\ \vee}} \psi) \equiv (\neg\varphi \mathbin{\substack{\vee \\ \&}} \neg\psi)$

8) $\models \forall x\varphi(x) \equiv \forall y(\varphi(x)\left\{{}^{x}/_{y}\right\})$; *y* не содержится в $\varphi(x)$
9) $\models \neg(\substack{\forall \\ \exists} x\varphi) \equiv \substack{\exists \\ \forall} x(\neg\varphi)$
10) $\models (\substack{\forall \\ \exists} x\varphi) \mathbin{\substack{\& \\ \vee}} \psi \equiv \substack{\forall \\ \exists} x(\varphi \mathbin{\substack{\& \\ \vee}} \psi)$ $\quad x \notin \mathrm{Var}_\psi$
11) $\models \forall x(\varphi \mathbin{\substack{\& \\ \vee}} \psi) \equiv (\forall x\varphi) \mathbin{\substack{\& \\ \vee}} (\forall x\psi)$

Определение. Запись $\varphi[\psi]$ означает, что *в формуле* φ *содержится подформула* ψ, запись $\varphi\left[{}^{\psi}/_{\chi}\right]$ означает *замену в формуле* φ *подформулы* ψ *на* χ.

Теорема: Пусть φ, ψ, χ - произвольные формулы. Тогда из равносильности ψ и χ следует, что $\models \varphi \equiv \varphi\left[{}^{\psi}/_{\chi}\right]$.

Определение. *Литера* – это либо атом $A = P(t_1, t_2, ..., t_n)$, либо $\neg A$.

Определение. *Дизъюнкт* – это либо дизъюнкция атомов $L_1 \vee L_2 \vee ... \vee L_n$, либо *пустой дизъюнкт* □ - тождественная ложь.

Определение. *Предваренная нормальная форма* – это формула вида $Q_1x_1Q_2x_2...Q_nx_nM(x_1, x_2, ..., x_n)$, где $Q_i \in \{\forall, \exists\}$ - кванторная приставка, а *M* – бескванторная формула (*матрица*), являющаяся КНФ вида $D_1 \& D_2 \& ... D_n$, где D_i – непустой дизъюнкт.

Теорема (о предваренной нормальной форме): Для любой замкнутой формулы φ существует равносильная ей предваренная нормальная форма ψ.

Определение. *Сколемовская стандартная форма* – это предваренная нормальная форма вида $\forall x_1 \forall x_2 ... \forall x_n M(x_1, x_2, ..., x_n)$.

Теорема (о сколемовской стандартной форме): Для любой замкнутой формулы φ существует сколемовская нормальная форма ψ, причем φ выполнима тогда и только тогда, когда выполнима ψ.

Лемма (о сколемизации): Пусть замкнутая формула φ имеет вид $\forall x_1 \forall x_2 ... \forall x_k \exists y \chi(x_1, x_2, ..., x_k, y)$ и пусть также имеется функциональный символ $f^{(k)}$. Тогда формула φ выполнима тогда и только тогда, когда выполнима формула $\forall x_1 \forall x_2 ... \forall x_k \chi(x_1, x_2, ..., x_k, f^{(k)}(x_1, x_2, ..., x_k))$. При этом $f^{(k)}$ называется *сколемовской функцией*.

Определение. Система дизъюнктов *S* называется *невыполнимой*, если не существует такой интерпретации, в которой будут выполнимы одновременно все дизъюнкты, входящие в *S*.

Теорема. Формула φ общезначима тогда и только тогда, когда невыполнима система дизъюнктов $S_{\neg\varphi}$.

Определение. *Основным термом* называется любой терм, не имеющий свободных переменных.

Определение. *Эрбрановским универсумом (H)* называется множество основных термов, которое строится следующим образом:
1. $H_0 = \begin{cases} \mathrm{Const}_\varphi \\ \{a\} \end{cases}$
2. $H_{i+1} = H_i \cup \{f^{(k)}(t_1, t_2, ..., t_k)\}$ для всех $f^{(k)} \in \mathrm{Func}_\varphi$ и $t_1, t_2, ..., t_k \in H_i$
3. $H = \bigcup_{i=0}^{\infty} H_i$

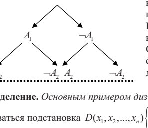
Определение. *Эрбрановской интерпретацией* для системы дизъюнктов S_φ называется интерпретация $I = < H, \overline{\mathrm{Const}}, \overline{\mathrm{Func}}, \overline{\mathrm{Pred}} >$, где *H* – эрбрановский универсум, а остальные множества определяются следующим образом:
$\overline{\mathrm{Const}} : c \in \mathrm{Const}; \overline{c} = c$
$\overline{\mathrm{Func}} : f^{(m)} \in \mathrm{Func}; \overline{f}^{(m)} : t_1, t_2, ..., t_m \in H \quad \overline{f}^{(m)} : (t_1, t_2, ..., t_m) \to f^{(m)}(t_1, t_2, ..., t_m)$
$\overline{\mathrm{Pred}} : P^{(n)} \in \mathrm{Pred}$

Теорема (об эрбрановских интерпретациях): Система дизъюнктов невыполнима тогда и только тогда, когда она невыполнима на эрбрановских интерпретациях.

Существует несколько способов представления эрбрановских интерпретаций:

1. Теоретико-множественный.
Назовем *основным атомом* атомарную формулу (без свободных переменных), полученную в результате подстановки в некоторый предикат *P* основных термов:
$P(t_1, t_2, ..., t_m) \in Atom; (t_1, t_2, ..., t_m) \in H$
Тогда *I* будет являться эрбрановской интерпретацией для множества основных атомов, если на ней будет выполним любой основной атом из этого множества.
2. Последовательный
Основной литерой называется либо основной атом, или его отрицание. Упорядоченное множество всех основных атомов будет называться *эрбрановским базисом B*. Тогда с эрбрановской интерпретацией можно связать последовательность литер $A_1^{\sigma_1}, A_2^{\sigma_2}, ..., A_l^{\sigma_l}, ...$, причем $A_i^{\sigma_i}$ будет равно A_i, если A_i выполнима в *I*, и $\neg A_i$ в противном случае.

Определение. Пусть $B_H = \{A_1, A_2, ..., A_l, ...\}$ - эрбрановский базис. *Семантическим деревом* называется бинарное корневое дерево следующего вида (см. рисунок слева).



В таком дереве каждая ветвь соответствует какой-нибудь эрбрановской интерпретации.
С помощью семантического дерева можно легко сформулировать критерии выполнимости системы дизъюнктов.

Определение. *Основным примером дизъюнкта* $D(x_1, x_2, ..., x_n) = L_1 \vee L_2 \vee ... \vee L_m$ будет называться подстановка $D(x_1, x_2, ..., x_n)\left\{{}^{x_1}/_{t_1}, ..., {}^{x_n}/_{t_n}\right\}$, где $t_1, t_2, ..., t_n \in H$.

Утверждение. Система дизъюнктов невыполнима тогда и только тогда, когда для любой эрбрановской интерпретации, которая задана в виде последовательности литер, существует такой основной пример D' дизъюнкта D из этой системы, что $D'(x_1, x_2, ..., x_n) = L'_1 \vee L'_2 \vee ... \vee L'_m$, что для некоторых его компонент будет выполняться равенство $L'_m = \neg A'^{\sigma_{i_m}}_{i_m}$, т.е. всегда будет находиться ложный основной пример.

Определение. Пусть v – вершина семантического дерева, в котором из корня ведет путь, помеченный литерами $L_1, L_2, ..., L_n$. Пусть также дизъюнкт D принадлежит системе S_φ. Говорят, что *дизъюнкт D опровергается в вершине v*, если существует такой основной пример D', состоящий из литер $L'_1, L'_2, ..., L'_n$, причем любая литера L'_i из D будет равна $\neg L_i$.

Определение. Вершина v называется *опровергающим узлом* для системы дизъюнктов S_φ, если

1) в вершине v опровергается какой-нибудь дизъюнкт D из S_φ;
2) никакая другая вершина, лежащая выше на пути из корня к v не опровергает ни одного дизъюнкта.

Лемма (о семантическом дереве): Система дизъюнктов невыполнима тогда и только тогда, когда в ее семантическом дереве каждая ветвь содержит опровергающий узел, и число таких узлов конечно.

Лемма (Кенига): Если есть бесконечное семантическое дерево, и из каждой его вершины выходит конечное число дуг, то это дерево содержит бесконечную ветвь.

Теорема (Эрбрана; об основных примерах): Система дизъюнктов S невыполнима (противоречива) тогда и только тогда, когда существует конечное множество S' основных примеров дизъюнктов из S, такое, что S' – тоже противоречивое множество.

Теорема Эрбрана сводит проблему выполнимости системы дизъюнктов к исследованию истинности булевых формул.

Алгоритм Девиса-Паттена проверки противоречивости системы дизъюнктов.

1. Строим семантическое дерево
2. Порождаем основные примеры дизъюнктов из системы.

После этого проверяется опровержимость основных примеров в вершине семантического дерева. Алгоритм завершается успешно, если построенная система опровергающих узлов пересекает все дерево.

Определение. *Композицией конечных подстановок* θ и η называется такая конечная постановка μ, что для любой переменной x $\mu(x) = (x\theta)\eta$. Композиция подстановок обозначается как $\eta\theta$.

Определение. Подстановка θ называется *унификатором* для логических выражений $E_1, E_2, ..., E_n$, если $E_1\theta = E_2\theta = ... = E_n\theta$.

Определение. Подстановка θ называется *наиболее общим унификатором (НОУ)* для логических выражений $E_1, E_2, ..., E_n$, если

1) θ - унификатор для $E_1, E_2, ..., E_n$;

2) для любой подстановки η, которая тоже унифицирует $E_1, E_2, ..., E_n$ найдется такая подстановка ρ, что $\eta = \theta\rho$ (т.е. η является частным случаем θ).

Определение. *Наиболее общим унификатором для системы уравнений* вида $\begin{cases} t_1 = s_1 \\ .... \\ t_m = s_m \end{cases}$ для атомарных формул $E_1 = P(t_1, t_2, ..., t_m)$ и $E_2 = P(s_1, s_2, ..., s_m)$ называется такая подстановка θ, что:

1) выполняется система равенств $\begin{cases} t_1\theta = s_1\theta \\ .... \\ t_m\theta = s_m\theta \end{cases}$;

2) любая другая подстановка, для которой эта система также выполняется, является частным случаем θ.

Утверждение. θ - НОУ для формул $E_1 = P(t_1, t_2, ..., t_m)$ и $E_2 = P(s_1, s_2, ..., s_m)$ тогда и только тогда, когда θ является НОУ для системы уравнений для этих формул.

Лемма (о связке): Пусть x – переменная, а t – терм, причем x не является свободной переменной для t. Тогда если конечная подстановка θ является унификатором x и t (т.е. $x\theta = t\theta$), то $\theta = \{x/t\}$.

Следствие. Подстановка θ из условия леммы является НОУ для x и t.

Теорема (о приведенной системе уравнений): Если $\begin{cases} x_1 = t_1 \\ .... \\ x_m = t_m \end{cases}$ - приведенная система уравнений, причем ни один x не является свободной переменной для любого t из системы, то НОУ такой системы $\lambda = \left\{ x_1/t_1, ..., x_m/t_m \right\}$

Алгоритм унификации Мортелло-Мортанари

Дана система термальных уравнений $\begin{cases} t_1 = s_1 \\ .... \\ t_m = s_m \end{cases}$. Алгоритм решает ее методом подстановки.

До тех пор, пока возможно, выполнить следующие действия:
Выбрать произвольное уравнение и применить к нему одно из 6 правил:

1) если уравнение имеет вид $f(t'_1, t'_2, ..., t'_k) = g(s'_1, s'_2, ..., s'_k)$, то оно преобразовывается в k уравнений $\begin{cases} t'_1 = s'_1 \\ .... \\ t'_k = s'_k \end{cases}$;
2) если уравнение имеет вид $f(t'_1, t'_2, ..., t'_k) = g(s'_1, s'_2, ..., s'_k)$, причем $f \neq g$, то алгоритм останавливается с ошибкой;
3) если уравнение имеет вид $c = k$, где c и k – константы, или x=x, где x – переменная, то такое уравнение надо удалить;

4) если уравнение имеет вид $t = y$, где y – переменная, а t – терм, то это уравнение заменяется на $y = t$;
5) если уравнение имеет вид $y = s_i$, причем y не является свободной переменной для s_i, но встречается в каких-нибудь уравнениях $t_j = s_j$, то в них надо произвести замену $\{y/s_i\}$.
6) если уравнение имеет $y = s_i$, и при этом y является свободной переменной для s_i, то алгоритм останавливается с ошибкой.

Теорема: Алгоритм унификации завершается и работает корректно, т.е.:

1. для любой системы уравнений l алгоритм останавливается после применения к этой системе;
2. если эта система унифицируема, то алгоритм выдает приведенную систему l', такую, что ее НОУ совпадает с НОУ исходной системы;
3. если эта система не унифицируема, то алгоритм останавливается с ошибкой.

Определение. *Правило резолюции* для двух дизъюнктов $D_1 = D'_1 \vee L$ и $D_2 = D'_2 \vee \neg L$ и их НОУ θ - это вывод их *резольвенты* – дизъюнкта вида $D_0 = (D'_1 \vee D'_2)\theta$.

Определение. *Правило склеивания* для дизъюнкта $D = D' \vee L \vee L'$ и НОУ θ для L и L' – это вывод *склейки D по паре L и L'* – дизъюнкта вида $D_0 = (D' \vee L)\theta$.

Определение. *Резолютивный вывод* для системы дизъюнктов S – это такая конечная последовательность дизъюнктов $D_1, D_2, ..., D_n$, что для любого номера i>1 выполняется одно из трех требований:

1) D_i - *вариант* D из S, т.е. D_i получен из D путем переименования переменных без отождествления;
2) D_i - резольвента двух дизъюнктов с меньшими номерами;
3) D_i - склейка дизъюнкта с меньшим номером

Резолютивный вывод называется *успешным* (*резолютивным опровержением*), если его последний дизъюнкт D_n - пустой.

Лемма (о резолюции): Если D_0 - резольвента дизъюнктов D и D', то он является их логическим следствием.

Лемма (о склейке): Если D_0 - склейка дизъюнкта D, то он является его логическим следствием.

Теорема (корректности резолютивного вывода): Если $D_1, D_2, ..., D_n$ - резолютивный вывод из семейства дизъюнктов S, то D_n - логическое следствие S.

Лемма (о подъеме для резолюции): Пусть D'_1 и D'_2 - основные примеры дизъюнктов D_1 и D_2 соответственно, а D'_0 - резольвента D'_1 и D'_2. Тогда дизъюнкт D_0, являющийся резольвентой D_1 и D_2, таков, что D'_0 - основной пример D_0.

Теорема (полноты для резолюций): Если S – противоречивая система дизъюнктов, то существует резолютивное опровержение S.

3. Логическое программирование

Логическая программа – это совокупность формул, причем программа позволяет доказать или опровергнуть общезначимость формул, которые в нее входят.

Определение. *Хорновским дизъюнктом* называется дизъюнкт, в который входит не более одной литеры без отрицания.

Определение. *Хорновская логическая программа* – множество хорновских дизъюнктов.

Определение. *Запрос к программе* – задача поиска всех правильных ответов в программе.

Определение. *Ответом на запрос* $G: ?C_1, ..., C_m$ с целевыми переменными $x_1, x_2, ..., x_n$ называется всякая подстановка $\theta = \left\{ x_1/t_1; ...; x_m/t_m \right\}$

Определение. Ответ η на запрос к логической программе P называется *правильным*, если результат подстановки $(C_1, C_2, ..., C_m)\eta$ является логическим следствием P.

Определение. Эрбрановская интерпретация I для логической программы P называется ее *моделью*, если она является моделью для любого хорновского дизъюнкта, входящего в нее.

Утверждение. I – модель для логической программы P тогда и только тогда, когда для любого основного примера $D_0 = A'_0 \leftarrow A'_1 \& A'_2 \& ... \& A'_n$, если любое A'_i принадлежит I, то и A'_0 также принадлежит I.

Лемма (о пересечении моделей): Пусть P – хорновская логическая программа, а M' и M'' - ее модели. Тогда эрбрановская интерпретация M, являющаяся пересечением M' и M'' также будет моделью для P.
Следствие 1: пересечение произвольного числа моделей для P также будет моделью.
Следствие 2: пересечение всех моделей для P также будет ее моделью. Такая модель будет называться *наименьшей Н-моделью (ННМ)* и обозначаться как M_P

Теорема: Пусть P – хорновская логическая программа, C - основной терм. C является следствием P тогда и только тогда, когда он принадлежит ННМ P.

Определение. Пусть P – хорновская логическая программа, $D = A_0 \leftarrow A_1 \& A_2 \& ... \& A_n$ - некоторое ее программное утверждение, $G: ?C_1, ..., C_m$ - запрос, а пересечение формальных параметров (Var_D) с фактическими (Var_G) пусто. Пусть C_i - подцель запроса G, а $\theta = НОУ(C_i, A_0)$. Тогда запрос $G': ?(C_1, ..., C_{i-1}, A_1, A_2, ..., A_n, C_{i+1}, ..., C_m)\theta$, полученный из G заменой C_i на $A_1, A_2, ..., A_n$ и последующей унификацией, называется *SLD-резольвентой запроса G и программного утверждения D с НОУ* θ. При этом C_i называется *выделенной подцелью*, а D – *активизированным программным утверждением*.

Определение. *SLD-резолютивным вычислением запроса G к хорновской логической программе P* называется последовательность пар (G_i, θ_i), конечная или бесконечная, такая, что

1) $G_0 = G$
2) Запрос G_{i+1} - SLD-резольвента запроса G_i и варианта D' программного утверждения D, такого $\mathrm{Var}_{G_i} \cap \mathrm{Var}_{D'} = \varnothing$, а θ_{i+1} - НОУ для этой резольвенты.
3) Возможно три варианта вычисления:

А) $G_l = ?\square$ - *успешное вычисление (SLD-резолютивное опровержение)*

Б) бесконечная последовательность – *бесконечное вычисление*

В) обнаруживается, что очередную SLD-резольвенту построить нельзя – *тупиковое вычисление*.

Определение. Пусть $(G_1, \theta_1), ..., (G_{l-1}, \theta_{l-1}), (\square, \theta_l)$ - успешное завершение запроса G к хорновской логической программе P. Тогда подстановка $\theta = \theta_1\theta_2...\theta_l |_{y_1, y_2, ..., y_n}$ называется *вычислимым ответом на запрос G к логической программе P*.

Определение. Пусть P – хорновская логическая программа. Тогда *множеством успехов P* будет называться следующее множество $Succ_P$ - это множество всех атомарных запросов, имеющих успешное SLD-резолютивное вычисление ($A \in B_P$ - эрбрановскому базису P).

Определение. Пусть 2^{B_P} - множество всех подмножеств B_P (множество всех эрбрановских интерпретаций). *Оператором непосредственного следования* для логической программы P называется функция $T_P: 2^{B_P} \to 2^{B_P}$, такая, что для любого подмножества B_P (обозначим его I) $T_P(I) = \{A : A \in B_P; A \leftarrow A_1, A_2, ..., A_n\}$, где A_i принадлежит I – т.е. все то, что следует из интерпретации I по правилам логической программы P.

Утверждение. Для оператора непосредственного следования выполняются свойства:

$I_1 \subseteq I_2 \Rightarrow T_P(I_1) \subseteq T_P(I_2)$

$T_P(I_1 \cup I_2) \supseteq T_P(I_1) \cup T_P(I_2)$

$I \in B_P; I \models P \Leftrightarrow T_P(I) \subseteq I$

Определение. Интерпретация I называется *неподвижной точкой оператора непосредственного следования*, если выполняется равенство $T_P(I) = I$. Множество всех неподвижных точек оператора непосредственного следования обозначим как f_{p_P}

Определение. I_0 называется *наименьшей неподвижной точкой* оператора T_P, если выполняются 2 свойства:

1) I_0 - неподвижная точка оператора T_P;
2) для любой другой неподвижной точки I выполняется свойство $I_0 \subseteq I$.

Наименьшая неподвижная точка обозначается как lf_{p_P}.

Теорема (о неподвижных точках). В хорновской логической программе P всегда существует наименьшая неподвижная точка, и $lf_{p_P} = \bigcup_{i=0}^{\infty} T_P^i(\varnothing) = M_P$ (т.е. наименьшая

неподвижная точка равна объединению всех следствий конечной кратности и является наименьшей эрбрановской моделью для логической программы P).

Теорема (корректности операционной семантики): Пусть P – хорновская логическая программа, G – запрос к ней, θ - вычисленный ответ на запрос G к P. Тогда вычисленный ответ θ - это правильный ответ.
Следствие. $Succ_P \subseteq M_P$ (т.е. множество успехов является подмножеством наименьшей эрбрановской модели).

Определение. Пусть P – хорновская логическая программа, G_0 - запрос к P. Тогда *квази-SLD-резолютивным вычислением* запроса G_0 к P называется последовательность пар $(G'_1, \theta'_1), ..., (G'_n, \theta'_n), (G'_{n+1}, \theta'_{n+1})$, которая удовлетворяет требованию: если $G'_{n-1} = ?C_1, ..., C_{i-1}, C_i, C_{i+1}, ..., C_m$, $D: A_0 \leftarrow A_1, ..., A_k \in P$ и θ'_n - <u>произвольный</u> унификатор C_i и A_0, то $G'_n = ?(C_1, ..., C_{i-1}, A_1, ..., A_k, C_{i+1}, ..., C_m)\theta'_n$.
Заметим, что SLD-резолютивное вычисление – частный случай квази-SLD-резолютивного вычисления, поэтому корректность квази-SLD-резолютивного вычисления представляет собой отдельную теорему.

Теорема (о квази-SLD-резолютивных вычислениях): Если атом C входит в наименьшую эрбрановскую интерпретацию ($C \in M_P$), то запрос $G_0: ?C$ имеет успешное квази-SLD-резолютивное вычисление.
Следствие. Если $C_1, C_2, ..., C_n \in M_P$, то запрос $G: ?C_1, C_2, ..., C_n$ имеет успешное квази-SLD-резолютивное вычисление.

Лемма (о подъеме для SLD-резолюций): Пусть G – запрос к хорновской логической программе P, η - конечная подстановка, а запрос $G\eta$ имеет успешное квази-SLD-резолютивное вычисление $G\eta, (G'_1, \theta'_1), ..., (\square, \theta'_n)$. Тогда запрос G имеет успешное SLD-резолютивное вычисление $G, (G_1, \theta_1), ..., (\square, \theta_n)$, причем существует такая конечная подстановка ρ, что $\eta\theta'_1\theta'_2...\theta'_n = \theta_1\theta_2...\theta_n\rho$. Иначе говоря, результат частного случая запроса для квази-SLD-резолютивного вычисления – это результат частного случая запроса для SLD-резолютивного вычисления.

Теорема (о полноте; ван Эмдена): Если атом $C \in M_P$, то $C \in Succ_P$.

Теорема (о полноте; Кларка): Пусть G – запрос к хорновской логической программе P, θ - правильный ответ на G. Тогда существует вычисленный ответ η, такой, что для некоторой конечной подстановки ρ выполняется равенство $\theta = \eta\rho$ (т.е. любой правильный ответ является частным случаем некоторого вычисленного ответа).

Машина Тьюринга:
Определение. *Ленточный алфавит* $A = \{a_0, ..., a_n\}$ - это множество допустимых значений ячеек памяти (символ a_0 называется *пустым*). Слово в алфавите A называется *ленточным словом*.
Определение. *Алфавит состояний* $Q = \{q_0, q_1, ..., q_m, ...\}$ - это множество состояний, причем состояние q_0 называется *заключительным*, а состояние q_1 - *начальным*.

Определение. *Ленточная конфигурация* – это тройка $d = < w_1, q, w_2 >$, где w_1 - ленточное слово, которое является содержимым ленты слева от обозреваемой ячейки, q - текущее состояние машины Тьюринга, w_2 - слово, состоящее из самой ячейки и содержимого ленты справа от нее.
Определение. *Команда машины Тьюринга* – это пятерка $< q, a, b, q', D >$, где q и q' – состояния, a, b – символы из алфавита A, $D \in \{L, R\}$.
Определение. *Машиной Тьюринга* называется тройка $\overline{T} = < A, Q_T, \Pi_T >$, где A – ленточный алфавит, Q_T – конечное подмножество множества состояний, Π_T – множество команд. Важное ограничение: для любого состояния, отличного от заключительного, и символа из ленточного алфавита существует единственная команда.

Определение. *Отношением непосредственного перехода* называется отношение $T: \alpha \to \beta$, где α, β - состояния машины Тьюринга. Обозначим $a \xrightarrow[T]{*} \beta$ транзитивное замыкание отношения непосредственного перехода.

Для каждой машины Тьюринга определим функцию $F_T: q_1 \to q_0$, причем если α - начальная конфигурация, то $F_T(\alpha) = \beta$ тогда и только тогда, когда β - заключительная конфигурация, и $a \xrightarrow[T]{*} \beta$.

Тезис Чёрча: Для любой функции $\varphi: q_1 \to q_0$ существует алгоритм вычисления этой функции тогда и только тогда, когда $\varphi = F_T$ для некоторой машины Тьюринга.

Введем дополнительные обозначения: *nil* – 0-местный функциональный символ (константа), • - 2-местный функциональный символ, и введем отображение множества машин Тьюринга на множество логических программ: $Compile: T \to P$

Теорема. Пусто T – машина Тьюринга, $Compile(T) = P$ - логическая программа. Тогда для любой начальной конфигурации $\alpha_0 = < a_{i1}, ..., a_{ik}; q_1; b_{j1}, ..., b_{jn} >$ существует заключительная конфигурация $\beta_0 = < a'_{l1}, ..., a'_{lm}; q_0; b'_{r1}, ..., b'_{rt} >$, такая, что $a \xrightarrow[T]{*} \beta$, тогда и только тогда, когда запрос $G_{\alpha_0}: ?R(\bullet(a_{ik}, \bullet(a_{ik-1}, ..., (a_{i1}, nil)...), q_1, \bullet(b_{j1}, ..., (b_{jn}, nil)...))$ имеет успешное SLD-резолютивное вычисление с ответом

$$\eta = \left\{ u/\bullet(a'_{lm}, \bullet(...(a'_{l1}, nil)...); \; v/q_0; \; w/\bullet(b'_{r1}, \bullet(...(b'_{rt}, nil)...)) \right\}.$$

Теорема (Чёрча; о неразрешимости проблемы общезначимости для классической логики предикатов): Не существует алгоритма, который для любой формулы φ позволял бы проверить ее общезначимость, т.е. не существует алгоритма, который выдавал бы на выходе 1, если формула φ общезначима, и 0 в противном случае.

Определение. *Правилом выбора подцелей* называется отображение $R: ?C_1, ..., C_n \to C_i$ (т.е. отображение, показывающее, какой атом должен быть выбран на текущем этапе вычисления).

Определение. SLD-резолютивное вычисление называется *R-вычислением*, если на каждом шаге вычисления очередная подцель выбирается по правилу R.

Определение. Вычисленный ответ, полученный в результате R-вычисления называется *R-вычисленным ответом*.

Лемма (переключательная): Пусть P – хорновская логическая программа, G_0 - запрос к ней, пусть также имеется SLD-резолютивное вычисление $G_0, (G_1, \theta_1), ..., (\square, \theta_n)$, причем G_1 и G_2 - этапы SLD-резолютивного вычисления, на которых i-ая подцель была раскрыта по правилу D_1 и D_2 соответственно. Тогда существует SLD-резолютивное вычисление $G_0, (G'_1, \theta'_1), (G'_2, \theta'_2), ..., (\square, \theta'_n)$, такое, что на этапе G'_1 раскрытие происходило по правилу D_2, а на этапе G'_2 - по правилу D_1, при этом для вычисленных ответов η и η' существуют конечные подстановки ρ и ρ', такие, что $\eta = \eta'\rho'$, $\eta' = \eta\rho$.

Теорема: Пусть P – хорновская логическая программа, G – запрос к ней, R и R' – различные правила выбора. Тогда если η - R-вычислимый ответ на G к P, то существует такой R'-вычислимый ответ η', такой, что $\eta = \eta'\rho'$ и $\eta' = \eta\rho$ для некоторых конечных перестановок (переименований) ρ и ρ'.

Теорема (сильной полноты): Пусть G – запрос к хорновской логической программе P, R – правило выбора, θ - правильный ответ на G к P. Тогда существует R-вычислимый ответ η, такой, что $\theta = \eta\rho$, где ρ - конечная подстановка (переименование).

Определение. Пусть G – запрос к хорновской логической программе P, в которой все программные утверждения упорядочены. Тогда *деревом SLD-резолютивных вычислений запроса G к P* называется корневое упорядоченное (т.е. выходящие из любого узла дуги пронумерованы) дерево, обладающее следующими свойствами:

1) каждой вершине приписан запрос;
2) корню дерева приписан запрос G;
3) из каждой вершины, помеченной запросом G_0, исходят дуги, помеченные натуральными числами $1 \le i_1 \le ... \le i_k \le N$, где N – количество программных утверждений в P, и идущие в вершины, помеченные $G_1, G_2, ..., G_k$ в том и только в том случае, когда $\forall l \in [1, k]$ запрос G_l получен из запроса G путем применения программного утверждения D_{r_l} и при этом других SLD-резольвент, инцидентных G_0, нет.

Таким образом, каждая ветвь такого дерева будет соответствовать вычислению ответа на запрос G к логической программе P.

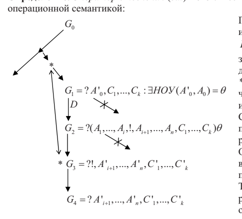
Определение. *Стратегией вычисления* называется стратегия обхода дерева SLD-резолютивных вычислений:

- стратегия обхода *в ширину*: поярусное построение дерева. Как только в одной из ветвей обнаруживается □, то ответ вычисляется путем «сбора» всех подстановок на данной ветви;
- стратегия обхода *в глубину с возвратом* – построение дерева по ветвям (на каждом этапе выбирается самая левая из непройденных ветвей)

Определение. Стратегия вычисления (обхода дерева SLD-резолютивных вычислений) называется *полной*, если для любого запроса G к логической программе P все вычисленные ответы будут построены.

Теорема. Стратегия обхода в ширину – полная, стратегия обхода в глубину с возвратом – неполная.

Определение. *Оператор отсечения* (*cat*) – это 0-местный предикат со следующей операционной семантикой:



Пусть в логической программе P имеется программной утверждение $D: A_0 \leftarrow A_1, ..., A_i, !, A_{i+1}, ..., A_n$, G_0 - запрос к этой программе. Строим дерево SLD-резолютивных вычислений. * - это парная метка, которая означает, что нас с момента ее появления интересует только одна альтернатива. Соответственно, после второго появления * ветвление снова разрешается. Откат при этом производится в вершину, предшествующую первому появлению *. Теорема полноты для SLD-резолютивного вычисления с операцией отсечения не выполняется.

Определение. *Оператор отрицания* (*not*) – это оператор логического программирования со следующей операционной семантикой:
Пусть P – логическая программа, а G – запрос вида *not*(A). Тогда реакция интерпретатора на данный запрос будет следующей:

1) если существует успешное SLD-резолютивное вычисление на запрос $G': ?A$, то результатом вычисления будет *fail*;
2) если SLD-резолютивное дерево вычислений для запроса $G': ?A$ представляет собой конечное число неудачных ветвей, то запрос G имеет успешное вычисление;
3) если интерпретатору не удалось обойти все ветви дерева за конечное число шагов (т.е. если в SLD-резолютивном дереве вычислений для запроса $G': ?A$ присутствует хотя бы одна бесконечная ветвь, а остальные неудачны), то ответом на запрос G будет ∞.

Таким образом, оператор *not* работает на основе «почти трехзначной» логики, что дает эффект немонотонности вывода.

4. Обзор разделов математической логики

4.1. Исчисление предикатов и элементарная теория

Исчисление предикатов состоит из системы аксиом и правил вывода. Существует много разных систем аксиом.

Система аксиом:

$Ax1: \varphi \to (\psi \to \varphi)$

$Ax2: (\varphi \to (\psi \to \chi)) \to ((\varphi \to \psi) \to (\varphi \to \chi))$ - закон сложного силлогизма

$Ax3: \varphi \& \psi \to \varphi$

$Ax4: \varphi \& \psi \to \psi$ - свойства конъюнкции

$Ax5: \varphi \to (\psi \to (\varphi \& \psi))$

$Ax6: \varphi \to \varphi \vee \psi$

$Ax7: \psi \to \varphi \vee \psi$ - свойства дизъюнкции

$Ax8: (\varphi \to \chi) \to ((\psi \to \chi) \to ((\varphi \vee \psi) \to \chi))$

$Ax9: \varphi \to (\neg\varphi \to \psi)$

$Ax10: (\varphi \to \psi) \to ((\varphi \to \neg\psi) \to \neg\varphi)$ - законы противоречия

$Ax11: \varphi \vee \neg\varphi$ - закон исключенного третьего

$Ax12: \forall x \varphi(x) \left\{ {}^{x}\!/_{t} \right\}$, где x свободна для t в φ.

$Ax13: \varphi(x) \left\{ {}^{x}\!/_{t} \right\} \to \exists x \varphi(x)$

$Ax14: \forall x (\varphi \to \psi) \to (\varphi \to \forall x \psi)$

$Ax15: \forall x (\psi \to \varphi) \to (\exists x \psi \to \varphi)$; $x \notin \mathrm{Var}_\varphi$

Правила вывода:

1. $\dfrac{\varphi \quad \varphi \to \psi}{\psi}$ - правило отделения (modus ponens); следствие импликации отделяется от посылки.
2. $\dfrac{\varphi}{\forall x \varphi}$ - правило обобщения.

Определение. *Логическим выводом* в исчислении предикатов из множества формул Γ называется конечная последовательность формул $\varphi_1, \varphi_2, ..., \varphi_n$, такая, что для любого $i \geq 1$

- либо $\varphi_i \subseteq \Gamma$;
- либо φ_i - одна из 15 аксиом;
- либо φ_i получается из предшествующих формул по правилу отделения или обобщения.

Логический вывод – формализация понятия математического доказательства.
Последняя формула φ_n из последовательности логического вывода называется *выводимой* из Γ и обозначается как $\Gamma \mid_{\overline{ИП}} \varphi_n$. Если Γ – пустое множество, то φ_n называется *теоремой исчисления предикатов*.

Теорема (Гёделя; корректности и полноты для исчисления предикатов): Формула является теоремой исчисления предикатов тогда и только тогда, когда она общезначима.

Следствие из первой теоремы Гёделя: Математика – неаксиоматизируемая наука.

4.2. Интуиционистская логика

Интуиционистская логика (логика Брауэра) – это классическая логика, из аксиом которой исключен закон «исключенного третьего».

Определение. *Интуиционистская интерпретация* – это тройка $I_{\text{int}} = \langle S, R, \xi \rangle$, где
S – непустое множество *альтернативных состояний* (*состояний знаний*);
$R \subseteq S \times S$ - отношение *нестрогого частичного порядка*, указывающее, что некоторое состояние знаний может быть достигнуто из другого и обладающее свойствами транзитивности, антисимметричности и рефлексивности;
$\xi : S \times P \to \{T, F\}$ - *оценка истинности* (способность решать задачи в определенном состоянии).
Отношение выполнимости для интуиционистской логики будет выглядеть следующим образом:

$$I_{\text{int}}, S \models p \Leftrightarrow \xi(S, p) = T$$

$$I_{\text{int}}, S \models p \& q \Leftrightarrow \begin{cases} I_{\text{int}}, S \models p \\ I_{\text{int}}, S \models q \end{cases}$$

$$I_{\text{int}}, S \models p \vee q \Leftrightarrow \left[\begin{array}{l} I_{\text{int}}, S \models p \\ I_{\text{int}}, S \models q \end{array} \right.$$

$$I_{\text{int}}, S \models p \to q \Leftrightarrow \forall S' \in S : R(S, S') \& I_{\text{int}}, S' \models p \Rightarrow I_{\text{int}}, S' \models q$$

$$I_{\text{int}}, S \models \neg p \Leftrightarrow \forall S' \in S : R(S, S') \Rightarrow I_{\text{int}}, S' \not\models p$$

Изменяется и понятие логического закона.

Определение. Формула φ называется *общезначимой с точки зрения интуиционистской логики* (*интуиционистски общезначимой*, обозначается как $\models_{\text{int}} \varphi$), если формула φ будет выполнима в любом состоянии S любой интерпретации $I_{\text{int}} = \langle S, R, \xi \rangle$.

Утверждение. Если формула φ интуиционистски общезначима, то она общезначима и с точки зрения классической логики предикатов.

Утверждение. Формулы $p \vee \neg p$ и $\neg\neg p \to p$ интуиционистски невыполнимы.

Замечание. Не существует ни одного содержательного математического утверждения, для которого нельзя было бы построить интуиционистского доказательства, однако все доказательства должны показывать правильность утверждения явно (конструктивно), поэтому все доказательства получаются более сложными, а результатом любого такого доказательства будет построение алгоритма.

Утверждение (дизъюнктивное свойство интуиционистской логики). Интуиционистская общезначимость формулы $\varphi \vee \psi$ эквивалентна интуиционистской общезначимости либо φ, либо ψ.

Утверждение. Формула $\exists x \varphi(x)$ общезначима тогда и только тогда, когда существует такой терм t, что формула $\varphi(t)$ будет интуиционистски общезначимой.

4.3. Логика Хоара

Пусть Π – программа, R_Π - отношение между ее входными и выходными данными (т.е. если при поступлении на вход значения x программа выдает y, то выражение $R_\Pi(x, y)$ является истинным).

Определение. Программа называется *правильной*, если на ее выходе всегда появляется результат, находящийся в определенном отношении с входными данными (т.е. $\models \forall x_{in} (\varphi_0(x_{in}) \to \exists x_{out} (R_\Pi(x_{in}, x_{out}) \& \varphi_1(x_{in}, x_{out}))))$.

Определение. *Тройкой Хоара* называется формула вида $\{\varphi\} < \Pi > \{\psi\}$, где φ, ψ - формулы, а Π – программа, причем если на входе программы Π выполняется утверждение φ, то на ее выходе выполняется отношение ψ. Формула φ называется *предусловием*, формула ψ - постусловием.

Определение. Программа Π называется *частично корректной относительно предусловия* φ *и постусловия* ψ, если общезначима формула $\{\varphi\} < \Pi > \{\psi\}$. Эту общезначимость следует понимать следующим образом: для любой допустимой интерпретации I и любого набора значений свободных переменных $\{\alpha_1, \alpha_2, ..., \alpha_n\} \in \mathrm{Var}_\varphi \cup \mathrm{Var}_\psi$ из выполнимости формулы φ ($I \models \varphi[d_1, d_2, ..., d_n]$) будет следовать, что после завершения программы Π переменные примут значения $d'_1, d'_2, ..., d'_n$, причем формула ψ будет выполнима на данных значениях переменных ($I \models \psi[d_1, d_2, ..., d_n]$).

Определим модельный язык программирования:

```
Π ::=   (x = t)
      | Π1, Π2
      | if A then Π1 else Π2 fi
      | while A do Π1 od
      | eps,
```

Где x – переменная, t – терм Π_i – программы, A – атом, `eps` – пустота.

Определение. *Правилом вывода* называется фигура следующего вида: $\dfrac{F_1, F_2, ..., F_n}{F_0}$, причем выражения в верхней части правила называются *предпосылками*, выражение в нижней части правила называется *заключением*. Семантика правила вывода такова: если истинным предпосылки, то истинно и заключения.

Рассмотрим следующую систему правил:

1. $\dfrac{\varphi_1 \to \varphi'_1, \{\varphi'_1\} < \Pi > \{\varphi'_2\}, \varphi_2 \to \varphi'_2}{\{\varphi_1\} < \Pi > \{\varphi_2\}}$ (ослабления)
2. $\dfrac{\varphi \to \psi \left\{ {}^{x}\!/_{t} \right\}}{\{\varphi\} < \mathtt{x := t} > \{\psi\}}$ (присваивания)
3. $\dfrac{\{\varphi\} < \Pi_1 > \{\chi\}, \{\chi\} < \Pi_2 > \{\psi\}}{\{\varphi\} < \Pi_1 ; \Pi_2 > \{\psi\}}$ (последовательного соединения)
4. $\dfrac{\{\varphi \& A\} < \Pi_1 > \{\psi\}, \{\varphi \& \neg A\} < \Pi_2 > \{\psi\}}{\{\varphi \& A\} < \mathtt{if\_}A\mathtt{\_then\_}\Pi_1\mathtt{\_else\_}\Pi_2\mathtt{\_fi} > \{\psi\}}$ (ветвления)

5. $\dfrac{\{\chi_{inv} \& A\} < \Pi >, \chi_{inv} \& \neg A \to \psi, \varphi \to \chi_{inv}}{\{\varphi\} < \mathtt{while\_}A\mathtt{\_do\_}\Pi > \{\psi\}}$ (итерации; χ_{inv} называется *инвариантом оператора итерации*, для проверки правильности работы цикла требуется его найти)
6. $\{\varphi\} < \mathtt{eps} > \{\varphi\}$ (аксиома)

Система таких правил называется *PVS* (Prototype Verification System) и позволяет упрощать проверку правильности программ.

4.4. Логики знаний

Рассмотрим *мультиагентную систему* (сложную систему, принимающую решения) $A_1, A_2, ..., A_n$ *программ-агентов*. При этом принятие решения может опираться на то, что знают или не знают другие участники процесса принятия решений.

Определение. *Модальным оператором* называется оператор $\Box$, который выражает отношение уверенности (необходимости). Его семантика такова:
$\Box_я \varphi$ означает «я знаю, что формула φ истинна».
Вопрос общезначимости формулы $\Box_я \varphi \to \varphi$ - вопрос *непротиворечивости* базы знаний.
Вопрос общезначимости формулы $\varphi \to \Box_я \varphi$ - вопрос *полноты* базы знаний.

4.5. Динамическая логика (логика программ)

Синтаксис динамической логики состоит из двух частей:

Программы	Формулы
$A = \{a_1, a_2, ..., a_n\}$ - набор *базовых действий* (*операторов*)	$P = \{\varphi_1, \varphi_2, ..., \varphi_n\}$ - набор *базовых условий* (*пропозиционных переменных*)
1) $a \in A$ - программа;	1) $\varphi \in P$ - формула;
2) Π_1, Π_2 - программа;	2) $\varphi_1 \& \varphi_2$, $\varphi_1 \vee \varphi_2$, $\varphi_1 \to \varphi_2$, $\neg\varphi$ - формулы;
3) $\Pi_1 \parallel \Pi_2$ - программа ($\parallel$ - недетермированный выбор);	3) если Π – программа, то $[\Pi]\varphi$ - формула.
4) $(\Pi_i)^*$ - программа;	
5) $\varphi?$ - программа (тест; φ - формула).	

Пример: выражение $\{\varphi\} < \mathtt{if\ A\ then\ \Pi_1\ else\ \Pi_2\ fi} > \{\psi\}$, записанное в терминах логики Хоара, в терминах динамической логики будет выглядеть как $\varphi \to [(A?, \Pi_1) + (\neg A?, \Pi_2)]\psi$.