Севастопольский государственный университет
Кафедра «Информационные системы»

Курс лекций по дисциплине “МЕТОДЫ И СИСТЕМЫ ИСКУССТВЕННОГО ИНТЕЛЛЕКТА" (МИСИИ)

Лектор: Бондарев Владимир Николаевич

Лекция 14

Дедуктивный вывод в исчислении предикатов. Принцип резолюции.

В.Бондарев

Задача дедуктивного вывода

Если обозначить через $\boldsymbol{\Phi_0}$ множество исходных утверждений и через $\boldsymbol{Fg}$ целевое утверждение, то задача, представленная в виде доказательства теоремы, формально может быть записана в виде

$$\Phi_0 \Rightarrow F_g$$

Формула $\boldsymbol{Fg}$ логически следует из $\boldsymbol{\Phi_0}$, если каждая интерпретация, удовлетворяющая $\boldsymbol{\Phi_0}$, удовлетворяет и $\boldsymbol{Fg}$.

Когда множество $\boldsymbol{\Phi_0}$ представлено формулами $\boldsymbol{F_1, F_2, \ldots, F_n}$, то задачей дедуктивного вывода исчисления предикатов является выяснение **общезначимости** формулы:

$$F_1 \wedge F_2 \wedge \ldots \wedge F_n \to F_g$$

Формулы, которые истины во всех интерпретациях называются ***тавтологиями*** или ***общезначимыми*** формулами. Для указания того, что формула является тавтологией, используется знак $\models$.

Например, $\models \; A \vee \overline{A}$.

Задача дедуктивного вывода

В ходе вывода часто используют метод доказательства от обратного, т.е. устанавливают не общезначимость приведенной выше формулы, а **невыполнимость формулы**

$$\overline{F_1 \wedge F_2 \wedge \ldots \wedge F_n \rightarrow F_g}$$

или

$$F_1 \wedge F_2 \wedge \ldots \wedge F_n \wedge \overline{F_g}$$

Т.е. доказывают, что объединение $\Phi_0 \cup \overline{F_g}$ **невыполнимо.** Процесс установления невыполнимости некоторого множества формул называют ***опровержением*.**

В 1965г. **Дж. Робинсон** разработал **принцип резолюции,** позволяющий автоматизировать процесс опровержения и являющийся теоретической базой для построения многих систем автоматического доказательства теорем.

Стандартизация предикатных формул

Для осуществления дедуктивного вывода на основе принципа резолюции, все формулы множества $\Phi_0 \cup \overline{F}_g$ должны быть представлены в виде дизъюнкций литералов.

Литералом называют элементарную формулу или ее отрицание. Дизъюнкция литералов называется ***предложением.*** Например, формула $P(x) \vee Q(x, y) \vee \overline{R}(x, y)$ является предложением.

Приведение формул к виду предложений выполняют в процессе указанных ниже **тождественных преобразований**.

1. **Исключение знаков импликации и эквиваленции**. Для этого используют равенства:

$$(P \rightarrow Q) = \overline{P} \vee Q \quad ,$$

$$(P \leftrightarrow Q) = (\overline{P} \vee Q) \wedge (\overline{Q} \vee P) \quad .$$

Стандартизация предикатных формул

2. Уменьшение области действия операции отрицания. При этом используют законы де Моргана:

$$\overline{(P\vee Q)}=(\overline{P}\wedge\overline{Q}), \quad \overline{(P\wedge Q)}=(\overline{P}\vee\overline{Q}),$$

$$\overline{\forall xP(x)}=\exists x\overline{P}(x), \quad \overline{\exists xP(x)}=\forall x\overline{P}(x).$$

Например,

$$\overline{\forall x(P(x)\wedge(Q(x)\vee R(x)))}=\exists x(\overline{P}(x)\vee(\overline{Q}(x)\wedge\overline{R}(x))).$$

3. **Стандартизация переменных.** В этом случае выполняют переименование переменных, связанных кванторами общности или существования. Переменные переименовываются так, чтобы каждый квантор связывал отдельную переменную, не встречающуюся в других кванторах. Например,

$$\forall xP(x)\vee\exists xQ(x)=\forall xP(x)\vee\exists zQ(z)$$

Стандартизация предикатных формул

4. Исключение кванторов существования. В простейшем случае это соответствует правилу экзистенциальной конкретизации, которое позволяет перейти от $\exists x P(x)$ к $P(a)$, где a – константа

Если квантор существования находится в области действия квантора общности, задача становится сложнее. Рассмотрим, например, формулу $\forall x \exists y (Q(y) \wedge R(x, y))$, которая может означать, что для каждого x существует такой конкретный y, что верно $Q(y)$ и $R(x,y)$.

В этой интерпретации значение переменной y зависит от значения переменной x. Такая зависимость представляется функцией $y=f(x)$, которую называют **функцией Сколема**. Функция Сколема позволяет исключить квантор существования и переписать рассматриваемую формулу в виде

$$\forall x (Q(f(x)) \wedge R(x, f(x))).$$

Стандартизация предикатных формул

5. Перемещение кванторов общности. Все кванторы общности записываются в начале формулы.

6. Приведение к конъюнктивной нормальной форме. Таким образом, любая формула записывается в виде $\forall x_1 \dots \forall x_n (K_1 \wedge K_2 \wedge \dots \wedge K_n)$. Здесь каждый член K_i представляет собой предложение (дизъюнкт), т.е. имеет вид $(L_1 \vee L_2 \vee \dots \vee \dots L_k)$, где L_i – литерал.

7. Исключение кванторов общности. Так как все переменные в формулах связаны кванторами общности, а порядок следования кванторов общности не имеет значения, то их можно не указывать в явном виде, то есть исключить.

8. Исключение конъюнкций. Каждая формула, приведенная к конъюнктивной нормальной форме, может быть заменена множеством формул

$$\{K_1, K_2, \dots, K_n\}.$$

Стандартизация предикатных формул

Рассмотрим **пример преобразования** формулы к множеству предложений. Пусть дана формула

$$\forall x\{P(x) \to \{\overline{\forall y[Q(x,y) \to R(y)]}\} \quad .$$

Исключим знаки импликации:

$$\forall x\{\overline{P}(x) \vee \{\overline{\forall y[\overline{Q}(x,y) \vee R(y)]} \quad .$$

Уменьшим область действия операции отрицания:

$$\forall x\{\overline{P}(x) \vee \{\exists y[Q(x,y) \wedge \overline{R}(y)]\} \quad .$$

Исключим квантор существования:

$$\forall x\{\overline{P}(x) \vee [Q(x,f(x)) \wedge \overline{R}(f(x))] \quad .$$

Приведем формулу к конъюнктивной нормальной форме:

$$\forall x\{(\overline{P}(x) \vee Q(x,f(x)) \wedge (\overline{P}(x) \vee \overline{R}(f(x)))\} \quad .$$

Исключив квантор общности и конъюнкцию, получим множество предложений:

$$\{\overline{P}(x) \vee Q(x,f(x)), \overline{P}(x) \vee \overline{R}(f(x))\} \quad .$$

Принцип резолюции

Доказательство теоремы состоит в том, чтобы показать, что ***расширенное множество формул***, образованное путем объединения множества аксиом $\boldsymbol{\Phi_0}$ и отрицания целевого утверждения ***Fg***, ***невыполнимо.*** Обозначим расширенное множество формул , приведенных предварительно к виду предложений (дизъюнктов), через ***S***.

Принцип резолюции представляет собой процедуру вывода, с помощью которой порождаются новые дизъюнкты из ***S***. Если в ходе применения этой процедуры ***выводится пустое предложение***, то ***S*** ***невыполнимо***.

Правило вывода, лежащее в **основе принципа резолюции**, по существу, представляет собой расширение правила **modus ponens** и **правила силлогизма**. Данные правила вывода можно записать в форме:

$$\frac{A, \overline{A} \vee B}{B} \quad \textbf{(modus ponens)}, \qquad \frac{\overline{A} \vee B, \quad \overline{B} \vee C}{\overline{A} \vee C} \quad \textbf{(силлогизм)}.$$

Принцип резолюции

В каждом из этих правил имеются дизъюнкты-посылки, содержащие *дополнительные (контрарные) пары литер* . Можно заметить, что следствие правил формально получается вычеркиванием в дизъюнктах-посылках контрарных пар литер и объединением оставшихся частей дизъюнктов.

Правило резолюции**:** если любые два дизъюнкта ***D1*** и ***D2*** содержат дополнительную пару литер, например L и $\overline{L}$, то, вычеркивая их, формируем новый дизъюнкт из оставшихся частей дизъюнктов ***D1*** и ***D2***. Вновь сформированный дизъюнкт называется ***резольвентой*** (следствием) исходных дизъюнктов.

Объединение исходного множества дизъюнктов с множеством всех резольвент, которые могут быть образованы из дизъюнктов, входящих в ***S***, будем обозначать ***R(S).*** Применяя принцип резолюции к ***R(S),*** получим $R(R(S))=R_2(S)$ и т.д. При этом $R_0(S)=S$.

Принцип резолюции

Теорема Робинсона: если ***S*** – произвольное конечное множество дизъюнктов, то ***S*** ***невыполнимо*** тогда и только тогда, когда $R_i(S)$ содержит ***пустой дизъюнкт***.

Данная теорема позволяет построить **процедуру опровержения**, которая для исходного множества ***S*** последовательно строит множества $R_1(S), R_2(S), \ldots R_i(S)$ и проверяет, содержит ли множество $R_i(S)$ пустой дизъюнкт.

При выполнении процедуры опровержения с целью построения контрарных пар выполняют различные **подстановки**, которые называют **унификаторами**. Унификатор, который позволяет получить все другие унификаторы, называется **наиболее общим унификатором (НОУ).**

Принцип резолюции

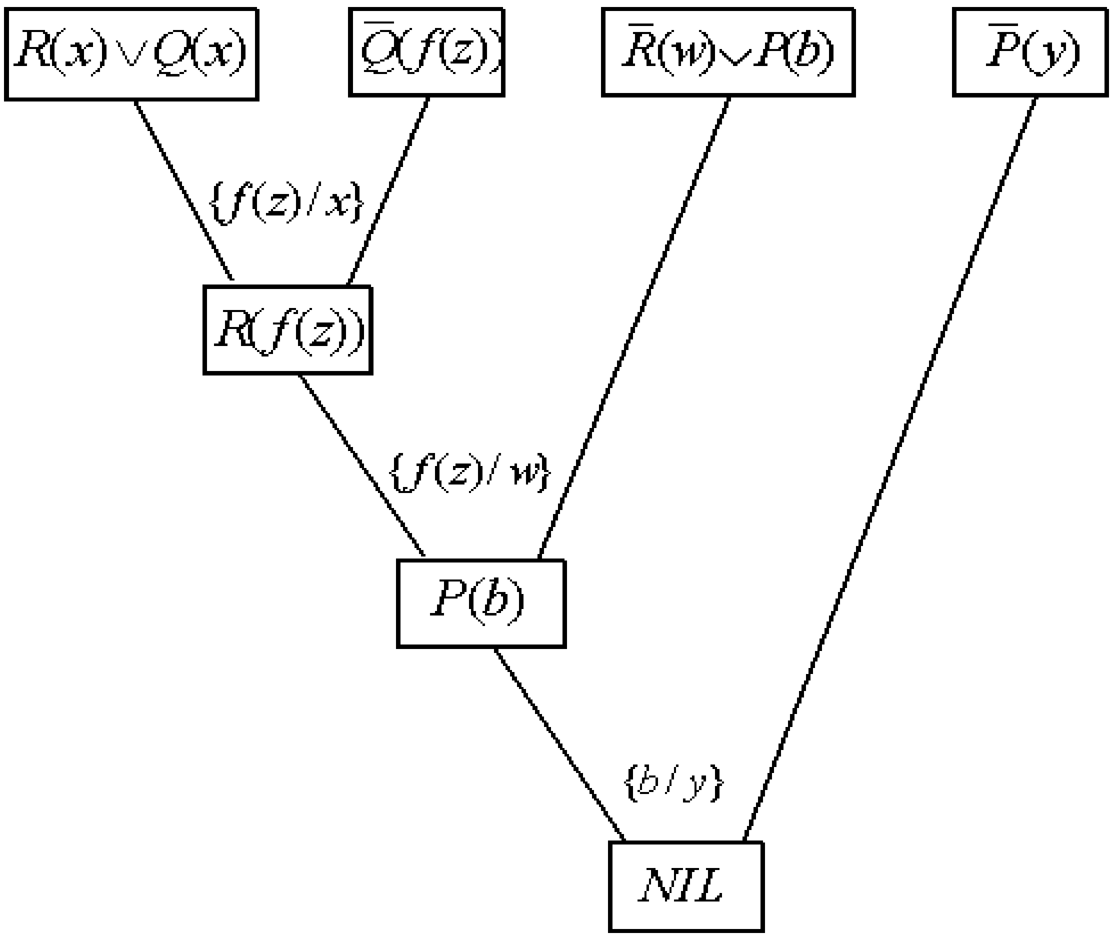
Т.о., **процедура нахождения резольвенты** двух предложений (дизъюнктов) сводится к следующим действиям:

1) переменные в предложениях переименовываются так, чтобы они не совпадали;
2) находится подстановка (НОУ), при которой литерал одного предложения становится дополнительным к какому-либо литералу другого предложения;
3) дополнительные литералы вычеркиваются;
4) если получены одинаковые литералы, то все они, за исключением одного в каком-либо предложении, вычеркиваются;
5) дизъюнкция оставшихся литералов каждого из предложений и есть резольвента.

Принцип резолюции

Вывод с помощью резолюции наглядно можно проиллюстрировать с помощью ***графа опровержения*.**

$$S = \{R(x) \vee Q(x),\quad \overline{Q}(f(z)),\quad \overline{R}(w) \vee P(b),\quad \overline{P}(y)\}$$



В.Бондарев

Принцип резолюции

В общем, процесс опровержения не протекает столь эффективно, как изображено на рисунке. Во многих случаях множество $R_i(S)$ очень быстро разрастается, так как непосредственное применение принципа резолюций соответствует “слепому” поиску.

Имеются **общие стратегии поиска**, которые, вне зависимости от контекста задачи, позволяют сократить число резолюций. В частности, линейная резолюция.

Линейная резолюция. Сначала находится следствие двух предложений исходного множества ***S*.** На следующих этапах в резолюции участвует резольвента ***Ci***, полученная на предыдущем шаге (называемая центральным дизъюнктом), и дизъюнкт ***B*** (называемый боковым), являющийся либо одним из дизъюнктов исходного множества *S*, либо центральным дизъюнктом *Cj*, который предшествует в выводе дизъюнкту *Ci* , т.е. $j < i$. Линейная стратегия является одной из наиболее эффективных в реализации.

Применения принципа резолюции

1. Информационный поиск

Пусть имеется база данных, содержащая сведения о химических соединениях: ***oxide(MgO)*** и ***white(MgO)*.**

Требуется установить, существует ли оксид белого цвета:

$$F_g(x) = \exists x(oxide(x) \wedge white(x))$$

Найдем отрицание **предположения:**

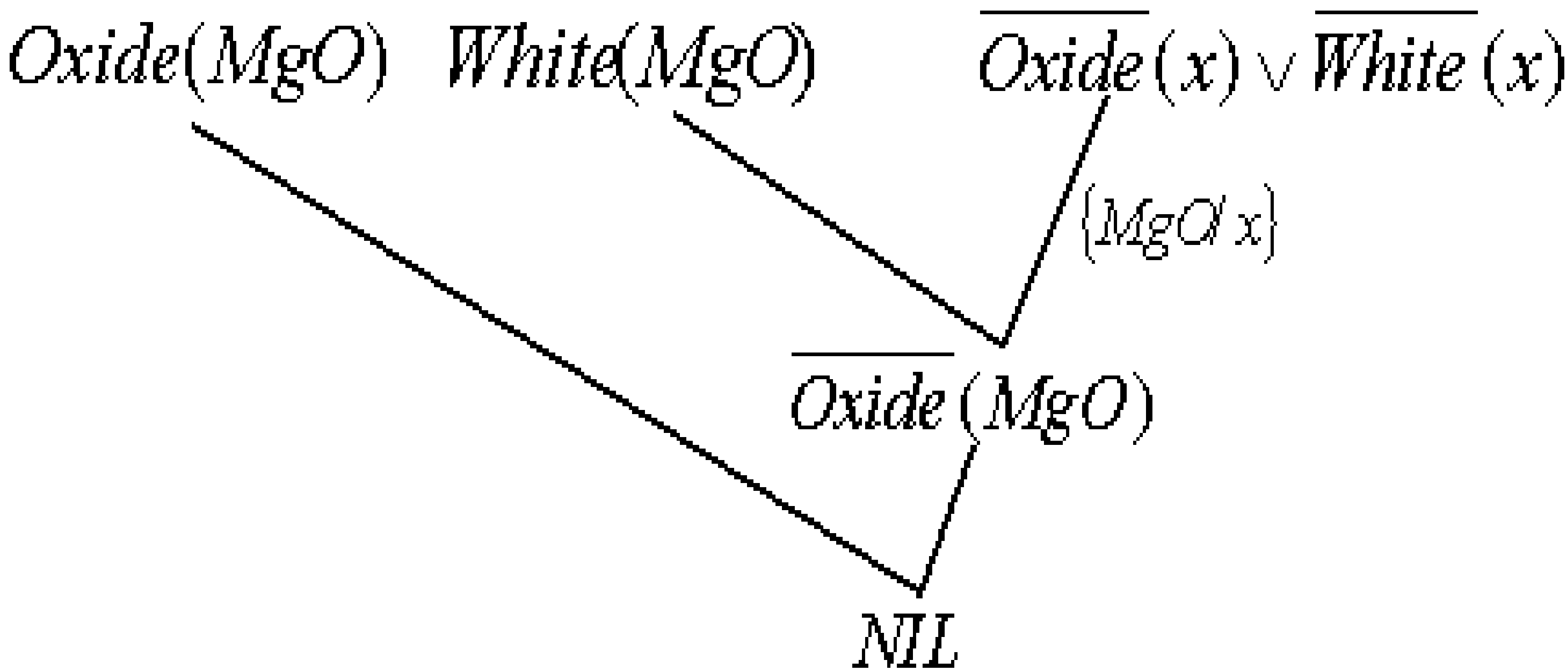
$$\overline{F}_g(x) = \forall x(\overline{oxide}(x) \vee \overline{white}(x))$$

Докажем, что множество ***S*** **невыполнимо**:

$$S = \{oxide(MgO), white(MgO), \overline{oxide}(x) \vee \overline{white}(x)\}$$

Применения принципа резолюции

Дерево опровержения



Применения принципа резолюции

Чтобы извлечь частное значение переменной, относящееся к квантору существования, которое и служит ответом на поставленный вопрос строят ***модифицированное дерево доказательства.***

$Oxide(MgO)$ $\quad$ $White(MgO)$ $\quad$ $\overline{Oxide}(x) \vee \overline{White}(x) \vee Oxide(x) \wedge White(x)$

$\{MgO/x\}$

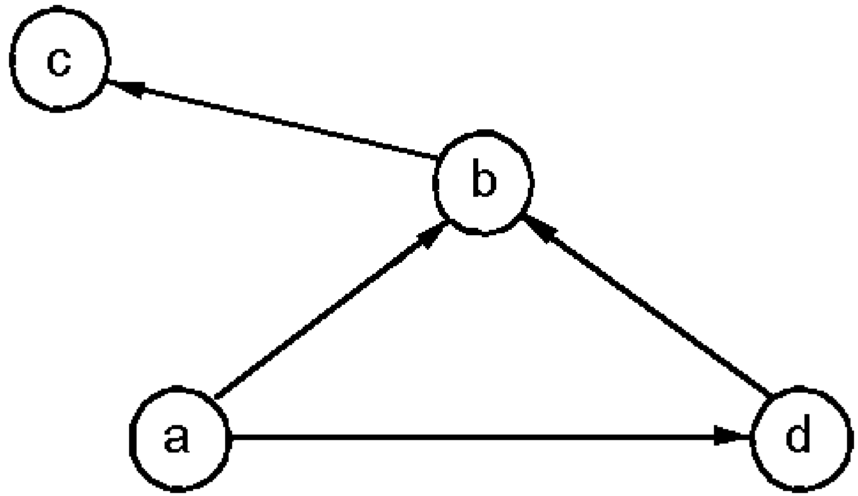
$\overline{Oxide}(MgO) \vee Oxide(MgO) \wedge White(MgO)$

$Oxide(MgO) \wedge White(MgO)$

Применения принципа резолюции

2. Планирование перемещений робота

Задача формулируется как задача поиска пути на графе состояний:



При этом преобразование состояния ***s1*** в состояние ***s2*** задается функцией ***s2=f(s1).*** Для участка графа состояний можно записать **аксиому** $P(s_1) \to Q(s_2)$ или $P(s_1) \to Q(f(s_1))$, где ***P(*s1*)*** – предикат, описывающий свойство начального состояния участка графа; ***Q(s2)*-** предикат, задающий свойство конечного состояния участка графа. Иными словами, если состояние ***s1*** обладает свойством ***P***, то состояние ***s2=f(s1)*** обладает свойством ***Q*.**

Применения принципа резолюции

Введем функцию ***move(x,y,s),*** которая соответствует перемещению робота из точки ***x*** в точку ***y*** в состоянии ***s***. В этом случае можно записать следующие **аксиомы**:

$$AT(a, s_0),$$

$$\forall s_1 (AT(a, s_1) \to AT(b, move(a, b, s_1))),$$

$$\forall s_2 (AT(a, s_2) \to AT(d, move(a, d, s_2))),$$

$$\forall s_3 (AT(b, s_3) \to AT(c, move(b, c, s_3))),$$

$$\forall s_4 (AT(d, s_4) \to AT(b, move(d, b, s_4))).$$

Целевое утверждение формулируется следующим образом: существует ли состояние задачи ***s***, при котором робот будет находиться в точке *"**c**"?*

$$\exists s AT(c, s)$$

Применения принципа резолюции

Дерево опровержения

$AT(b, s_3) \vee AT(c, move(b, c, s_3))$ $\qquad$ $\overline{AT}(c, s)$

$\{move(b, c, s_3)/s\}$

$\overline{AT}(a, s_1) \vee AT(b, move(a, b, s_1))$ $\qquad$ $\overline{AT}(b, s_3)$

$\{move(a, b, s_1)/s_3\}$

$AT(a, s_0)$ $\qquad$ $\overline{AT}(a, s_1)$

$\{s_0/s_1\}$

NIL

$s = move(b,\ c,\ move(a,\ b,\ s_0)).$

Принцип резолюции и язык Пролог

Доказательство теорем в Прологе основано на представлении предложений в форме дизъюнктов Хорна. ***Дизъюнкт Хорна*** – это предложение, содержащее не более одного положительного литерала. Например, $\overline{P} \vee \overline{Q} \vee \overline{R} \vee S$. В Прологе используется **три типа дизъюнктов Хорна**:

1) дизъюнкт, состоящий только из одного положительного литерала, например, Q;
2) дизъюнкт, содержащий один положительный и произвольное число отрицательных литералов, например, $Q \vee \overline{P}1 \vee \overline{P}2 \vee ... \vee \overline{P}n$;
3) дизъюнкт, состоящий только из отрицательных литералов, например, $\overline{P}1 \vee \overline{P}2 \vee ... \vee \overline{P}n$.

Приведенный выше дизъюнкт Хорна **второго типа** соответствует импликации , которая на языке Пролог выражается **правилом**:

q:-p1,p2,…,pn.

Принцип резолюции и язык Пролог

Рассмотрим пролог-программу, проверяющую вхождение элемента **H** в список:

member(H,[H|T]).
member(H,[X|T]):-member(H,T).

?- member(c,[a,b,c]).

В этом случае пролог-система будет доказывать невыполнимость множества:

$$S = \{member(H,[H\,|\,T]),$$
$$member(H,[X\,|\,T] \vee \overline{member}(H,T]),$$
$$\overline{member}(c,[a,b,c])\}.$$

Дерево опровержения для отношения member

$$member(H,[X\,|\,T]) \vee \overline{member}(H,T) \qquad \overline{member}(c,[a,b,c])$$

$$\alpha 1 = \{c/H, a/X, [b,c]/T\}$$

$$member(H',[X'|T']) \vee \overline{member}(H',T') \qquad \overline{member}(c,[b,c])$$

$$\alpha 2 = \{c/H', b/X', [c]/T'\}$$

$$member(H'',[H''|T'']) \qquad \overline{member}(c,[c])$$

$$\alpha 3 = \{c/H'', []/T''\}$$

$$NIL$$

append([ ],L,L).
append([H|T],Y,[H|W]):-append(T,Y,W).

?-append([a,b],[c],X).

Дерево опровержения для отношения append

$append([H\,|\,T],Y,[H\,|\,W]) \vee \overline{append}(T,Y,W)$

$\overline{append}([a,b],[c],X)$

$\alpha 1 = \{a/H,[b]/T,[c]/Y,[a\,|\,W]/X\}$

$append([H'|T'],Y',[H'|W']) \vee \overline{append}(T',Y',W')$

$\overline{append}([b],[c],W)$

$\alpha 2 = \{b/H',[]/T',[c]/Y',[b\,|\,W']/W\}$

$append([],L,L)$

$\overline{append}([],[c],W')$

$\alpha 3 = \{[c]/L,[c]/W'\}$

NIL

$X \leftarrow [H\,|\,W] = [a\,|\,W],$

$W \leftarrow [H'|\,W'] = [b\,|\,W'],$

$W' \leftarrow L = [c],$

X=[a | [b | [c]]]=[a | [b,c]]=[a,b,c].

В.Бондарев