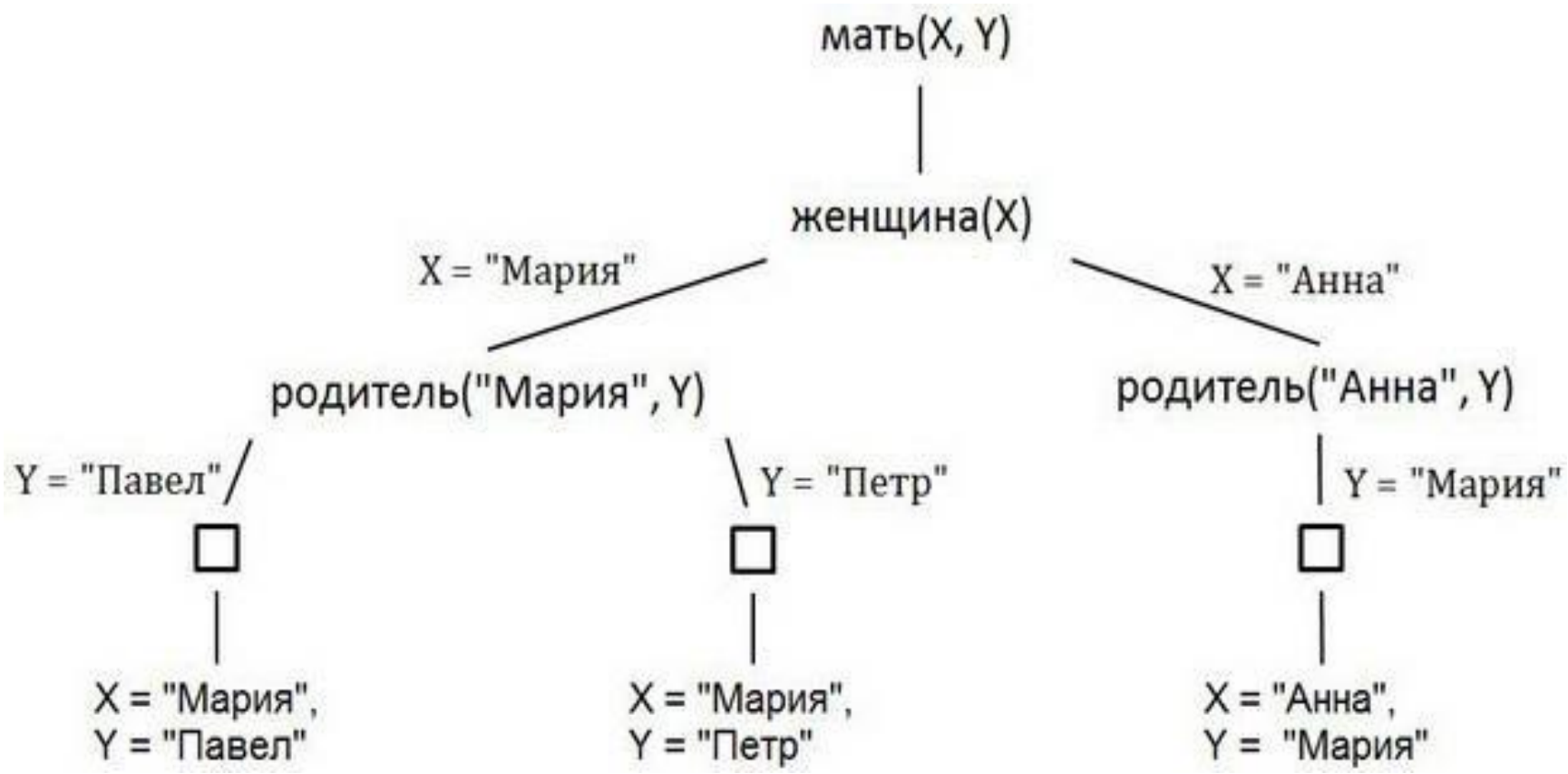
ПРОЛОГ

Управление поиском решения

Устройство вычислений в Прологе

clauses
родитель("Иван","Мария").родитель("Анна","Мария").родитель("Мария","Павел").
родитель("Мария", "Петр").
женщина("Мария"). женщина("Анна").
мать(X, Y):- женщина(X), родитель(X, Y).



Машина вывода Пролога использует для доказательства цели поиск в глубину.

Отрицание

```
супруг("Иван", "Анна").
мужчина("Иван").
мужчина("Петр").
мужчина("Степан").
```

Неженатыми мужчинами, которых можно найти с помощью запроса

```
goal мужчина(X), not(супруг(X, _)).
```

являются Петр и Степан.

Откат под знаком отрицания после достижения цели не производится (для другого доказательства цели), значения из-под него не возвращаются.

Под знаком отрицания неконкретизированные переменные считаются анонимными.

Логические задачи

Три молодые мамы Анна, Ирина и Ольга, гуляя в парке со своими малышами, встретили свою подругу. На вопрос, как зовут малышей, желая подшутить над подругой, они ответили:

Анна: моего малыша зовут Денис, а Кирилл – сын Ирины.

Ирина: моего сына зовут Максим, а Кирилл - сын Анны зовут.

Ольга: мой мальчик – Кирилл, а сына Анны зовут Максим.

Каждая из них один раз сказала правду и один раз солгала.

Как зовут мальчиков Анны, Ирины, Ольги?

```
        predicates
    name(symbol) son(symbol)
    result(symbol, symbol)
    solve(symbol,symbol,symbol,symbol,symbol,symbol)
        clauses
    name(anna). name(irina). name(olga). son(denis). son(kiril). son(maxim).
    result(X,Y):-name(X),X=anna,son(Y),Y=denis;
              name(X),X=irina,son(Y),Y=kiril.
    result(X,Y):-name(X),X=irina,son(Y),Y=maxim;
              name(X),X=anna,son(Y),Y=kiril.
    result(X,Y):-name(X),X=olga,son(Y),Y=kiril;
              name(X),X=anna,son(Y),Y=maxim.
    solve(X1,Y1,X2,Y2,X3,Y3):-X1=irina,result(X1,Y1),X2=anna,result(X2,Y2),
                        X3=olga,result(X3,Y3),Y1<>Y2,Y2<>Y3,Y1<>Y3.
    goal solve(N1,M1,N2,M2,N3,M3),write(N1,' ',M1,'\n',N2,' ',M2,'\n',N3,' ',M3),nl,fail.
```

Управление поиском с возвратом
Предикат fail

fail – это тождественно-ложный предикат, искусственно создающий ситуацию неуспеха. После выполнения этого предиката управление передается в точку отката и поиск продолжается. Использование предиката **fail** позволяет найти все решения задачи.

Пример.

База знаний содержит факты вида: ***student(имя, курс).*** Создать проект, позволяющий сформировать список студентов 1-го курса.

Решение:

```
PREDICATES
student(symbol,integer)
spisok
CLAUSES
student(vova,3). student(lena,1). student(dima,1). student(ira,2).
student(marina,1).
spisok:-student(X,1),write(X),nl,fail.
GOAL
write("Список студентов 1-курса"),nl,spisok.
```

Результат выполнения программы:

```
Список студентов 1-курса
lena
dima
marina
```

Управление поиском с возвратом
Предикат отсечения (cut) (!)

Чтобы ограничить пространство поиска и прервать поиск решений при выполнении какого-либо условия, используется предикат ***отсечения*** (обозначается **!**).

Однажды пройдя через отсечение, невозможно вернуться назад, т.к. этот предикат является тождественно-истинным.

Процесс может только перейти к следующей подцели, если такая имеется.

Например,

p :- p1, p2, **!**, p3.

Если достигнуты цели p1 и p2, то возврат к ним для поиска новых решений невозможен.

Предикат отсечения (cut) (!)

Пример.

Имеется база знаний, содержащая данные о спортсменах: имя и вид спорта. Определить возможные пары одного из спортсменов-теннисистов с другими теннисистами.

Решение:

```
DOMAINS
имя,вид_сп=string
PREDICATES
играет(имя,вид_сп)  спис_спортс
CLAUSES
играет("Саша",теннис).
играет("Аня",волейбол).
играет("Олег",футбол).
играет("Коля",теннис).
играет("Саша",футбол).
играет("Андрей",теннис).
спис_спортс:- играет(X,теннис),!,играет(Y,теннис),
              X<>Y,write(X,"-",Y),nl,fail.
GOAL
write("Пары теннисистов"),nl,
спис_спортс.
```

Результат выполнения программы:

Пары теннисистов
Саша-Коля
Саша-Андрей

Предикат отсечения (cut) (!)

Пример.

Студенту в зависимости от набранной в процессе обучения суммы баллов **Z** присваивается квалификация:

магистр (**M**), если 80<=Z<=100
специалист (**S**), если 60<= Z<80
бакалавр (**B**), если 40<= Z<60
неудачник (**N**), если 0<=Z<40

Составить программу, которая определит квалификацию в зависимости от введенного значения **Z.**

```
DOMAINS z=integer  r=string
PREDICATES grade(z,r)
CLAUSES
grade(Z,""):-Z<0,!, write("Неверный ввод данных!").
grade(Z,""):-Z>100,!,write("Неверный ввод данных!").
grade(Z,"M"):-Z>=80,!.
grade(Z,"S"):-Z>=60,!.
grade(Z,"B"):-Z>=40,!.
grade(_,"N").
GOAL
write("Z="), readint(Z), grade(Z,R),write(R).
```

Математические операции и функции

X + Y	Сумма X и Y
X – Y	Разность X и Y
X * Y	Произведение X и Y
X / Y	Деление X на Y
X mod Y	Остаток от деления X на Y
X div Y	Целочисленное деление X на Y
abs(X)	Абсолютная величина числа X
sqrt(X)	Квадратный корень из X
random(X)	Случайное число в диапазоне от 0 до 1
random(Int,X)	Случайное целое число в диапазоне от 0 до Int-1
round(X)	Округление X
trunc(X)	Целая часть X
sin(X)	Синус X
cos(X)	Косинус X
arctan(X)	Арктангенс X
tan(X)	Тангенс X
ln(X)	Натуральный логарифм X
log(X)	Логарифм X по основанию 10