Лекция 9. Логическое программирование. Синтаксис и значение программ Prolog

Ключевые понятия: ***атом***, ***число, структура, функтор, согласование.***

1 Объекты данных

На рис. 1 представлена классификация объектов данных в языке Prolog. Система Prolog распознает тип объекта в программе по его синтаксической форме. Это возможно благодаря тому, что в синтаксисе языка Prolog определены разные формы для объектов данных каждого типа. Ранее уже рассматривался способ проведения различий между атомами и переменными; переменные начинаются с прописных букв, а атомы - со строчных букв. Системе Prolog не требуется сообщать какую-либо дополнительную информацию (наподобие объявления типа данных) для того, чтобы она распознала тип объекта.

Рис. 2.1. Классификация объектов данных в языке Prolog

Атомы и числа

Атомы и переменные могут формироваться тремя способами:

•как строки букв, цифр и символов подчеркивания, начинающиеся с прописной буквы (tom, ann_123)

•как строки специальных символов (<->, ::- и т.д.), часть из которых (:-) уже имеет предопределенное значение;

•как строки символов в одинарных кавычках ('Sarah Jones') – атом начинается с заглавной буквы, но не является переменной

Числа, используемые в языке Prolog, подразделяются на целые числа и числа с плавающей точкой. Целые числа имеют простой синтаксис: : 1313 0 -97

Диапазон целых чисел ограничен интервалом между некоторым наименьшим и наибольшим числами, которые допустимо использовать в конкретной реализации Prolog.

Предполагается, что для представления чисел с плавающей точкой применяется простой синтаксис:

3.14 -0.0035 100.2

Обычно в программах на языке Prolog числа с плавающей точкой используются не очень часто. Причина этого состоит в том, что Prolog в основном предназначен для символьных, нечисловых вычислений. В символьных вычислениях часто применяются целые числа, например для подсчета количества элементов в списке, но необходимость в использовании чисел с плавающей точкой, как правило, возникает гораздо реже.

Кроме такого отсутствия необходимости использовать числа с плавающей точкой в типичных приложениях Prolog, есть еще одна причина, по которой следует избегать чисел с плавающей точкой. Как правило, необходимо стремиться к тому, чтобы смысл программ был как можно более очевидным. Но введение чисел с плавающей точкой иногда приводит к трудно диагностируемым нарушениям в работе программы из-за числовых ошибок, которые возникают при округлении во время выполнения арифметических операций. Например, при вычислении выражения 10000 + 0.0001 - 10000 может быть получен результат 0 вместо правильного результата 0. 0001.

Переменные

Имена переменных представляют собой строки, состоящие из букв, цифр и символов подчеркивания. Они начинаются с прописной буквы или символа подчеркивания:

X, Result, Object2, Participant_list, ShoppingList, _x23

Если переменная появляется в предложении только один раз, для нее не обязательно предусматривать имя. В этом случае можно использовать так называемую *анонимную* переменную, которая записывается как один символ подчеркивания. Например, рассмотрим следующее правило:

hasachild (X) :- parent (X, Y).

Это правило гласит: "Для всех X, X имеет ребенка, если X является родителем некоторого Y". Т.е., определено свойство hasachild, которое, как здесь подразумевается, не зависит от имени ребенка. Таким образом, переменная с обозначением ребенка - подходящее место для использования анонимной переменной. Следовательно, приведенное выше предложение может быть записано следующим образом: hasachiid(X) :- parent (X, _).

Каждый раз при появлении в предложении символа подчеркивания он представляет новую анонимную переменную. Например, можно утверждать, что в мире есть некто, имеющий ребенка, если существуют два объекта, один из которых является родителем другого:

somebody_has_child :- parent (_, _).

Это предложение эквивалентно следующему:

somebody_has_child :- parent (X, Y).

Но отличается от следующего:

somebody_has_child :- parent (X, X).

Если анонимная переменная встречается в предложении вопроса, то ее значение не выводится. Т.е. если требуются имена людей, имеющих детей, но не имена самих детей, то запрос можно задать в следующем виде:

?- parent(X, _)

Лексическая область определения имен переменных представляет собой одно предложение. Это означает, например, что если имя X15 встречается в двух предложениях, то оно обозначает две разные переменные. Но каждое вхождение X15 в одном и том же предложении соответствует одной и той же переменной. Для констант ситуация будет иной: один и тот же атом всегда обозначает в любом предложении (а следовательно, и во всей программе) один и тот же объект.

Структуры

Структурированными объектами (или *структурами)* называются объекты, которые имеют несколько компонентов. Сами компоненты, в свою очередь, также могут быть структурами. Например, дата может рассматриваться как структура с тремя компонентами: число, месяц, год. Несмотря на то что структуры состоят из нескольких компонентов, они рассматриваются в программе как целостные объекты. Для соединения компонентов в целостный объект необходимо выбрать *функтор.* В данном примере подходящим функтором является date. Например, дату "1 мая 2001 года" можно записать следующим образом:

date (1, may, 2001)

Рис. 2.2. Дата как пример структурированного объекта: а) представленного в виде дерева; б) рассматриваемого в той записи, которая применяется в языке Prolog

В этом примере все компоненты (два целых числа и один атом) являются константами. Компоненты могут также представлять собой переменные или другие структуры. Объект, который обозначает любое число мая, можно представить в виде следующей структуры: date (Day, may, 2001).

Обратите внимание, что Day является переменной и может конкретизироваться значением любого объекта на каком-либо из следующих этапов выполнения программы.

Этот метод структурирования данных является одной из причин того,

почему Prolog можно применять для решения проблем, связанных с символическими манипуляциями.

С точки зрения синтаксиса все объекты данных в языке Prolog представляют собой термы. Например, термами являются объекты may и date(1, may, 2001)

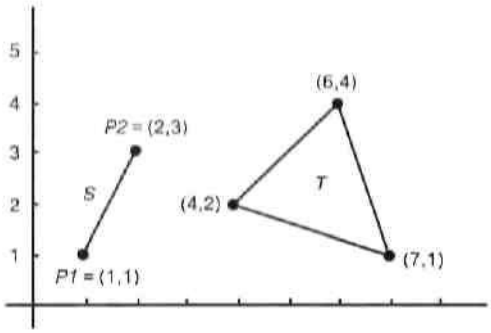
Все структурированные объекты можно представить графически в виде деревьев (см. рис. 2). Корнем дерева является функтор, а ветвями - компоненты. Если компонент представляет собой структуру, он становится поддеревом этого дерева, которое соответствует всему структурированному объекту.

В следующем примере показано, как могут использоваться структуры для представления некоторых простых геометрических объектов (рис. 3). Точка в двухмерном пространстве определена двумя своими координатами; отрезок прямой определен двумя точками, а треугольник можно определить тремя точками. Предположим, что выбраны следующие функторы:

• point, для обозначения точек;

• seg, для обозначения отрезков прямых;

• triangle, для обозначения треугольников.



Рис, 2.3. Некоторые простые геометрические объекты

В этом случае объекты, изображенные на рис.3, могут быть представлены следующим образом:

P1 = point (1,1)

P2 = point(2,3)

S = seg(P1, P2) = seg(point(1,1), point (2,3))

T = triangie(point(4,2), point(6,4), point (7,1))

Соответствующее древовидное представление этих объектов показано на рис. 4. Как правило, функтор, находящийся в корне дерева, называется *главным функтором терма.*

Если бы в той же программе были также точки в трехмерном пространстве, то для их представления можно было бы использовать другой функтор, скажем, point3: points (X, Y, Z)

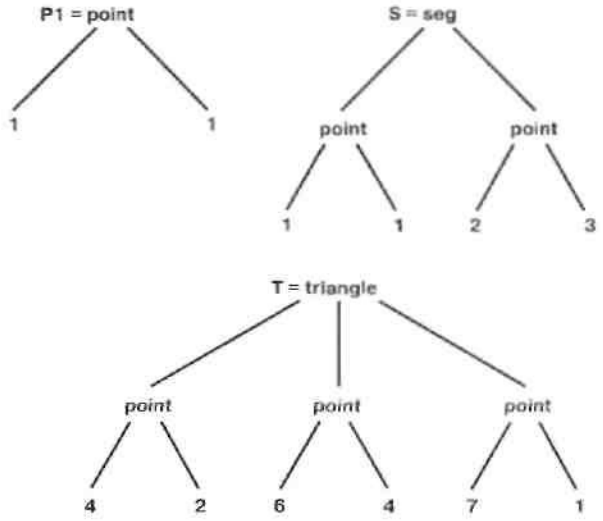
Но допускается использовать одно и то же имя, в данном случае point, для

точек в двух- и в трехмерном пространстве и, например, записывать: point;(X1, Y1) и point$_t$X, Y, Z)

Если одно и то же имя встречается в программе в двух разных ролях, как в приведенном выше случае с функтором point, система Prolog определяет различие между ними по количеству параметров и интерпретирует само это имя как два функтора, поскольку один из них имеет два параметра, а другой - три. Это связано с тем, что каждый функтор определяется следующими двумя признаками.

1.*Имя,* которое имеет такую же синтаксическую структуру, как и атомы.

2.*Арность,* т.е. количество параметров.



Рис, 2.4, Древовидное представление обьектов, показанныхна рис. 2.3

Как было описано выше, все структурированные объекты в языке Prolog представляют собой деревья, которые отображаются л программе с помощью термов. Рассмотрим еще один пример для иллюстрации того, как можно естественным образом представить с помощью термов Prolog сложные объекты данных. На рис. 2.5 показана древовидная структура, которая соответствует следующему арифметическому выражению: (а + Ь) * (с – 5)

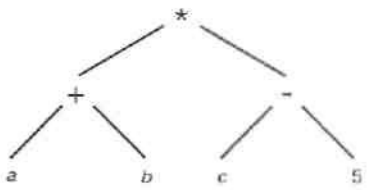


Рис. 2.5. Древовидная структура, соответствующая арифметическому ***выражению*** *(a + b) * (c - 5)*

В соответствии с синтаксисом термов, описанным выше, это выражение можно записать, используя символы -, + и * в качестве функторов, следующим образом:

* (+ (a,b) , - (c, *5))*

Это допустимый терм Prolog, но обычно такая форма записи вряд ли является приемлемой**.** Люди, как правило, предпочитают использовать общепринятую инфиксную систему обозначений, которая широко распространена в математике. В действительности язык Prolog также позволяет применять инфиксную систему обозначений таким образом, чтобы символы "-", "+ " и "*" записывались как инфиксные знаки операций.