

С. Григорьев

Йошкар-Ола

Графические средства системы Пролог-Д

Практически все современные учебные системы программирования предполагают использование компьютерной графики, она является мощным дидактическим средством. В традиционной компьютерной графике построение изображения на экране интерпретируется серией команд. Таким образом, всякое изображение ассоциируется с алгоритмом его построения и служит для иллюстрации алгоритмического подхода к работе ЭВМ. Однако эта концепция не укладывается в принципы декларативного программирования, принятые в системах логического программирования, к которым относится Пролог-Д. Если последовать концепции логического программирования, то необходимым становится описание элементов рисунка в виде совокупности графических объектов, соотношений и связей между ними. В этом случае описание последовательности действий художника-исполнителя становится излишним (а может ли настоящий художник быть исполнителем?).

В системе Пролог-Д определен набор графических примитивов, отображающих графические объекты и построенных таким образом, что с точки зрения синтаксиса каждый из них может быть только целью и принимает значение «истина» (выполняется), если на экране появляется графическое изображение объекта. При записи в правиле нескольких графических примитивов и выполнении данного правила на экране появляется объединение графических образов в той последовательности, как они описаны в данном правиле.

В системе Пролог-Д **БК-0010** предусмотрен один встроенный графический предикат $ОТР(x,y,z,t,c)$. Он выполняется, если на экране генерируется отрезок цвета c , соединяющий точки (x,y) и (z,t) . Аргументы x,y,z,t должны быть целыми или переменными, связанными с целыми, причем x, z должны быть в пределах от 0 до 256, а y,t - в пределах от 0 до 512.

В качестве примера приведем описание угла, вершина которого находится в точке (x, y) .

```
угол(x,y)<-ОТР(x,y,10,10,1),ОТР(x,y,50,50,1);  
?угол(100,100);
```

Сначала будет нарисован отрезок, соединяющий точки $(100,100)$ и $(10,10)$, а затем отрезок, соединяющий точки $(100,100)$ и $(50,50)$. Если бы пятым аргументом предикатов $ОТР$ было число, равное нулю, то точки отрезков были бы невидимы.

Не обязательно, чтобы описание всей картинки было записано в одном предложении. Часть описания может быть выделена в виде отдельного предложения. Программу предыдущего примера можно модифицировать:

```
угол(x,y)<-ОТР(x,y,10,10,1), продолжение(x,y);  
продолжение(x,y)<-ОТР(x,y,50,50,1);  
?угол(100,100);
```

Новая программа выполняет те же самые функции.

Система Пролог-Д допускает возможность использования переменных в графических примитивах. В качестве примера приведем описание вектора, выходящего из точки A с координатами (x,y) в точку B с координатами (s,t) :

```
вектор(A(x,y),B(s,t))<-ОТР(x,y,s,t,1);
```

Необходимо отметить особенность графических объектов, описываемых с помощью переменных. В процессе работы системы может оказаться, что какая-то переменная в описании графического примитива не определена. В этом случае графический примитив все равно будет выполнен, однако переменная принимает все допустимые для нее значения. Иными словами, на экране появится геометрическое место точек, задаваемое уравнением графического объекта. В качестве примера приведем вопрос

$?OTR(0,0,x,100,1);$

В результате ответа на этот вопрос на экране появится треугольник белого цвета. Причина этого в том, что величина абсциссы второй точки не определена, в этом случае абсцисса должна быть любым числом, в допустимых пределах. Как правило, область допустимых значений ограничена размерами экрана.

Языковые средства Пролога-Д обеспечивают возможность наращивать определения, естественным путем поддерживают структурность описания объекта. В качестве примера приведем описание построения домика. Домик можно определить как треугольник и квадрат, совмещенные одной стороной. Квадрат можно определить посредством четырех отрезков:

$квадр(x,y,z,t) \leftarrow -OTR(x,y,x,t,1), OTR(x,y,z,y,1), OTR(x,t,z,t,1), OTR(z,y,z,t,1);$

В этом выражении нет нарушения синтаксиса, однако длина этого предложения больше, чем допустимые 64 символа, и чтобы оно было выполнено в рамках системы Пролог-Д БК-0010, необходимо разделить выражение, например, на две части:

$квадр(x,y,z,t) \leftarrow -OTR(x,y,x,t,1), OTR(x,y,z,y,1), ч2(x,y,z,t);$
 $ч2(x,y,z,t) \leftarrow -OTR(x,t,z,t,1), OTR(z,y,z,t,1);$

Аналогично определяется треугольник:

$треуг(x,y,z,t) \leftarrow -OTR(x,y,t,y,1), СЛ(x,t,r), ДЕ(r,2,f), OTR(x,y,f,z,1), OTR(f,z,t,y,1);$

Предикат с именем ДЕ - предикат деления первого аргумента на второй, частное в третьем, а предикат с именем СЛ - сложение первого аргумента со вторым, результат в третьем аргументе. К сожалению, и это определение не входит в прокрустово ложе 64 символов. Вот окончательный вариант разделения:

$треуг(x,y,z,t) \leftarrow -OTR(x,y,t,y,1), ДЕ2(x,t,f), угол(x,y,z,t,f);$
 $угол(x,y,z,t,f) \leftarrow -OTR(x,y,f,z,1), OTR(f,z,t,y,1);$
 $ДЕ2(x,t,f) \leftarrow -СЛ(x,t,r), ДЕ(r,2,f);$

Определение дома длиной 20 единиц, высотой этажа 10 единиц и высотой крыши 20 единиц имеет вид

$дом(x,y) \leftarrow -крыша(x,y,r,f), этаж(x,y,f);$
 $этаж(x,y,f) \leftarrow -СЛ(y,20,z), квадрат(x,y,f,z);$
 $крыша(x,y,r,f) \leftarrow -СЛ(r,10,y), СЛ(x,20,f), треуг(x,y,r,f);$

Вновь необходимо разделить это описание на две части: в одной части определяется крыша, а в другой этаж (это тот случай, когда дом приходится начинать строить с крыши).

Использование рекурсивных определений дает возможность записать базу знаний более лаконично. Рекурсия может быть использована и для создания динамически изменяющегося графического объекта. Для этого на одном и том же месте

последовательно фиксируется образ объекта так, что цвет его элементов попеременно меняется от цвета фона до цвета, определяемого в базе знаний.

Пример - описание летящей птицы, машущей крыльями. В предикате *взмах* описан взмах вниз и затем взмах вверх. Первому положению соответствует горизонтальное положение отрезка белого цвета, затем этот отрезок гасится, становится цветом фона. Положению «вверх» соответствует угол, состоящий из двух отрезков белого цвета, которые затем гасятся. Каждое из понятий «вверх» и «вниз» описывается отдельно, при этом производятся арифметические операции, необходимые для вычисления координат начала и конца отрезков. Периодическое повторение взмаха вверх и вниз осуществляется с помощью рекурсивного обращения к одному и тому же предикату птица. (Внимание! В некоторых вариантах транслятора не предусмотрена обработка возникающей в данном примере эффекта концевой рекурсии типа петли. Следствием этого может быть аварийный останов после нескольких десятков взмахов). Полное описание птицы, машущей крыльями, приведено ниже.

```
птица(х,у)<-взмах(х,у),птица(х,у);
взмах(х,у)<-вниз(х,у,1),вниз(х,у,0),вверх(х,у,1),вверх(х,у,0);
вниз(х,у,с)<-сдв(х,у,з,т,у,в),ОТР(з,у,у,у,1);
вверх(х,у)<-сдв(х,у,з,т,у,в),ОТР(х,у,з,т,1),ОТР(х,у,у,в,1);
сдв(х,т,з,т,у,в)<-сдп(х,у,з,т),сдл(х,у,у,в);
сдп(х,у,з,т)<-СЛ(х,5,з),СЛ(т,5,у);
сдл(х,у,з,т)<-СЛ(з,5,х),СЛ(т,5,у);
```

Декларативность языка позволяет создавать достаточно сложные картины, используя известные принципы декомпозиции графического объекта на части и последующее их описание.

Например, картина, содержащая дом и летящую птицу, есть пример и описания сложного синтетического объекта, и использования рекурсии для определения динамического изменения объекта.

```
СЛ(а,б,с)<-ВЫЧ(1,а,б,с);
дом(х,у)<-крыша(х,у,г,ф),этаж(х,у,ф);
этаж(х,у,ф)<-СЛ(у,20,з),квадр(х,у,ф,з);
крыша(х,у,г,ф)<-СЛ(г,10,у),СЛ(х,20,ф),треуг(х,у,г,ф);
треуг(х,у,з,т)<-ОТР(х,у,т,у,1),ДЕ2(х,т,ф),угол(х,у,з,т,ф);
угол(х,у,з,т,ф)<-ОТР(х,у,ф,з,1),ОТР(ф,з,т,у,1);
ДЕ2(х,т,ф)<-СЛ(х,т,г),ДЕ(г,2,ф);
квадр(х,у,з,т)<-ОТР(х,у,х,т,1),ОТР(х,у,з,у,1),ч2(х,у,з,т);
ч2(х,у,з,т)<-ОТР(х,т,з,т,1),ОТР(з,у,з,т,1);
птица(х,у)<-взмах(х,у),птица(х,у);
взмах(х,у)<-вниз(х,у,1),вниз(х,у,0),вверх(х,у,1),вверх(х,у,0);
вниз(х,у,с)<-сдв(х,у,з,т,у,в),ОТР(з,у,у,у,1);
вверх(х,у)<-сдв(х,у,з,т,у,в),ОТР(х,у,з,т,1),ОТР(х,у,у,в,1);
сдв(х,т,з,т,у,в)<-сдп(х,у,з,т),сдл(х,у,у,в);
сдп(х,у,з,т)<-СЛ(х,5,з),СЛ(т,5,у);
сдл(х,у,з,т)<-СЛ(з,5,х),СЛ(т,5,у);
```

В ответ на вопрос:

```
?дом(70,110),птица(120,50);
```

на экране будет построен дом и нарисована птица, машущая крыльями. Обратите внимание на одно обстоятельство: предикат птица должен быть описан после предиката дом, так как в нем содержится концевая рекурсия типа петли.

В более мощной системе Пролог-Д **MSX** число встроенных графических предикатов больше. Полный их перечень приведен в таблице. Все аргументы - целые или переменные, конкретизированные целыми.

| Функция | Действие |
|------------------------|--|
| ФОН(Т, F) | Установка цвета текста и фона. Переменная конкретизируется целым, соответствующим установленному цвету |
| ТОЧКА(Х, Y, C) | Генерация точки на экране с координатами X, Y и цветом C |
| ЛИНИЯ(Х, Y, U, V, C) | Генерация линии из точки с координатами X, Y в точку с координатами U, V и цветом C |
| ОКРУЖНОСТЬ(Х, Y, R, C) | Генерация окружности радиусом R с центром в X, Y |

В качестве примера приведем описание двух синих окружностей с центром в точке с координатами 100 и 100, радиусами 10 и 50:

```
синие-окружности<-ОКРУЖНОСТЬ(100,100,10,5),ОКРУЖНОСТЬ(100,100,50,5);
?синие-окружности;
```

Сначала будет нарисована окружность с радиусом 10, а затем окружность с радиусом 50.

Не обязательно, чтобы описание всей картинке было записано в одном предложении. Часть описания может быть выделена в виде отдельного предложения. Программу предыдущего примера можно модифицировать:

```
синие-окружности<-ОКРУЖНОСТЬ(100,100,10,5),продолжение;
продолжение<-ОКРУЖНОСТЬ(100,100,50,5);
?синие-окружности;
```

Новая программа выполняет те же функции. Использование переменных в графических примитивах аналогично использованию в системе Пролог-Д БК-0010.

Обратившись к опыту художников-примитивистов, нарисуем дерево в виде ствола - отрезка линии и кроны - окружности.

```
дерево(х,у,н,г)<-СЛОЖЕНИЕ(г,н,у),ЛИНИЯ(х,у,г,г,
14),СЛОЖЕНИЕ(г,г,г),ОКРУЖНОСТЬ(х,г,г,3);
```

Если к этому определению в базе знаний задать вопрос

```
?дерево(50,50,40,10);
```

то на экране появится одинокое дерево с координатами комеля 50, 50, высотой 40 и радиусом кроны 10.

Языковые средства Пролога-Д обеспечивают возможность наращивания определения, естественным путем поддерживают структурность описания объекта. Использование рекурсивных определений дает возможность записать базу знаний более лаконично.

Построим поляну с деревьями.

```
сад(1, x, y, h, r) ← дерево(x, y, h, r);  
сад(m, x, y, h, r) ← смещение(x, y, z, t), дерево(z, t, h, r), СЛОЖЕНИЕ(n,  
1, m), сад(n, z, t, h, r);  
смещение(x, y, z, t) ← СЛОЖЕНИЕ(x, 10, z), СЛОЖЕНИЕ(y, 10, t);  
поляна ← ОКРУЖНОСТЬ(v,  
180, 50, 2), сад(6, 100, 150, 50, 10), сад(5, 85, 172, 50, 10);
```

Если к этой базе знаний задать вопрос:

?поляна;

то на экране появится искомое изображение.

Поляна изображена с помощью примитива ОКРУЖНОСТЬ, в описании которого не определена одна из координат центра. Это значит, что на экране должны быть нарисованы все возможные окружности, центры которых лежат на горизонтальной прямой, а радиус равен 50. В этом примере использована рекурсия для определения объекта «сад». В нашем понимании сад - несколько деревьев, расположенных на некотором расстоянии друг от друга. Чтобы нарисовать сад, необходимо воспользоваться рекурсивным определением. Оно содержит начальную, или базовую, ситуацию и описание некоторого рекурсивного соотношения. В примере:

- базовой ситуацией будет та, когда в саду одно дерево;
- рекурсивное определение получится тогда, когда будет несколько деревьев, смещенных относительно друг друга (смещение задано в виде специальной функции, именно она и определяет взаимное расположение деревьев).

Упражнения.

1. Напишите на языке Пролог-Д базу знаний, описывающую прямоугольный треугольник.
2. Используя рекурсивное определение, напишите базу знаний, описывающую многоэтажный дом.
3. Опишите на языке Пролог-Д построение улицы без учета и с учетом перспективы.