

С. Григорьев, М. Морозов

Йошкар-Ола

Давайте попробуем Пролог

О традиционных языках и нетрадиционных задачах

Когда рассматривают круг проблем, связанных с изучением и использованием языков программирования в школе, обычно имеют в виду традиционные языки - Бейсик. Рапиру, Паскаль и т.п. В чем главная особенность этих, так называемых процедурных, языков? В ориентации на вычислительную машину. Процедурные языки развивались как средство записи операций, выполняемых вычислительной машиной, и отражают непривычное для человека «машинное мышление». Такой подход предполагает: чтобы ЭВМ решила задачу, необходимо построить алгоритм, определяющий последовательность действий, и запрограммировать его, а машина, точно следуя по указанному пути, сделает то, что было задумано.

Однако подобный образ действий вне физико-математических дисциплин сталкивается с серьезными трудностями.

Ориентированный на математические расчеты Бейсик малопригоден для отражения предметного мира, скажем, географии, истории, иностранного языка. Даже большие возможности символьной обработки, заложенные в Рапире, оказываются недостаточными.

Есть и другая проблема, возникающая при обучении школьников программированию на традиционных языках: поиск конкретных реальных задач, решать которые удобно, используя программирование.

В вузе вопрос решается просто: студентов заставляют работать с численными методами решения математических задач. В младших классах школы проблему пытаются обойти, вводя исполнителя (например, «черепаху»), действующего как ЭВМ. В старших классах все-таки приходится обращаться к вычислению площадей, решению квадратных, нелинейных и других уравнений.

Но школьникам еще предстоит долгий путь до практического применения этих численных методов! Это произойдет, например, когда в вузе будут изучены математические модели реальных физических процессов. А главное, в развитии и использовании современной вычислительной техники наметился явный отход от вычислений. Вчерашнему школьнику предстоит столкнуться в своей практической деятельности с экспертными системами, системами обеспечения принятий решений, базами знаний и всем тем, что принято называть технологией обработки знаний или искусственным интеллектом. А это потребует и других навыков, и другого, «немашинного», мышления.

Возникает необходимость поиска новых, более адекватных средств общения с компьютерами; здесь нельзя не обратиться к активно развивающемуся сейчас логическому программированию и языку программирования Пролог.

Появившись в начале 70-х гг. в лаборатории искусственного интеллекта в университете г. Марселя, Пролог на протяжении почти десяти лет оставался известен лишь узкому кругу специалистов. Но тот факт, что логическое программирование и сам Пролог были положены в основу японского проекта ЭВМ пятого поколения, привлек к языку всеобщее внимание.

В отличие от традиционных языков, которые нужны для изложения способа решения задачи, на Прологе формулируется только ее постановка. Пользователь должен сообщить ЭВМ необходимые факты и несколько правил, описывающих, как эти факты соотносятся

друг с другом, а она из полученной информации сформирует ответ на поставленный вопрос.

В основу Пролога положен сложный математический аппарат логики предикатов первого порядка и теории логического вывода. Однако, чтобы изучить этот язык и работать с ним, нет необходимости вникать в тонкости математической логики, как нет необходимости вникать в принципиальную схему телевизора, прежде чем включить его вечером.

Пролог очень легок для изучения и использования, но, несмотря на это, намного мощнее, чем любой другой язык, используемый в настоящее время на микро-ЭВМ. Часто оказывается, что программа на Прологе почти на порядок короче, чем на процедурном языке программирования.

Против Пролога существует предубеждение, что его эффективная реализация возможна только на больших, мощных ЭВМ. Действительно, в недалеком прошлом это было так, и в СССР Пролог действовал только на ЕС ЭВМ и СМ ЭВМ. Теперь ситуация изменилась. Оказалось, что его можно приспособить для быстрой и эффективной работы на небольших машинах и при этом создать развитые средства программной поддержки, ориентированные на пользователя. Сейчас существует реализация Пролога для всех микро-ЭВМ с операционной системой CP/M, в Институте программных систем АН СССР разработана реализация для микро-ЭВМ «Ямаха».

Эти достижения укрепили позицию Пролога как практичного, простого языка, пригодного для использования в школьной информатике. Широкое применение языка такого типа планируется в болгарских школах для изучения гуманитарных наук; реализуется проект «Логика как язык компьютеров для детей». Аналогичные работы проводятся в СССР.

Программа на Прологе-Д

Опишем процесс программирования на упрощенном логическом языке, реализованном авторами и названном Пролог-Д.

В процедурных языках программа состоит из операторов. В совокупности операторы описывают этапы, необходимые для достижения заданной цели. Программа на Прологе-Д состоит из последовательности предложений (фактов и правил), порядок которых значения не имеет. Совокупность фактов и правил называют базой данных.

Вот пример простой программы:

МАМА (НАДЯ, АСЯ); (1)

МАМА (АСЯ, ДАША); (2)

БАБУШКА (А, В)←МАМА (А, С), МАМА (С, В); (3)

В этой программе предложения (1) и (2) - факты, а предложение (3) - правило. В конце каждого предложения ставится точка с запятой. Факты дают представление об объектах и действиях. Предложение «Надя - мама Аси», записанное в виде факта МАМА (НАДЯ, АСЯ), можно было бы записать и как МАМА (АСЯ, НАДЯ), (исправив при этом (2) на МАМА (ДАША, АСЯ), а в (3) записав БАБУШКА (В, А)).

Важно, чтобы заданная последовательность аргументов сохранялась на всем протяжении программы.

Правило (3) определяет зависимость одних объектов или действий от других и записывается в общем виде так:

СЛЕДСТВИЕ←ФАКТ 1, ФАКТ 2;

Его можно прочесть так: «Если ФАКТ 1 верен, ФАКТ 2 верен, то СЛЕДСТВИЕ тоже верно». Левая часть правила - СЛЕДСТВИЕ - называется головой, а составляющие правой части - целями.

В программе (1)-(3) имена и слова БАБУШКА, НАДЯ, АСЯ, МАМА, ДАША называются атомами. Атомы - это константы, они не меняют своего значения при выполнении программы. Записываются атомы в виде последовательности букв, цифр и пробелов и начинаются с буквы. Константой является и целое, представляющее собой последовательность цифр.

В правиле (3) А, В, С - переменные. Область действия переменных в Прологе ограничивается одним предложением. Если переменная в голове правила принимает какое-либо значение, то его же она принимает и в целях. Переменные могут обозначаться одной буквой или последовательностью букв (заканчивающейся апострофом или нижним подчеркиванием, когда значение переменной несущественно).

Выражения вида (1) и (2) (в частности, выражение БАБУШКА (А, В)) называются термами.

Программа должна заканчиваться одним или несколькими вопросами. Вопрос записывается как факт, но впереди должен стоять вопросительный знак. Если, например, к программе (1)-(3) добавить вопрос

? БАБУШКА (НАДЯ, А);

то машина ответит:

А=ДАША

Вопрос должен соответствовать ряду условий:

- атом, с которого начинается вопрос, должен стоять в голове хотя бы одного из правил базы данных;
- вопрос не должен содержать новых атомов и переменных, в противном случае Пролог-Д ответит «не знаю».

На вопрос, не содержащий переменных, программа ответит «да», если в базе данных есть факт, совпадающий с вопросом, или «нет» в противном случае; если вопрос содержит переменные, то будут напечатаны все возможные их значения. Например, на

? МАМА (А, -);

машина ответит

А = НАДЯ

А = АСЯ

Немного о реализации языка

Поиск ответа на вопрос в Прологе-Д осуществляется доказательством теоремы, основанной на логических предложениях, из которых состоит программа. Доказательство осуществляется методом резолюций и основано на алгоритме, предложенном в работе: Sammut R. A., Sammut C. A. Implementation of UNSW-PROLOG // Austr. Comp. Journ. V. 15. P. 57-90 - и отличающемся достаточно высокой эффективностью.

Интерпретатор языка реализован на Бейсике для ДВК-2 и занимает 8К байт. Этот выбор обусловлен тем, что Бейсик широко распространен. Кроме того, хотя реализация интерпретатора Пролога-Д на интерпретаторе Бейсика и уменьшает скорость работы, зато обеспечивается определенная экономия памяти, необходимой для хранения программы на Прологе-Д.

В настоящее время разрабатывается версия Пролога-Д на Бейсике MSX для «Ямахи» и БК-0010. В ней несколько повышена скорость интерпретации.

Простые программы для разных уроков

География. База данных с фактами вида ГРАНИЧИТ (СТРАНА 1, СТРАНА 2) и одним правилом позволит создать информационную систему, дающую сведения о соседях каждой страны. Например, для юго-запада Европы:

```
ГРАНИЧИТ (ПОРТУГАЛИЯ, ИСПАНИЯ);  
ГРАНИЧИТ (ИСПАНИЯ, ФРАНЦИЯ);  
ГРАНИЧИТ (ФРАНЦИЯ, ШВЕЙЦАРИЯ);  
ГРАНИЧИТ (ФРАНЦИЯ, БЕЛЬГИЯ);  
ГРАНИЧИТ (ФРАНЦИЯ, ФРГ);  
ГРАНИЧИТ (ФРАНЦИЯ, ИТАЛИЯ);  
ГРАНИЧИТ (ФРАНЦИЯ, ЛЮКСЕМБУРГ);  
ГРАНИЧИТ (А, В) ← ГРАНИЧИТ (В, А);
```

На вопрос

```
? ГРАНИЧИТ (ИСПАНИЯ, ФРАНЦИЯ);
```

машина ответит «да», а на вопрос

```
? ГРАНИЧИТ (ИСПАНИЯ, БЕЛЬГИЯ);
```

ответ будет отрицательный.

Можно узнать и всех соседей Испании, задав вопрос

```
? ГРАНИЧИТ (ИСПАНИЯ, А);
```

Ответ будет:

```
А = ПОРТУГАЛИЯ
```

```
А = ФРАНЦИЯ
```

Легко сформулировать базу данных по принадлежности стран частям света, используя факты вида

```
НАХОДИТСЯ (СТРАНА, ЧАСТЬ СВЕТА);
```

Аналогично можно заложить информацию о природных условиях и ресурсах каждой страны, городах, населении, промышленности и т. д.

История. Курс истории содержит большое количество фактов, поэтому историческая информация легко укладывается в конструкции Пролога-Д. Например, база данных с фактами вида

```
СОБЫТИЕ (НАИМЕНОВАНИЕ, СТРАНА, ГОД НАЧАЛА, ГОД КОНЦА)
```

позволит записать основные исторические события, относящиеся к той или иной эпохе. Формулируя различные вопросы, можно узнать, что, когда, где произошло. А добавив в вопросы операции сравнения над целыми числами, можно получить интересные сведения и по хронологии. База данных об исторических деятелях создается с использованием фактов вида

ИСТОРИЧЕСКИЙ ДЕЯТЕЛЬ (ИМЯ, ГОД РОЖДЕНИЯ, ГОД СМЕРТИ);

Иностранный язык. Достаточно просто написать программу, которая будет представлять собой англо-русский и русско-английский словари одновременно. С ее помощью отыскиваются эквиваленты русских и английских слов, проверяется словарный запас. База данных имеет вид

ЗНАЧЕНИЕ (МАМА, МАММУ);
ЗНАЧЕНИЕ (ПАПА, DADDY);
ЗНАЧЕНИЕ (БАБУШКА, GRANNY);
ЗНАЧЕНИЕ (МИР, PEACE);
ЗНАЧЕНИЕ (НЕБО, SKY);

и т. д.

Задав несколько вопросов, можно получить ответы:

? ЗНАЧЕНИЕ (МИР, А);
А = PEACE
? ЗНАЧЕНИЕ (А, SKY);
А = НЕБО
? ЗНАЧЕНИЕ (ПЕЛЯ, А);
НЕ ЗНАЮ
? ЗНАЧЕНИЕ (МАМА, МАММУ);
ДА

Таким образом, используя минимальные средства логического программирования и создавая простейшие базы данных, можно строить интересные, полезные компьютерные системы для самых разных школьных уроков. Если же привлечь мощные средства Пролога для обработки списков, аппарат встроенных функций, то нетрудно построить простые системы автоматического перевода с языка на язык, аналитических преобразований. И как знать, может быть, скоро, овладев аппаратом логического программирования, школьники будут строить свои маленькие экспертные системы? Давайте попробуем поработать с Прологом!