ЯЗЫК "Флэнг" И ЕГО РЕАЛИЗАЦИЯ

Манцивода А.В., Петухин В.А., Иркутск

Рассматривается функционально-логический язык "Флэнг" и его реализация. "Флэнг" разработан для решения задач искусственного интеллекта, для аналитических вычислений, для решения комбинаторных проблем,а также для использования в обучении методам программирования в искусственном интеллекте. "Флэнг" является обобщением таких известных языков,как "Пролог" и "Лисп". Это означает, что в среде "Флэнг" можно работать как в "прологовской",так и в "лисповской" технике. Однако "Флэнг" не является механическим объединением двух стилей программирования - функционального и логического, а представляет собой единое целое с ясной методологией и простой семантикой.

Приведем несколько примеров программ на "Флэнге". Первая программа - определение функции факториал:

```
0!  <= 1;
X!  <= X > 0   X * (X-1)!;
```

Во "Флэнге" функции и отношения не различаются. Любая функция может играть роль предиката. При этом считается, что предикат истинен на каких-то аргументах, если соответствующая функция на аргументах *определена*. Пример - описание генеалогического древа:

```
родитель(Олег, Петр) <= истина;
родитель(Олег, Федор) <= истина;
```

и т.д. Пользуясь отношением *родитель*, определим функцию *предок*, находящую по своим двум аргументам - предку и потомку - цепочку родственников между ними:

```
предок (X, Y)<= родитель(X, Y)   [X, Y];
предок(X, Y) <= родитель(X, Z)   [X,предок(Z,Y)];
```

Это уже функционально-логическое определение, поскольку, хотя *предок* есть функция, возвращающая значение, при ее вычисле - нии Флэнг-система использует возврат.

Возможны и другие - более изящные - применения функционально-логической "технологии". Приведем в качестве примера программу быстрой сортировки:

```
разб(X, [], [], []) <= истина;
разб(X,[Y|Z],[Y|W1], W2) <= X=<Y разб(X, Z, W1, W2);
разб(X, Y|Z , W1,[Y|W2]) <= X>Y разб(X, Z, W1, W2);
сорт([ ], X) <= X;
сорт([X|Y],Z) <= разб(X,Y,W1,W2) сорт(W1, [X|сорт(X2,Z)]);
```

Определение отношения *разб* стандартно, функция же *сорт* написана в духе, навеянном такой тонкой "прологовской" техникой, как разностные списки.

Во "Флэнге" реализован также мощный метод, предназначенный для решения комбинаторных задач и основанный на идеях из [1]. Хотя метод довольно популярен, известна лишь одна хорошая его реализация. Нами разработана абстрактная машина для эффективной реализации этого метода. Пока эта машина эмулируется с помощью интерпретатора. Тем не менее даже эта малопроизводительная система показала очень обнадеживающие результаты (например, расстановка 16 ферзей производится за 0.7 с).

Метод, реализованный во "Флэнге", позволяет решать огромное число комбинаторных задач (таких, как задачи составления расписаний, варианты задачи коммивояжера, оптимальной раскраски графа и других) с эффективностью, не уступающей специализированным программам, написанным на языках низкого уровня. При этом размеры и стоимость разработки программ на "Флэнге" по расчетам должны быть на порядок ниже, чем размеры и стоимость специализированных программ. Сила метода состоит в том, что он в процессе вычислений позволяет очень рано обнаружить, что данный путь решения ведет не к результату, а в тупик. Эта особенность метода резко сокращает пространство поиска решений.

Во "Флэнг" введено много иных средств, например, функции высших порядков, средства для аналитических вычислений и другие.

Наиболее трудной проблемой для языков искусственного интеллекта ("Лисп", "Пролог", "Рефал" и другие) явилось то, что системы, поддерживающие эти языки, обладают производительностью, значительно худшей, чем стандартные императивные языки (такие, как "Си" и "Паскаль"). Нами была разработана технология компиляции языков искусственного интеллекта, мало уступающая по производительности компиляторам с императивных языков. Основным новшеством, позволившим сделать это, стал глобальный анализ программ. Мы реализовали прототип компилятора с "Флэнга", генерирующий код, который исполняется в несколько раз быстрее, чем программы, разработанные и скомпилированные в таких известных системах, как Arity и Turbo Prolog, muLisp. Было проведено сравнение производительности прототипа компилятора с "Флэнга" с компиляторами Arity и Turbo Prolog, а также сравнение с программами, написанными на "Паскале" и откомпилированными в системе TurboPascal. Результаты (в секундах) приведены в таблице:

Задача	"Флэнг"	Arity/Prolog	TurboProlog	TurboPascal
QuickSort	11,86	69.60	20.21	6.04
Insort	12.75	-	26.64	8.29
Akkermann	5.56	95.71	5.88	4.67
Fun6	29.90	201.55	35.37	29.17

В настоящее время разрабатывается коммерческая версия Флэнг -системы.

Литература

1. HENTENRYCK P.,van. Constraint satisfaction in logic Programming. - Cambridge: The MIT Press, 1989.

ПРИМЕНЕНИЕ ЛОГИКИ ДЛЯ РЕШЕНИЯ ЗАДАЧ РЕАЛЬНОГО ВРЕМЕНИ

Мартьянов В.И., Иркутск

Программные комплексы решения задач:

- распознавания ситуаций в условиях неполной и нечеткой информации,
- организации взаимодействия менеджеров;
- поддержки принятия решений

предъявляют высокие требования к быстродействию систем обработки знаний. Современные реализации "Пролога" в значительной мере удовлетворяют этим требованиям, но только на небольших базах знаний (это верно по крайней мере для реализации "Пролога" на ПЭВМ). Решение задач диагностики (распознавания ситуаций) на "Прологе" может производиться только последовательным применением правил (моделирование прямого вывода [1]). Для некоторых предметных областей, где количество правил исчисляется тысячами и десятками тысяч, подобный подход неприемлем в принципе.

Наш опыт решения задач реального времени показывает неплохие перспективы расширений "Пролога" следующими средствами:

- ограниченным прямым выводом;
- правилом разбора случаев.

Не вдаваясь в аспекты создания единой методологической основы такого расширения, рассмотрим задачи, где данные дополнительные средства необходимы.

Прямой вывод необходим в задаче распознавания ситуаций, которую приведем в следующей облегченной от деталей постановке.