Пусть $\mathcal{L}_1$, $\mathcal{L}_2$ - классы схем, которые могут быть получены соответственно из схем Янова с одним одноповоротным счетчиком (из линейных унарных рекурсивных схем с конечноповоротными счетчиками) конечным применением операций суперпозиции, α-дизъюнкции и α-итерации [2].

ТЕОРЕМА 5. *Для схем класса* $\mathcal{L}_1$ *над бесконечной циклической группой неразрешима проблема пустоты.*

ТЕОРЕМА 6. *Для схем класса* $\mathcal{L}_2$ *разрешима проблема эквивалентности.*

Литература

1. ГЛУШКОВ В.М., ЛЕТИЧЕВСКИЙ А.А. Теория дискретных пре - образователей //Избранные вопросы алгебры и логики. - Новоси - бирск: Наука, 1973. - С. 5-39.

2. ГЛУШКОВ В.М., ЦЕЙТЛИН Г.Е., ЮЩЕНКО Е.Л. Алгебра, языки, программирование. - Киев: Наукова думка, 1974.

ЯЗЫК ДОКАЗАТЕЛЬНОГО ПРОГРАММИРОВАНИЯ ЯМС

Макаров В.П., Гомель

Одной из основных проблем доказательного программирования [1,2] является разработка удобных языков доказательного программирования. В заметке излагаются основные принципы построения языка доказательного программирования ЯМС (Язык Математических Спецификаций) - входного языка системы доказательного программирования СИНАЛ, разрабатываемой в настоящее время в Гомельском отделении Вычислительного центра Академии наук Белоруссии.

Используя терминологию [3, с. 82], язык ЯМС можно считать теоретико-множественной надстройкой языка многосортного исчисления предикатов с равенством. Это означает, что в языке ЯМС, кроме обычных термов, имеются также типовые термы, соответствующие основным теоретико-множественным обозначениям, например:

set(t) - "множество всех подмножеств" множества t (множеством называется некоторый типовой терм; простейшими типовыми термами являются имена сортов);

{x:t!P(x)} - подмножество всех таких x из множества t, для которых формула P(x) истинна;

(S -> T) - множество всех функций из S в T.

Конечно, точный смысл этих (и других) обозначений задается соответствующими аксиомами.

Имеются также модельные типовые термы, обозначающие множество всех (конечных) моделей некоторой теории (при фиксированных носителях).

Кроме того, в языке ЯМС возможны описания частично определенных функций и функций со скрытыми параметрами, а также возможно определение отношения наследования между сортами (см. ниже). Все это значительно усложнило проблему описания правильно построенных термов (тривиальную для обычного многосортного исчисления предикатов). В частности, оказалось необходимым при определении каждого терма определять его уровень - некоторое натуральное число, аналогичное уровню переменной в языке простой теории типов Рассела и Уайтхеда [4].

Описание теории T с сортами s1,...,sm в простейшем случае в языке ЯМС имеет вид: $T(s_1,\ldots,sm)\colon = d_1,\ldots,dk$; здесь di - описание некоторого функционального имени из сигнатуры теории T или некоторая аксиома теории T.

Возможно также наследственное описание теории [5] - некоторый аналог описания классов в объектно-ориентированном программировании [6].

Например, если ранее описаны теория групп GROUP с сортом elg (тип элементов группы) и теория линейного порядка LOD с сортом elog (тип элементов линейно упорядоченного множества), то описание теории линейно упорядоченных групп LOG может иметь следующий вид:

LOG(elog): = GROUP(elog); LOD(elog);

A[a,b,x: elog!a $\leq$ b -> a*x $\leq$ b*x & x*a $\leq$ x*b];;

Такое описание вводит отношение прямого наследования между сортами elog и elg и между сортами elog и elod (т.е. сорт elog считается прямым наследником сортов elg и elod и позволяет определить правильно построенные термы вида, например, max(x,y), где max - функция "наибольший из двух элементов", определенная в теории линейного порядка, x,y - некоторые термы типа t, причем t должен быть наследником (не обязательно прямым) сорта elod).

В языке ЯМС допускается определение частично определенных функций. Например, определение (т.е. спецификация) функции sqrt(x,e) - вещественный квадратный корень из вещественного числа x с точностью e может иметь следующий вид:

sqrt(x,e: real; x $\geq$ 0 & e $>$ 0) r: real + abs(r*r-x) $<$ e;

Здесь знак "+" используется как синтаксический разделитель. Формула x $\geq$ 0 & e $>$ 0 задает допустимое множество входных значений и называется условием применимости (соответствует конструкции PRE в языке спецификаций VDM [7]). Терм sqrt(z,e) считается правильно построенным, если системный интерактивный доказатель теорем (prover) сможет доказать истинность формулы $z \geq 0$ & $e > 0$ (возможно, с помощью пользователя).

Данная спецификация является типичным примером непроцедурной спецификации - из нее нельзя непосредственно извлечь алгоритм вычисления квадратного корня. Одним из методов вычисления квадратного корня является, как известно, следующий итерационный метод Ньютона:

$$z(0) = x;$$
$$z(k+1) = (z(k) + x/z(k))/2;$$

Итерации продолжаются, пока не будет достигнута необходимая точность.Так как данный метод сходится,то существует функция k(x,e) - наименьшее число k, при котором abs(z(k)*z(k)-x< < e. Функции z(k) и k(x,e) можно также определить в языке ЯМС и получить "более процедурную" спецификацию, из которой легко извлекается соответствующий алгоритм (в виде, например, программы на языке Паскаль). В языке ЯМС можно записать доказательство теоремы о "правильности" последней спецификации.

Аналогичным образом рассматривается более сложный пример - задача вычисления для некоторой КС-грамматики множества первых символов терминальных цепочек, выводимых из данного нетерминала - эта задача встречается при построении LR(1)- анализаторов.

Литература

1. ЕРШОВ А.П. Научные основы доказательного программирования //Вестник АН СССР. - 1984. -№ 10. -С. 9-19.

2. СВИРИДЕНКО Д.И. Проект СИГМА. Цели и задачи //Логические методы в программировании. - Новосибирск. - 1990. -Вып.133: Вычислительные системы. -С. 68-94.

3. КОЛМОГОРОВ А.Н., ДРАГАЛИН А.Г. Введение в математическую логику. -М.: Из-во МГУ. - 1982. - 120 с.

4. ФРЕНКЕЛЬ А.А., БАР-ХИЛЛЕЛ И. Основания теории множеств. М.: Мир, 1966. - 555 с.

5. МАКАРОВ В.П. Наследственное расширение многосортных теорий //Доклады Академии наук Белоруссии. - 1991. -№12.

6. MEYER B. Object-Oriented Software Construction. - N.J.: Prentice-Hall, 1988. - 534 p.

7. JONES C.B. Systematic Software Development Using VDM. - London: Prentice-Hall, 1986. - 300 p.