А. Т. АРУТЮНЯН, А. А. ЗАКАРЯН, Э. У. КАЗАРЯН

ГЕНЕРАТОР ПРОГРАММ СИСТЕМЫ АНИ-81

Один из основных компонентов современных инструментальных систем является генератор программ (ГП). В статье рассмотрены принципы построения ГП, реализованного в базовой диалоговой системе научных исследований АНИ-81.

Сущность ГП заключается в формировании текста программы на языке высокого уровня на основе спланированных расчетных цепочек модулей.

В процессе работы транслятора с языка описания данных (ЯОД) и планировщика решений формируются информационные таблицы, которые являются входными наборами данных для ГП. Используя эту информацию, ГП системы АНИ-81 (ГПП) генерирует исходную программу на языке ПЛ/1.

После динамического вызова соответствующего компилятора и редактора связей сгенерированная программа может храниться:

а) в базе знаний системы в виде объектного модуля для многократного использования;

б) в предметной области в виде загрузочного модуля, что позволяет автоматически расширять ее.

ГПП работает в двух режимах: однопрограммном и мультизадачном.

В однопрограммном режиме ГПП генерирует программу простой структуры, используя одну выбранную расчетную цепочку модулей. В отличие от этого ГПП в мультизадачном режиме генерирует программу ветвящейся структуры, используя при этом дерево расчетных цепочек.

1. Определения.

Пусть $M=\{\varphi_1, \varphi_2, \ldots, \varphi_n\}$ конечное упорядоченное множество модулей из предметной области.

$V=\bigcup_{i=1}^{n}[U(\varphi_i)\cup V(\varphi_i)]$ множество параметров, где $U(\varphi_i)$, $V(\varphi_i)$ соответственно входные и выходные параметры модуля φ_i. Каждый параметр системы АНИ-81 характеризуется наименованием, описывающим параметр на семантическом уровне. В дальнейшем для упрощения не будем отличать параметр от его наименования.

Параметр имеет следующую структуру: ключевое имя, описатель 1, описатель 2, . . ., описатель l.

Между двумя параметрами вводится отношение частичного

упорядочения. Будем говорить, что $v_1 \leqslant v_2$, $v_1, v_2 \in V$, если $k_1 = k_2$ и $\{O_1^1, \ldots O_{l_1}^1\} \subseteq \{O_1^2, \ldots O_{l_2}^2\}$, где K_1, K_2 ключевые имена, $\{O_1^1, \ldots O_{l_1}^1\}$, $\{O_1^2, \ldots O_{l_2}^2\}$ описатели параметров v_1 и v_2 соответственно.

Например, пусть v_1 = матрица, v_2 = матрица действительная, симметрическая. Тогда $v_1 \leqslant v_2$.

Если значение параметра $v_l \in V$ задано, то такое же значение присваивается всем параметрам, меньше v_l. Рассматриваются параметры трех типов: 1) скаляр, 2) вектор, 3) матрица.

Размерность параметра типа „вектор" или „матрица" может функционально зависеть от значений некоторых параметров.

Пусть T паспорт параметра, запись которого имеет следующий вид:

$$T = \{\alpha, \beta_1, \ldots \beta_l, \text{ПР1}, \text{ПР2}, A, \gamma_1, \ldots \gamma_m\},$$

где α ключевое имя параметра, $\beta_1, \ldots \beta_l$ список описателей, ПР1 признак типа параметра, ПР2 признак размерности, A характеристика (атрибут и размерность) параметра, $\gamma_1, \ldots \gamma_m$ список параметров, от которых зависит размерность параметра.

Значения параметров, задаваемые пользователем, хранятся в таблице G, которая представляет собой последовательность упорядоченных записей. При этом значение одного параметра может занимать несколько записей. Начало и конец списка значений параметра указаны в таблице D, каждая запись которой имеет следующий вид: $D = \{N, \text{НАЧ}, \text{КОН}, \text{ПЗ}, \text{ППР}\}$, где N системный номер параметра, НАЧ—номер записи начала списка значений, КОН—номер записи конца списка значений, ПЗ—признак значения параметра (числовые или выражения), ППР—признак, характеризующий зависимость размерности некоторых параметров от данного.

ППР используется для генерации соответствующего препроцессорного утверждения (3).

С каждым модулем связан некоторый предикат, ограничивающий значения входных параметров, синтаксис которого должен удовлетворять принятым в языке ПЛ/1 соглашениям.

Алгоритмы решений задачи хранятся в таблице *TREE*, запись которой имеет следующий вид: $\{\varphi_1^l, \varphi_2^l, \ldots \varphi_p^l\}$, и указанная последовательность модулей является расчетной цепочкой задачи (1). Две расчетные цепочки модулей из таблицы *TREE*, $\{\varphi_1^i, \ldots \varphi_n^i\}$, $\{\varphi_1^l, \ldots, \varphi_m^l\}$ назовем совпадающими порядка K $1 \leqslant K \leqslant \min(n, m)$, если $\varphi_1^i = \varphi_1^l, \ldots$ $\ldots, \varphi_k^i = \varphi_k^l$.

2. Описание принципа работы ГПП в однопрограммном режиме.

ГПП можно условно представить в виде следующих программных блоков: ГПП = {ОРТ, *GEN*}.

В однопрограммном режиме генерируются программы $D1$, $D2$, где $D1$ является главной процедурой, осуществляющей последовательный вызов модулей из расчетной цепочки.

На первоначальном этапе программный блок ОРТ, используя

признаки ПР1, ПР2 и список $\gamma_1, \ldots \gamma_m$, из паспортов параметров определяет ПЗ и ППР для каждой записи таблицы. По системному номеру параметра из D, ОРТ находит соответствующую ему характеристику из паспорта параметра. Найденная характеристика параметра генерируется в зависимости от значения признака ППР. Для генерации характеристики и начальных присвоений параметра, от значения которого зависят размерности других параметров, используется препроцессорное средство языка ПЛ/1. Динамический вызов компилятора ПЛ/1 формирует главную процедуру $D1$ в виде объектного модуля.

Процедура $D2$ генерируется с помощью программного блока *GEN*.

Опишем этапы его работы:

1) на основе расчетной цепочки модулей $\varphi_1^l, \ldots \varphi_k^l$ генерируются характеристики всех параметров множества $\bigcup [V(\varphi_j^l) \bigcup U(\varphi_j^l)]$;

2) для каждого модуля $\varphi_i^l \in \{\varphi_1^l, \ldots \varphi_k^l\}$ осуществляется генерация оператора проверки предиката и оператор вызова модуля φ_t^l.

Сконструированная программа проходит стадию трансляции. С помощью редактора связей объектные модули $D1$ и $D2$ совместно редактируются; тем самым формируется некоторая программа P в виде загрузочного модуля.

В ходе работы программы P данные пользователем параметры активизируются в программе и передаются процедуре в виде фактических параметров. В программе проверяется истинность каждого предиката модулей. В случае истинности предиката вызывается соответствующий модуль. Если предикат какого-то модуля ложный, то ГПП в диалоговом режиме сообщает пользователю об ошибке, предоставляя возможность модификации входных параметров. Получая новые значения параметров, ГПП для формирования загрузочного модуля P генерирует и транслирует только новую процедуру $D1$ с учетом внесенных пользователем изменений.

При истинности всех предикатов программа P завершает работу, выдавая пользователю ответ решения задачи.

В случае невозможности модификации значений входных параметров, ГПП автоматически приступает к генерации программы, выбирая новую расчетную цепочку из таблицы *TREE*, если, конечно, таковая существует. Вновь выбранная расчетная цепочка может быть совпадающей порядка k с ранее рассмотренной. Естественно возможно выполнимыми будут только те цепочки, для которых $k < l$. В повторно сгенерированной программе для указанных расчетных цепочек могут встретиться модули, отработанные в предыдущей сгенерированной программе. При повторном выполнении таких модулей произойдет потеря машинного времени. Этот недостаток устраняется предусмотренным в ГПП мультизадачном режиме.

Мультизадачный режим

Основной информацией для ГПП в мультизадачном режиме (ГППМ) является выдаваемое планировщиком дерево алгоритмов. Каждая ветвь этого дерева и каждая запись таблицы *TREE* находятся в взаимооднозначном соответствии.

Для описания алгоритма работы ГППМ дадим некоторые определения.

Назовем весом дерева количество входящих в него модулей.

Определим операцию разложения по ветви в дереве, как удаление всех ребер из данной ветви. Результатом этой операции является множество поддеревьев.

Осуществим операцию разложения по ветви $\varphi_1^i, \ldots \varphi_k^i$ в дереве решений D.

Поддерево, включающее вершину φ_j^i, назовем собственным поддеревом вершины φ_j^i данной ветви.

Опишем алгоритм работы ГППМ.

1. Включаем в множество M дерево решений D, $M=\{D\}$.
2. Выбираем из множества M все поддеревья, вес которых больше нуля.

   Если таковых нет, переход на пункт 8.
3. Рассмотрим множество L „левых ветвей".
4. Осуществим операцию разложения для всех ветвей множества.

   Каждому поддереву сопоставим уникальное имя.
5. Осуществляется генерация программы для каждой ветви множества L по алгоритму однопрограммного режима, причем:

   а) каждая сгенерированная программа получает имя поддерева, „левую ветвь" которого она реализует;

   б) каждая сгенерированная программа объявляется подзадачей;

   в) после генерации оператора вызова каждого модуля генерируются операторы вызова подзадач, соответствующих его собственным поддеревьям;

   г) предпринимаются меры по синхронизации выполнения программ путем добавления в соответствующих точках операторов WAIT, EXIT.
6. Образуем множество M из поддеревьев, вес которых больше или равно 1.
7. Переход на п. 2.
8. Конец.

Каждая программа (задача) является старшей относительно подзадач, порожденных ею.

Операторы WAIT, EXIT в пункте 5 предусмотрны таким образом, что

1° старшая задача попадает в состояние ожидания при невыполнении некоторого предиката и ждет завершения работы порожденных ею подзадач;

2° в случае истинности всех его предикатов удаляются все порожденные его подзадачи. Может случится, что все подзадачи оказались невыполненными. В этом случае ГППМ в диалоговом режиме сообщает пользователю о неверном задании входных параметров. После введения новых значений весь описанный процесс повторяется до полного успешного завершения.

Преимущество мультизадачного режима состоит в том, что при большом параллелизме операций в программе мультизадачный режим может оказаться более выгодным по времени выполнения за счет одновременного использования процессора и каналов, несмотря на то, что мультизадачность использует больше памяти и больше процессорного времени для каждой задачи в отдельности.

Ա. Թ. ՀԱՐՈՒԹՅՈՒՆՅԱՆ, Ա. Ա. ԶԱՔԱՐՅԱՆ, Է. Հ. ՂԱԶԱՐՅԱՆ

ԱՆԻ—81 ՀԱՄԱԿԱՐԳԻ ԾՐԱԳՐԱՅԻՆ ԳԵՆԵՐԱՏՈՐ

Ա մ փ ո փ ո ւ մ

Հոդվածում դիտարկված են ծրագրային գեներատորի կառուցման սկզբունքները՝ իրացված ԱՆԻ—81 ծրագրային համակարգում։

Ծրագրային գեներատորը ԱՆԻ—81 բազային երկխոսական համակարգի հիմնական ենթահամակարգերից մեկն է։ Այն, բացահայտված հաշվարկային սխեմաների հիման վրա, ձևակերպում է բարձր մակարդակի լեզվով ծրագրի տեքստը։

Նկարագրված ծրագրային գեներատորը աշխատում է միածրագրային և բազմախնդրային ռեժիմներով։

ЛИТЕРАТУРА

1. *Э. У. Казарян, С. С. Погосян.* Планирование вычислительных схем с учетом предикатных условий.—В настоящем сборнике.
2. *Ф. Харари.* Теория графов. М., Мир, 1973.
3. *Л. К. Гребенников, В. Н. Лебедев.* Решение задач на ПЛ/1 в ОС ЕС. М., Финансы и статистика, 1981.