А. А. АБРАМЯН, М. А. УЗУНОВ

ОРГАНИЗАЦИЯ ВЫЧИСЛЕНИЙ В СИСТЕМЕ АНИ-79

Организация вычислений по сформулированному алгоритму [3], [4] осуществляется следующим образом: модули цепочки должны последовательно выполняться с передачей данных между собой. Однако в конкретной системе программирования возникают практические сложности реализации. Ниже будет описан метод разрешения их в ДОС ЕС.

В рассматриваемой системе АНИ-79 [1] процесс планирования алгоритма отделен от процесса вычислений по сформированному алгоритму. Это дает возможность рассчитывать результаты при различных наборах исходных данных без повторного планирования. Согласно такому разделению в системе АНИ-79, наряду с языком описания запросов, создан язык описания задач. На этом языке пользователь задает исходные данные, т. е. значения параметров, которые были указаны в предложении ДАНО запроса на планирование алгоритма, и дополнительные требования к решению задачи.

На рис. 1 приведен пример задачи на языке описания задач, который охватывает все возможности языка:

Первое предложение (до символа*) определяет исходные значения параметров задачи.

В первой строке задано значение скалярного параметра, имя которого МАССА ТЕЛА.

Во второй строке заданы значения векторного параметра (в данном случае размерности 2) и использовано не имя параметра, а идентификатор, который должен быть присвоен данному параметру в описании запроса на планирование.

```
>данные :  'масса тела'=50.
           Я = 10, 20,
           'тензор напряжений'(2,3,31,22,23,31,33,34/3,3);
           'время'(1, 100)2 *
>если :    0<'ускорение'<20 *
>пока :    0<'давление'<100 *
>условие : 'скорость'=max *
>конец : *
```

Рис. 1

Далее приведен пример задания значения матричного параметра ТЕНЗОР НАПРЯЖЕНИЙ размерности 3×3.

Наконец, в четвертой строке задается скалярный параметр со множеством исходных значений (циклический параметр). В данном случае параметр ВРЕМЯ изменяется от 1 до 100 с шагом 2. Циклический параметр можно задавать также непосредственным перечислением его значений.

Далее в описании задачи следуют дополнительные требования.

В пятой строке—предложение ЕСЛИ—задан диапазон значений для вычисляемого параметра. При нарушении условия типа ЕСЛИ вычисления по алгоритму прекращаются, либо, в случае цикла, возобновляются с очередным значением циклически меняющегося параметра.

Предложение ПОКА аналогично предложению ЕСЛИ, однако вычисления прекращаются и при циклических задачах. Такого рода ограничения накладываются обычно на монотонно изменяющийся параметр.

В предложениях ПОКА и ЕСЛИ могут быть наложены ограничения на несколько различных параметров. Между этими ограничениями в каждом из предложений предполагается логическая конъюнкция. С другой стороны, для одного и того же параметра можно указать несколько непересекающихся интервалов. Очевидно, что в этом случае условие считается выполненным, если значение параметра принадлежит одному из указанных интервалов.

Предложение УСЛОВИЕ (строка 7) определяет требование системе произвести полный расчет, доставляющий экстремальное значение указанному в предложении параметру (в данном случае—максимизируется параметр СКОРОСТЬ). Для остальных значений параметра цикла выполняется лишь часть алгоритма, необходимая для нахождения экстремума [4]. Разумеется, предложения ПОКА и УСЛОВИЕ имеют смысл лишь при циклических задачах.

Транслятор с ЯОЗ на основании описания задачи формирует таблицы.

Исходные значения параметров (в случае циклических задач—для очередного шага цикла) вносятся в рабочее поле таблицы ТДРЗ (строго говоря, в таблице два рабочих поля с идентичным распределением памяти между параметрами. Второе используется при решении задач с экстремизацией параметра—см. блок-схему). Затем управление передается собственно интерпретатору—программе, организующей выполнение цепочки модулей.

Поскольку модули пользователя — суть подпрограммы типа SUBROUTINE, то поочередная загрузка модулей и их выполнение согласно сформулированному алгоритму вызывает определенные сложности в ДОС ЕС. Поэтому, например, в системе, описанной в [2], был использован следующий метод.

Все имена модулей, используемых системой, и их списки параметров считались заранее известными—до ввода системы в эксплуатацию.

Роль интерпретатора выполняла программа системы, структура которой показана на рис. 2.

Такая программа должна быть написана при вводе системы в эксплуатацию. Затем производится редактирование системы со всеми модулями пользователя, т. е. создается многофазная программа, где каждый модуль выделен в отдельную фазу.

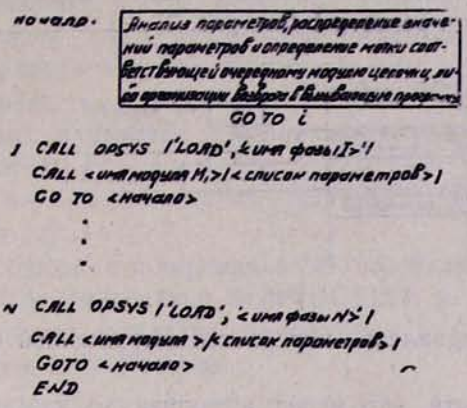

Рис. 2

Большой недостаток такого подхода в том, что при внесении нового модуля в систему требуется изменение (хотя и незначительное) программного обеспечения самой системы с последующей трансляцией и редактированием.

Кроме того, при большом количестве модулей пользователя программа интерпретатора будет занимать много места в памяти, что сократит объем памяти, выделяемой для проблемных модулей.

Другой метод состоит в том, чтобы интерпретатор формировал программу согласно цепочке модулей и выводил текст на ВЗУ—ленту или диск. После этого полученная программа обычным образом транслируется, редактируется и выполняется.

Подобный способ был реализован в первоначальном варианте системы.

Программа на языке ФОРТРАН состояла из последовательности операторов CALL<имя модуля>/<список параметров>/ и заканчивался оператором CALL RASP/<список параметров>/—вызывающим системную подпрограмму печати результатов. При этом внесение нового модуля в систему не сопровождалось изменением самой системы, однако осложнялась проверка значений параметров между выполнением модулей и, кроме того, размеры фазы, полученной после трансляции и редактирования исходного модуля, могли превышать объем доступной памяти. Это требовало дополнительных ухищрений при редактировании, что в целом делало метод неприемлемым.

Из сказанного ясно следует, что наиболее желательным методом выполнения цепочки является поочередная загрузка фаз модулей самой системой по именам, заданным в таблице ТИМ[3].

Средства ДОС ЕС в принципе позволяют осуществить такую загрузку—для этого служит макрокоманда АССЕМБЛЕРа LOAD.

При трансляции модулей пользователя в ДОС предполагается, что список адресов параметров будет передан подпрограмме в 1 регистре, а адрес возврата—в 14. Таким образом, перед вызовом модуля производится заполнение массива ADDR адресами значений параметров—эта информация берется из таблицы ТДПЗ, и затем адрес массива ADDR загружается в регистр 1 (см. рис. 3).

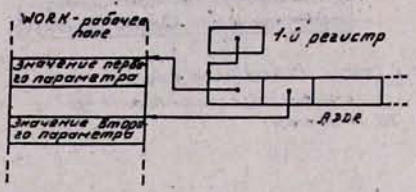

Рис. 3.

После этого управление передается загруженной фазе для выполнения модуля.

Следует отметить, что перед выполнением каждого модуля осуществляется как проверка условий, заданных в задаче, так и проверка допустимости работы модуля. Последняя включает в себя анализ диапазонов модуля и выполнение предмодуля.

В итоге получается следующая блок-схема (рис. 4).

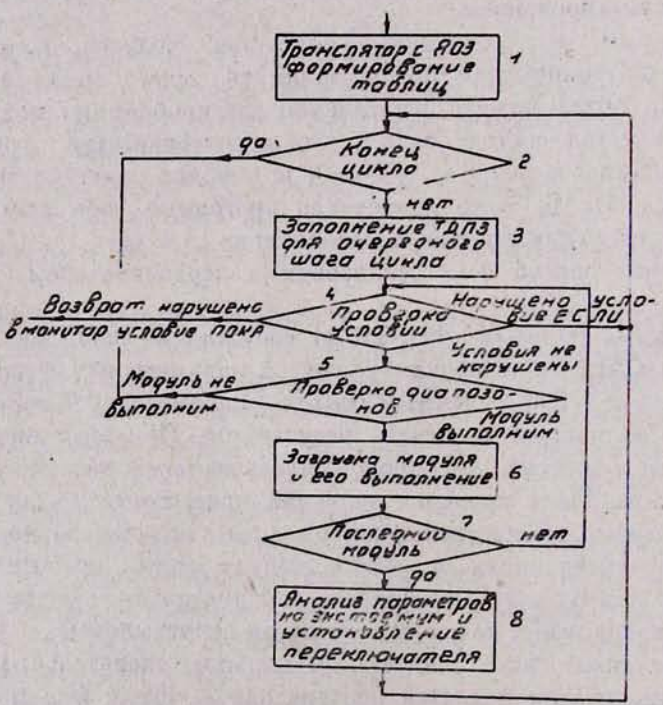

Рис. 4.

Здесь показано выполнение той части алгоритма, на которую влияет изменяемый в цикле параметр.

Первая часть алгоритма [4] выполняется по той же блок-схеме, но всего один раз до начала цикла.

8-й блок служит для выбора одного из двух рабочих полей—того, в котором значение максимизируемого (минимизируемого) параметра больше (меньше).

Для следующего шага цикла освобождается то рабочее поле, где хранится «худший» вариант. Таким образом, наряду со значением бо́льшего (меньшего) на данный момент значения максимизируемого (минимизируемого) параметра запоминаются также и значения остальных, вычисленных при соответствующем шаге цикла, параметров (в случае задач без цикла соответствующие участки блок схемы, очевидно, пропускаются).

Рассмотрим теперь формирование из подпрограмм пользователя отдельных фаз. При трансляции SUBROUTINE в объектном модуле, как правило, строятся команды обращения к стандартным подпрограммам ввода-вывода ФОРТРАНа:

ILFIBCOM, ILFFIOCS, ILFDIOCS

ILFADCON, ILFFINT, ILFUNTAB (1)

Кроме того, в программах для технических расчетов обычно используются такие подпрограммы, как ILFSQRT, ILFSSCN, ILFIXPI и т. д.

Набор последних, чаще всего используемых в модулях, определяется конкретной областью применения системы. Будем называть его набором вспомогательных подпрограмм. Разумеется, в нем могут быть не только стандартные подпрограммы ФОРТРАНа; но и часто используемые подпрограммы пользователя—например, модуль для численного интегрирования.

Все эти секции присоединяются редактором к модулю во время редактирования. Это значит, что в библиотеке абсолютных модулей будет храниться громадное количество дублируемой информации—только модули (1) занимают около четырех дорожек устройства 5052(!) Естественно, что такое положение неприемлемо.

Кроме того, хотя транслятор с ФОРТРАНа строит в объектных, модулях обращение к программам ввода—вывода, при трансляции SUBROUTINE обращение для открытия файлов (модули ILFFIOCS и ILFDIOCS) не формируются. Стало быть, при любой попытке произвести операцию ввода-вывода в модулях пользователя (может быть, неявно) будет возникать программное прерывание, что, конечно, накладывает на модули существенные ограничения.

Чтобы избежать перечисленных недостатков предлагается несколько необычный способ редактирования фаз пользователя. Суть его заключается в том, что часть программного обеспечения самой

системы—модули (1)—является одновременно частью модулей пользователя.

Такая организация фаз возможна, поскольку объектные модули (1) не содержат внешних ссылок, кроме как между собой.

Аналогично можно избежать включения набора вспомогательных подпрограмм в каждый модуль пользователя. Весь набор составляет лишь одну фазу А, секции которой могут использоваться всеми модулями.

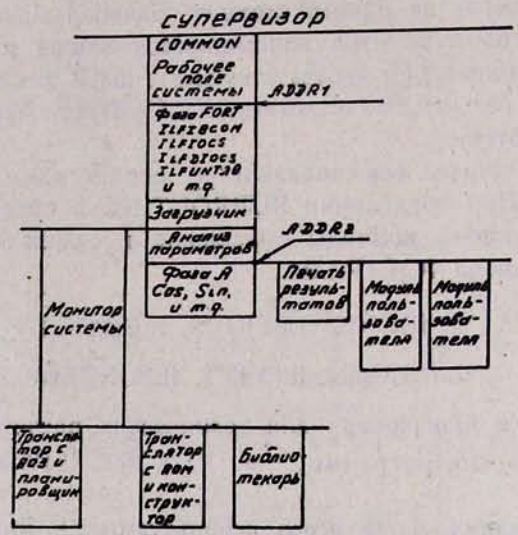

Рис. 5

Сама система таким образом представляет собой оверлейную структуру и модули пользователя являются отдельными ее фазами.

Для освобождения по возможности бо́льшего объема памяти для модулей пользователя предназначен загрузчик системы, который управляет загрузкой отдельных ее частей. В результате, при выполнении проблемных модулей в памяти находится только:

COMMON область системы—рабочее поле;

диапазоны параметров и значения условий;

фаза FORT—модули (1);

Загрузчик; Программа анализа параметров и фаза А (набор вспомогательных подпрограмм) (см. рис. 5).

Оверлейность системы, помимо сказанного, значительно уменьшает общий объем памяти, требуемый для работы системы.

На рис. 6 приведен пример внесения проблемного модуля в библиотеку.

Здесь адреса ADDR1 и ADDR2 сообщаются пользователю разработчиком системы.

В строке PHASE MOD1,*+2K смещение NK (может отсутство-

вать) служит для того, чтобы при необходимости можно было пополнить фазу A новыми модулями, не меняя при этом всей библиотеки модулей.

```
// JOB      пример
// OPTION   CATAL
// ASSGN    SYSCLB, X'Cnn'
   PHASE    FORT,  S+ADDR1
INCLUDE     ILFIBCON
INCLUDE     ILFFIOCS
INCLUDE     ILFDIOCS
INCLUDE     ILFFINI
INCLUDE     ILFADCON
INCLUDE     ILFUNTAB

   PHASE A,*+ADDR2
INCLUDE     ILFSQRT
INCLUDE     ILFFIXPI
            .
            .
            .
   PHASE  MOD1,*+2K
// EXEC   FFORTRAN
       SUBROUTINE MOD1(A,B,C)
            .
            .
            .
       END

/*
// EXEC  LNKEDT
// EXEC  MAINT
      CONDS CL

/*
/&
```

Рис. 6

Сжатие (CONDS CL) необходимо, так как при внесении каждого модуля в библиотеке уничтожаются две фазы—FORT и A, и заменяются тождественными (можно, конечно, производить сжатие по мере переполнения библиотеки).

Как видно из примера, управляющие операторы ДОС не меняются от модуля к модулю, и, стало быть, такой метод редактирования не вызовет затруднений для пользователей.

Ա. Ա. ԱԲՐԱՀԱՄՅԱՆ, Մ. Հ. ՈՒՋՈՒՆՈՎ

ՀԱՇՎԱՐԿՆԵՐԻ ԿԱԶՄԱԿԵՐՊՈՒՄԸ ԱՆԻ–79 ՀԱՄԱԿԱՐԳՈՒՄ

Հոդվածում դիտարկվում են մոդուլների պլանավորված շղթայով հաշվարկը կազմակերպելու մեթոդները և պարամետրերի արժեքների փոխան-

ցոմը մոդուլների միջև։ Նկարագրված է այդպիսի հաշվարկի կազմակերպման ծրագրային կենսագործումը ԷՀՄ ՄՀ գործողական համակարգերում։

ЛИТЕРАТУРА

1. *К. А. Абгарян.* Базовая система автоматизации научно-технических и опытно-конструкторских разработок. В настоящем сборнике.
2. *С. С. Гайсарян, А.А. Калентьев.* Планирование вычислений на обобщенных вычислительных моделях. Деп. рук. РЖ «Математика», 5В—642в, 5, 1977.
3. *А. А. Абрамян, К. А. Даниелян, Г. П. Казанчян, М. А. Узунов.* Общее описание программной реализации отдельных частей базовой системы АНИ-79. В настоящем сборнике.
4. *К. А. Даниелян, Г. П. Казанчян.* Автоматизация планирования алгоритма решения задачи. В настоящем сборнике.