СООБЩЕНИЯ
ОБЪЕДИНЕННОГО
ИНСТИТУТА
ЯДЕРНЫХ
ИССЛЕДОВАНИЙ
ДУБНА

11 - 9220

О Ен Ир, В.П.Шириков

ОБ ОПТИМИЗИРУЮЩЕМ ВАРИАНТЕ ОРТ-1 ОБРАБОТКИ ЦИКЛОВ ПРИ ТРАНСЛЯЦИИ С ЯЗЫКА ФОРТРАН НА ЭВМ БЭСМ-6

1975

11 - 9220

О Ен Ир, В.П.Шириков

ОБ ОПТИМИЗИРУЮЩЕМ ВАРИАНТЕ ОРТ-1 ОБРАБОТКИ ЦИКЛОВ ПРИ ТРАНСЛЯЦИИ С ЯЗЫКА ФОРТРАН НА ЭВМ БЭСМ-6

Цель создания варианта ОРТ-I

При трансляции DO -циклов программ, написанных на ФОРТРАНе, неоптимизирующий (стандартный) и оптимизирующий трансляторы ЭВМ БЭСМ-6 применяют разные способы реализации проверки конца повторений циклов в объектной программе /2,4,5/.

В неоптимизированной объектной программе при ее работе на каждом шаге цикла его управляющая переменная переопределяется и производится проверка, не превышает ли ее величина значение второго параметра (М2) DO - оператора.

Оптимизирующий транслятор (ОРТ-0)/2/ в случае, когда все параметры цикла являются константами, вычисляет число повторений цикла в процессе трансляции, и если оно не больше 32767, то в объектной программе такое число посылается в индекс-регистр и используется для проверки конца повторений цикла. Если число повторений цикла больше 32767 или по крайней мере один из параметров цикла является переменной, то организация проверки конца повторений цикла производится так же, как в стандартном варианте. В этом случае на каждом шаге переопределение значения управляющей переменной и проверка условия окончания цикла требуют выполнения 6 медленных операций (поскольку идет работа с числами в памяти, а не в регистре).

Потери времени здесь особенно обидны, если управляющая переменная внутри цикла не используется и ее переопределение не необходимо для выполнения внутренних операторов тела цикла.

Особого внимания заслуживают случаи, когда параметры цикла являются переменными. Хотя транслятор сам при своей работе не может в этой ситуации вычислять число повторений цикла, но рабочая программа может сделать это, послать вычисленное зна-

чение в индекс-регистр и после этого организовать цикл с использованием этого индекс-регистра. С точки зрения эффективности счета это является наилучшим решением, но , если его принять, то максимальное число повторений цикла ограничится на БЭСМ-6 величиной 32767 из-за малой разрядности индексных регистров.

Нужно сказать, что подобного рода ограничение существует в версиях входного языка ФОРТРАН многих машин, оно диктуется понятными соображениями и не выглядит чересчур сильным.

Обычно область представления чисел в индекс-регистре, используемом в арифметике вычисления адресных частей команд, определяется размером области математической памяти, выделяемой задачам .Это означает, что значения индексных функций, используемых при вычислении позиций переменных с индексами в массивах, как правило, не могут превышать максимальное число, представляемое в индекс-регистре.

Поскольку в большинстве случаев циклы в программах организуются для работы над элементами массивов и индексные переменные этих элементов явно или неявно ставятся в соответствие параметрам цикла, то в такой ситуации число повторений цикла оказывается допустимым для размещения в индекс-регистрах машины.

С учетом изложенного выше предложен данный вариант (ОРТ-I) оптимизирующего транслятора с ФОРТРАНа.

Если все параметры оператора DO являются константами, то трансляторы ОРТ-0 и ОРТ-I обрабатывают соответствующий цикл одинаковым образом.

Результаты работы вариантов ОРТ-0 и ОРТ-I отличаются в случае, когда удовлетворяются следующие условия:

(I) по крайней мере один из параметров цикла М1, М2 является переменной и параметр М3, как константа, равняется I или 2;

(2) внутри цикла нет необходимости переопределения управляющей переменной (см.ниже условия (I)-(4) раздела IУ, пункт 2);

(3) имеется резервный индекс-регистр, в который возможно заслать число повторений цикла (точнее, - (число повторений цикла - I), т.е. - $[(M2-M1)/M3]$, но для простоты будем называть это "число повторений цикла").

При удовлетворении условий (I)-(3) транслятор ОРТ-I генерирует команды, которые вычисляют число повторений цикла и посылают его в индекс-регистр. Кроме того, для организации повторения цикла ОРТ-I составляет специальную команду с индекс-регистром, предназначенную для повторения или завершения цикла.

Выбор условий (I)-(3) продиктован следующими соображениями (по-прежнему речь идет о случаях, когда не все параметры цикла являются константами). В рабочей программе, полученной через вариант транслятора ОРТ-I, по сравнению с рабочей программой, полученной через вариант ОРТ-О или стандартный вариант, в начальную часть программы цикла добавляется 3 команды (для вычисления числа повторений цикла и засылки его в индекс-регистр), если M3=4, или 4 команды, если M3=2; кроме того, в завершающей части первой из них отсутствуют 4 команды для переопределения управляющей переменной. В случае, когда параметр M3 больше 3 или является переменной, организация цикла с помощью индекс-регистра не эффективна, так как требуется I8 команд для вычисления выражения $[(M2-M1)/M3]$.

АЛГОРИТМ

В настоящей работе описывается процесс оптимизации самых внутренних DO -циклов. Эта оптимизация осуществляется рациональным определением значений индексных функций и эффективным использованием индекс-регистров машины.

Для оптимизации возможно использовать индексные регистры с номерами с 1-го по 6-й, так как регистр 7 предназначен для размещения базового адреса программ, регистры 8-12 используются как локальные и могут портиться при вызове подпрограмм, регистр 13 - для запоминания адреса возврата при переходах на подпрограммы, 14 - для запоминания адресов фактических параметров /3,5/.

Поэтому оптимизация DO -цикла проводится, когда число различных локальных и полулокальных индексных функций не больше 6, если нет глобальных (независимых от переменных цикла) индексных функций, или не больше 5, если они есть.

Если нет необходимости переопределения значения управляющей переменной цикла в шаге цикла и имеется резервный индекс-регистр, то проверка на конец цикла строится с использованием такого регистра.

Этапы процесса оптимизации

I. Проверка условий оптимизации и ее подготовка (делается почти так же, как в варианте OPT-0, см./2/).

II. Генерирование кодов начальной части цикла.

1 . Составляются команды засылки начального значения переменной цикла и обращения к части вычисления индексных функций, индексная переменная которых является переменной цикла (,XTA,M1;,ATX,1; ,NTR ,3; I3, VJM , * U<N>).

2 . В случае, когда в индекс-регистре возможно сохранять число повторений цикла ((RVFLAG)=0) и $\bigwedge_{j=1}^{3}$ (M_j = "константа") удовлетворяется, то

1) вычисляется значение N.C. = [(M2-M1)/M3] (при трансляции);

2) если N.C.≤32767, то ставится следующий флаг:

$$(RCFLAG)=\begin{cases} 1 : \text{нет необходимости переопределения значения переменной цикла по каждому переходу цикла } ((ALPHA) \neq 0) \\ 3 : \text{в противном случае } ((ALPHA) = 0) ; \end{cases}$$

3) если N.C.>32767, то выставляется флаг (RCFLAG) = 2 .

3. В случае, когда удовлетворяется условие

$(RVFLAG)\neq 0 \vee (RVFLAG)=0 \wedge \bigwedge_{j=1}^{3} (M_j \neq$ "константа"), работа следующая.

I) Если $\bigvee_{j=1}^{2}$ (M_j = "переменная"), то генерируются команды, проверяющие условие M2 < I (,X-A, M2;,NTR , I8; ,U1A , #T < DN >).

(I) Если удовлетворяется условие

(RVFLAG)=0∧(ALPHA)≠0∧(M3=1∨2) , то генерируется команда установки режима для операций с целыми числами (, NTR , 3). Затем генерируются команды, вычисляющие число - N.C. и посылающие его в индекс-регистр (если M3=I, то,X-A,= 640-0 ; ,ATI , N или если M3=2, то , ASN, 64+I ;,X-A,=0-0;,ATI,N ; здесь N -номер индекс-регистра,вычисленный при трансляции с учетом числа различных индексных функций внутри DO -цикла; число -N.C. засылается в этот регистр при выполнении рабочей программы). После генерирования команд ставится флаг (RCFLAG)=5.

(2) Если (RVFLAG)≠0 ∨ (ALPHA) = 0 ∨ (M3 ≥ 3 ∨ M3 = "переменная") удовлетворяется, то ставится флаг (RCFLAG)=6.

2) Если $\bigwedge_{j=1}^{2}$ (M_j ="константа"), то ставится флаг (RCFLAG)=2.

4. Составляются команды, засылающие начальные значения индексных функций в индекс-регистры (, #TC , #I<N>;IR, VTM,) :

I) заводятся команды посылки начальных значений локальных и полулокальных индексных функций в индекс-регистры;

2) если имеется резерв индекс-регистров, то заводятся команды посылки значений инвариантных глобальных индексных функций в индекс-регистры.

5. Когда $\bigwedge_{j=1}^{3}$ (M_j = "константа") истина, генерируется команда, которая посылает значение – N.C. , определенное при трансляции, в индекс-регистр (N , VTM , – N.C.).

Таким образом, содержание флага RCFLAG имеет следующий смысл:

$$(RCFLAG) = \begin{cases} 1 : -N.C. \in IR \\ 2 : I=I+M3 \wedge M2-I \\ 3 : -N.C. \in IR \wedge I=I+M3 \\ 5 : \blacksquare T<DN> \wedge -N.C. \in IR \\ 6 : \blacksquare T<DN> \wedge I+M3 \wedge M2-I \end{cases},$$

где IR – индекс-регистр; I=I+M3 означает, что необходимо переопределение переменной цикла; M2-I – что производится проверка необходимости повторений цикла, используя параметр M2 и переменную цикла I; ■T<DN> означает, что заводится команда, на которую передается управление, если MI > M2.

6. Организуется генерация команд вычисления приращений индексных функций.

Для этого с использованием таблицы индексных функций и характеристической таблицы локальных и полулокальных индексных функций составляется таблица приращений (локальных и полулокальных индексных функций [2]).

В обозначениях таблицы приращений получается следующая формула вычисления коэффициентов шага переменной цикла в приведенных индексных выражениях:

$$CV=\begin{cases}(ID_N) & : \quad Q=0\ ,\\ (ID_N)+(*M<N>) & : \quad Q=1\ ,\\ (ID_N)+(*K<N>) & : \quad Q=2\ ,\\ (ID_N)+(*K<N>)+(*M<N>) & : \quad Q=3\ ,\end{cases} \qquad (1)$$

(см. /2/).

Генерируются команды, реализующие формулу (I). При этом все постоянные составляющие формулы вычисляются в процессе трансляции.

В том случае (такой случай называется условием (5)), когда условия

$$Q = 0 \wedge M3 = \text{"константа"}\ , \qquad (2)$$

$$Q = 0 \wedge M3 = \text{"переменная"} \wedge (ID_N)=1\ , \qquad (3)$$

$$Q \neq 3 \wedge M3 = 1 \wedge (ID_N) = 0\ . \qquad (4)$$

не удовлетворяются, генерируется последовательность, состоящая из команд вычисления коэффициента, команды умножения M3 на результат их действий и команды посылки окончательного результата в соответствующий элемент массива *INCNT(6) приращений для шести индексных функций. Строки в таблице приращений, соответствующие индексным функциям, для которых сгенерированы команды вычисления приращений, стираются. В начале генерации команд вычислений приращений, если условие (RCFLAG) = 6 ∧ Q ≠0 удовлетворяется, то заносится команда режима сложения целых операндов (, NTR , 3) и устанавливается флаг (FLAGNTR) =1 ; в противном случае - (FLAGNTR) = 0 без генерирования команды режима.

7. Перед входом в тело цикла генерируется команда установки стандартного режима (, NTR , I8), причем не во всех случаях, а только в случае, когда условие (RCFLAG)=6 ∧ (FLAGNTR)=I удовлет-

воряется, для того, чтобы избежать ее повторного генерирования.

Ш. Генерирование команд, составляющих тело цикла

Заводится код метки начала тела DO -цикла для передачи управления на повторение выполнения цикла (*W<DN> :, BSS ,).

Эффективно то, что при определении позиций переменных из массивов используются индекс-регистры.

В приведенном индексном выражении может быть постоянная часть и переменная часть (индексная функция). Значение постоянной части вычисляется в процессе трансляции и входит вместе с номером индексной функции в PI-строку, содержащую адрес имени массива /5/.

Значение индексной функции (переменной части) определяется при выполнении рабочей программы. Поэтому и посылка значения индексной функции в индекс-регистр также должна производится при счете рабочей программы.

Адрес переменной с индексами представляет собой сумму адреса начала массива, значения постоянной части и содержимого индекс-регистра, имеющего значение индексной функции.

В стандартном (неоптимизирующем) режиме в случае, когда блок PUTAWAY , который генерирует код OUTLIST , встретится с операндом-переменной с индексами, то для определения адреса операнда этот блок просматривает маску PBMASK /4,5/. Если в PBMASKe нет номера индексной функции для обрабатываемого операнда, ее номер засылается в PBMASK на свободные места (один номер занимает 8 разрядов). После этого генерируется команда посылки значения индексной функции в индекс-регистр, который имеет номер, соответствующий порядковому номеру индексной функции в PBMASKe . Этот регистр затем используется для определения позиции элемента массива. Если же номер индексной функции операнда имеется в PBMASKe , то генерации коман-

ды посылки значения индексной функции в индекс-регистр не делается, а при формировании обращения к операнду используется соответствующий индекс-регистр.

В режиме оптимизации маска PBMASK формируется для всех индексных функций тела цикла (если хватает индексных регистров) еще в процессе составления команд начальной части цикла. Поэтому в теле цикла не будет генерирования посылки значений индексных функций в индекс-регистры. Если при проведении оптимизации имеется достаточное количество регистров для локальных и полулокальных функций, но остающегося резерва недостаточно для всех глобальных функций, то генерируются засылки в резервные регистры для части таких глобальных функций.

IУ. Генерирование кодов последней (завершающей) части цикла

I. Производится переопределение значений (локальных и полулокальных) индексных функций.

Если условие Q = 0 ∨ МЗ="константа" удовлетворяется, то в процессе трансляции вычисляется выражение (ID_N) * МЗ с использованием таблицы приращений.

Составляются команды, которые добавляют значения приращений индексных функций к содержимому соответствующих индекс-регистров.

Тип команд следующий:

(1) IR, UTM, ID_N*M : Q = 0∧МЗ = "константа"

(2) , WTC, *M<N>
IR, UTM, : Q = 1∧МЗ = 1∧(ID_N) = 0

(3) , WTC, *K<N>
IR, UTM, : Q = 2∧МЗ = 1∧(ID_N) = 0

(4) , WTC, M3 : Q = 0∧M3 = "переменная"∧(ID_N)=1
IR, UTM,

(5) , WTC,=INCNT+IR-1 : в остальных случаях
IR, UTM,

где IR - номер индексного регистра.

2. Производится переопределение значения переменной цикла.

В процессе выполнения цикла управляющая переменная (I) обязательно должна быть переопределена в следующих случаях:

(I)I используется в качестве операндов или фактических параметров (в теле DO -цикла),

(2) I - переменная типа COMMON ,

(3) I - формальный параметр,

(4) I - присутствует в операторе EQUIVALENCE ,

(5) невозможно сохранение числа повторений цикла в индексрегистре (это бывает в тех случаях, когда не хватает индексных регистров или выполнено условие M3 ⩾ 3 ∨ M3 = "переменная").

3. Генерируются команды проверки конца повторений цикла.

Эти команды строятся в соответствии с содержимым шкалы RCFLAG (т.е. генерируются следующие команды:

```
,X-A,M2  ; ,UZA,=W<DN>   : ((RCFLAG)∧1) = 0
           N,VLM,=W<DN>  : ((RCFLAG)∧2) = 0
,NTR,18  ;N,VLM,=W<DN>   : ((RCFLAG)∧3) = 3   ).
```

Затем в случае, когда ((RCFLAG)∧4) ≠ 0, генерируется строка, на которую передается управление, если MI > M2 (=T< DN >:, BSS ,).

Наконец, в случае необходимости переопределяется счетчик использованных в программе индексных регистров для того, чтобы допускать

оптимизацию и в случае, когда имеются вызовы подпрограмм (или функций) в теле DO-цикла.

Примечание. Транслятор ØPT-I на машине БЭСМ-6 в ОИЯИ временно записан на специальный дисковый файл, и для его вызова в колоду карт необходимо вставить следующие управляющие карты (перед текстом фортранной программы):

```
*NAME ...
  :
*DISC:3/SYSTEM,ØNIR
*FILE:FTN,87
*CØMMØN_LIBRARY
*TEMPØ
*CALL _ /ØPT
  [TEKCT]
  :

*END_FILE
```

В том случае, если карта *CALL /ØPT отсутствует или ставится карта *CALL /NØØPT , описанный транслятор OPT-I будет работать как неоптимизирующий.

Приложение

Рассмотрим программы умножения матриц в качестве типичного примера, связанного с применением циклов.

Прежде всего перечислим способы программирования умножения матриц.

I) Простой способ α-прямой метод (через обращение к подпрограмме): имена массивов, размерности массивов и порядок

зацаются через формальные параметры; обращения к массивам записываются программистом как обращения к двухмерным массивам.

Пример:

```
      SUBRØUTINE MUM1(A,B,C,L,N)
      DIMENSIØN A(L,L),B(L,L),C(L,L)
      DØ 10 I=1,N
      DØ 10 J=1,N
      C(I,J)=0.
      DØ 10 K=1,N
   10 C(I,J)=C(I,J)+A(I,K)*B(K,J)
      RETURN
      END
```

2) "Явный" способ – метод, при котором сам программист преобразует двухмерные массивы в одномерные. При этом имена массивов, размерность массивов и порядок – формальные параметры (используется обращение к подпрограмме умножения).

Пример:

```
      SUBRØUTINE MUM 2 (A,B,C,L,N)
      DIMENSIØN A(L,L),B(L,L),C(L,L)
      DØ 10 I=1,N
      JJ =1 $ IJ=I
      DØ 10 J=1,N
      IK= I $ KJ=JJ
      D=0
      DØ 5 K=1,N
      D=D+AC(IK)*B(KJ)
```

```
      IK=IK+L
    5 KJ=KJ+1
      C(IJ)=0
      IJ=IJ+L
   10 JJ=JJ+L
      RETURN
      END
```

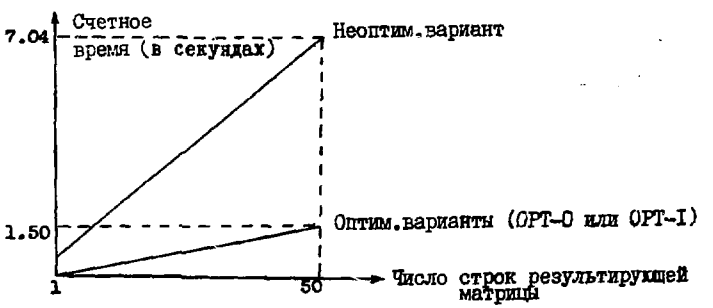
3) "Неявный" способ – метод, эквивалентный явному, но массивы, первая размерность (L) и порядок (N) матрицы – в операторе COMMON головной программы и подпрограммы умножения. Поэтому эта подпрограмма не имеет формальных параметров.

4) Простой способ β – метод, эквивалентный простому способу α, но имена, размерность и порядок матрицы – не формальные параметры и не COMMON (специальной подпрограммы умножения нет).

В примере I программы для случая 4 параметр N самого внешнего цикла умножения меняется от I до 50 с шагом I, а параметры N внутренних циклов равняются 50.

Пример I.

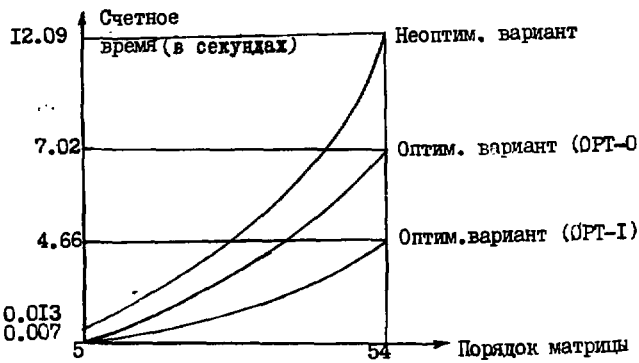
Для этого использовался простой способ β. Ниже показан график распределения счетного времени в моменты завершения вычисления очередной строки результирующей матрицы



Число строк	I	5	I0	I5	20	25	30	35	40	45	50
Неоптим. вариант	0.I4	0.7I	I.I4	2.II	2.82	3.52	4.23	4.93	5.63	6.34	7.04
Оптим. варианты	0.03	0.I5	0.29	0.45	0.60	0.75	0.89	I.05	I.I9	I.35	I.50

Пример 2.

Использовался простой способ α. Здесь показывается график счетного времени и таблицы для трех вариантов транслятора в случае, когда порядок матрицы меняется от 5 до 54.



Порядок матрицы	5	I0	I5	20	25	30	35	40	45	50	54
Неопт.	0.0I3	0.09	0.28	0.6I	I.I9	2.II	3.34	4.80	7.02	9.55	I2.09
OPT-0	0.007	0.05	0.I7	0.39	0.73	I.26	I.97	2.89	4.I0	5.59	7.02
OPT-I	0.007	0.04	0.I2	0.27	0.50	0.88	I.38	I.99	2.73	3.69	4.66

Пример 3.

Здесь использовались для сравнения простой способ α, явный и неявный способы тогда, когда порядок матрицы равен 50 (соответствующие программы пропускались через варианты транслятора).

способ / вариант	простой α	явный	неявный
неопт.	9.553	6.773	4.840
ЭРТ-0	5.593	5.907	4.073
ЭРТ-I	3.693	4.453	3.080

В заключение авторы выражают сердечную благодарность Н.Н.Говоруну, В.Г.Иванову, Г.А.Ососкову за поддержку, И.Н.Силину, Г.Л.Мазному, Н.С.Заикину за помощь и замечания.

ЛИТЕРАТУРА

1. Язык ФОРТРАН (Под редакцией В.П.Ширикова), ОИЯИ, II-4818, Дубна, 1969.
2. О Ен Ир, В.П.Шириков. Об одном оптимизирующем варианте обработки циклов при трансляции с языка ФОРТРАН на ЭВМ БЭСМ-6. ОИЯИ, II-9219, Дубна, 1975.
3. О Ен Ир, В.П.Шириков. Новые возможности использования логических выражений при программировании на языке ФОРТРАН для ЭВМ БЭСМ-6, ОИЯИ, II-7396, Дубна, 1973.
4. З.Бродцински и др. Транслятор с языка ФОРТРАН для системы математического обеспечения БЭСМ-6. Труды Первой всесоюзной конференции по программированию, г.Киев, 1968. Изд. ИК АН УССР.
5. З.Бродцински, Н.Н.Говорун и др. Система математического обеспечения ЭВМ БЭСМ-6. Транслятор с языка ФОРТРАН (части I-IУ), ОИЯИ, БI-II-7I60 и БI-II-7I62, Дубна, 1972.

Рукопись поступила в издательский отдел
9 октября 1975 г.