ВЫЧИСЛИТЕЛЬНЫЕ СИСТЕМЫ
1968 г. Выпуск 30

УДК.681.142.2.

ПРИМЕРЫ ИСПОЛЬЗОВАНИЯ ТАБЛИЦ ДЛЯ СОКРАЩЕНИЯ ВРЕМЕНИ СЧЕТА

Ю.Г. Косарев

С появлением вычислительных машин и систем с большими оперативными памятями (ОП) нередко возникает задача, как использовать свободные объемы ОП для сокращения времени счета. Один из возможных путей решения этой задачи - применение таблиц функций.

Ниже приводятся некоторые способы записи и употребления таблиц, обеспечивающие быструю выборку нужной информации.

I. Таблицы функций от одного аргумента $z=f(x), a \leqslant x < b$.

I.I. Шаг h задания аргумента x постоянен.

Расположим таблицу в ОП. Содержимым ячеек ОП будет значение функции z. Предполагается, что размеры таблицы $(b-a)/h$ позволяют разместить её в ОП, начиная с ячейки A_a, содержащей $z=f(x=a)$ (при $b>a\geqslant 0$), либо с ячейки A_b, содержащей $z=f(x=b)$ при $|b|<|a|, a\leqslant 0$). Тогда между аргументом x и адресом ячейки A_z, содержащей z, существует следующая очевидная зависимость:

$$A_z = A_a + \left[\frac{x-a}{h}\right], \qquad (1)$$

где $\left[\frac{x-a}{h}\right]$ - округленное до целого $(x-a)/h$.

Если a/h целое[*], то

$$A_z = A_0 + [x/h], \qquad (2)$$

где

$$A_0 = A_a - \frac{a}{h}. \qquad (3)$$

A_0 соответствует адресу ячейки, которая должна содержать $z_0=f(0)$.

[*] Если это не так, то расширим интервал задания x до a', кратного h, и заменим a на a'.

При $a \leqslant 0, b > 0$, $A_a \leqslant A_0 < A_b$ нужно положить $A_0 = A_{z=z_0}$.

Случай, когда аргумент x задан с фиксированной запятой, прост, и мы на нем останавливаться не будем.

Если $x = q_x \cdot 2^{P_x}, q_x < 1$, P_x - целое, то h удобно выбирать в виде $h = 2^d$, где d - целое. В этом случае

$$A_z = A_0 + \left[q_x \cdot 2^{P_x - d}\right] \qquad (4)$$

может быть получено сложением с плавающей запятой x со специально подобранной константой c. Для машины "Минск-2", при округлении до ближайшего целого, это будет выглядеть так[*]:

```
a+0   +I5 00   c+0 <x>        СМ:40 00 A_z a_9 a_10 (30+d)  (224мксек)
a+I   +72 00   c+I 0001       0001:00 00 A_z 0000           ( 72мксек)
                                                            (296мксек)
c+0   +20 00 0 A_0  I00 (3I+a)   для a +0,
c+I   +00 00 7777 0000           для a +I.
```

Здесь и далее $\langle x \rangle$ означает адрес ячейки ОП, содержащей x. Команда над числом z, например, сложение u и z, имеет вид:

```
b+0   +I5 0I 0000<u>          СМ: u+z.
```

Поясним, что происходит при выполнении команды $a+0$:

I) Выравнивание порядков. Мантисса числа x сдвигается вправо на $(3I+d-P_x)_8$ двоичных разрядов.[**]

2) Сложение. Число A_0, занимающее разряды I4 ÷ 25, складывается со сдвинутой мантиссой числа x и образует A_z. При этом так как у константы $c+0$ $\alpha_{26}=1$, то A_z округляется до ближайшего целого.

3) Нормализация. Мантисса сдвигается на один разряд влево, число A_z размещается в разрядах первого адреса.

4) Машинное округление. К 29-му дополнительному разряду мантиссы добавляется "I". Благодаря сдвигу при нормализации 29-ый разряд содержит нуль, и округление не изменяет результата. После этого команда a +I выделяет первый адрес и отправляет результат в индексную ячейку.

Для округления до меньшего целого[***] достаточно в констан-

[*] В "Минске-2" код операции занимает разряды 0,I-6, номер индексной ячейки 9 - I2, первый адрес - I3 - 24, второй - 25 - 36, мантисса - I - 28, порядок - разряды 3I-36, знак мантиссы - 0-ой разряд, знак порядка - 30-ый. Далее, при описании программ двоичные числа помечаются точкой сверху. Двоичный разряд с номером i (i =0,I,...,36) обозначается α_i, восьмеричный разряд с номером j (j =I,...,I2) a_j. В примечаниях к командам в скобках указывается среднее время их выполнения в микросекундах. Время выполнения команд сложения с плавающей запятой определялось согласно [I].

[**] Предполагается, что $P_\beta - d \leqslant 23$.

[***] Точнее, до наибольшего целого, не превышающего данного числа.

те $c+0$ положить $\alpha_{26}=0$. В этом случае, если x находится в сумматоре или индексной ячейке, обращение к таблице можно сформировать одной командой.

$a+0$ +I6 00 $c+0$ 0001 0001:4000 A_z 00 (30+d) (256мксек)

или

$a+0$ +I4 00 $c+0$ 0001 (280мксек)

$c+0$ +0I 00 0$\dot{0}\dot{0}$ A_0 $\dot{0}$(35+d) для $a+0$, A_0 в разрядах I8-29.

$b+0$ +I5 0I 0000<u>-(30+d) СМ: $u+z$ *)

При обращении к таблице по второму адресу можно употребить в качестве команды (a +I) сдвиг. Для округления до меньшего целого

$a+0$ +I5 00 $c+0$ <x>	СМ:4000 0$\dot{0}$ A_z $\dot{0}\dot{0}$(34+d)	(240мксек)
$a+I$ +66 00 $c+I$ 0001	0001:0010 0000 A_z	(I04мксек)
$c+0$ +20 00 0$\dot{0}\dot{0}$ A_0 $\dot{0}$ (35+$\dot{d}$)	для $a+0$	(344мксек)
$c+I$ +00 00 0000 0II0	для $a+I$	
$b+0$ +I5 0I <u> 0000	СМ: $z+u$	

При округлении до ближайшего целого время можно уменьшить на I2 мксек, изменив константу (c +0) на

+40 00 0$\dot{0}$ A_0 $\dot{0}\dot{0}$ (34+d) для $a+0$; A_0 в разрядах I7-28.

I.2. Для уменьшения размеров таблицы разобьем ее на части, в каждой из которых будет свой шаг. При небольшом числе частей можно установить, где находится нужное значение функции $z=f(x)$, сравнением величины x с граничными значениями частей. При большом числе частей может оказаться выгоднее следующий способ построения таблицы.

Насчитываем последовательные значения $a_0, a_1, \ldots, a_k, a_{k+1}, \ldots$ данной табличной функции с постоянным шагом $a_{k+1}-a_k=2^s$, затем каждый интервал (a_k, a_{k+1}) дробим на более мелкие интервалы с шагом $h_k=2^{d_k}$. При этом часть k будет содержать 2^{τ_k} ячеек, где $\tau_k=s-d_k$ (s, d_k - целые).

Кроме основной таблицы, построим вспомогательную, содержащую константы $C_k=\varphi(x)$ для каждой части. Тогда формирование обращения к таблице $z=f(x)$ при x, находящемся в сумматоре, займет три команды.

$a+0$ +I6 00 $c+0$ 0002	0002:4000 B_k 00 (30+S)	(256мксек)
$a'+0$ +I5 02 0000 <x> -(30+S)		(248мксек)
$a'+I$ +72 00 $c+I$ 0001	0001:0000 A_z 0000	(72мксек)
		(576мксек)

*) использовать такой же прием при округлении до ближайшего целого нельзя, так как в этом случае α_{25} - произвольное число.

$c+0$ +0I 00 0$\dot{0}\dot{0}$ B_0 $\dot{0}$ (35+S) для $a+0$, B_0 в разрядах I8-29.

$c+I$ +00 00 7777 0000 для $a'+I$.

Здесь $B_k=B_0+k$ - адрес константы C_k, B_0 - начальная ячейка таблицы констант. Эти константы имеют вид:

B_0+k) +2000 $\dot{0}$ A_0^k α_{26} 0$\dot{0}$ (3I+d_k),

где A_0^k получаются из (3) при $h=h_k$ и $a=a_k$ - значению x, соответствующего первой ячейке части k с адресом A_{a_k}. При округлении до меньшего (ближайшего) целого $\alpha_{26}=0$, ($\alpha_{26}=1$).

Чтобы при округлении до ближайшего целого избежать выхода за пределы части таблицы, первая ячейка части k должна быть последней ячейкой части k-I. Каждую часть основной таблицы можно также располагать в ОП отдельно, добавляя к ней при округлении до ближайшего целого $(2^{\tau_k}+I)$-ю ячейку.

I.3. Иногда, особенно при небольшом числе частей, выгодно так разбить таблицу на части, чтобы адрес первой ячейки части соответствовал значению x, равного степени 2. Тогда распознавать номер части можно по P_x - значению порядка x. Для чего строится таблица $(P_{max}-P_{min}) \leqslant$ I27 констант C_k, таких же, как и в п. I.2. Здесь (P_{min}, P_{max}) - интервал задания P_x. В этой таблице константы расположены соответственно следующим значениям порядка -77, -76, ..., ±00, ..., + 77.

Для "Минска-2" формирование обращения к таблице при x, находящемся в сумматоре, занимает около 400 мксек.

$a+0$ +72 00 $c+I$ 0001	0001:0000 0000 0$\dot{0}\dot{0}$ P_x	(72 мксек)
$a'+0$ +I5 0I <x> B_0		(248 мксек)
$a'+I$ +72 00 $c+2$ 0002	0002:0000 A_z 0000	(72 мксек)
$c+I$ +00 00 0000 0I77		(392 мксек)
$c+2$ +00 00 7777 0000		

Здесь B_0 - адрес ячейки, которая содержит константу, соответствующую порядку ±00.

I.4. Таблица функции $z=f(x)$ с кусочно-постоянным шагом для случая $x_1 \leqslant x_2 \leqslant \ldots \leqslant x_n$, где x_i - значение аргумента x при i-ом обращении к таблице, может иметь произвольные размеры частей.

При этом в отличие от п. I.2 вспомогательная таблица содержит для каждой части k, кроме константы C_k в ячейке B_0+2k, константу *) 0000 $A_{a_{k+1}}$ 0000 в ячейке B_0+2k+1 (k =0,I,..., $N-1$), а формирование обращения к таблице имеет вид:

*) Надобность в этой константе отпадает, если все части содержат одинаковое число ячеек.

```
a+0 +I5 02 000I ⟨x⟩ }                                   (248 мксек)
a+I +72 00 c+I 000I } 000I: 0000 A_z 0000               ( 72 мксек)
a+2 +23 02 0002 0000 }                                  ( 72 мксек)
a+3 -32 00 a+4 b+0   } A_z < A_{a_{k+1}}                ( 48 мксек)
a+4 -20 02 a+0 a+2                                      (1/q I20 мксек)
a+5 -00 00 0000 0000
c+I +00 00 7777 0000   для a+I                          (440+1/q I20мксек)
```

Предполагается, что перед первым обращением к таблице индексная ячейка 02 содержит N-I B_0-I 0000. Здесь q - среднее число обращений к одной части таблицы.

2. Таблицы функции от двух переменных $z=f(x,y)$.

2.I. Шаги $h_1=2^{d_1}$ по x и $h_2=2^{d_2}$ по y в области задания z $a\leqslant x<b$, $c\leqslant y<d$ постоянны.

Пусть в части k таблицы находятся значения $z=f(x_k,y)$

Интервалы задания x и y выбраны так, чтобы $(b-a)/h_1=r_1$,

$$(d-c)/h_2=r_2;\quad 2^{\tau_1-1}<r_1\leqslant 2^{\tau_1},\ r_2=2^{\tau_2};$$

d_1, d_2, τ_1, τ_2 - целые. В этом случае адрес ячейки, в которой хранится z,

$$A_z=A_0+x\cdot 2^{-d_1+\tau_2}+y\cdot 2^{-d_2}, \qquad (5)$$

где

$$A_0=A_a-a\cdot 2^{-d_1+\tau_2}-c\cdot 2^{d_2}. \qquad (6)$$

Для "Минск-2" при округлении до ближайшего целого

```
a+0 +I5 00 c+0 ⟨x⟩                                      (200 мксек)
a+I +72 00 c+I 000I 000I: 00 00 x·2^{-d1+τ2} 0000       ( 72 мксек)
a'+0 +I5 00 c'+0 ⟨y⟩                                    (224 мксек)
a'+I +73 00 c'+I 0000                                   ( 48 мксек)
a'+2 +I2 00 000I 000I 000I: 0000 A_z 0000               ( 72 мксек)
                                                        (6I6 мксек)
c+0 +20 00  0̇0̇...0̇I0̇...  0̇0̇0̇0 (3I+d1-τ2) для a+0
c+I +00 00  İ...İ0̇...0̇ (τ1, τ2)  0000  для a+I
c'+0 +20 00 0̇ (τ1) A0 (τ2)  İ0̇0 (3I+d2)  для a'+0
                  A0 занимает разряды I4 ÷ 25.
c+I +00 00  7777 0000                   для a'+I
```

Для округления до меньшего целого достаточно положить равными нулю в константе $(c+0)$ $\alpha_{26-\tau_2}$, в константе $(c'+0)$ $-\alpha_{26}$.

Время сократится на 48 мксек, если команды $(a+0)$ и $(a'+0)$ выполняются сразу же после вычисления x и y, соответственно, а при округлении до меньшего целого еще на 48 мксек, если не

очищать второй адрес, как это делалось в п. I.I.

2.2. Таблица для функции $z=f(x,y)$ с кусочно-постоянными шагами $h_x=2^{d_x}=\varphi(x)$ по x и $h_y=2^{d_y}=\varphi_2(x)$ по y. Кроме основной таблицы, такой же, как в п. I.2, построим таблицу констант для каждой части. Шаг этой таблицы, как и в п. I.2, постоянен и равен h_1 - наименьшему шагу по x. Эти константы имеют вид: B_0+k +20 00 0̇ A_0 α_{26} 0̇0̇ (3I+d_y). При округлении по y до меньшего (ближайшего) целого $\alpha_{26}=0$ ($\alpha_{26}=1$). Обращение к таблице констант такое же, как в п. I.I. При округлении по x до ближайшего целого

```
a+0 +I5 00 c+0 ⟨x⟩ }                                    (224 мксек)
a+I +72 00 c+I 0002 } 0002: 0000 B_x 0000               ( 72 мксек)
a'+0 +I5 02 0000 ⟨y⟩ }                                  (248 мксек)
a'+I +72 00 c+I 000I } 000I: 0000 A_z 0000              ( 72 мксек)
c+0 +20 00 0̇ B0 İ0̇0 (3I+d1)  для a+0,                   (6I6 мксек)
c+I +00 00 7777 0000          для a+I, a'+I.
```

При округлении по x до наименьшего целого достаточно в $c+0$ положить $\alpha_{26}=0$, либо, если x находится в сумматоре, поступить аналогично п. I.I, т.е.

```
a+0 +I6 00 c+0 0002          0002: 0000 B_x (30+d1)     (256 мксек)
a'+0 +I5 02 0000 ⟨y⟩ -(30+d1)                           (248 мксек)
a'+I +72 00 c+I 000I         000I: 0000 A_z 0000        ( 72 мксек)
c+0 +0I 00 0̇0̇0̇ B0 0̇(35+d1)      для a+0                (576 мксек)
c+I +00 00 7777 0000          для a'+I
```

Полученное время обращения меньше, чем в п. 2.I, т.е. и при постоянном шаге по x и по y имеет смысл использовать таблицу констант.

2.3. В случае, когда шаг по x $h_x=2^{d_x}=\varphi_1(x)$, а шаг по y $h_y=2^{d_y}=\varphi_2'(y)$ $(a\leqslant x\leqslant b,\ c\leqslant y\leqslant d)$ можно считать, что таблица функции $z=f(x,y)$ состоит из $(b-a)/h_x$ подтаблиц с кусочно-постоянным шагом для функции одного аргумента y.

Для выборки z нужно сначала по x найти адрес B_ℓ первой ячейки соответствующей подтаблицы, после чего задача сводится к случаю, рассмотренному в п. I.2. Для нахождения B_ℓ построим вспомогательную таблицу с кусочно-постоянным шагом по x (см. п. I.2). Для уменьшения времени обращения к таблице, если позволяет объем ОП, вспомогательные таблицы для B_ℓ и для констант $C_y=\varphi(y)$ удобно сделать с постоянным шагом.

2.4. Для уменьшения размеров вспомогательных таблиц можно применить тот же прием, что и в п. I.3. При этом некоторые граничные значения аргументов могут и не быть степенью 2, что не-

сколько увеличит объем вспомогательной таблицы, но она будет, как правило, значительно меньше таблиц в пп. 2.2 и 2.3. Кроме того, благодаря тому, что в этом случае размеры каждой части могут устанавливаться с большим произволом, чем в пп. 2.2 и 2.3, иногда может сократиться и объем основной таблицы.

3. Формирование обращения к таблице $z = f(x)$ с постоянным шагом h для переменной $x = x_1$ фактически означает, что сформировано обращение и для переменных $x_1 + \Delta \cdot n$, где Δ - число, кратное h, либо достаточно близкое к кратному h ($n = \pm 1, \pm 2, \ldots, \pm N$). Нужно только изменить на величину $[\Delta \cdot n]$ соответствующий адрес команды, выполняемой над величиной $f(x + \Delta \cdot n)$, и расширить таблицу на $2N$ ячеек.

Это свойство может быть использовано при интерполяции. Так при линейной интерполяции таблицу достаточно увеличить на две ячейки. Увеличение на 1/4 таблицы для $Sin\, x\ (0 \leq x < 2\pi)$ позволяет по той же таблице определять и $Cos\, x = Sin(x + \frac{\pi}{2})$. В командах, использующих $Cos\, x$, к соответствующему адресу добавляется $[\pi / 2h]$.

Все указанное выше справедливо и для таблиц с кусочно-постоянным шагом. При этом каждую часть таблицы нужно увеличить не более чем на $2N$ ячеек.

4. Таблицы функций, аргумент которых - случайное число. Датчики псевдослучайных чисел обычно вырабатывают случайную величину γ, равномерно распределенную в интервале (0,1). Случайная величина ξ с другим распределением вычисляется как функция γ. При табулировании этой функции удобнее, чтобы аргумент γ был в разрядах соответствующего адреса. Получение такого "случайного адреса" и отсылка его в индексную ячейку в некоторых датчиках [2] не требует увеличения времени, т.е. подпрограмма, вырабатывающая случайные числа, одновременно и формирует обращение к соответствующей таблице.

5. Выигрыш во времени δ при применении таблиц вместо непосредственного счета функции равен

$$\delta = \frac{t_{\alpha}}{t_{обр.} + t_{табл.} N/n}, \qquad (7)$$

где t_{α} - время вычисления функции без применения таблиц;
$t_{обр.}$ - среднее время обращения к таблице;
$t_{табл.}$ - среднее время вычисления одного значения функции для таблицы;
N - число значений функции в таблице;
n - среднее число обращений к таблице в процессе решения задачи.

Нередко $t_{табл.} < t_{\alpha}$ благодаря закономерному изменению аргументов и значений функции при счете таблицы. В практических задачах [3] - [5] нередко $N/n \ll 1$ и $t_{обр.} \ll t_{\alpha}$, т.е. $\delta \gg 1$. Это позволяет применять и более сложные способы обращения к таблице, чем описаны выше, а также размещать большие таблицы во вспомогательной памяти. Процесс использования больших таблиц при монотонном увеличении аргумента (см. п.1.4) особенно прост. Вводится очередная часть таблицы. Выполняется основная программа, пока значение аргумента не превысит наибольшее значение аргумента данной части. После этого вводится следующая часть и т.д. При размещении таблиц во вспомогательной памяти:

$$\delta = \frac{t_{\alpha}}{t_{обр.} + t_{табл.} N/n + t_{в} \cdot N/n_1}, \qquad (8)$$

где $t_{в}$ - среднее время ввода одной ячейки из вспомогательной памяти, n_1 - среднее число обращений к таблице за один ввод всей таблицы. Если n_1 мало, то нередко его можно увеличить одновременным счетом некоторого числа вариантов.

При произвольном изменении аргумента без одновременного счета вариантов размещение таблицы во вспомогательных памятях может оказаться целесообразным только при больших значениях t_{α}. Когда возможен одновременный счет вариантов, схема работы следующая. В ОП последовательно вводятся части таблицы. Считаются те варианты, для которых в таблице нашлись нужные значения, затем вводится очередная часть. После последней части таблицы вновь вводится первая. Варианты, счет которых закончился, замещаются новыми вариантами и т.д.

6. Использование таблиц в вычислительных системах типа "Минск-222". Рассмотрим основные случаи.

6.1. Схема параллельного алгоритма такова, что каждой ветви вычислений нужен свой набор таблиц. Например, при решении обыкновенных дифференциальных уравнений, когда таблицы составляются для правых частей уравнений [3]. В этом случае таблицы используются точно так же, как и в отдельной машине.

6.2. Конвейерная схема работы. Алгоритм разбивается на последовательно работающие блоки, каждый из которых находится в своей машине. Данная схема позволяет разместить в ОП необходимые для каждого блока таблицы и исключить (или уменьшить) обмен со вспомогательными памятями. Машины в этом случае обмениваются промежуточными результатами.

6.3. Если во всех машинах одновременно используется как-

либо таблица, то вычислительная система из ℓ машин позволяет сократить время ввода этой таблицы по сравнению с одной машиной. Для этого таблица разделяется на ℓ частей, каждая из которых вводится в свою машину, а затем машины по очереди посылают свои части во все остальные. Выигрыш во времени

$$\delta = \frac{\rho}{\rho + \ell}\,\ell\,, \qquad (9)$$

где ρ – отношение средних времен обмена ОП одним кодом машиной со своей вспомогательной памятью и с ОП всех остальных машин системы.

Для случая обмена с магнитной лентой в системе "Минск-222" $\rho = 8$.

Литература

1. Ю.Г. Косарев, С.В. Нагаев. О потерях времени на синхронизацию в вычислительных системах. Вычислительные системы, Новосибирск, Изд-во "Наука", Сибирское отделение, 1967 г., вып. 24, стр. 21-39.
2. М.В.Антипов, Ф.М.Израйлев, Б.В.Чириков. Статистическая проверка датчика псевдослучайных чисел – Данный сборник, 77-85.
3. А.И.Петрович, Ю.Г.Косарев. Исследование колебательных процессов автопоездов на системе "Минск-222". – Данный сборник, стр. 12-14.
4. Ю.Г.Косарев. Моделирование неупругого рассеяния электронов методом Монте-Карло.– Данный сборник, стр. 34-40.
5. М.А.Бахин, В.П.Бубнов, И.С.Захарова, Ю.Г.Косарев, В.Б.Нестеренко. Термодинамический расчет газовых АЭС циклов на диссоциирующих газах на системе "Минск-222".– Данный сборник, стр. 22-25.

Поступила в редакцию
20.VI.1967 г.