

УДК. 681.142.2.

ПРИМЕРЫ ИСПОЛЬЗОВАНИЯ ТАБЛИЦ ДЛЯ СОКРАЩЕНИЯ ВРЕМЕНИ СЧЕТА

Ю.Г. Косарев

С появлением вычислительных машин и систем с большими оперативными памятьми (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 $z=f(x), a \leq x < b$.

I.1. 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 $b > a > 0$), либо с ячейки A_b , содержащей $z=f(x=b)$ при $|b| < |a|, a \leq 0$. Тогда между аргументом x и адресом ячейки A_z , содержащей z , существует следующая очевидная зависимость:

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

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

Если a/h целое^{*}, то

$$A_z = A_0 + \left[x/h \right], \quad (2)$$

где

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

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 $a \leq 0, b > 0, A_a \leq A_0 < A_b$ 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

$$A_z = A_0 + [q_x \cdot 2^{p_x - d}] \quad (4)$$

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^{*)}:

$a+0$ +15 00 $c+0$ $\langle x \rangle$ CM:40 00 $A_z \alpha_9 \alpha_{10} (30+d)$ (224мксек)
 $a+1$ +72 00 $c+1$ 0001 0001:00 00 A_z 0000 (72мксек)

(296мксек)

$c+0$ +20 00 0 A_0 100 $(31+d)$ для $a+0$,
 $c+1$ +00 00 7777 0000 для $a+1$.

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

$b+0$ +15 01 0000 $\langle u \rangle$ CM: $u+z$.

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

1) Выравнивание порядков. Мантисса числа x сдвигается вправо на $(31+d-p_x)_8$ двоичных разрядов.^{**)}

2) Сложение. Число A_0 , занимающее разряды 14 ÷ 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 "1". Благодаря сдвигу при нормализации 29-ый разряд содержит нуль, и округление не изменяет результата. После этого команда $a+1$ выделяет первый адрес и отправляет результат в индексную ячейку.

Для округления до меньшего целого^{**)*)} достаточно в констан-

^{*)} В "Минске-2" код операции занимает разряды 0,1-6, номер индексной ячейки 9 - 12, первый адрес - 13 - 24, второй - 25 - 36, мантисса - 1 - 28, порядок - разряды 31-36, знак мантиссы - 0-ой разряд, знак порядка - 30-ый. Далее, при описании программ двоичные числа помечаются точкой сверху. Двоичный разряд с номером i ($i=0,1,\dots,36$) обозначается α_i , восьмеричный разряд с номером j ($j=1,\dots,12$) α_j . В примечаниях к командам в скобках указывается среднее время их выполнения в микросекундах. Время выполнения команд сложения с плавающей запятой определялось согласно [1].

^{**)*)} Предполагается, что $p_b - d \leq 23$.

^{**)*)} 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.

$\alpha+0 +16 00 c+0 0001 \quad 0001:4000 A_z 00 (30+d) \quad (256\text{мксек})$
или

$\alpha+0 +14 00 c+0 0001 \quad (280\text{мксек})$

$c+0 +01 00 000 A_0 \dot{0}(35+d)$ для $\alpha+0, A_0$ в разрядах 18-29.

$\beta+0 +15 01 0000 \langle u \rangle - (30+d) \text{ см: } u+z^*$

При обращении к таблице по второму адресу можно употребить в качестве команды ($\alpha+1$) сдвиг. Для округления до меньшего целого

$\alpha+0 +15 00 c+0 \langle x \rangle \quad \text{см: } 4000 00 A_z \dot{0}(34+d) \quad (240\text{мксек})$

$\alpha+1 +66 00 c+1 0001 \quad 0001:0010 0000 A_z \quad (104\text{мксек})$

$c+0 +20 00 000 A_0 \dot{0} (35+d)$ для $\alpha+0 \quad (344\text{мксек})$

$c+1 +00 00 0000 0110$ для $\alpha+1$

$\beta+0 +15 01 \langle u \rangle 0000 \quad \text{см: } z+u$

При округлении до ближайшего целого время можно уменьшить на 12 мксек, изменив константу ($c+0$) на

$+40 00 00 A_0 \dot{0} (34+d)$ для $\alpha+0; A_0$ в разрядах 17-28.

1.2. 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 $a_0, a_1, \dots, a_k, a_{k+1}, \dots$ данной табличной функции с постоянным шагом $a_{k+1} - a_k = 2^s$, затем каждый интервал (a_k, a_{k+1}) делим на более мелкие интервалы с шагом $h_k = 2^{d_k}$. При этом часть k будет содержать 2^{z_k} ячеек, где $z_k = s - d_k$ (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.

$\alpha+0 +16 00 c+0 0002 \quad 0002:4000 B_k 00 (30+s) \quad (256\text{мксек})$

$\alpha'+0 +15 02 0000 \langle x \rangle - (30+s) \quad (248\text{мксек})$

$\alpha'+1 +72 00 c+1 0001 \quad 0001:0000 A_z 0000 \quad (72\text{мксек})$

(576мксек)

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

$c+0 +01 00 000 B_0 0$ (35+S) для $\alpha+0$, B_0 в разрядах I8-29.
 $c+1 +00 00 7777 0000$ для $\alpha'+1$.

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

$$B_0 + k) + 2000 0 A_0^k \quad \alpha_{26} 00 (3I + d_k),$$

где A_0^k получаются из (3) при $h=h_k$ и $\alpha=\alpha_k$ - значению α , соответствующего первой ячейке части k с адресом A_{α_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 $k-1$. Каждую часть основной таблицы можно также располагать в ОП отдельно, добавляя к ней при округлении до ближайшего целого ($2^{2k}+1$)-ю ячейку.

1.3. 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 ($P_{max} - P_{min}$) ≤ 127 констант C_k , таких же, как и в п. 1.2. Здесь (P_{min}, P_{max}) - интервал задания P_x . В этой таблице константы расположены соответственно следующим значениям порядка $-77, -76, \dots, \pm 00, \dots, +77$.

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

$\alpha+0 +72 00$	$c+1 0001$	$0001:0000$	$0000 000 P_x$	(72 мксек)
$\alpha'+0 +15 01$	$\langle x \rangle B_0$	$0002:0000$	$A_z 0000$	(248 мксек)
$\alpha'+1 +72 00$				(72 мксек)
$c+1 +00 00$	$0000 0177$			(392 мксек)
$c+2 +00 00$	$7777 0000$			

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

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

При этом в отличие от п. 1.2 вспомогательная таблица содержит для каждой части k , кроме константы C_k в ячейке $B_0 + 2k$, константу*) $0000 A_{\alpha_{k+1}}$ 0000 в ячейке $B_0 + 2k + 1$ ($k=0, 1, \dots, N-1$), а формирование обращения к таблице имеет вид:

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

$a+0$	$+15$	02	0001	$\langle x \rangle$		(248 мксек)
$a+1$	$+72$	00	$c+1$	0001	} $0001: 0000 A_z 0000$	(72 мксек)
$a+2$	$+23$	02	0002	0000		(72 мксек)
$a+3$	-32	00	$a+4$	$b+0$	} $A_z \langle A_{\alpha_{k+1}}$	(48 мксек)
$a+4$	-20	02	$a+0$	$a+2$		$(\frac{1}{g} 120 \text{ мксек})$
$a+5$	-00	00	0000	0000		
$c+1$	$+00$	00	7777	0000	для $a+1$	$(440 + \frac{1}{g} 120 \text{ мксек})$

Предполагается, что перед первым обращением к таблице индексная ячейка 02 содержит $N-1$ B_0-1 0000. Здесь g - среднее число обращений к одной части таблицы.

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

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

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

Интервалы задания x и y выбраны так, чтобы $(b-a)/h_1 = n_1$,

$$(d-c)/h_2 = n_2; \quad 2^{z_1-1} < n_1 \leq 2^{z_1}, \quad n_2 = 2^{z_2};$$

d_1, d_2, z_1, z_2 - целые. В этом случае адрес ячейки, в которой хранится z ,

$$A_z = A_0 + x \cdot 2^{-d_1+z_2} + y \cdot 2^{-d_2}, \quad (5)$$

где

$$A_0 = A_a - a \cdot 2^{-d_1+z_2} - c \cdot 2^{d_2}. \quad (6)$$

Для "Минск-2" при округлении до ближайшего целого

$a+0$	$+15$	00	$c+0$	$\langle x \rangle$		(200 мксек)	
$a+1$	$+72$	00	$c+1$	0001	$0001: 00 00$	$x \cdot 2^{-d_1+z_2} 0000$	(72 мксек)
$a'+0$	$+15$	00	$c'+0$	$\langle y \rangle$		(224 мксек)	
$a'+1$	$+73$	00	$c'+1$	0000		(48 мксек)	
$a'+2$	$+12$	00	0001	0001	$0001: 0000$	$A_z 0000$	(72 мксек)
						(616 мксек)	

$c+0$ $+20$ 00 $\underbrace{\overset{\cdot\cdot}{0}\overset{\cdot\cdot}{0}\dots\overset{\cdot\cdot}{0}\overset{\cdot\cdot}{1}\overset{\cdot\cdot}{0}\dots\overset{\cdot\cdot}{0}\overset{\cdot\cdot}{0}\overset{\cdot\cdot}{0}}_{z_1}$ 0000 $(3I+d_1-z_2)$ для $a+0$

$c+1$ $+00$ 00 $\underbrace{\overset{\cdot\cdot}{1}\dots\overset{\cdot\cdot}{1}\overset{\cdot\cdot}{0}\dots\overset{\cdot\cdot}{0}}_{z_2}$ 0000 для $a+1$

$c'+0$ $+20$ 00 $\overset{\cdot\cdot}{0} \overset{z_1}{A_0} \overset{z_2}{i00}$ $(3I+d_2)$ для $a'+0$

A_0 занимает разряды $I4 + 25$.

$c+1$ $+00$ 00 7777 0000 для $a'+1$

Для округления до меньшего целого достаточно положить равными нулю в константе $(c+0) \alpha_{26-z_2}$, 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. 1.1.

2.2. Таблица для функции $z = f(x, y)$ с кусочно-постоянными шагами $h_x = 2^{d_x} = \varphi_1(x)$ по x и $h_y = 2^{d_y} = \varphi_2(y)$ по y . Кроме основной таблицы, такой же, как в п. 1.2, построим таблицу констант для каждой части. Шаг этой таблицы, как и в п. 1.2, постоянен и равен h_x - наименьшему шагу по x . Эти константы имеют вид:

$B_0 + \kappa + 20 \ 00 \ 0 \ A_0 \ \dot{\alpha}_{26} \ 00 \ (3I + d_y)$. При округлении по y до меньшего (ближайшего) целого $\alpha_{26} = 0$ ($\alpha_{26} = 1$). Обращение к таблице констант такое же, как в п. 1.1. При округлении по x до ближайшего целого

$a+0 \ +15 \ 00 \ c \ +0 \ \langle x \rangle$	(224 мксек)
$a+1 \ +72 \ 00 \ c \ +1 \ 0002$	(72 мксек)
$a'+0 \ +15 \ 02 \ 0000 \ \langle y \rangle$	(248 мксек)
$a'+1 \ +72 \ 00 \ c \ +1 \ 0001$	(72 мксек)
$c+0 \ +20 \ 00 \ 0 \ B_0 \ \dot{1}00 \ (3I + d_x)$ для $a+0$,	(616 мксек)
$c+1 \ +00 \ 00 \ 7777 \ 0000$ для $a+1, a'+1$.	

При округлении по x до наименьшего целого достаточно в $c+0$ положить $\alpha_{26} = 0$, либо, если x находится в сумматоре, поступить аналогично п. 1.1, т.е.

$a+0 \ +16 \ 00 \ c \ +0 \ 0002$	0002: 0000 $B_x (30 + d_x)$ (256 мксек)
$a'+0 \ +15 \ 02 \ 0000 \ \langle y \rangle - (30 + d_x)$	(248 мксек)
$a'+1 \ +72 \ 00 \ c \ +1 \ 0001$	0001: 0000 A_2 0000 (72 мксек)
$c+0 \ +01 \ 00 \ 000 \ B_0 \ \dot{0} (35 + d_x)$	для $a+0$ (576 мксек)
$c+1 \ +00 \ 00 \ 7777 \ 0000$	для $a'+1$

Полученное время обращения меньше, чем в п. 2.1, 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 $h_y = 2^{d_y} = \varphi_2(y)$ ($a \leq x \leq b, c \leq y \leq d$) 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_e первой ячейки соответствующей подтаблицы, после чего задача сводится к случаю, рассмотренному в п. 1.2. Для нахождения B_e построим вспомогательную таблицу с кусочно-постоянным шагом по x (см. п. 1.2). Для уменьшения времени обращения к таблице, если позволяет объем ОП, вспомогательные таблицы для B_e 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. 1.3. 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 $x = x_0$, фактически означает, что сформировано обращение и для переменных $x_0 + \Delta \cdot n$, где Δ - число, кратное h , либо достаточно близкое к кратному h ($n = \pm 1, \pm 2, \dots, \pm N$). 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 y , равномерно распределенную в интервале $(0, 1)$. Случайная величина ξ с другим распределением вычисляется как функция y . При табулировании этой функции удобнее, чтобы аргумент y 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

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

где t_{α} - время вычисления функции без применения таблиц;

$t_{\text{обр.}}$ - среднее время обращения к таблице;

$t_{\text{табл.}}$ - 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:

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

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

При произвольном изменении аргумента без одновременного счета вариантов размещение таблицы во вспомогательных memories 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 memories. 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

$$\delta = \frac{\rho}{\rho + e} e, \quad (9)$$

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" $\rho = 8$.

Л и т е р а т у р а

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. Бажин, 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
20.11.1967 г.