ВЫЧИСЛИТЕЛЬНЫЕ СИСТЕМЫ
1973 год Выпуск 57

УДК 681.142.2

О ПОСТРОЕНИИ ТАБЛИЦ С БЫСТРОЙ ВЫБОРКОЙ

Л.В.Головяшкина, Ю.Г.Косарев

Практическая реализация задач с большим объемом вычислений на данном вычислительном средстве часто оказывается невозможной без замены сложновычислимых функций более простыми, аппроксимирующими исходные функции с некоторой заданной точностью.

Такая замена связана с решением следующей задачи. Указаны классы $\Phi(c)$ быстровычислимых функций. Для любой $\varphi(\bar{x}) = \varphi(x_1, \ldots, x_n) \in \Phi(c)$, $\bar{x} \in X \subset R_n$, объем вычислений

$$t_{\varphi(\bar{x})} \leq c, \tag{1}$$

где c – заданное число операций. Требуется для произвольной функции $f(\bar{x})$ из некоторого достаточно широкого класса функций F и для заданного числа ε найти аппроксимирующую её функцию $\varphi(\bar{x}) \in \Phi(c)$ с мерой уклонения

$$\rho(f, \varphi) \leqslant \varepsilon. \tag{2}$$

Большой практический интерес представляют классы $\Phi(c)$ с малыми значениями c и среди них – классы табличных функций. Таблицей может быть представлена любая вычислимая функция, но для выполнения условий (1), (2), как правило, требуется запоминать большие объемы информации, если c и ε малы. Появление вычислительных систем и рост объема памяти ЭВМ значительно расширяет возможности использования табличных методов [1] и делает весьма актуальным их дальнейшую разработку.

Известны сложные задачи, имеющие большое практическое значение, решение которых стало возможным только благодаря использованию таблиц с быстрой выборкой [2-5].

Построение таблиц для основных функций, характеризующих элементарный акт рассеяния в задаче "Моделирование неупругого рассеяния электронов методом Монте-Карло" [2], вместе с табулированием элементарных функций, вычисляемых в основном цикле, позволило решить эту весьма трудоемкую задачу даже на машине малой производительности "Минск-22" [6].

§ I. Определение и классификация таблиц

Пусть $\varphi(\bar{x}) \in \Phi(c)$ аппроксимирует заданную функцию $f(\bar{x})$ ($\bar{x} \in X$) с мерой уклонения (2) для всех $\bar{x} \in X$. Упорядоченный набор чисел $u = (u_1, \dots, u_p)$, для которого указан алгоритм $\Gamma(u, \varphi)$, вычисляющий по набору u значения $\varphi(\bar{x})$, назовем т а б л и ц е й $T_{\varepsilon,c}(f)$ функции $f(\bar{x})$. Сложность $G(\Gamma)$ алгоритма $\Gamma(u)$ оценивается числом реализующих его операций.

При

$$\rho(f, \varphi) = \max_{\bar{x} \in X} |f(\bar{x}) - \varphi(\bar{x})| \tag{3}$$

абсолютная погрешность таблицы

$$\Delta_T = |f(\bar{x}) - \varphi(\bar{x})| \leqslant \varepsilon.$$

При

$$\rho(f, \varphi) = \max_{\bar{x} \in X} \frac{|f(\bar{x}) - \varphi(\bar{x})|}{|f(\bar{x})|} \tag{4}$$

относительная погрешность таблицы $\delta_T \leqslant \varepsilon$.

Объем V_T таблицы равен минимальному количеству двоичных разрядов, необходимых для записи набора чисел u.

Размерность W_T таблицы равна n, размерности вектора $\bar{x}$, аргумента функции $\varphi(\bar{x},)$.

Для функции одного переменного $f(x)$ при фиксированном ε можно построить различные таблицы $T_{\varepsilon,c}(f)$, отличающиеся предельным временем выборки табличного значения. Рассмотрим некоторые виды таблиц, для которых $1 \leqslant c \leqslant 7$.

Пусть таблица $T_{\varepsilon,c}(f)$ монотонной на $[a, b]$ функции $f(x)$ задана набором

$$u = (x_1 = a, x_2, \dots, x_N = b;\ \varphi(x_1), \dots, \varphi(x_N))$$

для которого

$$\varphi(x_i) = f(x_i) \pm \varepsilon_1 , \quad i = 1,\dots,N;$$
$$\rho(\varphi(x_i), \varphi(x_{i+1})) \leqslant \varepsilon_2 , \quad i = 1,\dots,N-1,$$

Здесь ε_1 – суммарная погрешность аппроксимации и машинной реализации функции $f(x)$; ε_2 – погрешность, связанная с табличной реализацией. Предполагаем, что $\varepsilon_1 < \varepsilon_2$.

Алгоритм $\Gamma(u)$ состоит в определении промежутка $[x_i, x_{i+1}) \in x$ и выборке соответствующего значения $\varphi(x) = \varphi(x_i)$. Если мера уклонения ρ определяется соотношением (3), то для заданной таблицы при $x \in [a, b]$

$$\rho(f(x), \varphi(x)) \leqslant \varepsilon_1 + \varepsilon_2 = \varepsilon.$$

На классификационной схеме представлены различные способы вычисления адреса табличного значения A_z в зависимости от шага h_x изменения аргумента.

I. Шаг h_x задания аргумента x постоянен.

I.I. При постоянном шаге $h_x = const = h$ адрес табличного значения $z = \varphi(x)$ определяется соотношением

$$A_z = A_0 + E\left(\frac{x}{h}\right), \tag{5}$$

где $E(x)$ – округленное до целого число x, $A_0 = A_{\varphi(a)} - E\left(\frac{a}{h}\right)$.

Соотношение (5) может быть реализовано тремя машинными операциями, среднее время выполнения которых на машине "Минск-22" – 589 мксек [5]. При этом в оперативной памяти машины необходимо хранить $N+2$ числа

$$u = (\varphi(x_1), \varphi(x_2), \dots, \varphi(x_N)), c_1, c_2 , \tag{6}$$

где $x_1 = a$, $x_N = b$, $N = E\left(\frac{b-a}{h_x}\right)$, c_1, c_2 – константы.

I.2. При $h_x = 2^d$ (d – целое), A_z определяется соотношением

$$A_z = A_0 + E(q_x \cdot 2^{p_x - d}),$$

где q_x – мантисса числа x, представленного в виде $x = q_x \cdot 2^{p_x}$.

Выборка табличного значения в этом случае может быть реализована одной-двумя операциями (224-296 мксек). Объем информации, хранящейся в ОП, определяется (6).

2. Шаг h_x есть функция x.

Из общего случая, когда A_z есть некоторая функция x, $A_z = \psi_1(x)$, интерес представляют те частные случаи, для которых эта зависимость реализуется простыми соотношениями.

2.I. Шаг h_x есть кусочно-постоянная функция. Точками $x_1^* = a, x_2^*, \ldots, x_{J+1}^* = b$ отрезок $[a, b]$ разбивается на промежутки $\alpha_j^* = [x_j^*, x_{j+1}^*), j = 1, \ldots, J$ такие, что для $x \in \alpha_j^*$ $h_x = h_j$. В этом случае адрес табличного значения определяется соотношением

$$A_z = A_{z_j^* = \varphi(x_j^*)} + E\left(\frac{x - x_j^*}{h_j}\right). \tag{7}$$

Пусть $A_{z_j^*}$ есть некоторая функция от x $A_{z_j^*} = \psi_2(x)$.

Если $x_{j+1}^* - x_j^* = h, j = 1, \ldots, J$, то значение $A_{z_j^*}$ определяется с помощью констант c_1, c_2 за 2-3 операции. Выборка табличного значения по формуле (7) требует 7 операций (I250 мксек.). В памяти машины необходимо хранить $N + 2J + 2$ чисел:

$$u = (\varphi(x_1), \varphi(x_2), \ldots, \varphi(x_N)); c_1, c_2, c_{1j}, c_{2j}, \tag{8}$$

$$N = \sum_{j=1}^{J} N_j, \quad N_j = E\left(\frac{x_{j+1}^* - x_j^*}{h_j}\right).$$

2.2. В частном случае, когда $x_{j+1}^* - x_j^* = h = 2^s$, а $h_x = h_j = 2^{d_j}$, значение $A_{z_j^*}$ определяется за I-2 операции, а выборка табличного значения может быть реализована двумя-тремя операциями (504-576 мксек.).

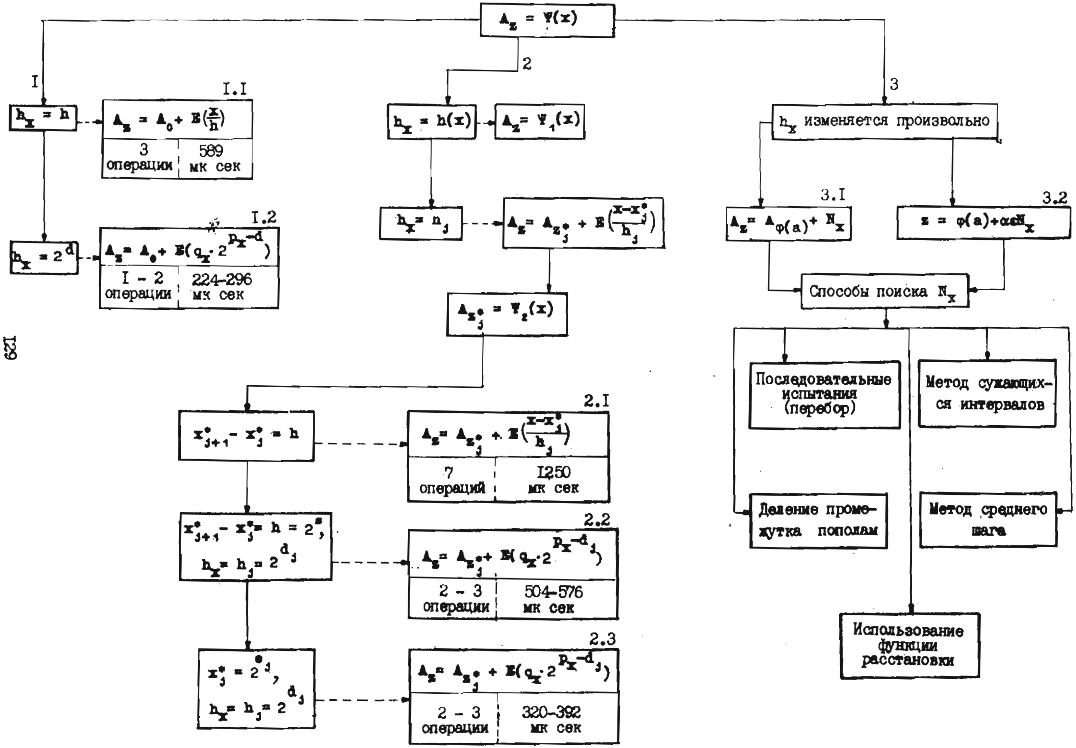
2.3. Когда J мало, иногда бывает выгодно разбить таблицу так, чтобы адрес первой ячейки части $A_{z_j^*}$ соответствовал значению x, равного степени 2. Тогда распознавать номер части можно по p_x – значению порядка x, для чего требуется всего лишь одна операция [I]. Выборка табличного значения A_z осуществляется за 2-3 операции (320-392 мксек.).

3. Произвольное изменение шага h_x.

3.I. При произвольном разбиении промежутка $[a, b]$ точками x_i в ОП машины хранятся числа $u = (x_1 = a, x_2, \ldots, x_N = b; \varphi(x_1), \varphi(x_2), \ldots, \varphi(x_N))$ и адрес A_z определяется зависимостью

$$A_z = A_{\varphi(a)} + N_x, \tag{9}$$

Классификационная схема таблиц функции одного переменного по способу задания шага изменения аргумента



где $A_{\varphi(a)}$ - адрес первой ячейки массива $\{\varphi(x_i)\}$, $i=1,2,\ldots,N$, N_x - номер того промежутка $[x_i, x_{i+1})$,которому принадлежит x.

3.2. Если $z=\varphi(x)$ не убывает ($\alpha=1$) или не возрастает ($\alpha=-1$) на $[a,b]$ и

$$\varphi(x_{i+1}) = \varphi(x_i) + \alpha\varepsilon,$$

то есть

$$\Delta_i = |\varphi(x_{i+1}) - \varphi(x_i)| \doteq \varepsilon, \quad i=1,2,\ldots,N-1,$$

значение функции в точке x определяется равенством

$$z = \varphi(a) + \alpha\varepsilon N_x. \tag{10}$$

Таблицы, для которых выборка табличного значения реализуется формулой (10), назовем таблицами минимального объема с заданной (абсолютной) погрешностью.

Ниже описываются некоторые способы поиска величины N_x, входящей в соотношения (9) и (10).

При методе последовательных испытаний (переборе) общая схема поиска N_x реализуется последовательностью трех операторов

$$A_1 \; \Phi_2 \; A_3, \tag{11}$$

где A_1 последовательно перебирает табличные узлы x_i, $i=1,2,\ldots,N$, и проверяет на каждом шаге выполнение условия $x_i \geqslant x$. При выполнении его управление передается оператору Φ_2 ,который выбирает необходимый набор узлов и отсылает их в рабочие ячейки оператора A_3 . A_3 - оператор интерполяции. Если для таблицы $T_{\varepsilon,c}(f)$ условие (2) выполнено заданием набора u, схема поиска N_x состоит из одного оператора A_1.

При методе сужающихся интервалов отрезок $[a,b]$ разбивается произвольно выбранными точками $x_1^*, x_2^*, \ldots$ на примерно равные (по числу содержащихся в них значений аргумента) интервалы. Оператор A_1 (11) сначала ищет промежуток $[x_i^*, x_{i+1}^*) \ni x$ (грубый поиск), после чего на этом промежутке применяется перебор (точный поиск).

При методе деления промежутка пополам (дихотомии) промежуток последовательно делится на две части, содержащих одинаковое (с точностью до единицы) число аргументов, и выбирается часть, содержащая x.

П р и   м е т о д е   с р е д н е г о   ш а г а   для заданной таблицы вычисляется средний шаг $\tilde{h} = \frac{b-a}{N}$, по которому находится начальная точка поиска $x_{\tilde{k}}$, где $\tilde{k} = E\left(\frac{b-a}{\tilde{h}}\right)$. Промежуток $[x_k, x_{k+1}) \ni x$ находится перебором, начиная со старших или младших x_k, в зависимости от соотношения между x и $x_{\tilde{k}}$.

При поиске и упорядочении элементов некоторого массива по признаку p длину перебора можно сократить, если значение признака несет в себе дополнительную информацию о возможном адресе элемента массива. Для этого определяется некоторая функция $R(p)$ – ф у н к ц и я   р а с с т а н о в к и [7], которая для каждого возможного значения признака p вырабатывает значение, прямо или косвенно связанное с номером элемента, обладающего данным значением признака. В этом случае помимо массивов $\{x_i\}$, $\{\varphi(x_i)\}$, $i = 1, 2, \ldots, N$, необходимы еще два участка памяти для оглавления и списков [7].

При $N \leqslant 320$ наиболее эффективным является метод сужающихся интервалов, при $N > 320$ – метод дихотомии [8]. Применение функции расстановки позволяет сократить поиск по сравнению с дихотомическим [7]. В некоторых случаях эффективным может оказаться и перебор. Именно, если известно, что значение аргумента x изменяется монотонно, то можно начинать поиск для очередного x с найденного перед этим промежутка $[x_k, x_{k+1})$. Для таблиц с незначительной в среднем вариацией шага метод среднего шага оказывается наиболее эффективным.

Для функций двух переменных (и более) используются различные комбинации описанных выше способов разбиения интервалов изменения переменных, некоторые из них рассмотрены в [I].

Для вычисляемой функции $f(x)$ может быть выбран один из трех способов её представления:

а) алгоритм $\Gamma_1(f)$, реализующий аналитическое представление $f(x)$ с погрешностью ε_1, сложность которого $G(\Gamma_1) \leq C_1$ и объем запоминаемой информации – V_1;

б) таблица $T_{\varepsilon_2, C_2}(f)$, состоящая из такого набора u значений некоторой близкой к $f(x)$ функции $\varphi(x)$ и соответствующих им значений аргументов, что алгоритм $\Gamma_2(u)$ выбора табличного значения состоит в определении промежутка, содержащего x, и выборе $\varphi_T(x)$, при этом $G(\Gamma_2) \leq C_2$; $V_T = V_2$;

в) таблица $\widetilde{T}_{\varepsilon_3, c_3}(f)$, для которой алгоритм $\widetilde{\Gamma}_3(\mathcal{U}, \varphi)$ реализует некоторую интерполяционную формулу, такую что $G(\widetilde{\Gamma}_3) \leqslant c_3$, $V_{\widetilde{T}} = V_3$.

Как правило, величины ε_i, $\dot{c}_i$, $V_i (i=1,2,3)$ связаны между собой следующей зависимостью

$$\varepsilon_1 < \varepsilon_3 < \varepsilon_2,$$

$$c_1 > c_3 > c_2,$$

$$V_1 < V_3 < V_2.$$

Выбор одного из трех способов представления диктуется условиями задачи, в которой вычисляется $f(x)$.

При наличии свободной памяти (не обязательно только оперативной) наиболее быстрое вычисление $f(x)$ достигается с помощью таблиц без применения интерполяционных формул. Специальная структура таких таблиц позволяет сделать выборку табличного значения за I-7 операций.

§ 2. Главная последовательность табличных алгоритмов

I. Для дальнейшего удобно говорить не о таблице, а об алгоритме Γ, который по данной таблице $T_\varepsilon(f)$ вычисляет требуемые значения $f(\bar{x})$ с заданной точностью ε. Табличный алгоритм Γ будем характеризовать двумя параметрами V и $\tilde{t}$, где V - объем таблицы (затратами на программу, реализующую Γ будем пренебрегать) и $\tilde{t}$ - средний объем вычислений, необходимый для получения одного значения $f(\bar{x})$.

Пусть $\{\Gamma_i\}$ - множество эквивалентных по точности табличных алгоритмов ($i=0,1, \ldots$). С их помощью значение заданной функции $f(\bar{x})$ может быть получено с точностью, не меньшей ε. Из этого множества аналогично [9] выделим главную последовательность табличных алгоритмов (ГП), составленную из р а ц и о н а л ь н ы х алгоритмов, которые по обоим параметрам V и t лучше всех остальных, не вошедших в ГП.

Укажем два примера пополнения и оптимизации ГП:

- изменение интерполяционных формул. Можно упрощать интерполяцию (уменьшать $\tilde{t}$) и компенсировать потерю точности повышением точности таблиц (увеличивать V);

- изменение процесса выборки табличного значения путем изменения структуры таблиц. Например, в таблицах с кусочно-постоянным шагом можно уменьшать число участков с различным шагом изменения аргументов (за счет увеличения объема V) и тем самым уменьшать $\tilde{t}$.

2. Ниже описывается алгоритм построения главной последовательности (рациональных) таблиц с кусочно-постоянным шагом, упорядоченных по убыванию $\tilde{t}_T$ и возрастанию $V_T \leq e$. Абсолютная погрешность таблиц не превосходит ε и определяется заданием самого набора табличных значений. Для преобразования таблиц применяется весьма общий итерационный метод возможных направлений, используемый при решении задач оптимизации [10]. Исходным решением является таблица T^o, для которой $E_{T^o} \approx \varepsilon$, $V_{T^o} \approx e_o$ (где E_T - погрешность таблицы, e_o - минимальный при заданных точности и структуре таблиц объем).

Для следующих приближений справедливы соотношения:

$$\begin{aligned} &t_{Ti} < t_{T i-1}, \\ &e > V_{Ti} > V_{T i-1}, \\ &E_{Ti} \leq E_{T i-1} \leq \varepsilon, \\ &i = 0, 1, 2, \ldots \end{aligned} \qquad (12)$$

Процесс построения таблиц T^i обрывается на некотором значении $i = J$, для которого $V_{TJ} \approx e$. В последовательности возникающих при этом таблиц

$$T^o_{\varepsilon, c_o, V_o}(f), \quad T^1_{\varepsilon, c_1, V_1}(f), \ \ldots, \ T^J_{\varepsilon, c_J, V_J}(f), \qquad (13)$$

при заданном ограничении на число операций выборки $c_i \ (i = 0, 1, \ldots, J)$ объем V_i близок к минимальному.

Рассмотрим этот алгоритм подробнее.

Пусть $\varphi(x)$ - монотонная на $[a, b]$ функция, аппроксимирующая $f(x)$ с точностью до $\varepsilon_1 \ll \varepsilon$:

$$\max_{x \in [a,b]} |f(x) - \varphi(x)| \leq \varepsilon_1 .$$

Последовательно находим наибольшие $h_k = 2^{d_k}$ (d_k - целые), $k = 1, 2, \ldots, K$, удовлетворяющие условию

$|\varphi(x_k)-\varphi(x_k+h_k)|\leq\varepsilon_2=\varepsilon-\varepsilon_1, \quad x_1=a, \; x_{k+1}=x_k+h_k=b.$

Возникающий при этом набор чисел

$$u=(x_1,\ldots,x_{k+1};\varphi(x_1),\ldots,\varphi(x_{k+1}))$$

вместе с алгоритмом $\Gamma(u)$ выборки табличного значения $\varphi(x)=\varphi(x_k)$ при $x\in[x_k,x_{k+1})$ образует таблицу $T^o_{\varepsilon,c}(f)$, для которой

$$\rho(f,\varphi)=\max_{x\in[a,b]}|f(x)-\varphi(x)|\leq\varepsilon.$$

При заданном ограничении на шаг $h_k(h_k=2^{d_k})$ таблица $T^o(f)$ имеет минимальный объем V_{T^o}. Последовательно объединяя смежные отрезки $[x_k,x_{k+1}]$ одинаковой длины, получаем участки с постоянным шагом. Для преобразования таблицы $T^o_{\varepsilon,c}(f)$ используем вспомогательную таблицу T_1, в которой каждому участку α_i, $i=1,2,\ldots,J$ отводится ячейка памяти для записи необходимой информации – веса участка ξ_k; $\Pi(h_k)$ – порядка шага $h_k=2^{d_k}$; констант, характеризующих тип участка. Вес участка определяется соотношением

$$\xi_k=\begin{cases} n_k, \text{ если } h_{k-1}>h_k<h_{k+1}, \\ \dfrac{h_k}{h_{k-1}}\cdot n_k, \text{ если } \begin{cases} h_{k-1}<h_k>h_{k+1}, h_{k-1}=h_{k+1}, \\ \text{или } h_{k+1}<h_{k-1}<h_k, \\ \text{или } h_{k-1}<h_k<h_{k+1}, \end{cases} \\ \dfrac{h_k}{h_{k+1}}\cdot n_k, \text{ если } \begin{cases} h_{k-1}<h_{k+1}<h_k, \\ \text{или } h_{k+1}<h_k<h_{k-1}, \end{cases} \end{cases}$$

где $n_k=\dfrac{\ell_k}{h_k}$, ℓ_k – длина участка α_k.

При преобразовании таблицы T_1 используется вспомогательная таблица T_2, в каждой ячейке $a+j)$ которой содержатся величины

$$a+j) \qquad \xi_{m_j} \qquad M_{m_j} \qquad N_{m_j} \qquad (j=1,2,\ldots,J),$$

где $\xi_{m_1}\leq\xi_{m_2}\leq\ldots\leq\xi_{m_J}$; M_{m_j} – адрес величины, указывающей, к какому из смежных участков может быть присоединен участок α_{m_j}; N_{m_j} – адрес ячейки в таблице T_1, из которой взято ξ_{m_j}.

Так как исходная таблица $T^{o}_{\varepsilon,c}(f)$ - таблица с кусочно-по - стоянным шагом изменения аргумента, то минимизировать время выборки $\tilde{t}_{T^o}$ можно уменьшением числа участков α_i с различным ша - гом в пределах свободного объема памяти $\Delta V = e - V_{T^o}$. Алгоритм минимизации числа участков состоит из следующих этапов:

Этап I. Построение таблицы T_1. Проверка условия

$$\sum_k n_k < e. \tag{I4}$$

При выполнении (I4) - этап 2, иначе задача не имеет решения.

Этап 2. Построение вспомогательной таблицы T_2.

Этап 3. Преобразование таблицы T_1: в зависимости от типа очередного участка таблицы T_2 присоединяем этот участок к участку таблицы T_1.

Этап 4. Проверка условия (I4). При невыполнении его - этап 5: если T_2 вся использована, то этап 2, иначе - этап 3.

Этап 5. Для таблицы $T_{\varepsilon, t_{min}, e}(f)$ с кусочно-постоянным шагом строим таблицу констант, с помощью которых выборка табличного значения реализуется минимальным числом операций [I].

Программа, реализующая этот алгоритм на машине "Минск-22", выполнена в дипломной работе А.С. Березовой.

3. Табулирование функций может стать источником значительного сокращения времени счета и увеличения эффективности различных вычислительных алгоритмов. Использование таблиц для заданной функции связано с оценкой выигрыша во времени.

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

$$\delta = \frac{t_{сч}}{\tilde{t}_T + \tilde{t}_{табл} \; N/k}, \tag{I5}$$

где $t_{сч}$ - время вычисления функции без применения таблиц; $\tilde{t}_T$ - среднее время выборки табличного значения; $\tilde{t}_{табл}$ - среднее время вычисления одного значения функции для таблицы; N - число значений функции в таблице; k - среднее число обращений к таблице при решении задачи.

При размещении таблиц во вспомогательной памяти

$$\delta = \frac{t_{сч}}{\tilde{t}_T + \tilde{t}_{табл} \cdot N/k + \tilde{t}_в / k_1}, \tag{I6}$$

где $\tilde{t}_в$ - среднее время ввода одного слова из вспомогательной памяти, k_1 - среднее число обращений к таблице за один ввод всей таблицы.

В практических задачах [2-5] нередко $N/k \ll 1$ и $\tilde{t}_T \ll t_{cz}$, то есть $\delta \gg 1$ (I5). Это позволяет применять не только таблицы с кусочно-постоянным шагом, но и более сложные по структуре, а также размещать большие таблицы во вспомогательной памяти. Использование больших таблиц значительно упрощается при монотонном увеличении аргументов. 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. Выигрыш во времени (I6) при этом может быть увеличен совмещением некоторого числа вариантов, то есть увеличением k_1. При произвольном изменении аргумента без одновременного счета вариантов размещение таблиц во вспомогательных памятях может оказаться целесообразным только при очень больших значениях t_{cz}.

Литература

I. КОСАРЕВ Ю.Г. Примеры использования таблиц для сокращения времени счета.-"Вычислительные системы", Новосибирск,1968,вып.30, с. 46-54.

2. КОСАРЕВ Ю.Г. Моделирование неупругого рассеяния электронов методом Монте-Карло. -"Вычислительные системы", Новосибирск, 1968, вып. 30, с. 34-40.

3. КОСАРЕВ Ю.Г., НАУМОВ В.А., РОЗИН С.Г., ЯРОШЕВИЧ А.А. Решение задачи о рассеянии энергии ионизирующего излучения методом Монте-Карло на системе "Минск-222".-"Вычислительные системы",Новосибирск, 1968, вып. 30, с. 41-45.

4. БАЖИН М.А., БУБНОВ В.П., ЗАХАРОВА И.С., КОСАРЕВ Ю.Г., НЕСТЕРЕНКО В.Б. Термодинамический расчет газовых АЭС циклов на диссоциирующих газах на системе "Минск-222". -"Вычислительные системы", Новосибирск, 1968, вып. 30, с. 22-25.

5. ПЕТРОВИЧ А.И., КОСАРЕВ Ю.Г. Исследование колебательных процессов автопоездов на системе "Минск-222".-"Вычислительные системы", Новосибирск, 1968, вып. 30, с. I2-I4.

6. ГОЛОВЯШКИНА Л.В., НАХОДКИН Н.Г. Прямое моделирование неупругого рассеяния электронов. -"Вычислительные системы", Новосибирск, 1972, вып. 52, с. 5-20.

7. ЛАВРОВ С.С., ГОНЧАРОВА Л.И.Автоматическая обработка данных. Хранение информации в памяти ЭВМ. М., "Наука", 1971.

8. ДЫМАРСКИЙ Я.С., ЛОЗИНСКИЙ Н.Н., МАКУШКИН А.Т., РОЗЕН - БЕРГ В.Я., ЭРГЛИС В.Р. Справочник программиста, 1963, т. I.

9. КОСАРЕВ Ю.Г. Об информационно- и пространственно-временных преобразованиях алгоритма. Настоящий сборник, с.149-160.

10. ЗОЙТЕНДЕЙК Г. Методы возможных направлений. М., 1963.

Поступила в ред.-изд.отд.
18 апреля 1973 года