С. Я. Виленкин, В. Н. Лифшиц, Формальная модель структур данных и абстрактных алгоритмов для многопроцессорных вычислительных систем,
Автомат. и телемех., 1984, выпуск 1, 146–154

УДК 681.32:65.01

ФОРМАЛЬНАЯ МОДЕЛЬ СТРУКТУР ДАННЫХ И АБСТРАКТНЫХ АЛГОРИТМОВ ДЛЯ МНОГОПРОЦЕССОРНЫХ ВЫЧИСЛИТЕЛЬНЫХ СИСТЕМ

ВИЛЕНКИН С. Я., ЛИФШИЦ В. Н.

(Москва)

Рассматриваются особенности представления данных в многопроцессорных вычислительных системах и дается формальная модель, позволяющая рассматривать программирование на них как отображение абстрактных алгоритмов на физически реализуемые алгоритмы, т. е. как отображение абстрактных структур данных на структуру памяти ЭВМ.

1. Введение

Для обеспечения эффективного использования аппаратных средств многопроцессорных вычислительных систем необходимо исследование различных структур данных и возможностей их представления в памяти вычислительных систем различных типов, а также способов описания параллельных вычислений и организации вычислительного процесса.

Как указано в [1], возможность и степень распараллеливания процессов вычислений определяется структурированием информационной среды и возможностью одновременного изменения в ней значений многих переменных. Но для каждой информационной среды можно определить множество структур, из которого в зависимости от конкретного алгоритма, реализующего данный вычислительный метод, следует выбрать оптимальную по некоторому критерию структуру. При выборе конкретной структуры каждому элементу информационной среды (индексу) ставится в соответствие положение в структуре (ячейка памяти) независимо от значения этого элемента. Следовательно, структурированная информационная среда характеризуется множеством элементов среды, множеством элементов памяти, на которую отображена эта среда, и множеством значений, которые могут принимать элементы памяти. При этом отношения на множестве элементов среды (множестве индексов или адресов) характеризуют абстрактную структуру данных, отношения на множестве элементов памяти — ее конкретную (физическую) структуру. Соответствие между элементами среды и элементами памяти устанавливает отображение абстрактной структуры данных на ее конкретную реализацию.

Таким образом, для абстрактных алгоритмов, не связанных с конкретной физической структурой, можно получить формальную модель структур данных, которая пригодна и для однопроцессорных систем, где достаточно успешно применялась классическая модель вида (имя, значение).

При использовании многопроцессорных систем с перестраиваемой структурой, например ПС-2000 [2], важен выбор структур для реализации конкретных алгоритмов, оптимальных не только при абстрактной их реализации, но в основном при реализации на одной из допустимых структур многопроцессорной системы. В связи с этим требуется дать формальные модели допустимых структур многопроцессорных систем и исследовать отображения абстрактных структур информационных сред на физически реализуемые структуры многопроцессорной вычислительной системы.

Такой подход связан с тем, что обычно рассматриваемая в программировании модель, состоящая из пары (имя, значение), не может учитывать отображения логических структур достаточно сложных информационных

объектов (многомерных матриц и т. п.) на (физические) структуры памяти многопроцессорных систем с разделенными ОЗУ. В сущности, даже обычная двумерная матрица $A=(a_{ij})$ требует различать положение элемента a_{ij} в матрице от его расположения в памяти. В линейной ЭВМ последовательного типа матрица может быть размещена по столбцам или по строкам. Образование адреса элемента a_{ij} по паре его индексов (i, j) требует своего алгоритма при каждом из этих вариантов размещения. Размещение матриц (а тем более, многомерных матриц) в n-мерной памяти многопроцессорной вычислительной системы (при $n\geqslant 2$), т. е. отображение множества $A=(a_{ij})$ в множество ячеек памяти ОЗУ процессорных элементов, можно произвести значительно большим числом способов. Поэтому возникает задача описания логической и физической структур информационных объектов; первая структура описывает математические (абстрактные) связи элементов информационной области; вторая — их размещение в памяти многопроцессорных вычислительных систем.

Развивая предложенную в [1] концепцию графа данных и модель памяти, рассмотренную в [3], в качестве носителя информации рассмотрим индексное множество (индексированное множество в смысле [4]), на котором различные бинарные отношения определяют ту или другую информационную структуру. Затем опишем действия операторов над индексными множествами и порождаемые операторами бинарные отношения между индексными множествами. Понятие индексного множества [4] позволяет уже на уровне модели информационной области отделить логическую структуру данных (структуру индексов) от ее физической структуры (структуры ячеек), отражающей специфику строения памяти конкретной многопроцессорной системы. В данной работе рассматриваются в основном логические структуры данных.

2. Индексные множества

Информационную среду будем рассматривать как совокупность индексных множеств (т. е. множеств, все элементы которых помечены), каждое из которых выступает как агрегат некоторых элементов памяти. Признак отнесения элементов памяти к одному индексному множеству может в различных случаях быть как «алгоритмическим» (например, компоненты одного вектора или матрицы), так и «аппаратным» (например, компоненты одного модуля памяти). **Индексным множеством** будем называть пятерку вида

$$\mathfrak{A}=\langle A, X, B, I, \varphi\rangle,$$

состоящую из следующих компонент: A — множество индексов, т. е. множество идентификаторов (адресов) всех элементов памяти, объединяемых в индексное множество $\mathfrak{A}$; индексы могут быть как цифровыми (массивы), так и более сложной структуры, например кортежи для многомерных матриц, нецифровые индексы; X — множество элементов памяти (ячеек, регистров) входящих в $\mathfrak{A}$; $I: A\to X$ — взаимно-однозначное отображение идентификации, сопоставляющее каждому индексу один и только один элемент памяти, этим индексом идентифицируемый; B — множество допустимых состояний элементов памяти, т. е. множество элементарных значений, которые могут принимать входящие в индексное множество элементы памяти; φ — отображение множества элементов памяти X в множество $\mathfrak{M}(B)$ всех подмножеств B. Отображение φ позволяет определить допустимые состояния $f: X\to B$ индексного множества $\mathfrak{A}$ следующим образом: f считается допустимым лишь при выполнении условия

$$xf\in x\varphi\subseteq B \quad \forall x\in X.$$

Множество $F_{\mathfrak{A}}$ всех определенных этим условием отображений $f: X\to B$ будем называть **множеством допустимых состояний** индексного множества $\mathfrak{A}$. Множество допустимых состояний следует рассматривать только на время выполнения оператора или группы операторов, не меняющих области допустимых значений каждого их элемента памяти X. Изменение до-

пустимых состояний в пределах области осуществляется операторами и рассмотрено в разделе 4 статьи.

Отображение f следует, вообще говоря, рассматривать как частичное, так как на различных этапах работы алгоритма часть элементов памяти из множества X может не использоваться в качестве рабочей области. Однако в этом случае их состояния легко могут быть доопределены. Поэтому, не ограничивая общности, можем рассматривать отображение f как всюду определенное.

Упорядоченную пару из индексного множества $\mathfrak{A}$ и его допустимого состояния f назовем **классом операндов**

$$G_{\mathfrak{A}} = \langle \mathfrak{A}, f \rangle.$$

Следует отметить, что в приведенной модели информационной среды в виде индексного множества $\mathfrak{A}$ задание отношений на множестве индексов A определяет абстрактную структуру данных, отношения на множестве элементов X — конкретную информационную структуру (физическую реализацию), а пара отображений (I, I^{-1}) устанавливает связь между абстрактной структурой данных и физической структурой, отвечающей реализации информационной среды.

Вид конкретного отображения I определяется как структурой множества индексов A, так и структурой памяти X. Примеры таких отображений подробно исследованы в ряде работ [5—7]. В разделе 3 этот вопрос будет рассмотрен применительно к многомерным матрицам.

Последнее обстоятельство позволяет рассматривать понятия согласованности абстрактной и физической структур, а также взаимно-индуцированные структуры.

3. Информационные структуры и отношения на индексных множествах

Для формального описания информационных структур введем на множестве индексов A набор бинарных отношений.

П-отношение есть бинарное отношение между элементами множества, удовлетворяющее следующим условиям:

а) сильная антисимметричность: $a<b \Rightarrow \neg(b<a)$;

б) антирефлексивность: $\neg(a<a)$;

в) ацикличность: $\forall n>2$

$$a_1<a_2<\ldots<a_n \Rightarrow \neg(a_n<a_1).$$

Этому отношению отвечают древовидные структуры и ациклические ориентированные графы.

ПЦ-отношение есть бинарное отношение, удовлетворяющее свойствам а), б) П-отношения, т. е. допускающее циклические замыкания. Этому отношению отвечают графы с циклами.

ЛП-отношением будем называть П-отношение $<$, удовлетворяющее следующим дополнительным условиям:

а) антитранзитивность: $\forall n>2$

$$a_1<a_2<\ldots<a_n \Rightarrow \neg(a_1<a_n);$$

б) начальный элемент: существует единственный $a_0 \in A$, такой, что

$$\forall b \in A \setminus \{a_0\} \quad \neg \ (b<a_0);$$

в) конечный элемент: существует единственный $\bar{a} \in A$, такой, что $a_0 \neq \bar{a}$ и $\forall b \in A \setminus \{\bar{a}\} \ \neg \ (\bar{a}<b)$;

г) непосредственное следование: $\forall b \in A \setminus \{\bar{a}\}$ существует единственный $b' \in A$, такой, что $b<b'$;

д) непосредственное предшествование: $\forall b \in A \setminus \{a_0\}$ существует единственный $b'' \in A$, такой, что $b''<b$.

Аналогично ЛПЦ-отношением будем называть ПЦ-отношение, удовлетворяющее дополнительно «условиям линейности» а) — д) ЛП-отношения.

Легко видеть, что ЛП-отношения описывают линейные структуры на множестве A без замыканий, а ЛПЦ-отношения — линейные структуры с замыканиями.

Перейдем теперь к формальному описанию структур типа многомерных матриц. Как уже указывалось, описание m-мерных матриц с помощью П-отношения (т. е. рассмотрение их как ориентированных графов) скрывает регулярную матричную структуру этого объекта. Поэтому представляется более целесообразным следующий способ описания.

Введем вначале еще один тип отношений.

Будем называть n — ЛП-отношением пару $(<, \rho)$ из П-отношения $<$ и разбиения (отношения эквивалентности) ρ множества A на n непересекающихся подмножеств $A_1, \ldots, A_n$, состоящих из равного числа элементов:

$$A = \bigcup_{i=1}^{n} A_i,\ A_i \cap A_j = \phi \text{ при } i \neq j,\ |A_i| = |A_j|$$

$$(i=1, \ldots, n;\ j=1, \ldots, n).$$

При этом дополнительно должны выполняться следующие два условия:

а) на каждом из A_i $(i=1, \ldots, n)$ отношение $<$ является ЛП-отношением;

б) никакие два элемента из различных подмножеств A_i, A_j $(i \neq j)$ несравнимы по отношению $<$:

$$a_i \in A_i,\quad a_j \in A_j,\quad i \neq j \Rightarrow \neg(a_i < a_j) \& \neg(a_j < a_i).$$

Легко видеть, что n — ЛП-отношение может быть задано на множестве A, число элементов которого кратно n.

Рассмотрим теперь, как использовать серии n — ЛП-отношений (с соответственным образом подобранными n) для описания взаимосвязей элементов памяти в структуре многомерной матрицы.

Рассмотрим вначале двумерную (прямоугольную) матрицу размера $k_1 \times k_2$:

$$\hat{A} = \begin{pmatrix} a_{11} \ldots\ldots a_{1k_2} \\ \ldots\ldots\ldots \\ a_{k_1 1} \ldots\ldots a_{k_1 k_2} \end{pmatrix}.$$

Через $A = \{a_{ij} \mid i=1, \ldots, k_1;\ j=1, \ldots, k_2\}$ обозначим множество идентификаторов ее элементов (множество индексов). Определим k_1 — ЛП-отношение $(<_1, \rho_1)$ и k_2 — ЛП-отношение $(<_2, \rho_2)$ так, что

1) ρ_1 есть разбиение $A = A_1^1 \cup \ldots \cup A_{k_1}^1$, где $A_i^1 = \{a_{i1}, \ldots, a_{ik_2}\}$ — i-я строка матрицы $\hat{A}$ $(i=1, \ldots, k_1)$;

2) ρ_2 есть разбиение $A = A_1^2 \cup \ldots \cup A_{k_2}^2$, где $A_j^2 = \{a_{1j}, \ldots, a_{k_1 j}\}$ — j-й столбец матрицы $\hat{A}$ $(j=1, \ldots, k_2)$;

3) отношение $<_1$ определено на A следующим образом:

$$a_{11} <_1 a_{12} <_1 \ldots <_1 a_{1k_2},$$
$$\ldots\ldots\ldots\ldots$$
$$a_{k_1 1} <_1 a_{k_1 2} <_1 \ldots <_1 a_{k_1 k_2},$$

т. е. упорядочивает элементы в пределах каждой строки;

4) отношение $<_2$ определено на A следующим образом:

$$a_{11} <_2 a_{21} <_2 \ldots <_2 a_{k_1 1},$$
$$\ldots\ldots\ldots\ldots$$
$$a_{1k_2} <_2 a_{2k_2} <_2 \ldots <_2 a_{k_1 k_2}.$$

Легко видеть, что пары $(<_1, \rho_1)$ и $(<_2, \rho_2)$ удовлетворяют условиям k — ЛП-отношений. Кроме того, они удовлетворяют следующим условиям согласованности (условия клетки):

а) для любых $a, b, c, d \in A$

$$a <_1 b,\ a <_2 c,\ c <_1 d \Rightarrow b <_2 d;$$

б) для любых $a, b, c, d \in A$

$$a <_1 b,\ b <_2 d,\ c <_1 d \Rightarrow a <_2 c;$$

в) для любых $a, b, c, d \in A$

$$a <_1 b,\ a <_2 c,\ b <_2 d \Rightarrow c <_1 d;$$

г) для любых $a, b, c, d \in A$

$$a <_2 c,\ b <_2 d,\ c <_1 d \Rightarrow a <_1 b;$$

В работах [5, 6] предложен прием скашивания n-мерных матриц (skewing), предназначенный для преобразования матриц к виду, удобному для размещения в модулях памяти, обеспечивающему бесконфликтный выбор данных из памяти (по строкам, столбцам, диагоналям). В соответствии с [6] скашивание n-мерной матрицы выполняется в два этапа:

«разрежение» матрицы по всем измерениям посредством сопоставления элементу $a_{i_1 \ldots i_n}$ номера $N_{i_1 \ldots i_n} = \Delta_1 (i_1 - 1) + \ldots + \Delta_n (i_n - 1)$, где $1 \leqslant i_j \leqslant k_j$ $(j = 1, \ldots, n)$; $\Delta_1 \ldots \Delta_n$ — коэффициенты скашивания;

запись элемента $a_{i_1 \ldots i_n}$ в модуль памяти с номером $N_{i_1 \ldots i_n} (\mathrm{mod}\, M)$, где M — общее количество модулей памяти. При этом порядок записи элементов принимается лексикографическим по их индексу.

Представление матрицы в скошенном виде можно задать как взаимно-однозначное отображение $A_{k_1 \times \ldots \times k_n}$ в множество $B_{l_1 \times \ldots \times l_m}$, удовлетворяющее следующему условию: если $a,\ a' \in A_{k_1 \times \ldots \times k_n}$ и $a < a'$, где $< \in \{<_1, \ldots, <_n\}$, то существует последовательность элементов $b_1, \ldots, b_p \in B_{l_1 \times \ldots \times l_m}$, такая, что $a \varphi <^1 b_1 <^2 \ldots <^p b_p <^{p+1} a' \varphi$, где $<^j \in \{<_1, \ldots, <_m\}$ $(j = 1, \ldots, p+1)$. При этом не допускается условие $<^j = <_1$ для всех $j = 1, \ldots, p+1$.

4. Алгоритмы над индексными множествами

Информационная область (область действия) любого алгоритма представляет собой некоторую совокупность индексных множеств. При этом описание действия алгоритма над индексными множествами без фиксации для каждого индексного множества его начального допустимого состояния соответствует абстрактной схеме алгоритма, как арифметико-логической функции переработки информации. Если же зафиксировать для каждого входящего в информационную область индексного множества одно из его допустимых состояний в качестве начального, то получим алгоритм, работающий над конкретными классами операндов.

Рассмотрим теперь формально действие операторов над совокупностью индексных множеств

$$\mathfrak{B} = \{\mathfrak{A}_1, \ldots, \mathfrak{A}_t\},\ \mathfrak{A}_i = \langle A_i, X_i, B_i, I_i, \varphi_i \rangle \quad (i = 1, \ldots, t),$$

причем $A_i \cap A_j = \emptyset$ при $i \neq j$.

Выделим операторы трех типов: оператор-преобразователь, осуществляющий переработку информации; оператор-распознаватель, осуществляющий наряду с переработкой информации изменение порядка выполнения операторов; оператор адресной арифметики, выполняющий операции преобразования элементов множеств индексов A_i (для вычисления адресов операндов операторов двух первых типов при помощи отображения I).

Будем предполагать, как и в случае элементов памяти индексных множеств, что в пределах каждого алгоритма все операторы некоторым образом идентифицированы, т. е. заданы множество $\mathscr{A}$ идентификаторов операторов алгоритма и набор $\mathscr{O}$ операторов алгоритма совместно с взаимно-однозначным отображением идентификации. Таким образом, идентифицированная совокупность операторов алгоритма рассматривается как упорядоченная тройка

$$\mathscr{P} = \langle \mathscr{A}, \mathscr{O}, \mathscr{I} \rangle,$$

где

$$|\mathscr{A}| = |\mathscr{O}|,\quad \mathscr{I} : \mathscr{A} \to \mathscr{O}.$$

Рассмотрим более подробно структуру операторов каждого из трех указанных типов.

Оператор адресной арифметики над $\mathfrak{B}$ представляет собой упорядоченную тройку вида

$$o_A=\langle SI(\mathfrak{B}),\ SO(\mathfrak{B}),\ \psi\rangle,$$

где $SI(\mathfrak{B})$ — область входа; $SO(\mathfrak{B})$ — область выхода; ψ — функция преобразования информации [8].

$SI(\mathfrak{B})$ и $SO(\mathfrak{B})$ представляют собой некоторые подмножества множества $\bar{A}=\bigcup_{i=1}^{t}\mathfrak{M}(A_i)$, где $\mathfrak{M}(A_i)$ — множество всех подмножеств (булеан) множества A_i $(i=1,\ldots,t)$. При этом требуется выполнение следующего дополнительного условия. Если $a\in SI(\mathfrak{B})$ (или $a\in SO(\mathfrak{B})$) и, таким образом, $a\in A_k$ для некоторого индекса $1\leqslant k\leqslant t$, то

$$(aI_k)f_k\in\bigcup_{j=1}^{t}A_j,$$

т. е. $(aI_k)f_k\in(aI_k)\varphi_k\subseteq B_k\cap\left(\bigcup^{t}A_j\right)$

для любого допустимого состояния f_k индексного множества $\mathfrak{A}_k$.

Другими словами, к области входа и выхода оператора адресной арифметики могут относиться лишь идентификаторы (адреса) таких элементов памяти, допустимыми состояниями которых являются идентификаторы (адреса) некоторых элементов памяти индексных множеств из совокупности $\mathfrak{B}$. Таким образом, операторы адресной арифметики выполняют преобразования над идентификаторами элементов памяти индексных множеств совокупности $\mathfrak{B}$.

Функция преобразования информации ψ представляет собой (n, m)-функцию, т. е. функцию n переменных с m значениями, где $n=|SI(\mathfrak{B})|$, $m=|SO(\mathfrak{B})|$. Так как использование таких функций предполагает строгую упорядоченность аргументов, дополнительно будем полагать, что на множествах $SI(\mathfrak{B})$ и $SO(\mathfrak{B})$ заданы ЛП-отношения $<_I$ и $<_O$, позволяющие рассматривать входы и выходы оператора O_A как упорядоченные последовательности аргументов.

Следует отметить, что типичным оператором адресной арифметики является оператор, обеспечивающий переход от идентификатора текущего элемента индексного множества к идентификатору следующего элемента в смысле транзитивного замыкания заданного на индексном множестве П-, ПЦ-, ЛП-, ЛПЦ-отношения.

Оператор-преобразователь над $\mathfrak{B}$ представляет собой упорядоченную пятерку вида

$$o_P=\langle PI(\mathfrak{B}),\ KI(\mathfrak{B}),\ PO(\mathfrak{B}),\ KO(\mathfrak{B}),\ \psi\rangle,$$

где $PI(\mathfrak{B})$, $PO(\mathfrak{B})$ — прямые области входа и выхода оператора; $KI(\mathfrak{B})$ — косвенная область входа; $KO(\mathfrak{B})$ — косвенная область выхода этого оператора; ψ — функция преобразования оператора.

$PI(\mathfrak{B})$, $KI(\mathfrak{B})$, $PO(\mathfrak{B})$, $KO(\mathfrak{B})$, как и в операторах адресной арифметики, представляют собой конечные подмножества множества $\bar{A}=\bigcup_{i=1}^{t}\mathfrak{M}(A_i)$.

Следует отметить, что содержимое (состояние) косвенных областей входа/выхода обычно формируется операторами адресной арифметики. Поэтому рассмотренная схема адресации является достаточно общей. Дополнительно требуется выполнение следующего условия. Если $a\in KI(\mathfrak{B})$ (или $a\in KO(\mathfrak{B})$) и, таким образом, $a\in A_k$ для некоторого $1\leqslant k\leqslant t$, то

$$(aI_k)f_k\in\bigcup_{j=1}^{t}A_j,$$

т. е. $(aI_k)f_k \in (aI_k)\varphi_k \subseteq B_k \cap (\bigcup_{j=1}^{t} A_j)$ для любого допустимого состояния f_k индексного множества $\mathfrak{A}_k$.

Другими словами, косвенные области входа и выхода оператора-преобразователя могут содержать лишь такие элементы памяти, любым допустимым состоянием которых являются идентификаторы (адреса) некоторых элементов памяти индексных множеств совокупности $\mathfrak{B}$. Кроме того, должны выполняться условия

$$PI(\mathfrak{B}) \cap KI(\mathfrak{B}) = \phi.$$

Функция преобразования информации ψ представляет собой (n, m)-функцию, где $n = |PI(\mathfrak{B})| + |KI(\mathfrak{B})|$, $m = |PO(\mathfrak{B})| + |KO(\mathfrak{B})|$.

Как и для случая операторов адресной арифметики, на множествах

$$PI(\mathfrak{B}) \cup KI(\mathfrak{B}), \quad PO(\mathfrak{B}) \cup KO(\mathfrak{B})$$

предполагаем заданными ЛП-отношения $<_I$ и $<_O$, позволяющие рассматривать входы и выходы оператора o_P как упорядоченные последовательности аргументов.

Оператор-распознаватель над $\mathfrak{B}$ представляет собой оператор-преобразователь совместно с $(n, 1)$-функцией перехода (выбора оператора) θ:

$$o_R = (o_P, \theta),$$

областью значений которого является множество $\mathfrak{M}(\mathscr{A})$ подмножеств идентификаторов операторов алгоритма. Оператор-распознаватель выполняет преобразование информации так же, как оператор-преобразователь, и дополнительно указывает идентификаторы некоторых операторов того же алгоритма.

Рассмотрим теперь формально процесс выполнения операторов каждого из трех типов. С этой целью зафиксируем для каждого индексного множества одно из допустимых состояний. Такие индексные множества можно рассматривать как классы операндов. При этом в случае пересечения элементов памяти индексных множеств дополнительно требуется выполнение условия согласованности допустимых состояний: если $x \in X_i \cap X_j$, где $\mathfrak{A}_i = \langle A_i, X_i, B_i, I_i, \varphi_i \rangle$ и $\mathfrak{A}_j = \langle A_j, X_j, B_j, I_j, \varphi_j \rangle$, то $xf_i = xf_j$.

Для операторов адресной арифметики меняется состояние лишь элементов памяти, идентификаторы которых входят в область выхода оператора. Корректность выполнения такого преобразования (заключающаяся в том, что идентификаторы, которые являются состояниями элементов памяти, отвечающих области входа оператора, преобразуются в идентификаторы, которые являются состояниями элементов памяти, отвечающих области выхода) обеспечивается дополнительными условиями, сформулированными при определении оператора адресной арифметики.

Перейдем теперь к описанию действия операторов-преобразователей. Здесь следует отметить две особенности:

ограничения на допустимые состояния, аналогичные ограничениям для операторов адресной арифметики, относятся лишь к косвенным областям входа и выхода оператора;

результирующие входная и выходная области определяются двумя путями, а не одним (как в случае операторов адресной арифметики), посредством прямой и косвенной адресации.

Пусть, как и ранее, заданы классы операндов $\langle \mathfrak{A}_i, f_i \rangle$ $(i = 1, \ldots, t)$ на семействе индексных множеств $\mathfrak{B}$, удовлетворяющие условию согласованности исходных состояний f_i. На основе этих классов операндов сформируем результирующие области входов и выходов оператора-преобразователя $o_P = \langle PI(\mathfrak{B}), KI(\mathfrak{B}), PO(\mathfrak{B}), KO(\mathfrak{B}), \psi \rangle$. В случае, если $KI(\mathfrak{B}) \neq \phi$, положим $SI(\mathfrak{B}, \bar{f}) = PI(\mathfrak{B}) \cup \{(aI_l)f_l \mid a \in KI(\mathfrak{B})$, если $a \in A_l$, $1 \leqslant l \leqslant t\}$. Иначе положим $SI(\mathfrak{B}, \bar{f}) = PI(\mathfrak{B})$. В случае, если $KO(\mathfrak{B}) \neq \phi$, положим $SO(\mathfrak{B}, \bar{f}) = PO(\mathfrak{B}) \cup \{(aI_l)f_l \mid a \in KO(\mathfrak{B})$, если $a \in A_l$, $1 \leqslant l \leqslant t\}$. Иначе положим $SO(\mathfrak{B}, \bar{f}) = PO(\mathfrak{B})$. Здесь $\bar{f} = \langle f_1, \ldots, f_t \rangle$ — совокупность согласованных со-

стояний индексных множеств $\mathfrak{B}$. Обозначим результирующие области входа и выхода $SI(\mathfrak{B}, \bar{f})$ и $SO(\mathfrak{B}, \bar{f})$, упорядоченные в соответствии с ЛП-отношениями $<_I$ и $<_O$, через $\langle a_1, \ldots, a_n\rangle$ и $\langle \bar{a}_1, \ldots, \bar{a}_m\rangle$ соответственно. Здесь необходимо отметить, что множества $\{(aI_l)f_l | a \in KI(\mathfrak{B})$, если $a \in A_l$, $1 \leqslant l \leqslant t\}$ и $\{(aI_l)f_l | a \in KO(\mathfrak{B})$, если $a \in A_i$, $1 \leqslant l \leqslant t\}$ находятся в однозначном соответствии с множествами $KI(\mathfrak{B})$ и $KO(\mathfrak{B})$. Поэтому ЛП-отношения $<_I$ и $<_O$ естественно переносятся с множеств $PI(\mathfrak{B}) \cup KI(\mathfrak{B})$ и $PO(\mathfrak{B}) \cup KO(\mathfrak{B})$ на множества $SI(\mathfrak{B}, f)$ и $SO(\mathfrak{B}, f)$.

Рассмотрим теперь, как преобразуется оператором-преобразователем o_P класс операндов $\langle \mathfrak{A}_i, f_i\rangle$ в класс операндов $\langle \mathfrak{A}_i, f_i'\rangle$. Как и выше, изменение состояния осуществляется лишь для элементов памяти, затрагиваемых областью выхода. При этом в отличие от операторов адресной арифметики результирующие области входа и выхода складываются из соответствующих прямых областей и состояний элементов памяти, указанных в косвенных областях входа и выхода.

Операторы-распознаватели, как было указано, состоят из операторов-преобразователей, а также функций перехода θ. Эта функция не влечет изменения состояний индексных множеств, но указывает некоторое подмножество операторов алгоритма. Выбор этого подмножества идентификаторов осуществляется функцией θ в зависимости от состояний элементов памяти, определяемых результирующей областью входа данного оператора. В обозначениях, принятых выше для оператора преобразователя o_P, действие оператора-распознавателя o_R дополнительно описывается следующим образом:

$$\theta((a_1 I_{k_1})f_{k_1}, \ldots, (a_n I_{k_n})f_{k_n}).$$

Это интерпретируется как указание семейства операторов, которые должны быть выполнены сразу после оператора o_R. Другими словами, пару $(\theta, \langle a_1, \ldots, a_n\rangle)$ можно интерпретировать как отображение

$$\langle \mathfrak{A}_1, f_1\rangle \times \ldots \times \langle \mathfrak{A}_t, f_t\rangle \to \mathfrak{M}(\mathscr{A}),$$

т. е. отображение совокупностей групп операндов по всем индексным множествам из $\mathfrak{B}$ в множестве всех подмножеств множества $\mathscr{A}$.

Введение в рассмотрение операторов трех указанных типов позволяет сделать описание алгоритма «машинно-ориентированным» со следующих позиций. При разработке алгоритмов для многопроцессорных систем возникает потребность в учете, наряду с операциями логико-арифметического типа, операций подготовки данных (вычислений адресов). Это обусловливает появление специальных операторов адресной арифметики и выделение в явном виде в операторах-преобразователях и операторах-распознавателях косвенных областей входа и выхода. При разработке алгоритмов для многопроцессорных систем важную роль играет не только высота алгоритма (в смысле [9]) без учета операторов формирования адресов данных, но и итоговая минимизация высоты алгоритма. Это связано с тем, что операторы адресной арифметики часто приобретают значительный вес в алгоритмах.

Если операторы-преобразователи и операторы-распознаватели составляют инвариантную часть алгоритма, то операторы адресной арифметики существенно зависят от структуры индексных множеств. Преобразование структур индексных множеств может существенно сократить как число, так и сложность этих операторов.

Рассмотренный подход к описанию индексных множеств и алгоритмов позволяет рассматривать программирование как отображение абстрактных алгоритмов на физически реализуемые алгоритмы, т. е. как отображение структур индексных множеств абстрактных алгоритмов на физическую структуру памяти многопроцессорной вычислительной системы.

ЛИТЕРАТУРА

1. *Глушков В. М., Капитонова Ю. В., Летичевский А. А.* Теория структур данных и синхронные параллельные вычисления.— Кибернетика, 1976, № 6, с. 2—15.

2. *Трапезников В. А., Прангишвили И. В., Реванов В. В.* Экспедиционные геофизические вычислительные комплексы на базе многопроцессорной ЭВМ ПС-2000.— Приборы и системы управления, 1981, № 2, с. 29—31.
3. *Лифшиц В. Н., Садовский Л. Е.* Алгебраические модели вычислительных машин.— Усп. матем. наук, 1972, т. XXVII, вып. 3 (165), с. 79—125.
4. *Бервтисс А. Т.* Структуры данных. М.: Статистика, 1974.
5. *Lawrie D. H.* Access and Alugnment of Data in Array Processors.— IEEE Trans. Comput., Dec. 1975, v. C-24, № 12, p. 1145—1155.
6. *Kuck D. J.* ILLIAC-IV. Software and Application Programming.— IEEE Trans. Comput., Dec. 1968, v. C-17, № 8, p. 758—770.
7. *Гришина И. М.* Применение метода динамического программирования к оптимизации задач статистической обработки данных на многопроцессорных вычислительных системах.— В кн.: Сб. трудов. Вып. 19. М.: Ин-т проблем управления, 1978, с. 79—84.
8. *Дейкстра Э.* Дисциплина программирования. М.: Мир, 1978.
9. *Фадеева В. Н., Фадеев Д. К.* Параллельные вычисления в линейной алгебре.— Кибернетика, 1977, № 6, с. 28—40.

Поступила в редакцию
17.III.1982

A FORMAL MODEL OF DATA STRUCTURES AND ABSTRACT ALGORITHMS FOR MULTI-PROCESSOR COMPUTER SYSTEMS

VILENKIN S. Ya., LIFSHITS V. N.

The article is concerned with specifics of data representation in multiprocessor computing systems and provides a formal models there the programming is regarded as the mapping of abstract algorithms on physically implementable algorithms, or as the mapping of abstract data structures on the computer memory structure.