ЛИТЕРАТУРА

1. **Иванников В. П., Бурдонов И. Б., Кузнецов С. Д. и др.** Технологический подход к организации информационных систем.— В кн.: Труды III Всесоюз. конф. «Банки данных». Таллин: Политехн. ин-т, 1985.
2. **Ritchie D. M., Thompson K.** The UNIX time-sharing system.— Commun. of the ACM, 1974, v. 17, N 7, p. 365—375.
3. **Бяков А. Ю., Иванников В. П., Кузнецов С. Д.** Архив центрального процессора.— М., 1980. (Препринт/ИТМиВТ; 5).
4. **McKusick M. K. e. a.** A fast file system for UNIX.— ACM Trans. on Computer Systems, 1984, v. 2, N 3, p. 181—197.
5. **RSX-11** Input/Output Operations Reference Manual, 1979, N 12.

Поступила в редакцию 4 ноября 1985 г.

УДК 519.682

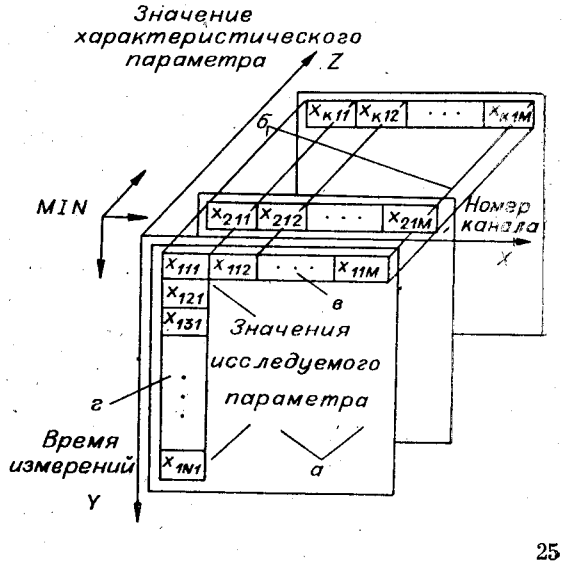
В. А. ВИТТИХ, П. О. СКОБЕЛЕВ

(Куйбышев)

ФУНКЦИОНАЛЬНЫЕ ЯЗЫКОВЫЕ СРЕДСТВА ИНТЕРАКТИВНОГО ВЗАИМОДЕЙСТВИЯ В АСНИ

Одним из путей повышения эффективности применения автоматизированных систем научных исследований (АСНИ) является разработка языковых средств для интерактивного взаимодействия исследователей с ЭВМ на этапе проведения экспериментов.

Необходимость применения языковых средств для АСНИ вызывается обычно требованиями обеспечить возможность оперативного изменения последовательности выполнения специализированных операций по сбору, обработке и отображению данных. Особенности построения указанных средств в значительной степени связаны с технологией проведения эксперимента. В ряде случаев эта технология требует сохранения информации о параметрах эксперимента и порядке их изменения, что предопределяет многомерность данных, поступающих в ЭВМ (рисунок). Операции при этом выполняются, как правило, для «сечений» (по времени и характеристическому параметру, отдельному каналу и т. п.) или фрагментов «сечений» (например, для заданного диапазона изменений параметра и интервала времени) данных. Число возможных направлений для построения таких «сечений» (и применения программ обработки, например *MIN*, на рисунке) обычно велико для многопараметрических объектов исследования, характеризуемых однородными (по физической природе) последовательностями данных измерений. При этом в процессе проведения экспериментов (особенно на ранних стадиях исследований) состав задаваемых (переменных) параметров и последо-



Представление данных эксперимента в координатах задаваемых параметров и «сечения» данных:

а — при выбранных значениях характеристического параметра; *б* — времени измерений; *в* — параметра и времени; *г* — параметра и номера канала

вательность их изменения могут существенно меняться; соответствующим образом изменяется и размерность обрабатываемых данных.

В целом при построении программ эксперимента это приводит к необходимости разбиения данных на различные по числу уровней вложенности и размеру блоки (выделение которых может быть также связано, например, с ограничениями на доступные ресурсы) и разработке средств для разделения или объединения таких блоков. Кроме того, в программах обработки получаемых последовательностей становится необходимым постоянное перевычисление индексов указанных блоков, характеризующих вложенность динамически создаваемых последовательностей, их физическое размещение и объем. В этих условиях программирование операций обработки с использованием традиционных (операторных) языков требует значительных усилий и, как правило, повторяется заново для каждого эксперимента.

В этой связи для создания языковых средств АСНИ, обладающих отмеченной спецификой, в работе предлагается использовать функциональный подход. Особенности его связаны с рассмотрением программ как функций, формируемых с помощью операций суперпозиции и рекурсии из некоторого числа заданных элементарных функций. За основу языка был выбран функциональный язык [1], обеспечивающий возможность представления многомерных данных эксперимента с помощью структурированных последовательностей произвольной вложенности; возможность структурного преобразования получаемых данных для применения типовых программ обработки на любом уровне вложенности; независимость этих программ от размерности обрабатываемых данных. Применение указанного подхода обеспечивает также алгебраические свойства и крайне простую семантику языка (отсутствуют переходы, индексы, ограничения на передачу параметров и т. п.), что упрощает разработку языковой системы и ее использование. Рассмотрим подробнее структуру и возможный состав операций базового языка.

Элементарные последовательности данных в языке имеют типы: вещественный, целый, байтовый — и размещаются (для сокращения объема структурных описаний данных) линейно в памяти ЭВМ. Из элементарных последовательностей вне зависимости от порядка их размещения в памяти образуются составные последовательности. Предельными случаями являются: Ф — пустая последовательность — и $\perp$ — неопределенность, вырабатываемая функциями при неверно заданных входных данных или аварийном завершении. Последовательности выделяются далее скобками: «‹» и «›».

Операции над объектами задаются с помощью функций, являющихся одноаргументными (аргументы объединены в последовательность) и строгими (если один из аргументов содержит неопределенность, результат также есть неопределенность).

Алгоритмический уровень языка составляют функциональные формы, определяющие схемы программ или способы применения функций к данным. К числу базовых форм относятся (здесь F_i обозначает произвольную функцию (или форму), x_i — произвольный объект (или последовательность) данных, «:» — оператор применения функции к объекту);

$\mathrm{SEQ}[F_N - F_{N-1} - \ldots - F_1] : x$ — последовательное применение функций, результат — $F_N : (F_{N-1} : (\ldots (F_1 : x) \ldots))$;

$\mathrm{PAR}[F_1, F_2, \ldots, F_N] : x$ — параллельное применение функций, результат — $\langle F_1 : x, F_2 : x, \ldots, F_N : x \rangle$;

$\mathrm{IF}[F_1 \rightarrow F_2; F_3] : x$ — применение функций F_2 или F_3 по условию F_1 (F_1 — функция-предикат, вырабатывающая объекты T (истина) или F (ложь)), результат — $F_2 : x$ или $F_3 : x$ соответственно.

Для блочно-итерационной и поблочной обработок, характерных для рассматриваемых систем, используются формы:

$\mathrm{ALL}[F] : x$ — применение функции F ко всем элементам входной последовательности $x = \langle x_1, x_2, \ldots, x_N \rangle$, результат есть $\langle F : x_1, F : x_2, \ldots, F : x_N \rangle$;

ITR$[F]:x$ — итерационное применение функции F к элементам входной последовательности, результат вырабатывается последовательным применением правила

$$\mathrm{ITR}[F]:x = F:\langle x_1,\ \mathrm{ITR}[F]:\langle x_2,\ \ldots,\ x_N\rangle\rangle,$$

причем $\mathrm{ITR}[F]:\langle x\rangle = x$.

Дополнительные параметры форм обычно либо параметры алгоритма формы (применение «слева» или «справа», кратность, шаг по аргументам и т. п.), либо параметры управления памятью (ограничивающие, например, копирование структурных описаний или элементарных последовательностей).

Для работы в реальном времени предназначены формы: $\mathrm{SQR} * P_1$, P_2, P_3, $P_4[F_1,\ F_2]:x$ — периодическое выполнение функции F_2 по шкале F_1 (по времени, перемещению и т. п.), где P_1 — общее число циклов выполнения F_2; P_2 — шаг по шкале; P_3 — допустимая погрешность обработки шага; P_4 — признак использования абсолютной (относительной) шкалы; функция F_2 выполняется, если значение величины, вырабатываемой по F_1, изменяется (с учетом допуска) на величину шага, результат есть $F_2:(F_2:(\ldots(F_2:x)\ldots))$; $\mathrm{SQL} * P_1[F_1 \to F_2]:x$ — выполнение F_2 по условию F_1, где P_1 — общее число обрабатываемых событий; F_2 выполняется в случае, если непрерывно проверяемое или ожидаемое условие F_1 вырабатывает T (например, при появлении синхроимпульса, при выходе сигнала за установленный диапазон и т. д.), результатом также является $F_2:(F_2:(\ldots(F_2:x)\ldots))$.

Для основных форм остаются справедливыми законы алгебры функциональных программ [2], позволяющие пользователю анализировать свойства программ, осуществлять их преобразования и т. п. Функции F и G при этом называются эквивалентными, если для любого x $F:x$ и $G:x$ определены и равны либо не определены. Ниже приводится несколько наиболее общих законов, применимых для любых функций F, G, H, P, Q, ... и любых объектов x:

$$\mathrm{SEQ}[\mathrm{PAR}[F,\ G]-H] = \mathrm{PAR}[\mathrm{SEQ}[F-H],\ \mathrm{SEQ}[G-H]]; \qquad (1)$$

$$\mathrm{SEQ}[\mathrm{ITR}[F]-\mathrm{PAR}[G_1,\ \ldots,\ G_N]] =$$

$$= \mathrm{SEQ}[F-\mathrm{PAR}[G_1,\ \mathrm{SEQ}[\mathrm{ITR}[F]-\mathrm{PAR}[G_2,\ \ldots,\ G_N]]]]; \qquad (2)$$

$$\mathrm{SEQ}[\mathrm{IF}[P \to F;\ G]-H] = \mathrm{IF}[\mathrm{SEQ}[P-H] \to \mathrm{SEQ}[F-H];\ \mathrm{SEQ}[G-H]]; \qquad (3)$$

$$\mathrm{SEQ}[H-\mathrm{IF}[P \to F;\ G]] = \mathrm{IF}[P \to \mathrm{SEQ}[H-F];\ \mathrm{SEQ}[H-G]]; \qquad (4)$$

$$\mathrm{PAR}[F,\ \mathrm{IF}[P \to G;\ H]] = \mathrm{IF}[P \to \mathrm{PAR}[F,\ G];\ \mathrm{PAR}[F,\ H]]; \qquad (5)$$

$$\mathrm{IF}[P \to \mathrm{IF}[Q \to R;\ S];\ \mathrm{IF}[T \to U;\ V]] =$$

$$= \mathrm{IF}[\mathrm{IF}[P \to Q;\ T] \to \mathrm{IF}[P \to R;\ U];\ \mathrm{IF}[P \to S;\ V]]. \qquad (6)$$

На основе форм и элементарных функций вводятся новые функции. Определение новой функции в общем случае имеет вид

$$\mathrm{DEF}\ F = E(F,\ G_1,\ G_2,\ \ldots,\ G_N),$$

где E — функциональное выражение, построенное из форм; G_1, G_2, ..., G_N — параметры форм; F — определяемый (возможно, рекурсивно) функциональный символ. В процессе вычислений $F:x$ символ F заменяется своим определением вне зависимости от контекста применения.

Среди элементарных функций наименее изменяемую часть составляют функции проверки и преобразования структуры последовательностей, позволяющие выделить необходимые «сечения» данных или объединить их в требуемом порядке, например:

NULL (проверка на равенство Ф): $\mathrm{NULL}:\Phi = T$, $\mathrm{NULL}:x = F$;

KDRS (полный выбор): $\mathrm{KDRS}:x = x$ (сохраняет последовательность);

KDR N или KDR $N:M$ (выбор N-го элемента последовательности или элементов от N до M): $\mathrm{KDR}\ L:\langle x_1 \ldots x_L \ldots x_N\rangle = x_L$ (иначе Ф);

TL (выбор остатка): $TL: \langle x_1 \dots x_N \rangle = \langle x_2 \dots x_N \rangle$;
APL (присоединение слева): $APL: \langle x \langle y_1 \dots y_N \rangle \rangle = \langle x y_1 \dots y_N \rangle$;
APR (присоединение справа): $APR: \langle \langle y_1 \dots y_N \rangle x \rangle = \langle y_1 \dots y_N x \rangle$;
DSL (распределение слева): $DSL: \langle x \langle y_1 \dots y_N \rangle \rangle = \langle \langle x y_1 \rangle \dots \langle x y_N \rangle \rangle$;
DSR (распределение справа): $DSR: \langle \langle y_1 \dots y_N \rangle x \rangle = \langle \langle y_1 x \rangle \dots \langle y_N x \rangle \rangle$.

Перечисленные функции применяются главным образом на уровне составных последовательностей. Дополнительно введены функции преобразования элементарных последовательностей в составные (каждый уровень которых строится из блоков одного объема) и составных в элементарные. При этом может быть указан номер уровня, начиная с которого создается или уничтожается структура последовательности и порядок изменения индексов последовательностей для получения требуемой развертки данных. Например, если данные на рисунке имеют развертку по $X - Y - Z$, можно получить развертку по $Y - X - Z$, $Y - Z - X$ и т. п. Данные могут быть представлены как составными (по $X - Y$, X, Y и т. п.), так и одной элементарной последовательностью.

Функции ввода-вывода ориентированы на использование стандартных периферийных устройств ЭВМ и аппаратуры КАМАК. Основными КАМАК-устройствами являются модули АЦП различных типов, модули отображения данных (на телевизионном приемнике, графопостроителе), входные и выходные регистры и т. п. Общий формат функции включает код операции (R — чтение данных с устройства, W — запись на устройство), тип устройства (например, TVM — устройство ввода изображения, ADC — АЦП), управляющие параметры (координаты считываемого фрагмента, параметры устройства, режима ввода и т. д.). Параметры могут применяться по умолчанию. Ввод или вывод данных на устройства происходит, как правило, с помощью элементарных последовательностей.

К числу специальных и арифметических функций отнесены традиционные (SIN, LOG, MIN, ADD, SUB и т. п.), а также подключаемые пользователем функции. За счет использования форм и функций структурных преобразований подключаемые функции содержат лишь элементарные вычислительные операции. При использовании параметров форм функции не требуют перераспределения данных на уровне элементарных последовательностей и позволяют осуществлять преобразования данных «на месте», не создавая промежуточных элементарных последовательностей.

Для примера рассмотрим функции, используемые в экспериментах по оптическому неразрушающему контролю.

В ряде задач обработки экспериментальных данных (например, фильтрации) становится необходимым «скользящее» применение функций обработки. Ниже приводится рекурсивная функция, реализующая подобную схему применения для попарного суммирования элементарных последовательностей (для переменного числа последовательностей произвольной длины):

DEF RES = IF[SEQ[NULL — TL] → KDR1; SEQ[APL —
— PAR[SEQ[ALL[ADD] — PAR[KDR1, KDR2]], SEQ[RES — TL]]]].

При этом элементы x_{ij} каждой (кроме последней) последовательности из $\langle \langle x_{11} \dots x_{1M} \rangle \dots \langle x_{N1} \dots x_{NM} \rangle \rangle$ заменяются на $x'_{ij} = x_{ij} + x_{i+1j}$, причем в процессе выполнения (если формы изменяют лишь структуру данных, а результат ADD сохраняется на месте левого аргумента) промежуточные элементарные последовательности могут не порождаться (соответствующие параметры полагаются заданными).

Обеспечив возможность выбора требуемого числа аргументов и передачи результата функции, рассматриваемой в качестве параметра (подставляемой вместо ALL[ADD]), можно получить форму, реализующую указанную схему.

Покажем применение приведенных ранее законов алгебры программ и теоремы линейного разложения [2] для анализа условий завершения

рассмотренной функции. Для этого требуется показать линейность формы, определяемой ее правой частью.

Обозначим HF = SEQ[APL − PAR[SEQ[...], SEQ[F − TL]]]. Линейность формы H означает главным образом выполнение условия HIF[$A \to B$; C] = IF[$H_tA \to HB$; HC] для любых A, B, C; H_t — преобразователь предиката, который должен быть определен. Последовательно применяя (3), (5) и (4), получим

$$\text{HIF}[A \to B;\ C] = \text{IF}[\text{SEQ}[A - \text{TL}] \to ,$$
$$\to \text{SEQ}[\text{APL} - \text{PAR}[\text{SEQ}[\text{ALL}[\text{ADD}] - \text{PAR}[\text{KDR1, KDR2}]],$$
$$\text{SEQ}[B - \text{TL}]]];\ \text{SEQ}[\text{APL} - \text{PAR}[\text{SEQ}[\text{ALL}[\text{ADD}] -$$
$$- \text{PAR}[\text{KDR1, KDR2}]],\ \text{SEQ}[C - \text{TL}]]]].$$

Таким образом, H линейна с преобразователем H_tA = SEQ[A − TL]. По условию теоремы, для всех уравнений вида DEF F = IF[$P \to Q$; HF], где P, Q — произвольные функции, а H — линейная форма, решение есть

$$F = \text{IF}[P \to Q;\ \text{IF}[H_tP \to HQ;\ \ldots \text{IF}[H_t^NP \to H^NQ;\ \ldots$$

Отсюда условие завершения на втором шаге H_tP = SEQ[P − TL] = = SEQ[NULL − TL − TL], на третьем — H_t^2 = SEQ[SEQ[P − TL] − TL] = = SEQ[NULL − TL − TL − TL] и т. д.

Следующая функция обеспечивает ввод кадров стандартного размера (параметры также заданы по умолчанию) с линейки фотоприемников ПЗС (PZS) в привязке к перемещению микрометрического стола (MMT). Далее первый кадр (фон) вычитается из остальных и удаляется:

$$\text{DEF INP} = \text{SEQ}[\text{ALL}[\text{SUB}] - \text{DSR} - \text{PAR}[\text{TL, KDR1}] -$$
$$- \text{SQR} * 10,\ 5,\ 1,\ \text{ABS}[\text{RMMT},\ \text{SEQ}[\text{APR} - \text{PAR}[\text{KDRS} - \text{PRZS}]]]].$$

Применяя далее ALL[ITR[ADD]], получаем последовательность сумм элементов по каждому кадру, ITR[ALL[ADD]] — поэлементную сумму кадров и т. д.

Приводимая ниже функция производит ввод стандартного блока данных с АЦП по появлению синхроимпульса (SYN) и суммирование получаемых блоков в процессе ввода:

$$\text{DEF MDP} = \text{SQL} * 100[\text{RSYN} \to \text{SEQ}[\text{ITR}[\text{ALL}[\text{ADD}]] -$$
$$- \text{APR} - \text{PAR}[\text{KDRS, RADC}]]].$$

Для создания языковых средств рассматриваемого типа используется разработанная на основе [3] инструментальная система, позволяющая автоматизировать основные этапы построения специализированных интерпретирующих систем. Основными перестраиваемыми компонентами базовой интерпретирующей системы при этом являются блок синтаксического анализа и разметки, интерпретатор функциональных форм, набор элементарных функций.

На первом этапе исходные данные для построения интерпретирующей системы задаются в виде грамматики входного языка, описывающей требуемый состав функциональных форм, элементарные функции и их параметры. Для исходной грамматики по методу [4] строится программа анализатора, являющаяся основой функционирования блока синтаксического анализа и разметки. Обеспечивается возможность послойной детализации грамматики без необходимости полной перестройки анализатора (базовая грамматика форм обычно дополняется описанием элементарных функций и их параметров). Далее проверяются условия контекстной независимости исходной грамматики, гарантирующие правильность разбора, и формируются таблицы символов, необходимые для дальнейших построений. Полученная программа анализатора доопределяется средствами ввода и преобразования входных предложений к ин-

терпретируемому виду, а также средствами их разметки (в основном для конструкций «форма» и «элементарная функция»).

На втором этапе при необходимости изменяется состав базовых функциональных форм. Подключение имеющихся форм производится автоматически. Новые формы (как и определения рекурсивных функций, задающие схему вычислений подобно ALL или ITR) добавляются к рекурсивной программе управления выполнением функциональных выражений [5], предоставляющей средства для ведения динамической памяти. Особенности реализации базовой системы связаны с возможностью использования автоматического и «ручного» управления памятью, сочетанием списочного и линейного размещения данных.

Элементарные функции объединены в библиотеки подпрограмм обработки экспериментальных данных и взаимодействия с используемыми модулями КАМАК. Настройка на обращение к новой функции осуществляется расширением таблицы функций интерпретатора: Дополнительно в этой функции требуется обеспечить прием стандартных управляющих параметров.

Предоставляются средства отладки перечисленных компонент разрабатываемой системы, включая трассировку основных структур данных (стек автомата, таблицы разметки, стек интерпретатора форм, параметры базовых функций).

ЛИТЕРАТУРА

1. **Backus J.** Can programming be liberated from von Neumann style? A functional style and its algebra of programs.— Comm. ACM, 1978, v. 21, N 8, p. 613—641.
2. **Backus J.** The algebra of functional programs: Functional level reasoning, linear equations and extended definitions.— Lect. Notes in Comp. Sci., 1981, v. 107, p. 1—43.
3. **Куликов В. В., Скобелев П. О.** Система автоматизированного построения проблемно-ориентированных диалоговых процессоров.— В кн.: Интерактивные системы: Материалы V школы-семинара. Тбилиси: Мецниереба, 1983.
4. **Куликов В. В., Скобелев П. О.** Об одном алгоритме построения $LR(k)$-анализаторов.— В кн.: Автоматизация научных исследований. Куйбышев, 1984.
5. **Inone K., Tanizawa T., Taniguchi K., Okamoto T.** Implementations of a functional programming language FPL.— Systems Computers Control., 1982, v. 13, N 3, p. 1—10.

Поступила в редакцию 20 марта 1986 г.

УДК 681.3.06

Р. ВЕРБОВА, А. НЯГОЛОВ, О. ПОПОВ, Д. СОФКОВА

(София, Болгария)

ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ МИКРОКОМПЬЮТЕРНОЙ СИСТЕМЫ АВТОМАТИЗАЦИИ ЦЛАНП 0270

1. Общее описание микрокомпьютерной системы ЦЛАНП 0270. Микрокомпьютерная система автоматизации ЦЛАНП 0270 является двухкрейтовой системой модульного типа в стандарте КАМАК [1, 2]. Ее можно применять для решения широкого круга задач, связанных с созданием проблемно-ориентированных комплексов автоматизации научных исследований и технологических процессов.

Состав технических средств базовой конфигурации включает микрокомпьютер, крейт-контроллеры двух крейтов, оперативную память до 48 Кбайт, модуль управления клавиатурой (дисплеем) и модуль управления гибкими магнитными дисками типа ЕС 5014 [3].