СООБЩЕНИЯ
ОБЪЕДИНЕННОГО
ИНСТИТУТА
ЯДЕРНЫХ
ИССЛЕДОВАНИЙ
ДУБНА

10 - 7795

В.А.Загинайко

ОБРАБОТКА СПИСКОВ В КОМПИЛИРУЮЩЕЙ СИСТЕМЕ БЭСМ-4

1974

ЛАБОРАТОРИЯ ВЫЧИСЛИТЕЛЬНОЙ ТЕХНИКИ И АВТОМАТИЗАЦИИ

10 - 7795

В.А.Загинайко

ОБРАБОТКА СПИСКОВ В КОМПИЛИРУЮЩЕЙ СИСТЕМЕ БЭСМ-4

В настоящей работе рассматривается разработанный и реализованный автором в виде серии подпрограмм метод обработки информации, представленной некоторым специальным образом, и возможные применения этого метода.

Работа состоит из трех глав. В первой главе дается описание представления информации в виде списковой структуры (списка) и его машинная реализация. Далее рассматривается перечень подпрограмм, выполняющих со списками как элементарные, так и более сложные операции.

Во второй главе рассматривается приложение аппарата списков к задачам обработки арифметических выражений.

В третьей главе показывается возможность приложения аппарата списков к некоторым физическим задачам на примере задачи расчета параметров электрической цепи.

I. Описание аппарата списков.

Под списком или списковой структурой в данной работе понимается связный линейный ориентированный граф. Вершины графа будем называть узлами списка, а отрезки (ребра) графа, ориентированные так, что данный узел есть начало этих отрезков, будем называть направлениями (исходящими из данного узла). Аналогично определяются направления, входящие в данный узел. С каждым направлением в списке

связано некоторое целое число, называемое номером этого направления. На нумерацию направлений накладывается следующее требование: из данного узла не могут исходить два направления с одинаковыми номерами.

Будем говорить, что узел A соединен с узлом B, если в списке существует серия узлов $A=A_0, A_1, A_2 \ldots A_n = B$ и направлений с номерами w_0 , w_1 , $\ldots$ w_n таких, что направление w_i ($i = 0, \ldots n$) исходит из узла A_i и входит в узел A_{i+1}. При этом серия направлений $w_0 \ldots w_n$ определяет путь, которым узел A соединен с узлом B.

Узел A будет называться базисным узлом списка, если существуют пути, соединяющие его со всеми остальными узлами списка. В дальнейшем будут рассматриваться только списки, обладающие базисными узлами. Следует заметить, что, если узел A соединен с узлом B, вообще говоря, неверно, что узел B соединен с узлом A.

Рассмотрим теперь реализацию понятия списка на ЭВМ. Поле памяти ЭВМ, отведенное под задачу, обрабатывающую списки, разделено на 2 части – поле задачи и поле списков. Каждый узел списка представляет собой группу ячеек памяти, расположенных в поле списка и притом некомпактно (ячейки одного узла могут чередоваться с ячейками других узлов). Каждая ячейка узла содержит две инструкции (левую и правую). Первая инструкция узла может быть только левой. Инструкции имеют следующий вид:

поле флагов	номер	адрес

На ЭВМ БЭСМ-4 поле флагов содержит 3 бита, номер – 3 бита, адрес-I2 бит. В поле флагов содержатся следующие признаки:

I. Признак того, что данная инструкция является последней инструкцией узла;

2. Признак того, что адрес является ссылкой на ячейку, находящуюся в поле задачи (является адресом этой ячейки);

3. Признак того, что адрес является адресом следующей (левой) инструкции того же самого узла (признак продолжения узла). В противном случае (если признак нулевой) адрес указывает на первую инструкцию какого-либо другого узала или на ячейку программы.

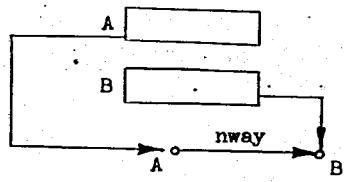
Любой список задается ячейкой в поле задачи, содержащей информацию об адресе первой инструкции какого-либо базисного узла этого списка, т.е. с ссылкой на базисный узел списка.

Обработка списка может быть реализована на языке ФОРТРАН в виде обращений к элементарным подпрограммам обработки списков, подобно тому, как это сделано в работе /I/. Эти подпрограммы могут быть написаны на языке нижнего уровня (автокоде или макроязыке) из соображений экономии памяти и времени счета. На ЭВМ БЭСМ-4 они реализованы в рамках компилирующей системы на языке "МАКРОС" /6/.

Рассмотрим более подробно эти подпрограммы.

I. Оператор `call nil(A)` заводит новый узел в поле списка (пустой узел), и в ячейке A возникает ссылка на него (адрес первой инструкции этого узла).

2. Оператор `B = step(A, nway )`. В данном случае предполагается, что узел, указанный в ячейке A ("узел A"), связан с узлом B направлением, номер которого содержится в ячейке `nway` . Оператор вычисляет в ячейке B адрес одноименного узла. Действие этого оператора может быть представлено в виде следующей схемы:



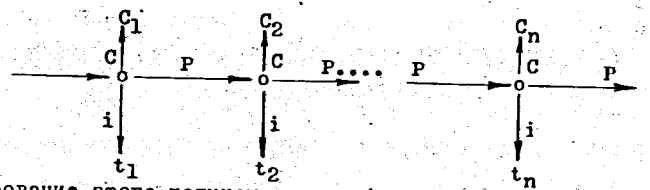
3. Оператор call cons (A,B, nway) соединяет узлы A и B направлением nway . Ссылки на эти узлы находятся в соответствующих ячейках. nway имеет тот же смысл, что и в предыдущем пункте.

4. n = nway (A,flag) Этот оператор вычисляет последовательно номера направлений, исходящих из узла A. Если перед этим оператором flag=1 ,то будет вычислен номер направления первой инструкции узла A, в противном случае вычисляется номер инструкции, следующей после той, направление которой вычислялось в этом узле раньше. После обращения к функции nway flag принимает значение I, если вычисляемый номер был в последней инструкции списка, в противном случае - flag=0 .

5. call wip(A) . Этот оператор "уничтожает" узел списка, на который есть ссылка в ячейке A программы. При этом производится увеличение свободной части поля списков на количество ячеек, занятых данным узлом. Свободная часть поля списков задается следующим образом: если после ячейки свободной части следует ячейка занятой части поля списков, то в этой ячейке (свободной части поля списков) располагаются адреса начала и конца следующего незанятого массива поля списков.

Перечисленные программы позволяют проделывать ряд более сложных операций со списками.

Рассмотрим в качестве примера задачу приведения подобных членов в полиноме вида $A = \sum_{i=1}^{n} C_i \cdot t_i$, где C_i – числовой коэффициент, t_i – текстовая константа, обозначающая идентификатор при коэффициенте. В этом случае полином может быть представлен списком следующего вида:



Формирование этого полинома может быть реализировано следующей программой (на языке ФОРТРАН):

```
      subroutine pol(A,n)
      integer c,p,term,coef, A ,term1
      c=1  $ i=2 $ p=3
      call nil(A) $ term=A
      do 1 i1=1,n
      call nil(ident) $ call nil(coef) $ call nil(term1)
      call cons(term,coef,c) $ call cons(term,ident,i)
      call cons(term,term1,p) $ call read(coef,ident)
    1 term=term1
      return
      end
```

Программа read (coef , ident) заносит значение коэффициента и текстовой константы в ячейки, связанные с узлами coef и ident . Для приведения подобных членов теперь достаточно двойным циклом сравнить всевозможные пары t_i , t_j , (i j) и, если $t_i = t_j$, то $C_i = C_i + C_j$, узлы соответствующие t_{j-1}

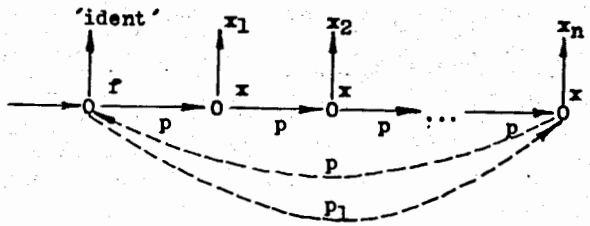
и t_{j+1} , соединяются направлением p .

Подобного рода программа была реализована автором в качестве тестовой задачи, проверяющей работы элементарных подпрограмм работы со списками.

В заключение этой главы следует заметить, что методы работы со списками, изложенные здесь, близки к методам, описанным в работе /I/.

2. Применение аппарата списков к задачам обработки арифметических выражений

Рассмотрим использование рассмотренного в главе I аппарата к задаче преобразования арифметического выражения в виде списка, представляющего прямую польскую запись этого выражения. Прямая польская запись арифметического выражения (арифметическое выражение понимается в смысле языка ФОРТРАН) получится, если арифметические или логические операции записать в виде функций от своих операндов. Например, выражение a+bжc принимает вид +(a, ж(b,c)). Прямую польскую запись арифметического выражения удобно представлять в виде списка следующим способом. Если подвыражение представляет собой функцию от нескольких переменных вида ident (x_1 , ... x_n), то соответствующий элемент списка примет вид:

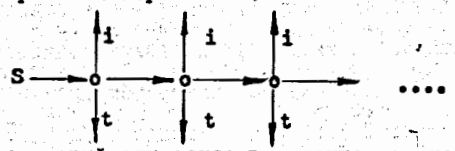


Направление f есть ссылка на текстовую константу-название (идентификатор) функции, направление p определяет список параметров (группу направлений, см. гл.3).

Если переменная представляет собой первичную переменную, соответствующий элемент списка имеет вид:

В данном случае 'ident'- идентификатор, тип (характеристика переменной) может быть списком или машинным словом. Тип содержит, в частности, информацию о том, является ли переменная числом, идентификатором или разделителем (знак действия, запятая, скобки и т.д.).

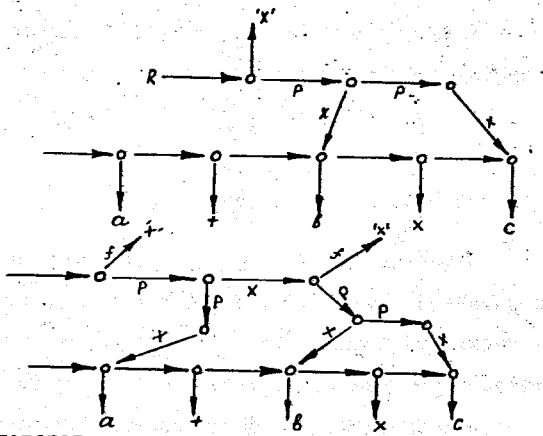
Вначале арифметическое выражение предполагается заданным в виде списка первичных переменных:



В случае, если арифметическое выражение задано в текстовом виде, такой список может быть получен, например, программой лексического анализа текста /2/.

Из списка первичных переменных может быть получен список прямой польской записи способом, аналогичным трансляции арифметического выражения в последовательность элементарных арифметических операций. При этом выполнение каждой арифметической операции интерпретируется как наращивание результирующего списка элементом, соответствующим функции-арифметическому действию. Для случая выражения a+b ж c соответствующее выполнение элементарных действий примет следующий вид (для простоты линии направления и указатели опущены):

R = b x c



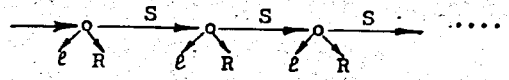
a + R

результат

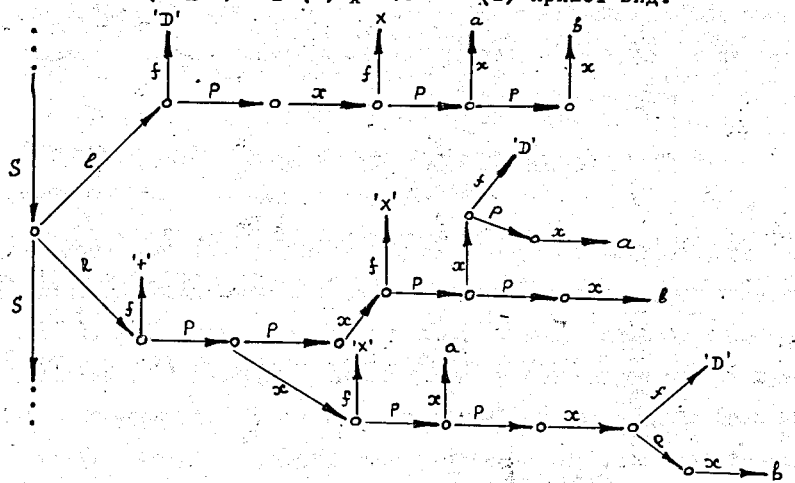
Для получения последовательности элементарных арифметических действий можно использовать алгоритм Дикстры /2/ преобразования арифметического выражения в инверсную польскую запись с использованием магазина операций с последующей интерпретацией полученной последовательности с помощью магазина операндов. Следует заметить, что с помощью инверсной записи можно преобразовывать выражения вида: f(-x, y), если их предварительно преобразовать в следующую форму f (, 0 - x, y) (0 - арифметическая переменная функциональной композиции имеет приоритет выше, чем степень) операции, "запятая" (параметр) имеет низкий приоритет /2/.

Преобразование арифметического выражения в список, представляющий прямую польскую запись, полезно использовать в качестве подпрограммы в задачах эквивалентных преобразований арифметических выражений, например, в задаче аналитического дифференцирования. Процесс дифференцирования можно представить себе как работу некоторого макроассемблера, причем макроопределениями ("таблицей соответствий" в смысле /3/) являются правила аналитического дифференцирования. Левая и правая части макроопределений преобразуются в

прямую польскую запись и формируется таблица поиска в виде списка следующего вида:

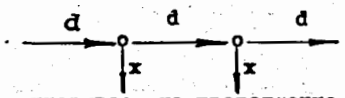


Здесь направления ℓ и R указывают на польскую запись левой и правой частей макроопределения. Например, фрагмент таблицы поиска для правила D (a x в) = D (a) x в+a x D(в) примет вид:



Для проведения дифференцирования арифметическое выражение следует предварительно преобразовать в вид списка польской записи (прямой). В процессе дифференцирования ведется линейный служебный список-указатель на те узлы, из которых по направлению f есть выход на текстовую константу 'D' . Дифференцирование представляет собой просмотр этого списка. При этом с помощью таблицы поиска соответствующий фрагмент списка заменяется на результат своего дифференцирования. Если правая часть соответствующего макроопределения содержит операторы дифференцирования, служебный список наращивается.

После обработки узла служебного списка он уничтожается, и базисный узел служебного списка смещается по направлению этого списка. Служебный список имеет вид:



Направление d указывает на продолжение списка, направления x указывают на узлы дифференцируемого списка, связанные с 'D' . Процесс дифференцирования кончается в случае, если служебный список исчерпан, либо если соответствующее правило дифференцирования отсутствует в макроопределениях.

Применение аппарата списков к электротехническим расчетам.

Рассмотренный в главе I аппарат списков может быть применен к задачам расчета электрических цепей. В данной главе рассматривается метод решения одной из таких задач. Электрическую цепь этой задачи удобно определить в топологических терминах /4/. Под электрической цепью будем понимать ориентированнный одномерный симплициальный комплекс, причем подкомплекс циклов совпадает со всем комплексом (каждый симплекс принадлежит хотя бы одному циклу (замкнутому электрическому контуру). С каждым симплексом (отрезком) цепи связано одно или несколько электрических сопротивлений и источников тока, задающих перепады напряжения в определенных участка цепи. Обычно в таком случае по заданным сопротивлениям и источникам тока ищутся токи в одномерных симплексах и потенциалы в нульмерных симплексах (узлах) цепи. Нашей задачей будут описание возможного построения функции y = eq(x) , где аргумент представляет собой список – электрическую цепь, а y– список, описывающий

соответствующую систему уравнений для токов. Аргумент и функция в машинном представлении – ячейки программы, содержащие ссылки на базисные узлы соответствующих списков.

Система уравнений для токов задается правилами Кирхгофа /5/:

$$\sum_{(k)} R_{\ell k} j_k + \sum_{(k)} E_{\ell k} = 0 \qquad (\ell = 1, \ldots n)$$

$$j_k = \sum_{(s)} i_{ks} \qquad (s = 1, \ldots m(k))$$

При этом n–число гомологически независимых между собой циклов (одномерное число Бетти комплекса). С каждым таким циклом связан элементарный ток j_{ks}. Полный ток, протекающий через симплекс k , является алгебраической суммой токов всех циклов, которым этот симплекс принадлежит. Количество этих циклов есть функция m (k). В уравнениях для напряжений суммирование проводится по всем симплексам, входящим в данный цикл. Для описания электрической цепи в виде списка удобно ввести понятие группы направлений, изображаемых в виде списка следующей структуры:

Направление n будем называть направлением группы, а саму группу направлений будем называть группа направлений n . Направления n и n_1 для всех групп суть одни и те же. Направление n1 удобно ввести для экономии времени при наращивании числа узлов в группе.

Каждый нульмерный симплекс электрической цепи есть узел списка, из которого исходят две группы направлений: группа C и группа i . Узлы группы i содержат ссылки на соседние узлы (нульмерные симплексы) электрической цепи или на соседние элементы цепи (сопротивления или источники тока). Количество узлов в группе i

совпадает с количеством циклов, в которые входит нульмерный симплекс. Направление, исходящее из группы C в соседний узел электрической цепи.совпадает с ориентацией (направлением обхода) сооответствующего цикла.

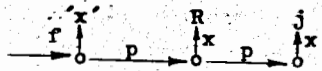
Направления, исходящие из группы C ,суть ссылки на узлы, описывающие циклы, включающие данный узел электрической цепи. В свою очередь,из узла цикла исходит группа направлений на нульмерные симплексы соответствующего цикла, причем порядок ссылок в группе направлений определяется направлением обхода или направлением элементарного тока соответствующей цепи.

Элементы цепи (сопротивления или источники тока) описываются узлом списка с тремя направлениями, Первое направление (R или E) есть ссылка на ячейку программы, в которой содержится величина соответствующего сопротивления или источника тока. Величина может быть представлена как числом, так и списком, если речь идет о получении аналитических выражений для параметров цепи. Два других направления (i , il) задают ссылки на элементы или узлы цикла вдоль и против элементарного тока.

Вся электрическая цепь характеризуется величиной, из которой исходит группа направлений t на узлы циклов цепи.

Аналитическое выражение для системы уравнений Кирхгофа представляет величину, из которой исходит группа направлений e , количество направлений которой совпадает с количеством направлений группы t . Группа e ссылается на узлы, представляющие собой отдельные уравнения системы. Из каждого такого узла исходит группа направлений plus . Количество направлений группы plus совпадает с суммарным количеством узлов и элементов цикла. Элемент цикла в прямой польской записи имеет вид функции x (R , j) (т.е. R x j)

и, следовательно, представляется элементом списка вида:



Узел, обозначающий ток в симплексе согласно правилу Кирхгофа для токов, представляет собой группу plus с направлением на пары вида + i или - i .

Таким образом, искомая программы-функция eq (x) включает в себя три вложенных друг в друга цикла. Внешний цикл по группе t формирует группу e . Средний цикл есть просмотр узлов и элементов цикла электрической цепи и формирование группы plus . Внутренний цикл представляет собой просмотр группы c, связанной с началом симплекса цикла. В результате возникает группа plus суммы элементарных токов.

В заключение этой главы следует заметить, что описанная здесь схема получения формул расчета цепей пригодна для электрических схем, состоящий из транзисторов, сопротивлений, конденсаторов и индуктивностей. Для конденсаторов и индуктивностей величины сопротивлений имеют вид операторов вида $R_C = (cd)^{-1}$; $R_L = LD$, где D – дифференциальный оператор времени, что приводит в результате вычисления функции eq (x) к системе интегродифференциальных уравнений. Транзистор может быть представлен эквивалентной схемой в виде источника тока и переменного сопротивления, в результате чего в уравнениях для токов возникают нелинейные члены вида R (i) x i .

ЛИТЕРАТУРА

I. Weizenbaum J., Symmetrical List Processor Comm. A.C.M, 6, 1963.

2. Болье. Методы построения компиляторов (в сб. "Языки программирования". "Мир", М., 1972 г.)

3. Загинайко В.А. Сообщения ОИЯИ, Р11-3993, Дубна, 1968.

4. П.Хилтон, С.Уайли. Теория гомологий, "Мир", М., 1966 г.

5. Гришин Ю.А., Дворецкий В.И., Родионов А.И., Тюриков О.А. Сообщения ОИЯИ, Р11-5185, Дубна, 1970.

Рукопись поступила в издательский отдел
12 марта 1974 года.