Ц8406
А-84

СООБЩЕНИЯ
ОБЪЕДИНЕННОГО
ИНСТИТУТА
ЯДЕРНЫХ
ИССЛЕДОВАНИЙ
ДУБНА

4/III.74

10 - 7586

893/2-74

Д.Д.Арнаудов

ОБРАБОТКА СПИСОЧНОЙ ИНФОРМАЦИИ ПРИ ПРОГРАММИРОВАНИИ ОСНОВНЫХ АЛГОРИТМОВ ИПС НА КОБОЛе

1973

ЛАБОРАТОРИЯ ВЫЧИСЛИТЕЛЬНОЙ ТЕХНИКИ И АВТОМАТИЗАЦИИ

10 - 7586

Д.Д.Арнаудов

ОБРАБОТКА СПИСОЧНОЙ ИНФОРМАЦИИ ПРИ ПРОГРАММИРОВАНИИ ОСНОВНЫХ АЛГОРИТМОВ ИПС НА КОБОЛе

Объединенный институт ядерных исследований БИБЛИОТЕКА

Основой эффективности управления различными сферами общественной жизни является своевременный сбор, переработка и использование информации самого различного характера (не только научно-технической, поток которой весьма значителен, но и информации политической, военной, медицинской и т.д.). Так как сущность управления состоит в принятии оптимальных решений на базе определенной информации, то проблемы анализа, обработки, сохранения и поиска информации являются основными при решении данного вопроса.

Информационно-поисковая система (ИПС) - это система, предназначенная для информационного поиска, хранения и обновления информации. Под "информационным поиском" понимается область обработки данных, включающая совокупность ряда логических операций, конечной целью которых является выявление по заданным признакам отбора всех данных (в виде документов, фактических справок), содержащих требуемую информацию и представляющих собой ответы на вводимые запросы .

Автоматизированная ИПС - это информационно- поисковая система, созданная на базе современной вычислительной техники. Наряду с техническими средствами (ЭВМ, терминалы, быстропечатающие устройства и т.п.) в состав автоматизированных ИПС входит комплекс программ, обеспечивающий функционирование системы, информационный массив (содержащий сведения, учитываемые в системе) и различные виды слу-

жебных массивов (речники, таблицы и т.д.).

В зависимости от решаемых задач, ИПС бывают документальными (библиографическими) и фактографическими /I/.

Математическая теория ИПС дана Н.А.Криницким в работе /2/. Каждую информационно-поисковую систему можно понимать как совокупность

$$I = \{P, A, F(p), H(p,F(p)), Q, V(q)\},$$

где $P = \{p\}$ – конечное множество признаков p ;

$A = \{a\}$ – конечное множество элементов информации a. При этом каждое a имеет свойство, выраженное соответствующим признаком p ;

$F(p)$ – алгоритм поиска информации, однозначно отражающий P в A;

$H(P,F(p))$ – алгоритм обработки, который каждой паре P, F(p) ставит в соответствие $H(P,F(p)) \subset R$, где R – это конечное множество результатов решения информационной задачи;

$Q = \{q\}$ – конечное множество сообщений, на основе которых совершается изменение информации в ИПС;

$V(q)$ – алгоритм для внесения изменений в ИПС.

Имея в виду это математическое описание, мы можем определить основные алгоритмы ИПС.

I. При создании автоматизированной ИПС существенное место занимает алгоритм формирования структуры информационного языка (в частности, дескрипторного речника), в терминах которого выражается основное смысловое содержание документов и информационных запросов /3/. Информационный язык определяет не только структуру, но и эффективность работы ИПС.

2. Основными являются алгоритмы формирования информационного массива в памяти машины /4/, поиска и вывода нужной информации. Эффективность этих алгоритмов определяется прежде всего организационной структурой ИПС /I,3/. От правильного функционирования этих алгоритмов зависит экономическая эффективность ИПС: рациональное использование памяти вычислительной машины, удобство пополнения ИПС новой информацией и такие технические параметры, как скорость поиска необходимой информации, число одновременно обрабатываемых запросов и т.д.

3. Создание ИПС требует разработки эффективных алгоритмов обновления и корректировки информационных массивов /5/. Кроме того, важное место занимают и алгоритмы контроля и дублирования информационных массивов /6/.

При создании и программировании этих алгоритмов существует множество различных проблем. Одна из самых важных проблем – вопросы организации информационных данных в памяти ЭВМ – состоит в том, что сущность функционирования основных алгоритмов ИПС сводится к так называемой категоризации объектов, т.е. к их разделению на виды, типы, классы и т.д. в зависимости от их свойств и признаков. При этом запись новой информации об объектах в память машины или ее поиск осуществляется по значениям этих признаков. Для эффективного решения подобных задач необходима специальная организация работы запоминающих устройств машин, которая позволяла бы осуществлять поиск информации не только по ее адресам, но и по ее признакам.

Один из способов такой организации называют программно-ассоциативным /I,7/. Он основан на использовании обычных адресных систем памяти и заключается в том, что ассоциативные связи между различны-

ми данными, хранящимися в памяти, осуществляется либо путем специального упорядоченного расположения этих данных (или их наименований) в памяти машины, либо путем объединения их в последовательные цепочки при помощи специальных адресов связи, коды которых хранятся в тех же ячейках памяти совместно с данными. Группы последовательно расположенных данных или данных, связанных адресами связи, называются списками. При этом необходимые признаки информации могут храниться либо во всех объединяемых членах, либо только в отдельных ячейках, помещаемых в начале списков. Некоторые члены списков в общем случае могут указывать на ответвления к другим спискам и т.д. Любой список со всеми отходящими от него подсписками называется списочной структурой.

Практическое применение программно-ассоциативного способа особенно эффективно при больших объемах переменной информации, т.к. не требует разработки специальной ассоциативной памяти большого объема. Данный способ обладает гибкостью и обеспечивает возможность изменения в широких пределах алгоритмов поиска информации, состава и характера признаков, по которым производится поиск, в том числе реализацию многоступенчатых и рекурсивных поисковых процессов.

Существуют следующие способы построения списков: последовательный, цепной, гнездовой, узловой. Все они хорошо описаны в работе /I/. Последовательный список можно рассматривать как частный случай цепного, а гнездовой – как комбинацию последовательного и цепного. Узловой список – это некоторый способ комбинирования цепных и гнездовых списков.

Поэтому особый интерес представляет программная реализация цепного и узлового списков. Нами предложена методика работы со списками, использующая системы автоматизации программирования

АЛГЭМ – СТЭ /8/. Эта система имеет транслятор для ЭВМ "Минск-22" и широко используется в СССР /9/.

В данной работе мы предлагаем методику построения цепного списка и ее программную реализацию на КОБОЛе. Для практической проверки алгоритмов использован вариант КОБОЛа для ЭВМ СДС-6200 /IO/.

Алгоритмический язык КОБОЛ является достаточно хорошим языком для обработки списочной информации. Он имеет две основные характеристики, которые отличают его от известных алголоподобных языков.

Первая состоит в том, что КОБОЛ специально создан для многократного повторения однотипных операций над последовательными группами данных. Поэтому первостепенную важность приобретают вопросы, связанные с их обработкой.

Вторая характеристика связана с тем фактом, что сами данные, находящиеся в записях, отделены от программы, которая должна их обрабатывать, а не рассматриваются как часть этой программы. Частью программы могут быть только данные в форме литералов.

В КОБОЛе записи – это основные обрабатываемые единицы данных. Группы однотипных записей могут образовать массивы.

Операторы КОБОЛа в общих чертах аналогичны операторам АЛГОЛ-60. Специфика КОБОЛа проявляется в способе организации данных, а также в методике управления перемещением данных. Методика организации данных является самой важной частью КОБОЛа. Данные состоят из совокупностей записей. В описании записей машине сообщаются такие сведения об организации и структуре записи, как уровни и подуровни, обозначения этих уровней и подуровней, разрядность данных, класс или тип каждой единицы данных и т.д.

В общем случае отдельные единицы данных (не содержащих в себе более мелких подразделений) называются элементарными единицами данных . Очень часто названия элементарных единиц нельзя употреблять непосредственно, а нужно уточнять с помощью названий данных, имеющих меньший уровень (т.е. с помощью величин с более высоких уровней).

Необходимо отметить, что практически нет ограничений на количество уровней иерархии при построении данной записи.

При помощи КОБОЛа можно также описывать однородные, подряд расположенные данные, представляющие, например, строки или столбцы документа. Эта возможность может быть реализована с помощью фразы OCCURS (повторяется). Фраза может быть употреблена на уровне группового данного и указывает в этом случае, что описание применимо к повторяющейся группе данных. Фразы описания, используемые в описании пункта наряду с фразой OCCURS, распространяются на каждое повторение этого пункта.

Используя эти возможности языка КОБОЛ, мы можем представить цепной список в виде многоуровневой записи, состоящей из однородных, подряд расположенных данных, повторяющихся некоторое число раз. Программным путем все эти данные могут быть объединены в цепной список. Таким образом, в частности ,можно объединить в цепной список свободные ячейки памяти, что помогает записать на КОБОЛе программу распределения ячеек памяти машины для списковой области. Из этой области программным путем можно получать ячейки и формировать любые списки.

Фиксатор цепного списка

Количество членов списка	Адрес первого члена списка
6	I08

а)

Цепной список

Адрес члена списка	Машинное наименование	Адрес связи
I02	I003	I05
I03	I004	I07
I04	I005	КС
I05	I002	I04
I07	I00I	I02
I08	I000	I03

б)

Рис. I

Пример некоторого цепного списка показан на Рис. I.Чтобы войти в список, необходимо иметь т.н. "фиксатор списка", который содержит информацию о количестве членов списка и об адресе его первого члена. Фиксаторы списков всегда являются многоуровневыми записями. Эти фиксаторы должны быть известны программисту (чтобы войти в нужный список), и поэтому они могут быть описаны как некоторые записи в DATA DIVISION.

Итак,для того чтобы построить цепной список,необходимо иметь две основные части: его списочную область и фиксатор.

Рассмотрим пример и покажем, как с помощью алгоритма входного языка системы КОБОЛ-СДС можно получить списочную область свободных ячеек, осуществить процесс выделения ячейки из этой области, включения ячейки в список и провести корректировку списка свободных ячеек.

```
IDENTITICATION DIVISION.
PROGRAM -ID. EXAMP.
ENVIRONMENT DIVISION.
CONFIGURATION SECTION.
SOURCE-COMPUTER .6200.
OBJECT - COMPUTER .6200.
DATA DIVISION.
WORKING - STORAGE SECTION.
77  P PICTURE 9(4) VALUE 0.
77  N PICTURE 9(4) VALUE 0.
77 KC1 PICTURE 9(4) VALUE 9999.
77  F PICTURE 9(4) VALUE 0.
01  CJ1.
    02  KCH  PICTURE 9(4).
    02  ACF  PICTURE 9(4).
01 SP.
    02  C1 PICTURE 9(4) OCCURS 50 TIMES.
    02  J1 PICTURE X(25) OCCURS 50 TIMES.
    02  AC1 PICTURE 9(4) OCCURS 50 TIMES.
```



```
PROCEDURE DIVISION.
START.
  PERFORM NACH-1 VARYING N FROM 1
  BY 1 UNTIL N=51.
  GO TO MARCH.
NACH-1.
  MOVE 0 TO C1(N).
  MOVE 0 TO J1(N).
  MOVE 0 TO AC1(N).
MARCH.
  MOVE 2 TO N.
  MOVE 1 TO F.
NAT.
  MOVE N TO AC1(F).
  IF N EQUELS 50 GO TO PIT-1 ELSE
  NEXT SENTENCE.
  ADD 1 TO N.
  ADD 1 TO F.
  GO TO NAT.
PIT-1.
  MOVE 50 TO KCH.
  MOVE 1 TO ACF.
  IF ACF EQUELS KC1 GO TO POSL
  ELSE NEXT SENTENCE.
  MOVE ACF TO P.
  MOVE AC1(P) TO ACF.
  SUBTRACT 1 FROM KCH.
. . . . . .
PIT-2.
  MOVE ACF TO AC1(P).
  MOVE  P  TO ACF.
  ADD 1 TO KCH.
POSL.
  STOP RUN.
```

С помощью этой программы мы организовали в цепной список 50 ячеек памяти(они- свободные ячейки),показали, как можно получить ячейку из этого списка, присваивая номер ячейки некоторой величине P,как корректировать список и включать вновь освободившуюся ячейку в список свободных ячеек.Рассмотрим более подробно данный алгоритм.

В DATA DIVISION описаны некоторые рабочие величины: P , N , F , KC1. KC1 в данном случае является признаком конца списка и имеет некоторое значение - 9999. Запись с идентификатором CJ1 является фиксатором списка свободных ячеек. В эту запись входят величины KCH (количество членов списка) и ACF (адрес связи, указывающий на первый член списка). Сами списочные члены расположены в записи SP . Они имеют следующую структуру:

C1	J1	AC1

C1 - некоторый признак (поисковый),

J1 - наименование объекта,

AC1 - адрес связи.

Сам список состоит из 50 членов. Процедура START заполняет все эти элементы нулями, чтобы в дальнейшем использовать их как свободные ячейки (например,для построения других списков). Процедуры MARCH и NAT организуют эти ячейки в цепной список. Процедура PIT-1 оформляет фиксатор списка,и с ее помощью выделяется свободная ячейка из этого списка с присвоением ее номера величине P. Происходит и корректировка списка. Процедура PIT-2 включает некоторую ячейку с номером P в список свободных ячеек. Все это может быть иллюстрировано следующими рисунками:

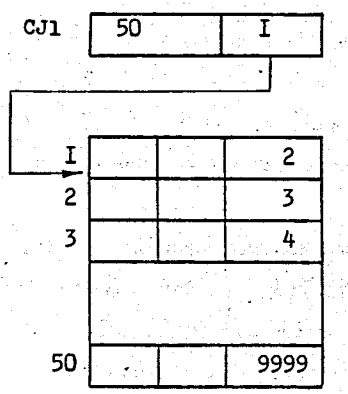

Рис. 2.

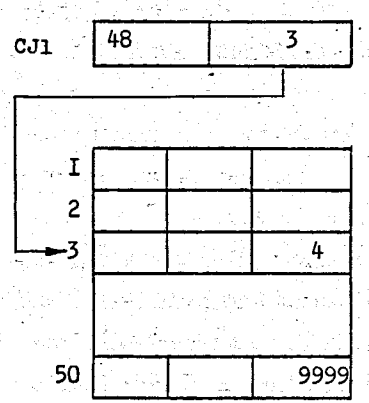

Рис. 3.

На Рис. 2 показан цепной список, имеющий 50 членов. На Рис. 3 из этого списка получены две свободные ячейки (они могут быть использованы для другого списка), и тогда в нем имеются 48 членов. В общем случае можем иметь,например,следующую картинку(см.Рис.4). На этом рисунке показана простейшая многосписочная структура.

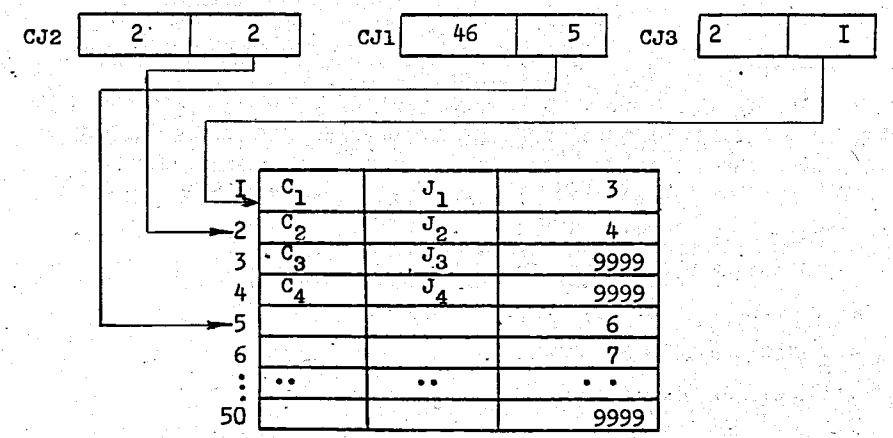

Рис. 4

Здесь, в частности, видно, что четыре ячейки, полученные из списка свободных ячеек, являются членами некоторых других списков, которые для примера обозначены с фиксаторами СЈ2 и СЈ3 .

Интересно, что несмотря на принадлежность ячеек другим спискам, они фактически остаются расположенными в уже определенной списочной области.

В дальнейшем эта методика может быть использована в алгоритме формирования информационного массива некоторой ИПС. Сам информационный массив, состоящий из ряда списков, будет сегментирован таким образом, чтобы каждый список представлял определенную зону для поиска информации. Все это приведет к созданию такого поискового массива, в котором время поиска нужных документов практически не будет зависеть от объема информационного массива.

Л И Т Е Р А Т У Р А

I. Китов А.И. "Программирование информационно-логических задач". Сов. радио, М., 1967.

2. Криницкий Н.А. "О некоторых ИПС", сб. Цифровая вычислительная математика и программирование", вып. 2, Сов. радио, М., 1967.

3. Арнаудов Д.Д. "Об одном способе организации последовательного массива в библиографической ИПС". Сб. Цифровая вычислительная техника и программирование", вып.7, Сов. радио, М., 1967.

4. Китов А.И. "Основные принципы построения ИПС для медицины", сб. Цифровая вычислительная техника и программирование", вып.6, Сов. радио, М., 1971.

5. Арнаудов Д.Д. "Автоматическо построение на управляващия списък при моделиране на сложни системи на ЦЕИМ". Сб. Наукознание и управление на науката, Наука и Искусство, София, 1971.

6. Глушков В.М. и др. " Обработка информационных массивов в АСУ", Наукова Думка, Киев, 1970.

7. Prywes N.S,GrayH.J. "The organisation of a multilist type associative memory", IEEE, 1963.

8. Арнаудов Д.Д. "Применение системы автоматизации программирования АЛГЭМ-СТ3 для работы со списками", сб. Цифровая вычислительная техника и программирование", вып. 6, Сов. радио, Москва, 1971.

9. Ригмонт М.Г. "Обработка данных статистической отчетности в системе АЛГЭМ "Минск-22", Статистика, Москва, 1973.

10. Control data 6000 Computer Systems, COBOL, USA, 1971.

Рукопись поступила в издательский отдел
6 декабря 1973 года.