СООБЩЕНИЯ
ОБЪЕДИНЕННОГО
ИНСТИТУТА
ЯДЕРНЫХ
ИССЛЕДОВАНИЙ
ДУБНА

10 - 8784

Д.Д.Арнаудов, Н.Н.Янев

МЕТОДЫ ФОРМИРОВАНИЯ И КОНТРОЛЯ ДЕСКРИПТОРНОГО СЛОВАРЯ ИНФОРМАЦИОННО-ПОИСКОВОЙ СИСТЕМЫ ОИЯИ

1975

10 - 8784

Д.Д.Арнаудов, Н.Н.Янев

МЕТОДЫ ФОРМИРОВАНИЯ И КОНТРОЛЯ ДЕСКРИПТОРНОГО СЛОВАРЯ ИНФОРМАЦИОННО-ПОИСКОВОЙ СИСТЕМЫ ОИЯИ

В информационно-поисковой системе (ИПС) ОИЯИ особое место занимает тезаурус системы, который можно рассматривать как конечное множество элементов (ключевые слова, дескрипторы), использующихся для индексирования документов (доклады, патенты, фильмы, книги и др.). Индексирование подобных документов - это процесс представления смыслового содержания документа конечным набором дескрипторов. Чаще всего индексирование производится вручную и носит в некоторой степени субъективный характер.Для индексации документов обычно используются дескрипторы, включенные в тезаурус. С течением времени, однако, содержимое тезауруса может измениться,например, при добавлении новых дескрипторов и отбрасывании устаревших.

Основой тезауруса ИПС ОИЯИ выбран тезаурус INIS (1974 г.). В дальнейшем развитие этого тезауруса может идти самостоятельным путем (независимо от тезауруса INIS). Все дескрипторы тезауруса пока заданы как слова английского языка с максимальной длиной слова 30 символов.

Как было написано в[I], в ИПС ОИЯИ выбрана списочная организация основных массивов и доступ к информации ведется в направлении: дескриптор → документ. При такой организации неизбежны преобразования вида:

а) дескриптор → код дескриптора;

б) код дескриптора → дескриптор.

Код дескриптора (в ИПС ОИЯИ это целое положительное число) является в то же время адресом памяти, где хранится информация о данном дескрипторе. Введение кодов дескрипторов дает возможность быстрого доступа к хранимой информации и, кроме этого, позволяет сократить объем памяти, отведенной для хранения этой информации. Например, как уже отметили, в ИПС ОИЯИ каждому дескриптору тезауруса с максимальной длиной слова 30 символов поставлено в соответствие пятизначное число - код дескриптора. Имея в виду, что каждый документ индексируется в среднем десятью дескрипторами, легко можно подсчитать выигрыш в памяти вследствие использования пятизначных кодов дескрипторов вместо самих дескрипторов.

Чтобы осуществить преобразования вида а) или б), необходимо использование словарей. Для преобразования типа а) этот словарь можно назвать "дескриптор-код" (ДК). Словарь ДК используется с большой интенсивностью в процессе добавления документов к ИПС, а также в процессе поиска в ИПС, а словарь "код-дескриптор" - при выдаче библиографических справок. Эти справки выдаются на последнем этапе поиска, а также и при поиске недескрипторного вида, который ведется в массиве библиографических справок по характеристикам, не являющимся дескрипторами, например; год издания, фамилия автора, предметная классификация и др.

Кроме функции преобразования вида а) или б), словарь ДК выполняет существенную функцию контроля поступающей информации. Может случиться так, что документы, поступающие в ИПС, индексированы дескрипторами, не включенными в словарь. Так, например, в ИПС ОИЯИ поступает из INIS как библиографическая информация, так и информация

для коррекции тезауруса. Часто бывает так, что информация для пополнения тезауруса опаздывает по сравнению с библиографической. В таком случае, если не предусмотреть меры для контроля словаря, все документы, индексированные дескрипторами, которых нет в словаре, не будут добавлены к ИПС программой формирования ИПС. Однако наличие словаря ДК позволяет с помощью специальной программы для контроля изъять из поступающей информации все дескрипторы, не включенные в словарь, и добавить их туда с соответствующим кодом.

Далее остановимся подробнее на способе формирования подобных словарей при помощи ЭВМ.

I. Формирование словаря типа "код-дескриптор" (КД)

Реализовать словари такого типа на устройствах с прямым доступом (магнитные диски, магнитные барабаны и т.п.) при наличии соответствующего математического обеспечения для организации массивов прямого доступа нетрудно, благодаря тому, что аргумент функции (код-дескриптор) пробегает лишь целочисленные точки данного интервала $[a,в]$ и всегда удается осуществить соответствие этого интервала интервалу $[1, L,]$ (здесь и при дальнейшем рассмотрении имеется в виду дискретное соответствие, т.е. только целочисленные точки интервалов). Пусть $i \in [a,в]$ означает код некоторого дескриптора, f - целочисленная функция, определенная для всех целочисленных точек $[a,в]$, со значением в $[1,L]$. Тогда, если говорить о словаре КД как о таблице с L элементами, задача о нахождении дескриптора с кодом i сводится к чтению $f(i)$ -го элемента таблицы. Можно рассмотреть два основных случая:

а) Когда L равняется числу дескрипторов в словаре, вышеописанная процедура сразу выдает требуемый дескриптор.

б) Если элементы таблицы суть не просто дескрипторы, а группы из К дескрипторов, тогда данная процедура выдает ту группу из К дескрипторов, в которой находится искомый.

В операционной системе (СДС, серия 6000) SCOPE 3.4 предусмотрены такие способы организации массивов прямого доступа, которые позволяют легко реализовать как случай а), так и б). В SCOPE 3.3 реальная возможность существует только для случая б).

Рассмотрим две конкретные реализации случаев а) и б) в ИПС ОИЯИ. Напомним, что в ИПС ОИЯИ код дескриптора составной и имеет вид: XY, где X занимает одну позицию и принимает значения от 1 до 9, а Y - четыре позиции и принимает значения от 1 до 4480. Организация массивов, которая позволяет реализовать случай б), называется STANDARD и имеет следующие особенности:

- доступ к каждой записи в массиве совершается с помощью ключа (целое положительное число);

- все ключи хранятся в специальной таблице, называемой "индекс", которая в момент работы с массивом находится в оперативной памяти;

- с каждым ключом ассоциирован адрес, который указывает на начало записи на диске;

- для осуществления доступа к массиву по заданному ключу производится просмотр "индекса" до нахождения нужного ключа или до установления его отсутствия.

Итак, если словарь состоит из 12000 дескрипторов и каждый дескриптор рассматривается как отдельная запись, то для работы с таким словарем необходимо по крайней мере 12000 слов оперативной памяти для "индекса" и в среднем 6000 операций сравнения для осуществления

доступа к отдельному дескриптору. Из-за явной неэффективности такой организации (SCOPE 3.3) словарь ИПС ОИЯИ реализован как в случае б) при использовании организации STANDARD. Словарь организован как массив из К записей по m дескрипторов в каждой. К и m связаны: $Km=L$, где L - максимальный объем словаря. Число m надо выбирать, имея в виду вышеуказанные особенности метода STANDARD и , кроме того, конкретные особенности ввода/вывода. Для СДС (серия 6000) m надо выбирать так, чтобы длина одной записи была кратной длине одной физической записи (PRU).

В этом случае декодирование (нахождение дескриптора по коду) кода XY осуществляется следующим образом:

а) Вычисляется ключ записи, где находится искомый дескриптор, по формуле:

$$\text{ключ} = \left[\frac{(x-1)4480+y-1}{m}\right]+1 ,$$

здесь [A] обозначает функцию целой части A. Получаем как раз функцию f , о которой шла речь раньше.

б) Читается запись с этим ключом.

в) Вычисляется местонахождение M дескриптора в этой записи по формуле:

$$M = У - (\text{ключ} - 1)m ,$$

и из этого места извлекается нужная информация.

Как мы уже сказали, в SCOPE 3.4 (а также в математическом обеспечении IBM, РЯД и др.) возможна и организация массивов прямого доступа типа RELATIVE . Для этой организации характерны следующие особенности:

- все записи имеют одинаковую длину,

– доступ к каждой записи в массиве совершается по ключу (целое положительное число),

– система вычисляет адрес на диске по ключу записи, длине записи и начальному адресу массива.

Для декодирования кода XY надо только вычислить функцию f по формуле: ключ=(X-1)4480+Y, и совершить операцию чтения по этому ключу.

Необходимо отметить, что в этих случаях необязательно присутствие кодов дескрипторов в самой записи. Присутствие кода в этом словаре оправдывается только тогда, когда производится контроль правильности доступа к словарю.

2. Формирование словаря типа "дескриптор-код" (ДК) и доступ к его элементам

Использование рассмотренных выше способов организации массивов (STANDARD, RELATIVE) для формирования словаря типа КД обусловливалось возможностью легко преобразовать код дескриптора в соответствующий адрес для доступа к словарю. К элементам словаря ДК доступ совершается по дескрипторам (произвольная последовательность буквенно-цифровых символов), которые в принципе нельзя использовать в качестве ключей для организации массивов типа RELATIVE или STANDARD.

Осуществление быстрого однозначного преобразования типа дескриптор-адрес тоже не представляется возможным, так что рассмотренные выше методы прямого доступа по целочисленным ключам не рациональны. Поэтому для решения этой проблемы выбираем другой путь.

Пусть α_i означает i-ый дескриптор, а K_i – код этого дескриптора. На основе исходного массива с элементами (α_1, K_i) (элементы массива расположены в восходящем порядке по α_1) заполняем таблицы

одинаковой структуры с емкостью t элементов (α_i, K_i) каждая (рис.I). Первые t элементов (α_i, K_i) записываем в первой таблице, следующие – во второй и т.д. В специальной таблице (назовем ее индексной) с емкостью L/t элементов (L –максимальное число дескрипторов в словаре), записываем на i –е место элемент (α_{it}, i) (α_{it} является последним дескриптором i –й таблицы).

Чтобы найти код произвольного дескриптора, например α, поступим там: просматриваем индексную таблицу с начала до конца или до установления неравенства $\alpha \leq \alpha_{it}$ (максимальное число таких сравнений – L/t). В первом случае дескриптор α не записан в словаре, а во втором случае в i^{*} таблице ищем элемент, в котором участвует α. Эта таблица тоже просматривается с начала до конца или до установления равенства $\alpha = \alpha_p^*$, или до установления неравенства $\alpha > \alpha_p^*$.

Поскольку максимальное количество сравнений здесь равно t, то для нахождения кода одного дескриптора будет проделано максимум t+L/t сравнений. Чтобы минимизировать эту форму, достаточно выбрать $t = [\sqrt{L}]$. Например, при L=22500, t=150 максимальное количество сравнений – 300. Это число сравнений можно заметно уменьшить, если построить индексы на более высоком уровне (рис.2). Процедура поиска по этой схеме состоит из поиска в индексной таблице на 2 уровне, затем поиска в одной из индексных таблиц на I уровне и, наконец, поиска в одной из таблиц, содержащих (α_i, K_i). При L=22500, t=50 и C=50 (рис.2) максимальное количество сравнений по этой схеме получается I50, т.е. в два раза меньше, чем в первом примере. Но прежде чем отдать предпочтение первой или второй схеме,

рассмотрим, как они могут быть реализованы при использовании организации массивов RELATIVE. Если рассматривать словарь как массив прямого доступа, состоящий из $S+1$ записей (рис.I), где первые S записей будут ТАБ-I, ТАБ-2, ...ТАБ- S, а ($S+1$)-ая запись - это индексная таблица, то получается четкая схема формирования (поиска) словаря:

а) формирование - на основе поступающей информации (α_i, K_i) (напоминаем, что она рассортирована в нарастающем порядке по α_i) заполняется последовательно зона для записи рассматриваемого массива (словаря). Каждый раз, когда число (i) поступивших дескрипторов становится кратным t , в i/t - е место в специальной зоне памяти, отведенной для индексной таблицы, записывается последний дескриптор из зоны записи и число i/t , которое является номером (ключом) этой записи. После этого информация из зоны для записи заносится в массив с ключом i/t . При достижении конца входового массива по ключу $S+1$ записывается информация из зоны, отведенной для индексной таблицы.

б) на процедуре поиска в словаре не будем останавливаться, поскольку она уже была рассмотрена, только отметим, что она будет намного эффективнее, если индексная таблица будет находиться постоянно в оперативной памяти во время поиска.

Сейчас уже видны преимущества и недостатки приведенных выше схем организации словаря.

По первой схеме (один уровень - рис.I) тратится больше времени для сравнений в оперативной памяти, однако читается только одна таблица (если не считать однократного чтения индексной таблицы), по второй схеме (два уровня - рис.2) теряется в среднем в два раза меньше времени для сравнений в оперативной памяти, однако осуществля-

ются два доступа к внешней памяти (одна индексная таблица на I-ом уровне).

При конкретных реализациях для выбора параметров (размер таблицы, размер индекса) надо учитывать такие факторы, как объём оперативной памяти, отведенный для работы со словарем, и время поиска в словаре, которое, в свою очередь, можно рассматривать как время центрального процессора и время для ввода-вывода.

Выберем для единицы измерения времени время центрального процессора для совершения операции сравнения двух дескрипторов. Обозначим через $V(t)$ время передачи t элементов словаря с диска в оперативную память (в это время входит и среднее время доступа). Функция $V(t)$, выраженная при помощи принятых единиц измерения времени, является линейной, т.е. $V(t)=at+b$, где a, b - константы для конкретного типа дисковых устройств (для СДС-6200, см.[2]). Если L - объем дескрипторного словаря, то максимальное время для поиска одного дескриптора по первой схеме задается функцией

$$T(t) = L/t + t + V(t) = \frac{L}{t} + (a+1)t + b$$

(напомним, что t выражает емкость таблицы , L/t - емкость индексной таблицы).

Естественное стремление к минимизации времени поиска в словаре приводит к решению следующей задачи математического программирования:

минимизировать: $T(t) = \frac{L}{t} + (a+1)t + b$

при следующих ограничениях

$$0 \leqslant t \leqslant K_1$$

$$\frac{L}{t} \leqslant K_2$$

для целочисленных t и L/t .

Здесь K_1 и K_2 выражают ограничение, связанное с емкостью используемой оперативной памяти. Для решения этой задачи можно отыскать

абсолютный минимум T(t) и, если он не удовлетворяет данным ограничениям, совершить наименьший сдвиг от этой точки до попадания в область ограничений. Точка минимума этой функции получается для

$$t = \sqrt{\frac{L}{a+1}} .$$

Для второй схемы (двухуровневых таблиц индексов) функция T(t) имеет вид $T(t)=L/tc+c+(a+1)t + b$, где t имеет вышеуказанный смысл, c задает количество элементов индексной таблицы первого уровня, $\frac{L}{tc}$ - количество элементов индексной таблицы 2-го уровня. Соответствующая задача для минимизации формулируется так:

минимизировать $T(t) = L/tc + c + (a+1)t+b$

при ограничениях $0 \leqslant t \leqslant K_1$

$0 \leqslant c \leqslant K_2$

$L/tc \leqslant K_3$

для целочисленных $t, c, L/tc$,

где K_1, K_2 и K_3 выражают ограничения, связанные с емкостью используемой оперативной памяти.

Если в памяти необходимо сохранить только индексную таблицу второго уровня, то первые два ограничения редуцируются в

$$0 \leqslant \max(t,c) \leqslant K_1 .$$

Приближенное решение этой задачи получается при

$$c = \sqrt[3]{(a+1)L} \quad \text{и} \quad t = \sqrt[3]{L/(a+1)^2} .$$

Если время доступа к внешней памяти не учитывается, то в приведенных формулах надо положить a=0, и тогда для словаря с емкостью I2000 дескрипторов получается, что оптимальный размер таблицы будет 23 элемента.

В SCOPE 3.3 и 3.4 существует так называемая индексно-последовательная организация массивов, которая,по существу,реализует приведенные выше схемы (подробно об индексно-последовательной организации массивов см в[2]). В ИПС ОИЯИ этот словарь (ДК) реализован при помощи индексно - последовательной организации, где все параметры индексных таблиц и таблиц для данных вычислены с использованием библиотечной программы ESTMATE.

Надо отметить, однако, что, если потребитель не намерен использовать все дополнительные возможности, которые дает эта организация, тогда лучше реализовать словарь по приведенной выше схеме с использованием организации RELATIVE.

Индексно-последовательная организация массивов, кроме произвольного доступа по ключу, дает возможность последовательного чтения информации с произвольным началом, указанным потребителем. Основное достоинство этого метода, в связи с применением словаря, состоит в том, что добавление новых дескрипторов к словарю совершается автоматически, т.е. все проблемы в отношении определения места новой записи решаются системой. Рассмотрим этот процесс подробнее. Например, таблица (рис.2), в которой надо записать новый дескриптор, определяется таким же образом,как и при поиске. Затем, если в этой таблице есть свободное место, то дескриптор записывается туда, при этом сохраняется порядок следования (т.е. дескрипторы в каждой таблице записываются в восходящем порядке). Однако, если таблица заполнена, то запись засылается в новую таблицу, а в таблице, в которой он должен логически присутствовать, создается связь с новой таблицей.В таком случае при поиске этого дескриптора время поиска удлиняется из-за дополнительного чтения таблицы переполнения. Подобное может случиться и с индексными таблицами.

Можно предотвратить такие ситуации, резервируя место как в таблицах для данных, так и в индексных таблицах, которые в будущем будут использованы при пополнении словаря. Процент неиспользованных мест в таблицах надо вычислять с учетом ожидаемого количества новых дескрипторов. Поскольку все-таки создание таблиц переполнения предотвратить нельзя, после добавления определенного количества новых дескрипторов к словарю надо реорганизовать весь массив, т.е. создать его заново. Это легко сделать, используя возможности последовательного чтения индексно-последовательного массива,т.е. массив читается последовательно (при этом дескрипторы поступают в восходящем порядке),и после чтения каждой записи делается запись в новом словаре. Эта реорганизация освобождает место на диске, где хранится словарь, и в то же время улучшает характеристики поиска.

Подобные процедуры могут быть сделаны и при RELATIVE организации словаря. Здесь, однако, добавление новых дескрипторов к словарю не совершается автоматически, а должно быть предусмотрено в алгоритмах программы.

После реальных экспериментов со словарем типа ДК и ИПС ОИЯИ (SCOPE 3.3) получены следующие результаты.

Астрономическое время для формирования словаря, состоящего из I2000 дескрипторов, равно 30 с.

Среднее время для кодирования дескриптора при кодировании 6000 дескрипторов меньше чем 0.0I с.

В заключение надо отметить, что в ИПС в качестве постоянного словаря можно хранить только один из словарей двух типов,ДК и КД,- другой можно программно формировать во время поиска. Это позволяет сэкономить внешнюю память, отведенную для задач ИПС.

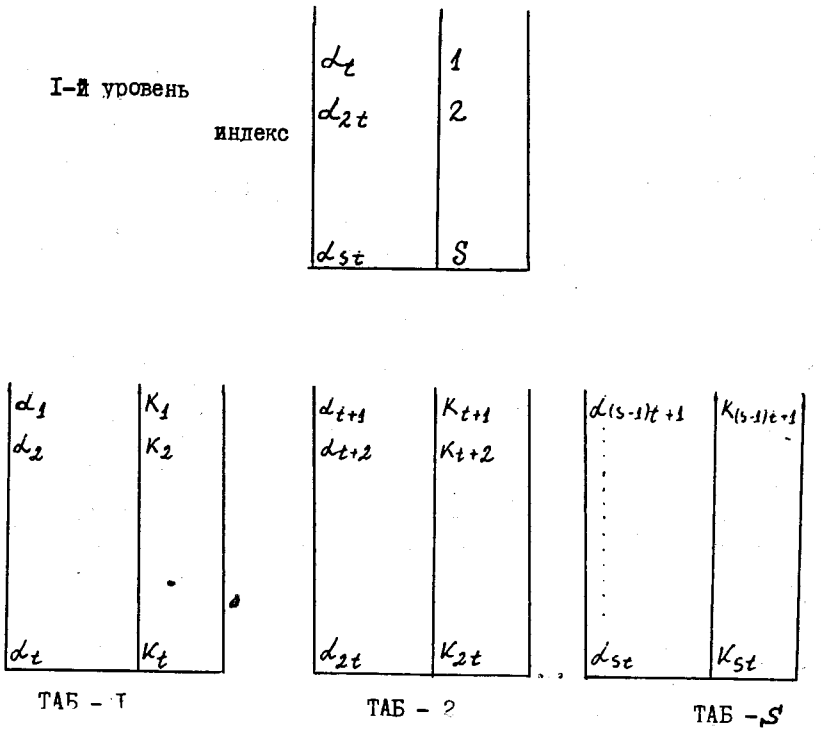

Рис. I

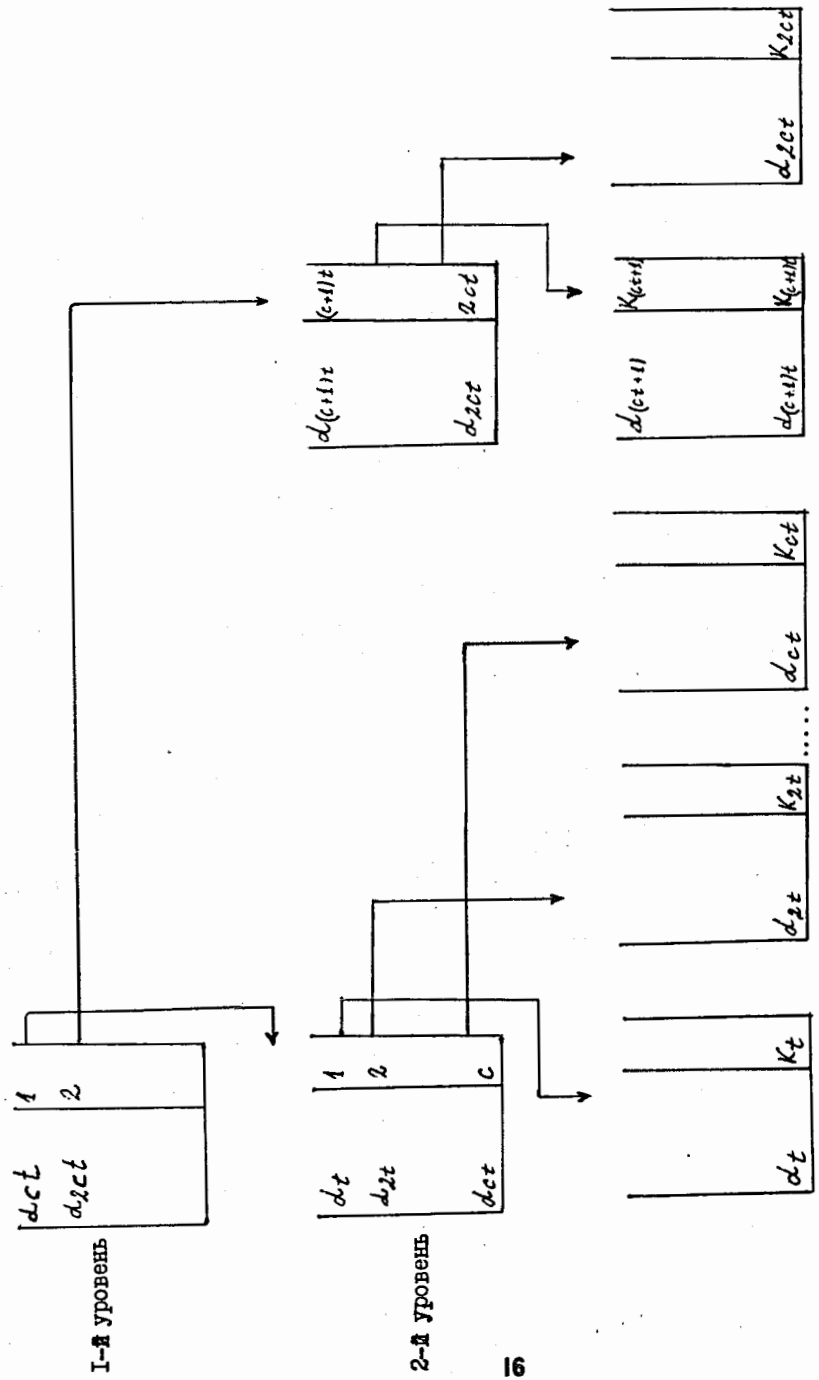

Рис.2

ЛИТЕРАТУРА

I. Д.Д.Арнаудов. Структурно-функциональная организация основных массивов ИПС ОИЯИ . Сообщение ОИЯИ, Р10-8621, Дубна, 1975.

2. Д.Д.Арнаудов. Анализ методов доступа информации к внешним ЗУ на магнитных дисках при работе информационно-поисковой системы (ИПС), реализованной на ЭВМ третьего поколения. Препринт ОИЯИ, 10-7953, 1974.

3. Scope Reference Manual 3.3., 1970, CDC,USA.

4. Thesaurus INIS.,Vienna,1971.

Рукопись поступила в издательский отдел
10 апреля 1975 г.