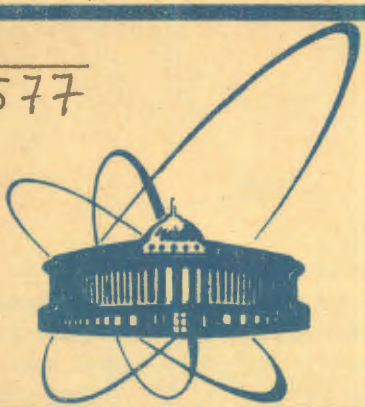

сообщения
объединенного
института
ядерных
исследований
дубна

20/VII-81

10-81-353

Н.Н.Говорун, В.И.Никитина, Г.Н.Тентюкова

ПОИСК ИНФОРМАЦИИ В СИСТЕМЕ "КАДРЫ"

1981

Поиск информации является одной из центральных функций любой информационной системы. При проектировании систем большое внимание уделяется разработке методов поиска, позволяющих удовлетворять требованиям достаточного быстродействия. Эти требования особенно важны в системах, работающих в оперативном режиме.

Разработка алгоритма поиска включает в себя решение вопросов организации информационных массивов, методов доступа, создания языка запроса.

Организация информационных массивов системы "Кадры"[1] и методы доступа к памяти проектировались с целью обеспечения быстрой выборки необходимой информации. Для организации файла документов применен метод частично-инвертированных списков[2]. Документы, являющиеся записями этого файла, содержат набор сведений об объектах системы и характеризуются набором признаков.

В системе введено понятие "линейки". Для определенного значения некоторого признака формируется "линейка". Это двоичное число, i -ый разряд которого равен единице, если i -ый документ содержит данное значение признака, и нулю в противном случае. Число единиц в линейке назовем ее длиной. Значения признаков, для которых система составляет линейки, являются ключевыми словами, входящими в справочник ключей.

Поиск информации в файле с инвертированной структурой выполняется в два этапа: поиск заданных элементов в справочнике ключей и вызов линейки из внешней памяти. Таблица справочника ключей упорядочена по ключам, поэтому применяется метод бинарного поиска заданных элементов. Его реализация не представляет труда, и поиск не требует больших затрат времени.

Система "Кадры" рассчитана на использование дисковых пакетов в качестве внешней памяти устройств прямого доступа. Вызов линейки из внешней памяти выполняется системой программ прямого доступа[3].

Для поиска информации пользователь задает поисковое предписание[4]:

$$ПП = T_1 \vee T_2 \vee \dots T_\ell \ , \qquad (1)$$

где

$$T_i = t_{i1} \wedge t_{i2} \wedge \dots t_{iN_i} \ . \qquad (2)$$

t_{ij} – термины запроса,

N_i – количество терминов запроса в конъюнктивной группе.

В качестве t_{ij} могут выступать:

- значение признака (интервал значений, сумма интервалов значений),
- некоторые функции от значений признаков,
- имя документа (часть имени),
- СФСЗ – скобочная форма списковой записи, в которой участвуют элементы разных экземпляров записей нескольких полей (т.е. нескольких подпризнаков) одного и того же признака.

Каждый t_{ij} (кроме имени документа) может иметь отрицание, т.е. знак логической инверсии.

В основе поисковой системы лежит принцип работы с линейками. Каждому t_{ij} сопоставляется линейка. Если t_{ij} является ключевым значением, то для него уже есть линейка, сосчитанная заранее и хранящаяся во внешней памяти ЭВМ. Линейки для неключевых значений вычисляются непосредственно при работе поисковой системы. В случае задания функций от значений признаков вычисление этих функций выполняется при составлении линеек.

Каждому t_{ij}, входящему в поисковое предписание, соответствует множество π_{ij} номеров документов, которому сопоставлена линейка L_{ij}.

Требуется найти π –множество номеров документов, соответствующих поисковому предписанию, заданному выражением (I). Очевидно,

$$\pi = \bigcup_{i=1}^{\ell} \bigcap_{j=1}^{N_i} \pi_{ij} . \qquad (3)$$

Если бы для каждого t_{ij} существовала линейка, записанная в память ЭВМ, то вся процедура поиска свелась бы к вычислению линейки:

$L = (L_{11} \wedge L_{12} \wedge \dots L_{1M1}) \vee (L_{21} \wedge L_{22} \wedge \dots L_{2M2}) \dots \vee (L_{\ell 1} \wedge \dots \wedge L_{\ell M\ell})$.

Но, поскольку в системе применен метод частично инвертированной организации массивов, в памяти ЭВМ хранятся линейки лишь для некоторого подмножества P из множества всех значений признаков. В это подмножество входят наиболее часто используемые в запросах значения признаков. Его размер определяется наличием свободной памяти на диске для хранения линеек.

Термины запроса t_{ij} могут быть заданы не обязательно значениями признаков. Для таких t_{ij}, а также для значений признаков, не принадлежащих P, нет заранее сформированных линеек, и их нужно получить в процессе поиска.

Т.к. для создания линеек требуется просмотр фонда документов, то нужно построить алгоритм поиска таким образом, чтобы минимизировать как количество обращений к внешней памяти ЭВМ, так и количество сопоставлений значений признаков в запросе и в документах для установления релевантности документа заданному запросу.

Рассмотрим выражение (2). Обозначим через π_i множество документов, соответствующих запросу, заданному посредством T_i, а через π_{oi} – пересечение множеств документов, соответствующих тем t_{ij}, для которых имеются линейки. Если ни для одного t_{ij} нет линеек, то в качестве π_{oi} выступает U – множество всех документов фонда

$$\pi_{oi} = \begin{cases} \bigcap_{j \in M_{oi}} \pi_{ij} \\ U \end{cases} .$$

Здесь M_{oi} – множество значений j, для которых t_{ij} имеют линейки. Тогда, очевидно,

$$\pi_i = \pi_{oi} \bigcap_{j \in M_i} \pi_{ij} = \bigcap_{j \in M_i} (\pi_{oi} \cap \pi_{ij}) .$$

Здесь M_i – множество значений j, для которых t_{ij} не имеют линеек.

Вместо вычисления линеек, соответствующих π_{ij}, достаточно вычислить линейки для $\pi_{oi} \cap \pi_{ij}$, т.е. требуется рассмотреть не весь фонд документов, а некоторое подмножество π_{oi} (за исключением случая $\pi_{oi} = U$). Если $\pi_{oi} = \emptyset$, то это означает, что множество документов, релевантных запросу, является пустым, и дальнейший поиск прекращается.

Процесс поиска информации, соответствующей поисковому предписанию (I), выглядит следующим образом. По таблице справочника ключей определяются адреса линеек во внешней памяти для всех t_{ij}, имеющих линейки. Эти линейки (L_{ij}) считываются в оперативную память ЭВМ. Затем вычисляются линейки, соответствующие π_{oi}, посредством логического умножения (пересечения) линеек L_{ij} для $j \in M_{oi}$ и для каждого i.

Если все t_{ij} имеют линейки, то на этом процесс поиска заканчивается. В противном случае вычисляются линейки, соответствующие множествам $\pi_{oi} \cap \pi_{ij}$ для $j \in M_i$. Для этого прежде всего определяется принадлежность очередного документа соответствующему множеству π_{oi}. Если он принадлежит π_{oi} хотя бы для одного значения i, то выполняется вызов из внешней памяти физической записи, содержащей этот документ. Затем производится распаковка полей признаков, значения которых заданы в t_{ij}, и засылка 0 или 1 в массивы, зарезервированные для формирования линеек.

Поскольку одна физическая запись содержит несколько документов с номерами, образующими возрастающую последовательность, то в процессе формирования линеек обращение к внешней памяти для выборки документов выполняется, практически, в режиме последовательного доступа. Вызываются только те физические записи, которые содержат номера документов, принадлежащих π_{oi} . Каждая физическая запись читается из внешней памяти не более одного раза для всего фонда документов и для всех формируемых линеек.

Формирование линеек для t_{ij} , выраженных значениями признаков, а также функциями значений признаков, выполняется в результате сопоставления значений, заданных в поисковом предписании, и значений, содержащихся в документах.

Для t_{ij} , заданных через СФСЗ, вычисление линеек выполняется по особому алгоритму. В этом случае требуется сравнение запроса с документом не для значений каждого подпризнака в отдельности, а по совокупности значений разных подпризнаков каждой записи признака. Но т.к. линейки для каждого значения подпризнака сосчитаны независимо, то нельзя искать линейку, соответствующую СФСЗ, простым пересечением линеек, соответствующих отдельным компонентам СФСЗ.

Если существуют линейки для некоторых значений подпризнаков, входящих в СФСЗ, то из них выбирается та, которая имеет наименьшую длину. Затем выполняется пересечение этой линейки с линейкой для π_{oi} . Линейка, полученная в результате, содержит номера документов, потенциально релевантных части поискового предписания, выраженного через СФСЗ. Далее производится вызов в оперативную память документов с номерами, выбранными из этой линейки, и сравнение предписания, заданного в СФСЗ, с соответствующими значениями подпризнаков, составляющими отдельную запись в документе.

Например, признак "Знание языков" включает два подпризнака: "Название языка" и "Степень знания языка". Каждая запись для признака "Знание языков" состоит из двух полей. Каждому названию языка соответствует своя степень знания. Если требуется найти номера анкет сотрудников, свободно владеющих английским языком, то из двух линеек, соответствующих значениям "Английский язык" и "Свободно владеет" следует выбрать линейку с наименьшей длиной. Затем, вызвав в оперативную память анкеты с номерами, входящими в полученную линейку, отобрать те, которые хотя бы в одной записи признака "Знание языков" имеют значения "Английский язык" и "Свободно владеет".

При наличии отрицаний в поисковом предписании выполняется вычисление линеек, обратных тем, которые соответствуют t_{ij} со знаком логической инверсии.

Рассмотрим теперь алгоритм поиска для случая задания t_{ij} через имя документа. Поскольку запрос по имени задается довольно часто, а имена документов не являются ключевыми значениями, то для этого случая разработан специальный алгоритм поиска.

Программа ввода и редактирования документов заносит начальный фрагмент (I3 символов) имени в таблицу имен, упорядоченную по возрастающему ключу, а остальная часть имени остается в поле документа. Поиск имени выполняется сначала по таблице. Если в таблице отсутствует искомый фрагмент, то поиск заканчивается. Если найдены одна или несколько записей, совпадающих с заданным текстом запроса, то далее выполняется вызов соответствующих документов из внешней памяти и сравнение заданного имени с именами документов. Результатом этого процесса является линейка с номерами документов, имеющих заданное имя.

Операции с линейками (сложение, пересечение, отрицание, вычисление длины линейки и значения элемента) выполняются системой программ, разработанной Ю.П.Залаторюсом [5].

Литература

I. Говорун Н.Н. и др. ОИЯИ, I0-II05I, Дубна, I977.

2. Мартин Дж. Организация баз данных в вычислительных системах. М., "Мир", I978.

3. Мазный Г.Л. ОИЯИ, II-9845, Дубна, I976.

4. Говорун Н.Н. и др. ОИЯИ, I0-II052, Дубна, I977.

5. Говорун Н.Н. и др. ОИЯИ, I0-I095I, Дубна, I977.

Рукопись поступила в издательский отдел
26 мая I98I года.