ОБЪЕДИНЕННЫЙ
ИНСТИТУТ
ЯДЕРНЫХ
ИССЛЕДОВАНИЙ
ДУБНА

Р11 - 9770

Д.Д.Арнаудов, Н.И.Янев

ОБ ОДНОМ СПОСОБЕ ПРИМЕНЕНИЯ ЧАСТИЧНО-ЦЕЛОЧИСЛЕННОГО ЛИНЕЙНОГО ПРОГРАММИРОВАНИЯ ДЛЯ ОПТИМИЗАЦИИ ПОИСКА В МНОГОУРОВНЕВЫХ ИПС

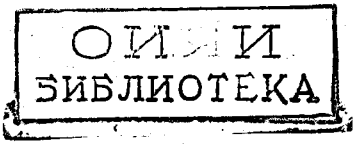
1976

P11 - 9770

Д.Д.Арнаудов, Н.И.Янев

ОБ ОДНОМ СПОСОБЕ ПРИМЕНЕНИЯ ЧАСТИЧНО-ЦЕЛОЧИСЛЕННОГО ЛИНЕЙНОГО ПРОГРАММИРОВАНИЯ ДЛЯ ОПТИМИЗАЦИИ ПОИСКА В МНОГОУРОВНЕВЫХ ИПС

Направлено в журнал " Программирование"



Арнаудов Д.Д., Янев Н.И. Р11 - 9770

Об одном способе применения частично-целочисленного программирования для оптимизации поиска в многоуровневых ИПС

Проблема оптимизации структуры многоуровневой информационно-поисковой системы рассматривается как задача частично-целочисленного линейного программирования большой размерности. Предложен эффективный алгоритм типа "ветвей и границ" для решения подобных задач.

Работа выполнена в Лаборатории вычислительной техники и автоматизации ОИЯИ.

Препринт Объединенного института ядерных исследований
Дубна 1976

Arnaudov D.D., Yanev N.I. P11 - 9770

An Application of Mixed Integer Programming for Minimizing Retrieval Time in Multilevel IRS

A large scale mixed integer programming model is used for representing a problem of multilevel information retrieval system structure optimizing. An effective branch and bound algorithm is proposed for solving such problems.

The investigation has been performed at the Laboratory of Computing Techniques and Automation, JINR.

Preprint of the Joint Institute for Nuclear Research
Dubna 1976

Теоретические вопросы организации многоуровневой ИПС подробно изложены в работе[I]. Там показано, что для оптимизации процесса поиска необходимо произвести уплотнение узловых списков основного информационного массива. Этот процесс не приводит к тому, что система будет отвечать на запросы с определенной комбинацией дескрипторов более эффективным образом, чем на запросы с любой другой комбинацией дескрипторов, а скорее приводит к оптимизации поиска в среднем, независимо от взаимосвязи ключевых терминов в запросе. Необходимо отметить, что весь этот процесс может быть проведен путем регистрации в течение некоторого периода времени фактических взаимосвязей, которые имеют место в запросах, и последующего изменения длины списков применительно к этим взаимосвязям без использования принципа равномерных взаимосвязей, но для этого необходимо накопить довольно большую статистику[2].

Здесь рассмотрим способ автоматической сортировки, который основан на применении некоторых методов частично-целочисленного линейного программирования.

I. Постановка задачи и ее особенности

Пусть задано конечное множество T объектов- $t_i (i=1,...n)$, множество D признаков $d_j (j=1,...k)$ и однозначное соответствие $\varphi: T \to 2^D$. Пусть M - произвольное подмножество T с мощностью $|M|=m < n$. Определим $\varphi(M) = \bigcup_{t_i \in M} \varphi(t_i)$ ($\cup$ обозначает операцию объединения множеств). Ставится задача о нахождении $\min(\varphi(M))$ при ограничениях: $M \subset T$, $|M| = m$. Другими словами, из $\binom{n}{m} = \frac{n!}{m!(n-m)!}$ множеств необходимо найти то множество, которое имеет наименьшее число элементов. Отметим сразу, что в смысле интересующих нас проблем ($n, m \sim$ нескольких тысяч) использование метода полного перебора для решения этой задачи исключается. Представим ее как задачу дискретного программирования следующим образом. Пронумеруем все элементы множества D числами от I до K . Каждый $t_i \in T$ представим как бинарный вектор $t_i = (a_{i_1}, a_{i_2}, \ldots, a_{i_k})$, где

$$a_{ij} = \begin{cases} 1, & \text{если } d_j \in \varphi(t_i) ; \\ 0, & \text{если } d_j \notin \varphi(t_i) . \end{cases}$$

Сопоставим каждому t_i двоичную переменную x_i , с помощью которой будем строить всевозможные выборки $M \subset T$, т.е. если $x_i=1$, то t_i входит в выборку M , иначе $x_i=0$. При этих обозначениях получаем следующее.

Задача А.

Найти $$\min \to Z = \sum_{i=1}^{K} y_i - \sum_{i=1}^{K} \max(0, y_i - 1) \qquad (I)$$

при ограничениях:

$$\sum_{j=1}^{n} a_{ij} x_j = y_i, \quad i=1,...k; \qquad (2)$$

$$\sum_{j=1}^{n} x_j = m ; \qquad (3)$$

$$x_j \in \{0,1\} . \qquad (4)$$

Ограничение (3) (учитывая (4)) задает требуемый размер выборки (мощность множества M). В условии (2) y_i показывает, сколько раз i-ый элемент множества D участвует в описании элементов $t_i \in T$, попавших в выборку M . Целевая функция (I) задает мощность (число различных элементов) M . Чтобы освободиться от нелинейности целевой функции (I), сформулируем задачу А следующим образом.

Задача В.

Найти $$\min \to V = \sum_{i=1}^{K} v_i \qquad (I')$$

при ограничениях:

$$\sum_{j=1}^{n} a_{ij} x_j = \ell (v_i - z_i), \quad i=1,...k ; \qquad (2')$$

$$\sum_{j=1}^{n} x_j = m ; \qquad (3')$$

$$0 \le z_i \le 1 - \frac{1}{\ell} , \quad i=1,...K ; \qquad (4')$$

$$x_j, v_i \in \{0,1\} , \quad \begin{matrix} i=1,...K ; \\ j=1,...n . \end{matrix} \qquad (5')$$

Здесь $\ell = \max\limits_{i \in \{1,...K\}} \sum\limits_{j=1}^{n} a_{ij}$.

Заметим сразу, что из условий (2'), (4'), (5') следует, что если $v_i=0$, то и $z_i=0$.

Покажем эквивалентность задач А и В, т.е. если вектор $X=(x_1, x_2 \ldots x_n)$ оптимален для задачи А, то он оптимален и для задачи В, и наоборот. Для каждого возможного вектора X значение целевой функции (I) равняется числу ненулевых компонент вектора $Y = (y_1, y_2 \ldots y_k)$, а значение целевой функции (I') равняется числу ненулевых компонент вектора $V = (v_1, v_2 \ldots v_n)$. Утверждение эквивалентности следует из следующей связи между компонентами векторов Y и V при фиксированном векторе X , удовлетворяющем условию (3):

Если $y_i > 0$, то $v_i = 1$; и если $v_i = 1$, то $y_i > 0$;

Если $y_i = 0$, то $v_i = 0$; и если $v_i = 0$, то $y_i = 0$.

Поскольку задача B принадлежит классу задач частично-целочисленного линейного программирования, то для ее решения можно использовать некоторые из методов, рассмотренных в /5/.

Однако использование алгоритмов с общим предназначением для решения задач, имеющих структуру задачи B и размерность $K \geq 2000$, $n \geq 2000$, пока является иллюзорным. Это видно из результатов, опубликованных в /6/ для экспериментов с большими линейно-целочисленными задачами. Поэтому для решения специальных классов задач требуется создание специализированных алгоритмов, учитывающих особенности решаемой задачи. Ниже рассматривается алгоритм для решения поставленной задачи. Прежде всего займемся теоретическим обоснованием алгоритма.

2. Исследование непрерывной задачи

Рассмотрим задачу B_R, которая получается из задачи B заменой ограничения (5) неравенствами: $0 \leq X_j \leq 1$ для $j=1,\dots n$; $0 \leq V_i \leq 1$ для $i=1,\dots K$. Введем следующие обозначения:

$$C_j = \sum_{i=1}^{K} a_{ij} \quad \text{для} \quad j=1,\dots n.$$

Расставим стобцы матрицы $A = \| a_{ij} \|$ так, чтобы $C_1 \geq C_2 \geq \dots C_n$. Пусть векторы C, X, Y, X^*, Y^*, Y_o равны, по определению:

$$C = (C_1, C_2, \dots C_n);$$

$$X = (X_1, X_2, \dots X_n);$$

$$Y = (Y_1, Y_2, \dots Y_{n+2K});$$

где $Y_i = \begin{cases} V_i & \text{для } i = 1,\dots K; \\ X_i & \text{для } i = K+1,\dots,K+n; \\ Z_i & \text{для } i = K+n+1,\dots,n+2K; \end{cases}$

если все $Z_i = 0$, обозначим Y через Y_o.

$X^* = (X_1^*, \dots, X_n^*)$,
$Y^* = (Y_1^*, Y_2^*, \dots Y_{n+2K}^*)$, где $X_i^* = \begin{cases} 0, & i=1,\dots n-m; \\ 1, & i=n-m+1,\dots n. \end{cases}$

где $Y_i^* = \begin{cases} \frac{1}{\ell} \sum_{j=n-m+1}^{n} a_{ij}, & i = 1, \dots K; \\ X_{i-k}^*, & i = K+1,\dots K+n; \\ 0, & i = K+n+1,\dots n+2K. \end{cases}$

Определим задачу B_R'.

Найти $\min Z = \sum_{j=1}^{n} C_j X_j$

при ограничениях:

$$\sum_{j=1}^{n} X_j = m;$$

$$0 \leq X_j \leq 1; \quad j = 1,\dots n.$$

Теорема I. Оптимальный план задачи B_R задается вектором Y^*, и оптимальное значение целевой функции равняется

$$V^* = \frac{1}{\ell}(C, X^*).$$

Доказательство: доказательство теоремы начнем с установления эквивалентности задач B_R и B_R', т.е. если Y оптимален для B_R, то X оптимален для B_R', и обратно: если X оптимален для B_R', то Y_o оптимален для B_R.

Действительно, пусть X оптимален для B_R' со значением целевой функции $Z(X)$. Тогда Y_o является возможным планом для задачи B_R и значение целевой функции $V(Y_o) = \frac{Z(X)}{\ell}$. Если, допустим, что существует $Y_o^1 \neq Y_o$, для которого $V(Y_o^1) < V(Y_o)$, то есть $\frac{Z(X_o)}{\ell} < \frac{Z(X)}{\ell}$, то X_o будет планом задачи B_R' лучшим, чем X, что противоречит оптимальности X. Эквивалентность в обратном направлении доказывается аналогично.

Из основной теоремы линейного программирования /7/ следует, что оптимальные планы для задачи B_R' следует искать среди опорных планов множества ограничений, которые для задачи B_R являются бинарными n-мерными векторами с m ненулевыми компонентами.

Доказательство теоремы I уже непосредственно следует из Леммы I (см.ниже).

Действительно, целевая функция задачи B'_R задается скалярным произведением вектора C, компоненты которого расставлены в убывающем порядке, и вектора X, который принадлежит множеству n-мерных бинарных векторов с $m < n$ ненулевыми компонентами. Из Леммы I следует, что минимум этого скалярного произведения достигается для вектора X, компоненты которого расставлены в возрастающем порядке, т.е. это вектор X^*.

Лемма I. Пусть числа $X_1, X_2 \ldots X_n$ и $Y_1, Y_2, \ldots Y_n$ удовлетворяют условиям:

$$X_1 \leq X_2 \leq \ldots \leq X_n ;$$

$$Y_1 \geq Y_2 \geq \ldots \geq Y_n .$$

Тогда для каждой перестановки $K_1, K_2 \ldots K_n$ выполняется неравенство

$$\sum_{i=1}^{n} X_i Y_i \leq \sum_{i=1}^{n} X_i Y_{k_i} .$$

Доказательство: в /8/ эта лемма доказана для неотрицательных чисел; здесь дадим более простое доказательство, которое выполняется для произвольных чисел.

Отметим, что

а) $X_s - X_{s-1} \geq 0$ для $S = 2,3,\ldots n$;

б) $[Y_s + Y_{s+1} + \ldots + Y_n - (Y_{k_s} + Y_{k_2} + \ldots + Y_{k_n})] \leq 0$, $S=2,3\ldots n$.

Тогда получаем:

$$\sum X_i Y_i - \sum X_i Y_{k_i} =$$
$$= X_1 [Y_1 + Y_2 + Y_3 + \ldots + Y_n - (Y_{k_1} + Y_{k_2} + \ldots Y_{k_n})] +$$
$$+ (X_2 - X_1)[Y_2 + Y_3 + \ldots + Y_n - (Y_{k_2} + \ldots + Y_{k_n})] +$$
$$+ (X_3 - X_2)[Y_3 + \ldots + Y_n - (Y_{k_3} + \ldots Y_{k_n})] +$$
$$\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots$$
$$(X_n - X_{n-1})[Y_n - Y_{k_n}] \leq 0 .$$

Из теоремы I видно, что оптимальное решение непрерывной задачи B_R существенно зависит от ℓ (частота наиболее встречаемого признака из множества D, который использован для идентификации объектов t_i). Поскольку ℓ прямо пропорционально n (число объектов множества D, из которого делается оптимальная выборка), может оказаться, что v^* намного меньше оптимального значения целевой функции целочисленной задачи B. Поэтому использование решения непрерывной задачи как критерия "усечения" множества возможных решений в некоторых задачах будет неэффективно.

Чтобы улучшить нижнюю оценку для оптимального решения дискретной задачи, модифицируем модель B, вводя в ограничение (2') вместо ℓ константу ℓ_i, где $\ell_i = \sum_{j=1}^{n} a_{ij}$ для $i=1,\ldots K$. Таким образом, получаем:

Задача F :

Найти $\min \rightarrow W = \sum_{i=1}^{K} V_i$ (I")

при ограничениях:

$$\ell_i (V_i - Z_i) = \sum_{j=1}^{n} a_{ij} x_j , \quad i=1,\ldots K; \qquad (2")$$

$$\sum x_j \leq m; \qquad (3")$$

$$0 \leq Z_i \leq \frac{\ell_i - 1}{\ell_i} , \quad i=1,\ldots K ; \qquad (4")$$

$$X_j, V_i \in \{0,1\} , \quad j=1,\ldots n ; \quad i=1,\ldots K. \qquad (5")$$

Отметим, что утверждения теоремы I остаются в силе и для релаксированной задачи F_R (которую получаем из F, опуская требования для целочисленности V_i и X_j), если положим, что

$$c_j = \sum_{i=1}^{K} \frac{a_{ij}}{\ell_i} , \quad j=1,\ldots,n.$$

Следующая теорема показывает связь между оптимальными решениями задач B_R и F_R.

Теорема 2. Если V^* и W^* – оптимальные значения целевых функций задач B_R и F_R, то $V^* \le W^*$.

Доказательство: поскольку $\ell = \max_{i \in \{1,\dots K\}} \ell_i$ (по определению), то $\ell \ge \ell_i$ для всех $i=1,\dots,K$; отсюда для $j=1,\dots,n$, $\frac{1}{\ell}\sum_{i=1}^{K} a_{ij} \le \sum_{i=1}^{K} \frac{a_{ij}}{\ell_i}$, что вместе с утверждением теоремы I дает $V^* \le W^*$.

Отметим, что достижение равенства возможно в очень редких случаях, которые характеризуются равенством почти всех ℓ_i.

Полученные результаты показывают:

а) в алгоритмах, использующих решения непрерывных задач в качестве механизма усечения, лучше применить модель B (которая более проста для программной реализации) для решения задач, в которых распределение признаков по объектам одинаково;

б) во всех остальных случаях использование модели F намного предпочтительнее.

Особенно сильно проявляется преимущество модели F при решении информационных задач с распределением признаков по документам по закону Ципфа. Напомним, что по этому закону вероятность P_i появления дескриптора α_i в случайно выбранном документе равняется $P_i = \frac{1}{i(\ell n n + \gamma)}$, где i – ранговый номер дескриптора (т.е. дескрипторы пронумерованы по убыванию частоты встречаемости), n – мощность тезауруса, γ – константа Эйлера.

3. Алгоритм решения задачи

Алгоритм удобно представить состоящим из двух частей.

I часть.

Этот алгоритм вычисляет начальную оценку для целевой функции задачи F, находит первое возможное решение и задает порядок фиксации переменных v_i.

Шаг I. Решаем непрерывную задачу F_R (см. алгоритм SELECT).

Шаг 2. Вычисляем $R = \sum_{i=1}^{n} \lceil v_i \rceil$, S=0.

Шаг 3. Перенумеруем переменные V_i следующим образом:

Все нулевые переменные V_i в оптимальном решении F_R получают номера от 1 до K-R.

Все ненулевые переменные V_i получают номера от K-R+1 до K.

Шаг 4. Для $i=1,\dots,K-R-1$ фиксируем $V_i = 0$;

положим L=K-R; если $L = 0$, то

положим L=1; фиксируем $V_L = 1$, S=S+1; переходим ко П части (шаг I).

Шаг 5. Если возможна фиксация $V_L = 1$, то фиксируем $V_L = 1$, S=S+1; переходим ко П части (шаг I).

Шаг 6. L=L-1; аннулируется фиксация $V_{L+1} = 0$; переходим к шагу 5.

П часть

Этот алгоритм решает задачу F.

Шаг I. Решаем непрерывную задачу F_{CR}. Если $V^*_{CR} \ge R$ и $V_L = 0$, то переходим к шагу 5, иначе ($V_i > 0$) – к шагу 7.

Шаг 2. Если $R_C < R$, вычисляем $R = R_C$, запоминаем (печать) R, X^*_{CR};

Шаг 3. L=L+1, если $L > K$, то переходим к шагу 6.

Шаг 4. Если $V_L = 0$, переходим к шагу 3, иначе, если возможна фиксация $V_L = 0$, то фиксируем $V_L = 0$ и переходим к шагу I.

Шаг 5. Если $S + 1 < R$, фиксируем $V_L = 1$, S=S+1, переходим к шагу I.

Шаг 6. L=L-1, если $L = 0$ – конец.

Шаг 7. Если $V_L=1$, вычисляем $S=S-1$, переходим к шагу 6, иначе – переходим к шагу 5.

Данный алгоритм принадлежит к классу комбинаторных алгоритмов типа ветвей и границ. В нём существенно учитывается специфика задачи. Алгоритм позволяет реализацию в двух вариантах:

а) компактный – без сохранения симплексных таблиц;

б) с сохранением симплексных таблиц.

Замечания к I части алгоритма

Шаг I. Решение непрерывной задачи F_R, согласно Теореме I, сводится к отысканию m наименьших чисел из n. Для решения этой задачи ниже приводится специальный алгоритм SELECT. Оптимальное решение непрерывной задачи характеризуется тем важным свойством, что все X_i – целочисленные (Теорема I), т.е. мы сразу получаем возможное решение задачи F. Действительно, если $Y=(V_1,V_2...V_k,X_1,X_2...X_n,0,...0)$ – оптимальное решение F_R, то $Y^*=(\lceil V_1\rceil,\lceil V_2\rceil,..\lceil V_k\rceil,X_1,X_2,...X_n,\lceil V_1\rceil-V_1,..\lceil V_k\rceil-V_k)$ – возможное решение задачи F со значением целевой функции $\sum_{i=1}^{K}\lceil V_i\rceil$. (Здесь X означает наименьшее целое число, которое больше или равно X).

Шаг 2. Значение целевой функции задачи F запоминается в R. В S будут подсчитаны все V_i, которые фиксируются равными единице.

Шаг 3. Определяется порядок фиксирования переменных на основе оптимального решения задачи F_R. Заметим, что фиксированию (ветвлению) подлежат только переменные V_i, так как в оптимальном решении соответствующих подзадач F_{CR} все X_j – целочисленные (F_{CR} – это подзадача, которая получается из F_R при фиксировании некоторого V_i). Первыми будут фиксированы все V_i, которые в оптимальном решении F_R имеют значение 0. Число этих переменных равно $K-R$. Внутри этой группы переменных

порядок фиксации определяется по мере убывания ℓ_i. Заметим, что таким образом уменьшаются шансы правостороннего ветвления переменных этой группы. Вторая группа переменных, подлежащих ветвлению, – это те V_i, которые в оптимальном решении F_R имеют ненулевые значения. Заметим, что с получением первого возможного решения задачи F множество исследуемых узлов уменьшается на 2^R элементов (если $R<K$).

Шаг 4. Все V_i (кроме последней $i=L-R$) первой группы фиксируем на 0.

Шаг 4 и Шаг 5. Начиная с $i=L-R$, с шагом -1 ищем первую возможную фиксацию $V_i=1$.

Шаг 6. Аннулируем все фиксации переменных первой группы, для которых невозможно правостороннее ветвление ($V_L=0$).

Замечания ко II части алгоритма

Шаг I. Решается непрерывная задача F_{CR}, которая получается из задачи F_R после фиксации переменной V_L.

Если минимальное значение целевой функции V^*_{CR} больше или равно R (верхняя граница для оптимального значения целевой функции V задачи F), то фиксация следующих переменных $V_i (i>L)$ бесперспективна (см, например, /5/). Имеем два возможных случая:

а) если $V_L=0$, будем фиксировать её на I;

б) если $V_L=1$, возвращаемся назад (уменьшаем L), пока не найдётся L, для которого $V_L=0$.

Шаг 2. Оптимальному плану X^*_{CR} задачи F_{CR} соответствует возможное решение задачи F (см. шаг I, часть I) со значением целевой функции R_C. Если R_C меньше верхней границы R, то R улучшается ($R=R_C$), а параметры возможного решения X^*_{CR} (напоминаем, что компоненты этого вектора задают выборку из множест-

ва T) и R_C запоминаются (чаще всего выдаются на печать).

Шаг 3. Переходим к фиксации следующей по порядку переменной V_i. Если все переменные V_i уже фиксированы, переходим к шагу 6 (возврат).

Шаг 4. Если переменная V_L, подлежащая фиксации, равна нулю в результате фиксации переменных V_i для $i < L$; то сразу переходим к фиксации следующей переменной. В противном случае делается попытка левостороннего ветвления, т.е. фиксация на 0. Если это возможно, фиксируем $V_L = 0$ и переходим к получению новой нижней оценки (решение непрерывной задачи).

Шаг 5. В этом шаге переменная V_L фиксируется на I. Эта фиксация имеет смысл только, в том случае, если число S переменных V_L, фиксированных на I, остаётся меньше R.

Шаг 6 и 7. Обращение к этим шагам алгоритма происходит только в тех случаях, когда множество возможных значений текущей переменной V_L исчерпано или $L > K$, т.е. все переменные фиксированы. В таких случаях необходимо вернуться назад (уменьшить L) до первой $V_L = 0$, для которой фиксация $V_L = I$ возможна.

$L = 0$ соответствует установлению оптимальности последнего отпечатанного возможного решения. Остановимся на некоторых существенных особенностях данного алгоритма. Здесь, как и в большинстве алгоритмов типа "ветвей и границ", основным решающим механизмом для углубления ветвления (см. терминологию в /5/) является решение непрерывной задачи. При этом, чем ближе оптимальное решение непрерывной задачи к оптимальному решению дискретной проблемы, тем больше шансов для уменьшения числа исследуемых узлов. Однако оптимальное решение непрерывной задачи раз и навсегда определяется выбором параметров модели. Поэтому в

задачах минимизации как эвристического критерия для выбора переменной, подлежащей фиксированию, чаще всего используется максимизация ожидаемого наращивания оптимального решения непрерывной задачи. В этом смысле модель F оказывается очень гибкой, поскольку ожидаемым наращиванием оптимального решения непрерывной задачи можно управлять не только выбором целочисленных переменных, а также и изменением величины ℓ_i. Действительно, пусть возможна фиксация $V_p = 0$.

Тогда мы получим следующую подзадачу F_C :

найти $\min \rightarrow V_C = \sum_{i \in I \setminus p} V_i$

при ограничениях:

$$V_i = \frac{1}{\ell_i} \sum_{j \in J \setminus I_p} a_{ij} x_j + Z_i , \qquad i \in I \setminus p ;$$

$$\sum_{j \in J \setminus I_p} x_j = m$$

$$x_j , V_i \in \{0,1\} , \quad j \in J \setminus I_p ;$$

$$0 \le Z_i \le 1 - \frac{f}{\ell_i} , \quad i \in I \setminus p ,$$

где

$I = \{1,2,\ldots,k\}$;

$J = \{1,2,\ldots,n\}$;

$I_p = \{j / a_{p,j} = 1\}$;

$\ell_i = \sum_{j=1}^{n} a_{i,j}$; $\quad i \in I$.

Как уже доказали, оптимальное решение непрерывной задачи получается для $X = (x_1, x_2, \ldots x_n)$, причём

$$x_j = \begin{cases} 1, & \text{если } j \in J_{min} ; \\ 0, & \text{если } j \notin j_{min} . \end{cases}$$

Множество J_{min} содержит m индексов наименьших элементов c_j^p, где

$$c_j^p = \sum_{i \in I|p} \frac{a_{ij}}{\ell_i} \quad , \quad j \in J \setminus I_p .$$

Минимальное значение целевой функции на множестве ограничений в этом случае равно:

$$V_{CR}^{*} = \sum_{j \in J_{min}} c_j^p \quad .$$

Смысл подзадачи F_c не изменится (см. I), если вместо ℓ_i в модель подставим величину

$$\ell_i' = \sum_{j \in J \setminus I_p} a_{ij} \quad , \quad i \in I .$$

Тогда минимальное значение целевой функции на множестве ограничений будет:

$$V_{CR}' = \sum_{j \in J'_{min}} c_j' , \quad \text{где}$$

$$c_j' = \sum_{i \in I|p} \frac{a_{ij}}{\ell_i'} \quad , \quad j \in J \setminus I_p .$$

(Множество J'_{min} определено аналогично J_{min} на множестве чисел c_j'. Однако $\ell_i' \le \ell_i$ для $i \in I$, следовательно, $c_j' \ge c_j^p$, отсюда следует, что $V_{CR}' \ge V_{CR}^{*}$.

Следует отметить, что поскольку реализация этой модификации связана с бо́льшим числом операций, чем реализация модели с постоянным ℓ_i, то её использование будет эффективным, когда вероятность выполнения равенства $V_{CR}' = V_{CR}^{*}$ мала. Однако основное преимущество модели с переменными ℓ_i состоит в возможности получать целочисленные решения непрерывной задачи. Для модели с постоянными ℓ_i это является маловероятным событием

(см. модель задачи F). Заметим, что вероятность получения целочисленного решения стремится к I, когда число элементов множества $J \setminus (I_p \cup I_q \ldots \cup I_t)$ приближается к m (p,q,....,t - индексы переменных V , которые фиксированы на 0). Когда $|J \setminus (I_p \cup I_q \ldots \cup I_t)| = m$, оптимальное решение задачи F_{CR} - целочисленное. Это приводит к значительному уменьшению числа исследуемых узлов.

Рассмотрим, как влияют на модель правосторонние фиксации. Итак, пусть возможна фиксация $V_p = 1$. С добавлением этого ограничения к множеству ограничений задачи F после соответствующих преобразований получим следующую подзадачу F_c :

найти

$$\min \rightarrow V_c = 1 + \sum_{i \in I \setminus p} V_i$$

при ограничениях:

$$V_i = \frac{1}{\ell_i} \sum a_{ij} x_j + Z_i \quad \text{для} \quad i \in I \setminus p ;$$

$$\sum_{j \in J} x_j = m \ ;$$

$$\sum a_{pj} x_j + Z_p = 1 \ ;$$

$$x_j, V_i \in \{0,1\} \quad , \quad j \in J \ ;$$

$$0 \le Z_i \le 1 - \frac{1}{\ell_i} \quad , \quad i \in I \setminus p .$$

Если опустим ограничения на дискретность, получаем непрерывную задачу F_{CR}, оптимальное решение которой можно получить, решив следующую задачу (Теорема I):

найти

$$\min \rightarrow \sum_{j \in J} c_j^p \, x_j$$

при ограничениях

$$\sum_{j \in J} x_j = m \ ;$$

$$\sum_{j \in I_p} x_j \ge 1 \ ;$$

$$0 \le x_j \le 1 \ ; \quad j \in J .$$

Эту задачу можно решить, используя утверждение Теоремы I следующим образом.

Разобьём множество J на подмножества $Jmin$ и $Jmax$ (с мощностью m и $n-m$) такие, чтобы $Jmin \cap Jmax = \emptyset$, $Jmin \cup Jmax = J$, причём, чтобы для произвольных $r \in Jmin$, $q \in Jmax$ выполнялось неравенство $C_r^p \leq C_q^p$. Если $Jmin \cap I_p \neq 0$, тогда множество J_{min} определяет оптимальное решение задачи. Если $Jmin \cap I_p = \emptyset$, тогда оптимальное решение задаётся множеством J'_{min}, получаемым из множества J_{min}, в которое вместо элемента j^* ($C_{j^*}^p = \max C_j^p$ $j \in Jmin$) вводится элемент q^* ($C_{q^*}^p = \min C_q^p$ $q \in I_p$).

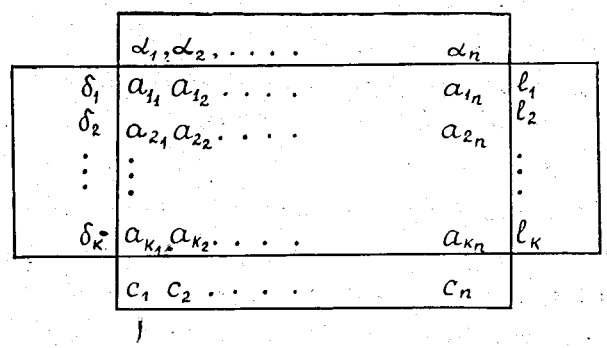
4. Программная реализация и экспериментальные результаты

Анализ приведённого алгоритма для задачи P показывает особую роль решения непрерывной задачи. Так как это наиболее часто употребляемая процедура, трудоёмкость алгоритма определяется числом операций, выполняемых при решении непрерывной задачи. Поэтому ниже приводится быстродействующий алгоритм для решения такой задачи.

Согласно Теореме I, решение задачи F_R (см. 2) сводится к решению следующей задачи выбора: переставить элементы множества $X[1:N]$ так, что для заданного $1 \leq M \leq N$ $X[M]$ будет содержать M-ый минимальный элемент, т.е. для $1 \leq I \leq M$ $X[I] \leq X[M]$ и для $M \leq I \leq N$ $X[I] \geq X[M]$. Оказывается, что для решения этой задачи можно использовать идею одного из методов внутренней сортировки, т.н. QUICK SORT (быстрая

сортировка) /9/. Первый алгоритм [FIND] для решения этой проблемы опубликован в /10/. Позднее был опубликован /11/ алгоритм [SELECT], характеристики которого приблизительно в два раза лучше, чем у FIND. Показано, что число операции сравнения в алгоритме SELECT асимптотически пропорционально $N + \min(N-M, M)$. Здесь приводится формальное описание алгоритма на языке Algol 60 (в /11/ алгоритм использует рекурсивное обращение к себе, которое делает его неприменимым для ряда ЭВМ).

Расположение информационных массивов в оперативной памяти ЭВМ показано на рис. I.



Массив $\|a_{ij}\|$ $i = 1,\ldots K$; $j = 1,\ldots n$ организован как одномерный с использованием мультисписковых структур. Каждый элемент $a_{ij} = I$ записывается как следующая логическая структура:

i ; j ; P(i), где i – номер строки (признака); j – номер столбца (объекта, документа); P(i) – относительный адрес следующего ненулевого элемента в i-ой строке. P(i) = 0 означает конец строки. Ненулевые элементы a_{ij} записываются по

столбцам. Каждое δ_i ассоциируется с переменной V_i и несёт следующую информацию: а) указывает адрес первого ненулевого элемента i-ой строки матрицы $\|a_{ij}\|$ и б) указывает значение переменной V_i, т.е. если $\delta_i < 0$, то V_i фиксировано на нуле; если $\delta_i > 0$, то V_i фиксировано на I.

Аналогичную функцию выполняют числа α_j по отношению к переменным X_j. α_j указывает относительный адрес первого ненулевого элемента j-ого столбца. Кроме этого, если $\alpha_j < 0$, то это означает, что X_j фиксировано на 0, если $\alpha_j > 0$, то X_j ещё не фиксировано. Числа ℓ_i и c_j имеют смысл, указанный в 3. Представление матрицы $\|a_{ij}\|$ в виде мультисписковой структуры осуществляется специальной программой CONDENSE, написанной на языке КОБОЛ.

Программа, реализующая алгоритм для решения задачи оптимального выбора, написана на языке FORTRAN IV и предназначена для выполнения на ЭВМ СДС. Константы типа INTEGER в этой машине записываются 60-битными словами, что является лишней затратой памяти для тех типов операций и значений переменных или констант, которые применяются в алгоритме. Поэтому, чтобы оперативная память использовалась рационально, приходится программно расщеплять машинные слова на сегменты, которые служат для представления программных переменных типа INTEGER. Такая редукция длины слова всегда приводит к увеличению времени работы программы, однако позволяет существенно увеличить размерность решаемых задач.

Для решения непрерывной задачи используется описанный выше алгоритм SELECT с некоторой модификацией. Дело в том, что числа c_j, среди которых ищется m наименьших, связаны

с конкретной переменной X_j. Это означает, что в процессе выполнения алгоритма SELECT необходимо следить за перемещением индекса j, связанного с константой c_j. Поэтому все операции, которые указаны в алгоритме оператором

exchange(X[I], X[j]), выполняются не над самими числами $c_j(X[j])$, а над их индексами (j).

При решении задач больших размерностей часто возникают проблемы, затрагивающие скорее эффективность программы, чем эффективность алгоритма. Одной из таких проблем здесь является определение числа ненулевых переменных V_i в результате фиксации переменных X_j (см. шаг 2 второй части алгоритма). Формально эта задача выглядит так:

Для заданных множеств: $I_i = \{ j | a_{ij} = 1 \}$, $i = 1,...t$

$J = \{ j / X_j = 1 \}$ найти количество тех индексов i, для которых $I_i \cap J \neq \emptyset$.

Для решения этой задачи (и других подобных ей) очень удачным оказалось применение т.н. метода "hash-coding", или рандомизационной техники. Впервые эта техника использовалась в /I2/. В программе был реализован алгоритм, указанный в /I3/.

В таблице I приведены некоторые результаты машинного эксперимента. Для оценки эффективности алгоритма выбрано число фиксаций переменных V_i (напоминаем, что для решения задач с K переменными V_i методом полного перебора число таких фиксаций будет 2^k). Отдельные столбцы таблицы представляют собой:

Z – номер задачи,

K – число ограничений,

N – число столбцов непрерывной задачи,

NZ – число ненулевых элементов a_{ij},

$N_ц$ – число целочисленных переменных (двоичные),

I_H – число фиксаций переменных V_i, соответствующих решению непрерывной задачи,

I_o – число фиксации переменных V_i до получения оптимального решения,

$I_{оД}$ – число фиксаций переменных V_i до установления оптимальности.

Таблица I

Z	K	N	NZ	$N_ц$	I_H	I_O	$I_{оД}$
I	34	II6	350	83	27	30	II7
2	4I	I30	370	90	3I	3I	I48
3	4I	I30	350	90	26	28	I20
4	264I	6780	I4223	4I4I	I780	I90I	I273I
5	2906	78I0	I6I47	4906	2I0I	2I0I	оптимальность не доказана

Задачи I, 2, 3 искусственно генерировались для проведения анализа эффективности алгоритма. Задачи 4, 5 являются реальными и получены на основе документов, входящих в ИПС ОИЯИ.

Для ИПС ОИЯИ вполне удовлетворительными оказались параметры первого возможного решения, на основе которого меняется порядок документов ИПС.

Рассмотренный метод использовался для создания алгоритма автоматической сортировки элементов узловых списков основного информационного массива, который применялся для уплотнения сегментов списков в зонах информационного массива ИПС ОИЯИ (размер зоны указывается параметром m рассмотренной выше модели).

Результаты проведённых машинных экспериментов на ЭВМ СДС/6400 приведены в таблице 2, где:

L – средняя длина узлового списка;

V – количество документов основного информационного массива (общее количество узлов во всех уровнях списков);

ε – полученная плотность списков в зонах основного информационного массива;

t – время работы программы автоматической сортировки(в сек.).

Полученные результаты дают возможность эффективного применения данного метода при организации адаптирующейся структуры многоуровневой ИПС, где время поиска существенно меньше времени поиска в инверсной и ассоциативно-адресной структурах /4/.

Таблица 2

L	V	ε	t *)
9	3500	5,7	20
I2	6000	8,6	25
I4	I0000	I2,I	3I

*) В этом времени не учитывается подготовка данных для ввода.

```
Algorithm
procedure SELECT(X,L,R,K);
value L,R,K; integer L,R,K; array X;
begin  integer N,I,J,S,SD,LL,RR; val Z,T;
while R > L do  begin if R-L > 600 then begin
N:=R-L+1; I:=K-L+1; Z:=ln(N); S:=.5xexp(2xZ/3);
SD:=.5xsqrt(ZxSx(N-S)/N)xsign(I-N/2);
LL:=max(L,K-IxS/N+SD); RR:=min(R,K+(N-I)xS/N+SD);
while RR > LL do  SELECT1(X,L,R,K); end;  SELECT1(X,L,R,K); end;
end SELECT
procedure SELECT1(X,L,R,K);
integer I,J; realT;
begin  T:=X[K]; I:=LL; J:=RR;
exchange(X[L],X[R]);
if X[R]>T then exchange (X[R], X[L]);
while I < J do
begin
exchange (X[I], X[J]) ; I=I+1; J:=J-1;
while X[I]<T do I:=I+1;
while X[J] > T do J:=J-1;
end;
if X[L]=T then exchange (X[L] , X[J])  else
begin J:=J+1; exchange (X[J] , X[R]) end;
if J ≤ K then L:=J+1;
if K ≤ J then R:=J-1;
end SELECT1
```


Литература

I. Арнаудов Д.Д. "Об одном способе организации адаптирующейся многоуровневой информационно-поисковой системы". РІО-8І78, І975, Дубна.

2. Лефковиц Д. "Структура информационных массивов оперативных систем". М., І973, Энергия.

3. Арнаудов Д.Д. "Структурно-функциональная организация основных поисковых массивов ИПС ОИЯИ". РІО-862І, І975, Дубна.

4. Арнаудов Д.Д. "Вопросы теории и программной реализации ИПС ОИЯИ". Автореферат диссертации на соискание учёной степени доктора технических наук. ОИЯИ, І976, Дубна.

5. Geoffrion A.M.Marsten R.E. "Integer programming: framework and state-of-the-art survey" Management science,v.18,N9,1972.

6. Forest J.J.H., Hirst J.P.H.,Tomlin J.A."Practical solution of large mixed integer programming problems with UMPIRE"-Management science, v.20, №5,74.

7. Данциг Дж. Линейное программирование, М., І966, "Прогресс".

8. Харди Т.Г., Литлвуд Дж., Полиа Г., "Неравенства", изд.ИЛ,МІ948

9. Knuth D.E. The art of computer programming V3/Sorting and Searching/.

10. Hoare C.A.R. Algorithm 63 (PARTITION) and Algorithm 65 (FIND) CACM 4 (July 1961), 321.

11. Floyd R.W., Rivest R.L. Expected Time Bounds for Selection; Algorithm 489(SELECT)-CACM V18, No.3, 1975.

12. Королев Л.Н. Кодирование и свертывание кодов. ДАН,т.ІІ3,№ 4, І957.

13. Brent R.P. Reducing the Retrieval Time of Scatter Storage Techniques - CACM,V.16, No. 2, 1973.

Рукопись поступила в издательский отдел
6 мая І976 года.