ОБЪЕДИНЕННЫЙ
ИНСТИТУТ
ЯДЕРНЫХ
ИССЛЕДОВАНИЙ
ДУБНА

P11 - 9756

М.Д.Иванчев, В.М.Сумароков, Н.И.Янев

МЕТОД ВЕТВЕЙ И ГРАНИЦ ДЛЯ НАПРАВЛЕННОГО СИНТЕЗА ЭКОНОМНОЙ СТРУКТУРЫ ФАЙЛА

1976

P11 - 9756

М.Д.Иванчев, В.М.Сумароков, Н.И.Янев

МЕТОД ВЕТВЕЙ И ГРАНИЦ ДЛЯ НАПРАВЛЕННОГО СИНТЕЗА ЭКОНОМНОЙ СТРУКТУРЫ ФАЙЛА

Направлено в журнал "Научно-техническая информация"

В ряде прикладных задач, относящихся к организации данных в памяти ЭВМ, часто возникает проблема размещения информации "лучшим" образом. Основные критерии качества, которым должна удовлетворять структура данных, диктуются реальными условиями функционирования информационных систем и обычно связываются с максимальным быстродействием доступа к необходимой информации, минимальными затратами памяти на ее размещение, развитыми динамическими свойствами структуры данных (эффективность алгоритмов включения и изъятия), соображениями надежности математического обеспечения и другими факторами.

В общем случае разработчику приходится решать обратную многокритериальную задачу исследования операций, где показатели эффективности могут быть коррелированы, противоречивы и очень часто - трудно формализуемы. Поэтому принятие окончательного решения до сих пор остается прерогативой системного программиста, полагающегося в своем выборе во многом на эвристические соображения.

Разноаспектность требований к структуре информационных массивов породила широкий спектр методов их организации. Наиболее развитыми из них являются наращиваемые признаковые деревья /I,2/. Они сочетают малое время поиска данных при экономном их размещении с высоким быстродействием алгоритмов обновления файла.

В задачах исследования признаковых деревьев далеко не всегда удается построить выразительную математическую модель, формально связывающую основные параметры анализируемой структуры. Обширная и теперь уже труднообозримая литература дает много характерных примеров. В тех же случаях, когда выбор и формализация параметров дерева приводят к построению целевой функции, определяющей эффективность его структуры, обычно (см.,например, /I-3/) приходится решать экстремальную задачу дискретного нелинейного программирования. Последнее объясняется характерными свойствами любых иерархий, которым присущи конечное число уровней, а также ветвлений в узлах растущего дерева. Общих аналитических методов решения подобных задач пока не существует, поэтому приходится прибегать к специальным вычислительным приемам.

В последние годы для точного решения дискретных задач в комбинаторной постановке стали применяться различные способы направленного перебора, которые являются частными интерпретациями общей формальной схемы метода ветвей и границ/4,5/. Содержательное переложение этого метода на конкретный класс решаемых задач, вообще говоря, позволяет ускорить сходимость процедуры оптимизации. В данной работе предлагается алгоритм, укладывающийся в общую концепцию метода и существенно использующий структурные особенности и постановку задач оптимизации на односвязанных признаковых деревьях/7,8/. Заметим, что в вычислительном аспекте эти две задачи, с сильно выраженной нелинейностью, не проще многих из/I/. Мы надеемся также, что описание алгоритма проиллюстрирует характерные приемы учета специфики задач и на других признаковых деревьях, применяемых при организации информационных массивов.

Для полноты изложения коротко остановимся на важных для нас особенностях общей формальной схемы метода ветвей и границ. В основе метода лежит, с одной стороны, процесс ветвления множества, на котором ищется экстремум, на дерево подмножеств, а с другой стороны, реализуется процедура вычисления оценок подмножеств, т.е. прогнозирования целевой функции.Такой подход оставляет большую свободу для учета специфики решаемой задачи и позволяет заменить полный перебор всех элементов множества (часто неприемлемый) их частичным перебором ввиду того, что в дальнейшем ветвлению подлежат лишь "преспективные" подмножества. Выбор правил ветвления и вычисления оценок может быть в большей или меньшей степени удачным - в зависимости от специфики задачи и уровня понимания ее тонкостей. Конечно же, то обстоятельство, что задачу оптимизации на признаковых деревьях мы ставим, используя древовидную интерпретацию направленного перебора, является не больше чем курьезом. Введем предварительно еще три понятия: ветвления,

релаксации и фатоминг-критерия, а затем сформулируем алгоритм направленного перебора.

I. Ветвление. Для оптимизационной задачи Z обозначим через $M(Z)$ множество ее возможных решений. Будем говорить, что задача Z разветвлена на наборе $\{Z_1, Z_2, \dots, Z_q\}$ более "легких" задач, если выполнены следующие условия:

I.I. Каждое возможное решение задачи Z является возможным и для точно одной из задач $\{Z_i\}$, $i=\overline{1,q}$;

I.2. Любое из возможных решений каждой из задач $\{Z_i\}$, $i=\overline{1,q}$, является возможным и для Z.

Эти условия выполняются, когда $M(Z_i) \subseteq M(Z)$;

$$M(Z_i) \,\&\, M(Z_j) = \varnothing, \quad i=\overline{1,q}, \quad j=\overline{1,q}, \quad i \neq j.$$

В этом случае каждая из задач $\{Z_i\}$ является преемником задачи Z. В результате последующих ветвлений, "поощряемых" при соответствующих оценках по фатоминг-критерию, преемники приобретают своих преемников и т.д. Создание преемников преемников эквивалентно перерассмотрению разбиения исходного множества $M(Z)$.

2. Релаксация. Каждая исходная задача Z может быть релаксирована в задачу Z^r путем снятия некоторых исходных ограничений (например, целочисленности и пр.), что упрощает Z. Единственное условие корректности релаксации:

$$M(Z) \subseteq M(Z^r).$$

Пусть для определенности Z- задача минимизации целевой функции $\varphi(Z)$. Тогда релаксация предполагает выполнение следующих условий:

2.I если Z^r не имеет возможного решения, то это справедливо и для Z;

2.2 минимальное значение для $\varphi(Z)$ не меньше минимального значения $\varphi(Z^r)$;

2.3 если оптимальное решение для Z^r является возможным для Z, то это решение является оптимальным и для Z.

3. Фатоминг-критерий (от fathom - углубляться). Этот критерий можно рассматривать как решающее правило, позволяющее установить для каждого подмножества, полученного в результате раз-

ветвления, перспективу его последующих ветвлений, т.е. перспективу углубления в дерево вариантов.

Пусть $\varphi(z_j)$ – значение целевой функции для задачи z_j, полученной на некотором шаге ветвления от исходной задачи z; $\varphi^*(z)$ – значение целевой функции исходной задачи z, лучшее из всех, что были найдены до сих пор (текущий "рекорд"). Очевидно, что имеет смысл ветвить только задачи, для которой значения целевой функции меньше текущего "рекорда". Поэтому фатоминг-критерий устанавливает активные точки растущего дерева и границы, где ветвление вариантов прекращается. В любой ситуации возможна одна из трех альтернатив:

3.I анализ релаксированной задачи z_j^r показывает, что z_j не имеет решений, т.е. $M(z_j^r) = \emptyset$;

3.2 анализ z_j^r показывает, что z_j не имеет возможного решения, которое было бы лучше текущего "рекорда": $\varphi(z_j^r) \geqslant \varphi^*(z)$;

3.3 анализ z_j^r показывает, что оптимальное решение для z_j^r является возможным для z_j.

Общая схема алгоритма может быть представлена теперь следующим образом:

шаг I [начало] – начать список задач-преемников; присвоить начальное значение рекорда $\varphi^*(z)$;

шаг 2 [проверка на окончание алгоритма] – если список задач-преемников не исчерпан, перейти к шагу 3, иначе – конец;

шаг 3 [эвристическое правило] – выбрать задачу для ветвления;

шаг 4 [релаксация] – релаксировать рассматриваемую задачу-преемник;

шаг 5 [решение z_j^r] – найти $\varphi(z_j^r)$;

шаг 6 [проверка альтернативы 3.I.] – если $M(z_j^r)=\emptyset$, перейти к шагу 2, иначе – к шагу 7;

шаг 7 [проверка альтернативы 3.2.] – если $\varphi(z_j^r) \geqslant \varphi^*(z)$, перейти к шагу 2, иначе – к шагу 8;

шаг 8 [проверка альтернативы 3.3.] – если оптимальное решение $\varphi(z_j^r)$ является возможным для z_j, перейти к шагу I2, иначе – к шагу 9;

Шаг 9 [проверка необходимости продолжения решения выбранной z_j] – если хотим продолжить решение z_j, то углубить релаксацию z_j и перейти к шагу IO, иначе – к шагу II;

шаг IO [модификация релаксации] – модифицировать релаксацию z_j и перейти к шагу 5;

шаг II [ветвление] – разветвить z_j и занести ее преемников в список задач-преемников и перейти к шагу 2;

шаг I2 [возможное решение] – найдено возможное решение для z. Если $\varphi(z_j) < \varphi^*(z)$, заменить текущий "рекорд" $\varphi^*(z) = \varphi(z_j)$ и очистить список задач-преемников, для которых оценка хуже $\varphi^*(z)$, перейти к шагу 2.

Заметим еще, что в качестве эвристического правила выбора задачи для ветвления чаще всего используется оценка ее "перспективности". Метод получения таких оценок не является общим.

Для реализации ветвления задач в настоящее время применяются два основных метода:

а) одновременное ветвление, когда любое из подмножеств имеющихся разбиений на следующем шаге может быть подвергнуто ветвлению;

б) одностороннее ветвление, когда разбиению подлежит подмножество из числа полученных на предыдущем шаге и перейти к ветвлению других подмножеств можно, лишь закончив расчеты с выбранным подмножеством.

Реализация схемы одностороннего ветвления существенно экономит память ЭВМ, т.к. не требует запоминания решений по всему "фронту" ветвления. Однако при этом увеличивается перебор по сравнению со схемой одновременного ветвления. В связи с этим некоторые авторы/5/ предлагают в процессе отыскания экстремума компромиссное чередование обеих схем. В задаче, рассматриваемой ниже, интересы экономии памяти заставили нас воспользоваться схемой одностороннего ветвления.

Поставим теперь нашу задачу синтеза оптимального наращиваемого поискового дерева/7,8/. Дадим несколько определений. Сразу заметим, что в соответствии с обусловленной целью настоящей статьи мы ориентируем их на понимание математической схемы экстремальной задачи, т.к. полное представление было бы слишком громоздким; методически последовательное изложение с акцентом на прикладные аспекты структуры деревьев можно найти в /6-8/.

Дерево – это связный ациклический орграф, в каждый узел которого, кроме корня, входит одна дуга и из каждого узла которого, кроме терминальных, исходит несколько дуг. Совокупность исходящих из узла дуг образует пучок, а их количество называется размером пучка. Номер уровня определяется числом дуг, соединяющих узел этого уровня с корнем. Будем считать, что корень находится на нулевом, а терминальные узлы – на m-ом уровне. В деревьях [6–8], а также в некоторых из [1] узлы, принадлежащие одному уровню, имеют одинаковые размеры пучков. Структура рассматриваемых деревьев однозначно определяется вектором

$$\vec{Q}(m, n_0, n_1, \ldots, n_{m-1}),$$

где m – количество уровней,

n_i – размер пучков узлов i-ого уровня, $i=\overline{0,m-1}$.

Узлы, содержащие, помимо адреса связи и признакового ключа объекта, также информацию о самих объектах (или их адресах), называются объектными. В [6] исследовано растущее односвязное дерево, в котором объектными являются лишь терминальные узлы. В [7] оно модифицировано некоторым перераспределением объектных узлов по всей структуре дерева, начиная с корня. Эта перестройка позволила увеличить быстродействие алгоритмов поиска, включения и изъятия данных; а анализ полученной структуры показал, что условием ее оптимальности является редукция размеров пучков от корня к терминальному уровню на представительном множестве трех номиналов $\rho\{4,3,2\}$. Лучшие поисковые свойства такого дерева, "выпуклого" к корню, объясняются большим содержанием поисковых треков меньших длин, чем в равноемких деревьях другой структуры. В [8] предпринята дальнейшая эволюция односвязанных сбалансированных деревьев, в результате которой каждому из

$$N(\vec{Q}) = 1 + \sum_{j=0}^{m-1} \prod_{i=0}^{j} n_i \qquad (1)$$

узлов сообщен объектный смысл. Возникает задача оптимизации структуры такого предельно насыщенного дерева.

В [8] доказывается, что сумма Φ длин поисковых треков всех объектных узлов задается формулой

$$\Phi(\vec{Q}) = \sum_{k=1}^{m} \left\{ \prod_{i=0}^{k-1} n_i \left(1 + \sum_{j=0}^{k-1} \frac{n_j+1}{2}\right)\right\} + 1 .$$

В качестве критерия выбора компонентов вектора $\vec{Q}$ следует взять минимальную трудоемкость поиска, характеристикой которой может служить длина поискового трека искомого объектного узла (цели поиска). Но эта величина сильно зависит от положения цели поиска в ряду последовательных объектных узлов дерева, изменяясь в широких пределах по закону неправильной пилы. Поэтому в качестве характеристики трудоемкости поиска принимается средняя длина поискового трека:

$$\varphi(\vec{Q}) = \frac{\Phi(\vec{Q})}{N(\vec{Q})} .$$

Задача состоит в нахождении такого вектора $\vec{Q}^*$, чтобы

$$\varphi(\vec{Q}^*) = \min_{\vec{Q}\in\Omega} \frac{\sum_{k=1}^{m} \left\{ \prod_{i=0}^{k-1} n_i \left(1 + \sum_{j=0}^{k-1} \frac{n_j+1}{2}\right)\right\} + 1}{\sum_{j=0}^{m-1} \prod_{i=0}^{j} n_i + 1} , \qquad (2)$$

где $\Omega = \left\{ \vec{Q} \mid E \le N(\vec{Q}) \le \delta E \right\}$, (3)

E – прогнозируемое требуемое число объектных узлов дерева,

$\delta \ge 1$ – допуск на превышение E, разрешенный наличным резервом памяти,

а компоненты вектора Q должны удовлетворять естественным требованиям целочисленности, причем

$$n_i \ge 2 , \qquad i = \overline{1, m-1} , \qquad (4)$$

$$1 \le m \le [\log_2(\delta E)] . \qquad (5)$$

Ограничения (4), (5) отражают тот очевидный факт, что наименьший размер пучка, реализацию которого можно обеспечить, не дискредитируя самой идеи древовидной структуры, равен 2; максимальный – при дереве, состоящем из одного пучка,– равен ($N-1$). Заметим также, что превышение свободного резерва памяти над E, всегда обеспеченное реальными условиями синтеза, вообще говоря, частично используется для размещения дерева, т.к. в силу целочисленности размеров пучков в отношении $N(\vec{Q}^*) \ge E$ наиболее вероятно неравенство. Несколько забегая вперед, отметим, что отношение $(N-E)/E$ оказывается незначительным и убывает с увеличением E, т.к. в файлах большей емкости число составляющих вектора $\vec{Q}$ возрастает, увеличивается возможность выбора (I) ближе к E и (2) достигается при малых значениях $(N-E)/E$.

Опишем теперь алгоритм, использованный для решения задачи (2)-(5). Он реализует схему одностороннего ветвления по методу направленного перебора, представленному выше. Некоторые комментарии к реализованному алгоритму приведены после его записи.

Шаг I. Вычисление значения $m^* = [\log_2(\delta E)]$; $m=2$;

$$n_0 = [\sqrt{E}] ; \quad n_1 = n_0 - 1 ;$$

$$\varphi^* = \frac{1/2[n_0(n_0+1)+n_0 n_1(n_0+n_1+2)]+1}{n_0 n_1 + n_0 + 1} ; \quad k=0; n_j=1, j=\overline{0,m^*}.$$

Шаг 2. Если $(m+1) > [\log_2(\delta E)]$, конец; иначе перейти к шагу 3.

Шаг 3. $m = m+1$.

Шаг 4. $n_k = n_k + 1$; $n_j \geqslant 2$, $j = \overline{k+1, m^*}$.

Шаг 5. Вычисление $\varphi = \frac{\Phi}{\delta E}$, где $\vec{Q}(m, n_0, n_1, \ldots, n_k, 2, 2, \ldots, 2)$.

Шаг 6. Если $N(Q) > \delta E$, то $k=k-2$; иначе перейти к шагу 7. Если $k < 0$, перейти к шагу 2, иначе – к шагу 4.

Шаг 7. Если $\varphi(\vec{Q}) < \varphi^*$, перейти к шагу 4; иначе $k=k-2$. Если $k < 0$, перейти к шагу 2, иначе – к шагу 4.

Шаг 8. Стандартный.

Шаг 9. Всегда к шагу IO.

Шаг IO. $k=k+1$. Если $k \leqslant m$, перейти к шагу 4, иначе $k=k-1$ и перейти к шагу 4.

Шаг II. Ветвление одностороннее.

Шаг I2. Стандартный.

В шаге I оценка для текущего "рекорда" φ^* получается как первое грубое приближение при аналитическом решении задачи (2)-(5) для $m=2$. Здесь, как и всюду прежде, $[x]$ означает целую часть x.

Для проверки условия в шаге 6 используется формула

$$\left(\delta E - 1 - \sum_{j=0}^{k-1} \prod_{i=0}^{j} n_i\right) \Big/ \prod_{i=0}^{k-1} n_i + 1 \geqslant 2^{m-k+1} ,$$

непосредственно следующая из условий (2) и (4).

Оценка φ в шаге 5 вычисляется по формуле (2), где еще не зафиксированные переменные имеют минимально возможные значения. Последовательность выбора переменных, подлежащих фиксации, строго детерминирована в порядке возрастания индекса компонент вектора $\vec{Q}$. Поскольку целевая функция (2) относительно переменной n_{m-1} является возрастающей в области (4), то выбор значений этой переменной однозначно определяется из условия (3).

Описанный алгоритм был запрограммирован на языке ФОРТРАН-IУ, и программа приведена в приложении. Ввиду этого не будем обсуждать его более подробно. Отметим только, что все априорные знания о решаемой задаче могут быть использованы непосредственно и приводят к существенному сокращению перебора.

<u>ТАБЛИЦА</u>

$E=2^k$ k	$N(\vec{Q}^*)$	m	$\vec{Q}^*$ $n_0, n_1, \ldots, n_{m-1}$	$\varphi(\vec{Q}^*)$	t /c/
8	269	5	4,3,3,2,2	9,335	0,68
9	557	6	4,3,3,2,2,2	IO,7I3	2,02
IO	IO77	6	4,4,3,3,2,2	II,825	6,06
II	2229	7	4,4,3,3,2,2,2	I3,207	9,87
I2	4309	7	4,4,4,3,3,2,2	I4,32I	I5,I3
I3	89I7	8	4,4,4,3,3,2,2,2	I5,706	30,42
I4	I6402	9	3,3,3,3,3,3,3,2,2	I6,80I	72,29
I5	33898	IO	3,3,3,3,3,3,3,2,2,2	I8,I94	I97,3I
I6	65609	IO	4,3,3,3,3,3,3,3,2,2	I9,300	4II,76
I7	I3I593	II	4,3,3,3,3,3,3,3,2,2,2	20,694	907,5I

В прилагаемую таблицу сведены некоторые результаты машинного эксперимента по синтезу наращиваемых поисковых деревьев для различных значений E . В крайней справа колонке указано время работы программы синтеза на ЭВМ СДС-6400 (номинальное быстродействие $2 \cdot 10^6$ оп/с). Следует заметить, что оно может быть и еще значительно сокращено (в 50-IOO раз – в зависимости от E) при еще более кумулятивном переборе, возможности для которого открываются в связи с ограниченностью области фактических значений компонентов вектора $\vec{Q}^*$ в оптимальных структурах. Действительно, из таблицы видно, что эта область локализована множеством $\rho\{4,3,2\}$ и что, как и в /7/, имеет место эффект редукции размеров пучков от корня к терминальному уровню. Экстраполяция этого результата из /7/ на данную задачу, ставшая теперь фактом, могла прогнозироваться в силу оправданности геометрической модели дерева, "выпуклого" к корню. Однако для предложенного здесь алгоритма не были фактом ни представительность множества ρ, ни редукция составляющих вектора $\vec{Q}^*$, т.к. необходимо было в этом убедиться.

Заметим еще, что изредка (в наших экспериментах 2 случая из 73) среди компонентов вектора $\vec{Q}^*$ появляется $n_0 = 5$, что не меняет значения сформулированного результата и подтверждает "флюктуационную" природу дискретности. Поэтому при кумулятивном переборе множество ρ может быть расширено до $\{5,4,3,2\}$. В этих редких случаях значения целевой функции $\psi(\vec{Q})$ на соседних допустимых векторах, представленных в ρ, отличались от $\psi(\vec{Q}^*)$ на величины порядка 10^{-3}. При машинном моделировании задачи /7/ с помощью алгоритма, родственного описанному, в 67 оптимальных реализациях подобных флюктуаций не обнаружено.

ПРИЛОЖЕНИЕ

```
      PROGRAM TREE(INPUT,OUTPUT)
      DIMENSION A(21),N1(21),N2(21),NC(21)
      DIMENSION N(21)
      EPS=1.0E-1
      A(1)=
      A(2)=4
      B=
      DO 22 J=3,21
      B=B*2
22    A(J)=2*A(J-1)+B
      N1(1)=1
      N2(1)=1
      READ 1,M,L,ICF,FI,N
1     FORMAT(I5,2I15,F15.0,/,20I3)
      NC(1)=L
      IF(ICF.GT. )GO TO 23
      ICF=1.0E16
       IF(FI.GT. ) GO TO 23
      FI=ICF
      IF(M.LE.2)GO TO 20
      IX=(SQRT(L-1.5)-EPS)+1
      IY=IX-1
      ICF=1+IX+IX*IY+(IX*(IX+1)+IX*IY*(IX+IY+2))/2
      N1(3)=1+IX+IX*IY
      FI=(ICF+0.0)/N1(3)
      N(1)=IX
      N(2)=IY
      PRINT 16,ICF,N1(3),FI,N
      IF(M.LT.3) M=3
20    DO 2 I=1,M
2     N(I)=1
23    PRINT 21,M,L
21    FORMAT(1X,60(1H*),≠M=≠,I5,5X,≠L=≠,I15)
      K=0
      IS=0
3     K=K+1
      IF(K-M)4,5,5
4     N(K)=N(K)+1
14    N2(K+1)=N(K)*N2(K)
      N1(K+1)=N1(K)+N2(K+1)
      X=1.5*L-N1(K+1)
      X=X/N2(K+1)
      IS=IS+N(K)
      IF(K.EQ.M)GO TO 31
      IF(X.LT.(2.0**(M-K+1)-2.0))GO TO 1
31    CONTINUE
      ICS1=(N2(K+1)*(IS+K))/2
      NC(K+1)=NC(K)+ICS1
      X=((2.0**(M-K+1)-2.0)*N2(K+1)*(IS+K+1)+A(M-K+1)*N2(K+1))/2.0
      IF((NC(K+1)+X).GE.ICF)GO TO 10
      GO TO 3
10    K=K-1
      IF(K)12,12,13
13    IS=IS-N(K+1)-N(K)
      N(K+1)=1
      GO TO 4
5     X=L-N1(M)
```

ПРИЛОЖЕНИЕ
(продолжение)

```
      N(M)=AINT(X/N2(M)-EPS)+1
      IF(N(M).GE.2)GO TO 14
      N(M)=2
      GO TO 14
 6    J=NC(M+1)-L+N1(M+1)
      FI1=(J+0.0)/N1(M+1)
      IF(FI1.GE.FI)GO TO 11
      FI=FI1
      PRINT 16,J,N1(M+1),FI1,N
16    FORMAT(1X,≠VAZMOJNO RECHENIE CF=≠,I10,3X,≠PI=≠,I10,3X,≠FI=≠,F15.3
     *  ,/,(20I5))
11    CONTINUE
      K=K-1
      GO TO 10
12    X=L**(1./M)
      PRINT 17,M,X
17    FORMAT(≠ PRI M=≠,I5,5X,≠OPTIMALNOST USTANOVENA  L**(1./M=≠,F10.2)
      M=M+1
      IF((2.**M).GE.(1.5*L))STOP
      GO TO 20
      END
```

Замечание: оператор **READ** предполагает наличие двух перфокарт с указанным в программе форматом данных:

I.

I) M – исходное число уровней дерева;

2) L=K ;

3) ICF=Φ ;

4) FI= φ.

П. N – целочисленный кортеж размеров пучков.

Программа допускает прерывание решения. Для продолжения оптимизации достаточно подготовить 2ПК со значениями параметров, соответствующими последнему выданному текущему "рекорду".

ЛИТЕРАТУРА

I. D.E.Knuth. The Art of Computer Programming ,vol.3:Sorting and Searching;Addison-Wesley,Reading,Mass., 1973,p.406-479.

2. С.С.Лавров, Л.И.Гончарова. Автоматическая обработка данных. Хранение информации в памяти ЭВМ. М., "Наука", I97I.

3. L.E.Stanfel .Optimal Trees for a Class of Information Retrieval Problems." Information Storage and Retrieval", 1973, 9, № 1, 43-59.

4. E.L.Lawler, Wood D.E. Branch-and-Bound Methods. "Operation Research", 1966, 14, № 4.

5. A.M.Goeffrion, R.E.Marsten. Integer Programming Algorithms: a Framework and State-of-the-Art Survey. "Management Science", 1972, 18, № 9, 465-491.

6. W.I.Landauer. The Balanced Tree and its Utilization in Information Retrieval. " IEEE Transactions on EC", 1963, EC-12, № 6, December, 863-871.

7. В.М.Сумароков. Некоторые комбинаторные свойства одного класса ассоциативных структур. "Программирование", I976, № 2, I9-28; см.также препринт ОИЯИ PII-9I03, Дубна, I975.

8. М.Д.Иванчев, В.М.Сумароков, Н.И.Янев. Препринт ОИЯИ, PII-9777, Дубна, I976.

Рукопись поступила в издательский отдел
3 мая I976 года.