ИТОГИ НАУКИ И ТЕХНИКИ.
Современная математика и ее приложения.
Тематические обзоры.
Том 224 (2023). С. 97–108
DOI: 10.36535/0233-6723-2023-224-97-108

УДК 514.75

ИЕРАРХИЧЕСКИЕ СТРУКТУРЫ И КОМБИНАТОРНЫЕ ЗАДАЧИ ИНФОРМАЦИОННОГО ПОИСКА

© 2023 г. **О. В. КУЗЬМИН**

Аннотация. Изучаются комбинаторные объекты пирамидальной структуры. Рассмотрен один из способов представления правил в иерархической, последовательной структуре — метод деревьев принятия решений, где каждому объекту соответствует единственный узел, дающий решение. Предложен алгоритм построения дерева решений на основе обобщенной пирамиды Паскаля. Предложен метод построения поискового индекса, который отображает долю релевантного материала и позволяет производить сравнения во множестве терминов, исходя из весовых коэффициентов терминов и путей.

Ключевые слова: иерархическая структура, частично упорядоченное множество, обобщенная пирамида Паскаля, задача принятия решений, дерево решений, комбинаторный алгоритм.

HIERARCHICAL STRUCTURES AND COMBINATORIAL PROBLEMS OF INFORMATION RETRIEVAL

© 2023 **O. V. KUZMIN**

Abstract. We examine combinatorial objects of pyramidal structure. We consider one of the ways of representing rules in hierarchical, sequential structures: the method of decision trees, where each object corresponds to a single node that provides a solution. An algorithm for constructing a decision tree based on the generalized Pascal pyramid is suggested. Also, we propose a method for constructing a search index, which displays the proportion of relevant material and allows one to perform comparisons in the variety of terms based on the weight coefficients of terms and paths.

Keywords and phrases: hierarchical structure, partially ordered set, generalized Pascal pyramid, decision-making problem, decision tree, combinatorial algorithm.

AMS Subject Classification: 06E30, 94D10, 15B34, 05B20

1. Введение. Комбинаторные задачи алгоритмического характера на дискретных конечных математических структурах встречаются в практике постоянно. В последние годы неуклонно растет интерес к теории «больших систем», с которыми приходится иметь дело в самых различных областях науки и техники. Важным направлением исследования таких «больших» или «сложных» систем является рассмотрение их как многоуровневых систем или систем с иерархической структурой (см. [13, 14]). Процесс поэтапного построения решения многокритериальных задач с иерархическими структурами часто может быть интерпретирован как траектория на конечной решетке (см. [4, 15]), описывающей соответствующее частично упорядоченное множество (см. [2]). Подобные задачи нередко встречаются при разработке методов автоматического анализа больших массивов данных в информационных системах и обработке сетей (см. [11, 27]).

В монографии [8] предложена схема построения комбинаторных чисел и полиномов на основе иерархической пирамидальной структуры с весами, названной обобщенной пирамидой Паскаля. В [22]) широко известная техника теории частично упорядоченных множеств Рота—Стенли

(см. [15]) применяется для исследования целого ряда комбинаторных объектов, описываемых этой схемой.

Деревья решений являются одним из наиболее эффективных инструментов интеллектуального анализа данных и предсказательной аналитики, которые позволяют решать задачи классификации и регрессии.

Основополагающие идеи, послужившие толчком к появлению и развитию деревьев решений, были заложены в 1950-х гг. в области исследований моделирования человеческого поведения с помощью компьютерных систем. Среди них следует выделить работы К. А. Ховланда (см. [19]) и Э. Б. Ханта и др. (см. [20]).

Дальнейшее развитие деревьев решений как самообучающихся моделей для анализа данных связано с Дж. Р. Куинленом (см. [28,29]), который разработал алгоритм ID3 и его усовершенствованные модификации C4.5 и C5.0, и Л. Брейманом (см. [18]), предложившим алгоритм CART и метод случайного леса.

Деревья решений используются преимущественно в процессе принятия решений. В отличие от них деревья поиска предназначены в основном для управления поиском. Задачи поиска в различных постановках не являются новым предметом исследования (см. [1]). Первые системы, реализующие информационный поиск, были созданы в 1950-е гг. По мере развития информационных систем размеры и количество хранящихся документов постоянно росли. Помимо самого текста, стали хранить параметры его форматирования, гипертекстовые связи между документами и другую информацию. В связи с этим встает вопрос сравнения близости информационного содержимого в некоторой понятийной плоскости. Организацию баз данных разделяют на три уровня: физический, логический (см. [5]) и концептуальный (см. [16]). Задачи поиска выделяют, как правило, на физическом и логическом уровнях.

Основными моделями логической организации баз являются три модели: иерархическая, сетевая и реляционная. В иерархической модели данные представлены деревом иерархии, с отношениями типа «родитель-потомок», причем у потомка может быть только один «родитель», а корневой элемент не имеет «родителя». В сетевой модели нет таких жестких ограничений, однако это делает ее более сложной как в использовании, так и программной реализации. Реляционная модель, описывающая данные с помощью плоских таблиц, предложенная Е. Коддом, позволяет, используя операции разрезания и склеивания отношений, извлечения и объединения столбцов, а также процедуру нормализации, представлять как линейные, так и древовидные и сетевые структуры. Такая большая гибкость реляционной модели, вкупе с ее строгой математической моделью — реляционной алгеброй, сделали ее особенной популярной. Почти все крупные базы данных на данный момент реализованы на основе реляционной модели.

Однако несмотря на превалирование реляционных баз данных и систем управления ими, в последнее время стали развиваться альтернативные подходы к организации данных, такие как XML (Extensible Markup Language — расширяемый язык разметки), который является в некотором роде, развитием идей иерархической модели, а также объектные базы данных, которые несмотря на их определенные отличия, во многом повторяют преимущества сетевой модели.

В данной работе, относящейся к области разработки методов анализа иерархических систем и их приложений в задачах поиска и принятия решений, формулируются алгоритмы построения иерархической классификационной модели — дерева принятия решений. Рассматриваются вопросы отыскания новых способов построения индексов релевантности информации, и предлагается способ построения такого индекса с помощью обобщенных пирамид Паскаля.

2. Основные понятия и соотношения. *Обобщенной пирамидой Паскаля* (или V-пирамидой) (см. [8]) называется иерархическая трехгранная пирамидальная структура V с весами, элементы которой удовлетворяют рекуррентным соотношениям

$$V(n,k,l) = \alpha_{n,k-1,l}\,V(n-1,k-1,l) + \beta_{n,k,l-1}\,V(n-1,k,l-1) + \gamma_{n,k,l}\,V(n-1,k,l) \qquad (1)$$

с граничными условиями

$$V(0,0,0) = V_0, \qquad V(n,k,l) = 0, \text{ если } \min(n,k,l,n-k-l) < 0.$$

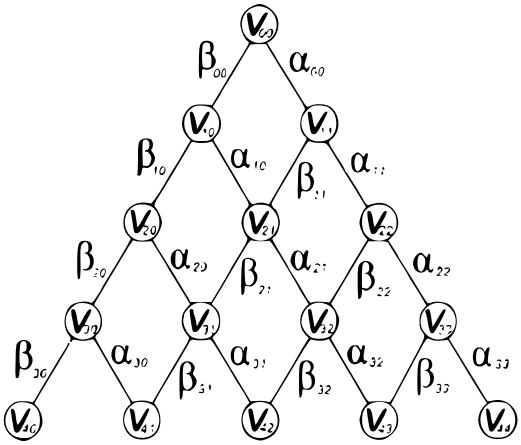

Рис. 1. Иерархическая структура V-треугольника

Число V_0, стоящее в вершине (нулевом слое) V-пирамиды, оговаривается особо; во многих случаях можно считать $V_0 = 1$. Величины $\alpha_{n,k,l}$, $\beta_{n,k,l}$, $\gamma_{n,k,l}$, называют весовыми коэффициентами, или *весами*. Ряд свойств V-пирамид и их частных случаев приведен в [3, 10, 17, 25].

Верхним N-отсечением V-пирамиды (V_N-пирамидой) называем (см. [10]) конечный трехгранный пирамидальный массив элементов, составляющий верхнюю часть V-пирамиды, причем основанием V_N-пирамиды служит плоское N-сечение V-пирамиды, а номера рассматриваемых V_N-пирамид совпадают с номерами соответствующих сечений V-пирамиды.

Важным частным случаем обобщенной пирамиды Паскаля является *обобщенный треугольник Паскаля* (или V-треугольник) (см. [8]), определяемый как иерархическая треугольная структура V с весами, элементы которого удовлетворяют рекуррентным соотношениям

$$V(n,k) = \alpha_{n,k-1}\, V(n-1,k-1) + \beta_{n,k}\, V(n-1,k) \tag{2}$$

с граничными условиями

$$V(0,0) = V_0, \qquad V(n,k) = 0, \text{ если } \min(n,k,n-k) < 0.$$

На рис. 1 изображена иерархическая структура V с весами, которую описывает соотношение (2).

При задании соответствующих наборов значений весов соотношения (1) и (2) допускают перечислительные интерпретации в терминах решеточных путей, которые, в свою очередь, могут быть использованы при анализе решеточных структур (см. [23, 24]).

Далее нам потребуются следующие операции над графами (см. [26]).

Удалением вершины $v \in V$ графа $G = G(V, E)$ называется операция, состоящая в удалении этой вершины вместе со всеми инцидентными ей ребрами. Для обозначения графа, полученного после удаления вершины v графа G используется символ $G \setminus v$.

Удалением ребра $e \in E$ графа $G = G(V, E)$ называется операция, состоящая в удалении этого ребра при сохранении всех вершин графа. Для обозначения графа, полученного после удаления ребра e графа G используется символ $G \setminus e$.

Стягиванием ребра $e = (u, v)$ графа $G(V, E)$ называется операция, состоящая в отождествлении смежных вершин u и v и удалении образовавшейся петли.

Напомним (см. [7]), что *высота* корневого дерева — это максимальное количество дуг, отделяющих листья от корня. Если дерево не взвешенное, то его высота — это просто расстояние от корня до самого удаленного листа.

В приложениях нередки *бинарные деревья* решений. В этом случае корень и каждая внутренняя вершина имеют не более двух потомков. На *уровне* h бинарного дерева имеется максимум 2^h вершин. Бинарное дерево высотой h, имеющее максимальное количество узлов, является *совершенным* бинарным деревом.

Также применяются *тернарные деревья*, в которых корень и каждая внутренняя вершина имеют не более трех потомков. Общие приложения для троичных деревьев поиска включают проверку орфографии и автозаполнение.

3. Дерево принятия решений. Дерево решений является способом организации данных в виде иерархической структуры, включающей в себя элементы двух типов — *узлов* или *внутренних вершин* и *листьев* или *висячих вершин*, отличных от *корня*. Лист определяет решение для каждого попавшего в него примера, в нем содержится не правило, а подмножество объектов, удовлетворяющих всем правилам ветви, которая заканчивается этим листом. К каждому листу есть только один путь, поэтому пример или утверждение могут попасть только в один лист, что обеспечивает единственность решения.

В настоящее время деревья решений стали одним из наиболее популярных методов *интеллектуального анализа данных* (Data Mining), используемых при решении задач *классификации*.

Перейдем к описанию разработанного алгоритма построения дерева принятия решений на основе обобщенной пирамиды Паскаля (см. [21]).

Алгоритм 1.

BEGIN

Шаг 1. Определяем параметры для нахождения размеров V_N — верхнего N-отсечения V-пирамиды.

Шаг 2. Определяем параметры для построения множества запрещенных вершин $V' \subset V$.

Шаг 3. Определяем параметры для построения множества запрещенных ребер E'.

Шаг 4. Определяем множество запрещенных вершин $V' \subset V$.

Шаг 5. Определяем множество запрещенных ребер $E' \subset E$.

Шаг 6. Удаляем из множества V вершины, принадлежащие V', т.е. заменяем V на $V \diagdown V'$.

Шаг 7. Удаляем из множества ребер E ребра, принадлежащие $E'' \subset E'$, т.е. запрещенные ребра, оставшихся после выполнения шага 5.

Шаг 8. Если найдутся три вершины u, w, v, которые образуют (в указанном порядке, считая от корня) (простую) цепь, причем $\deg w = 2$, то переходим к шагу 9, иначе выполняем шаг 10.

Шаг 9. Ребро (w, v) стягиваем к вершине v и переходим к шагу 7.

Шаг 10. У всех оставшихся в V_N ребер полагаем веса равными 1.

Шаг 11. Дерево D построено.

END

Алгоритм 1 может быть использован, в частности, при построении тернарного дерева T.

При построении бинарных деревьев можно использовать Алгоритм 2, который отличается от Алгоритма 1 тем, что в нем Шаг 1 заменяется следующим:

Шаг 1′. Определяем входящие параметры для нахождения размеров V_N — верхнего N-отсечения V-треугольника.

Алгоритм 2 может, в частности, использоваться при построении бинарного дерева B. При этом для определения входящих параметров для нахождения размеров V_N — верхнего N-отсечения V-треугольника можно воспользоваться следующими утверждениями (см. [21]).

Утверждение 1. *Для построения совершенного бинарного дерева $\mathcal{B}$ высоты n с корнем V_0 необходимо и достаточно V_N-треугольника, где $N = 2^n$.*

Утверждение 2. *Для построения бинарного дерева B высоты n с корнем V_0 достаточно V_N-треугольника, где $N = 2^n$.*

Для того чтобы показать, как Алгоритм 2 применяется к конкретным задачам, рассмотрим пример построения бинарного дерева решений B на основе V-треугольника.

Посредством задания весов, множеств $V' \subset V$ запрещенных вершин (запрещенных позиций) и $E' \subset E$ запрещенных ребер, удаления всех элементов указанных множеств V' и E' и последующего стягивания ребер, иерархическая треугольная структура, описываемая соотношением (2), может быть преобразована в соответствующее бинарное дерево решений (рис. 2 и 3).

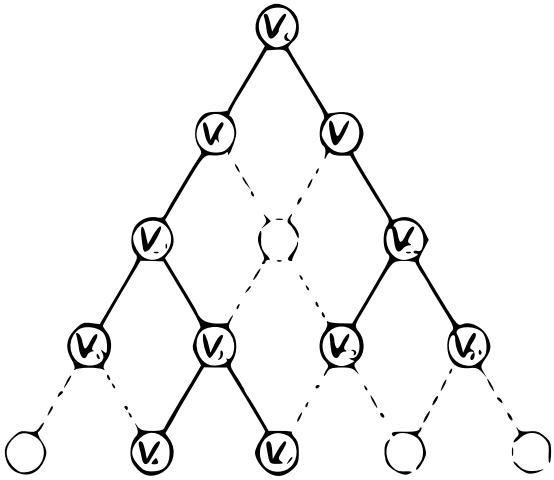

Рис. 2. V-треугольник с запрещенными ребрами и вершинами

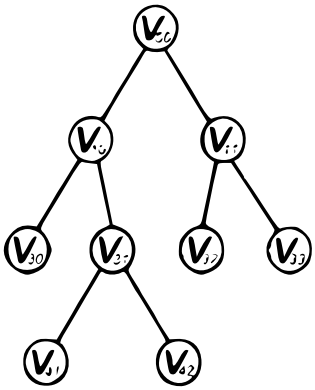

Рис. 3. Бинарное дерево решений B

Рисунок 2 получен из рис. 1, где запрещенные вершины отмечены серым, запрещенные ребра — штриховыми линиями, а остальные веса полагаются равными единице. На рис. 3 приведено бинарное дерево решений B, полученное из изображенного на рис. 2 V-треугольника удалением запрещенных вершин из множества $V' = \{V_{2,1}, V_{4,0}, V_{4,3}, V_{4,4}\}$, удалением ребер $(V_{3,0},\ V_{4,1})$ и $(V_{3,2},\ V_{4,2})$ из E' и последующим стягиванием ребер $(V_{1,1},\ V_{2,0})$ и $(V_{1,0},\ V_{2,2})$ в вершины $V_{1,0}$ и $V_{1,1}$ соответственно.

Простейшие деревья решений хороши своей наглядностью. Они не оперируют вероятностями или весами. Для решения реальных задач часто используют усложненные и дополненные модификации деревьев решений.

Метод дерева решений применяется в задачах *классификации* и *прогнозирования*, когда решения приходится принимать в условиях риска, неопределённости и исход событий зависит от вероятностей. На каждое решение влияют какие-то определённые факторы, и у каждого решения есть свои последствия, которым присущ вероятностный характер.

Замечание 1. При задании соответствующих наборов значений весов (и с учетом условия нормировки), соотношения (1) и (2) допускают вероятностные интерпретации (см. [8]) при решении реальных задач в терминах модифицированных деревьев решений.

При этом могут быть использованы модификации приведенных алгоритмов — Алгоритм 1* или Алгоритм 2*, в которых Шаг 10 заменяется на следующий:

Шаг 10′. У всех оставшихся в V_N ребер значения весов полагаем равными соответствующим $p_i > 0$ с учетом условия нормировки по уровням дерева.

4. Методы поиска, использующие деревья решений. Согласно [7] *основной принцип*, на котором базируются методы поиска с деревом решений, состоит в *декомпозиции* начальной задачи P_0 на некоторое число подзадач $P_1, \ldots, P_k$ (в целом представляющих всю задачу P_0) с последующей попыткой разрешить каждую из них:

(i) найти оптимальное решение задачи P_i, если оно находится очевидным образом;
(ii) показать, что решать задачу P_i не имеет смысла, поскольку значение оптимального решения для P_i заведомо хуже, чем для наилучшего из ранее найденных решений;
(iii) показать, что подзадача P_i не является допустимой.

Это разбиение описывается деревом, представленным на рис. 4, где вершины $P_1,\ \ldots, P_k$ изображают подзадачи.

Идея декомпозиции задачи P_0 на некоторое число подзадач $P_1, \ldots, P_k$ состоит в том, что или эти подзадачи проще решить, или они имеют меньший размер, или обладают структурой, не

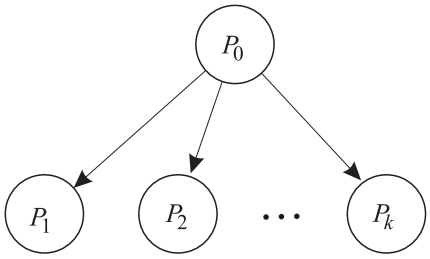

Рис. 4. Декомпозиция задачи P_0 на подзадачи

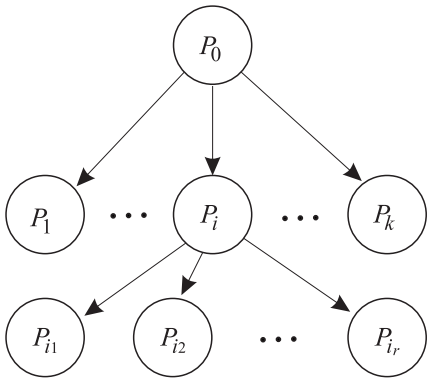

Рис. 5. Дерево после ветвления в вершине P_i

присущей первоначальной задаче P_0. Если подзадачу P_i нельзя решить, то она также разбивается на новые подзадачи $P_{i_1}, P_{i_2}, \ldots, P_{i_r}$, как это показано на рис. 5. Это разбиение (называемое также ветвлением) повторяется для каждой подзадачи, которая не может быть решена.

На любом этапе полное множество подзадач, требующих решения, представляется множеством *концевых* вершин (т. е. вершин степени 1) всех цепей, исходящих из корня (начальной задачи P_0) дерева поиска. Эти вершины называются *висячими* вершинами; на рис. 5 это вершины P_1, …, P_{i-1}, P_{i_1}, P_{i_2}, …, P_{i_r}, P_{i+1}, …, P_k.

Если поиск исчерпан, то множество подзадач, на которые разбита задача, должно представлять все пространство подзадач исходной задачи. Таким образом, если задача P_i разбита на r подзадач P_{i_1}, P_{i_2}, …, P_{i_r}, то

$$\{P_{i_1}\} \cup \{P_{i_2}\} \cup \cdots \cup \{P_{i_r}\} = P_i, \tag{3}$$

где $\{P\}$ обозначает множество всех допустимых решений задачи P.

Так как соотношение (3) должно быть применено к каждому разбиению, то

$$\{P_0\} = \bigcup \Big\{ P(i \;\Big|\; P(j) \text{ — висячая вершина дерева} \Big\}. \tag{4}$$

В случаях, когда требуется перебрать все решения задачи P_0 (а не только найти оптимальное в некотором смысле решение), желательно уметь перебирать решения с помощью вышеприведенной декомпозиции задачи на подзадачи и перебирать решения каждой из этих подзадач. В этом случае нужно избежать дублирования построенных решений, т.е. нужно разбивать задачу P_i на подзадачи $P_{i_1}, P_{i_2}, \cdots, P_{i_r}$ так, чтобы

$$\{P_{i_s}\} \cap \{P_{i_q}\} = \varnothing \tag{5}$$

для любых двух подзадач P_{i_s} и P_{i_q}, для которых $s \neq q$.

Соотношение (5) определяет собственное разбиение задачи P_i. Хотя условие (5) не является необходимым для полноценного поиска с деревом решений, оно, тем не менее, имеет большие выгоды с вычислительной точки зрения, поскольку

(i) для задачи оптимизации P_0 оптимальное решение является решением одной и только одной подзадачи, представляемой висячей вершиной;
(ii) для задачи полного перебора объединение множеств решений подзадач, представляемых висячими вершинами, дает множество всех решений задачи P_0 без дублирования.

Замечание 2. Дерево решений при поиске обычно не задается априори. Оно строится непосредственно в процессе поиска: когда возникает некоторая ситуация, тогда и определяются возможные направления процесса, представляемые совокупностью дуг, исходящих из вершины, соответствующей данной ситуации. Естественно стремиться по возможности сокращать число этих дуг, чтобы быстрее найти решение. Способы этого сокращения строятся с учетом особенностей конкретных задач.

Двумя основными факторами, которые делают бинарное дерево поиска, часто сокращенно называемое BST, оптимальным решением любых реальных проблем, являются скорость и точность. Алгоритмы, основанные на BST, часто используется в реальных решениях, таких как игры, автозаполнение данных и графика.

5. Задачи информационного поиска. В связи с возникновением и развитием сети Интернет стали особенно популярны службы и технологии, позволяющие пользователям информационных систем отыскивать необходимую информацию. Такие службы подразделяются на службы общего назначения, к примеру, системы поиска во всей сети Интернет: Google, Yahoo!, Яндекс, и специализированные порталы, содержащие информацию по каким-либо областям знания, к примеру, Download.Com (поиск программного обеспечения), GettyImages (поиск изображений) — причем это только самые крупные системы, более или менее полный список же занял бы много страниц. Все они предоставляют хорошие — в некотором смысле — результаты.

Системы поиска имеют широко разработанные технологии и связанный с ними математический аппарат для получения наилучших результатов поиска в больших массивах информации. Эти технологии основаны на данных о структуре естественных языков (законы Зипфа), а также на особенностях структуры Интернет (системы взаимных ссылок, популярность сайтов). Специализированные системы, в свою очередь, используют классификаторы, сходные с библиотечными, составляемые систематизаторами — профессионалами какой-либо узкой области.

Так, большая популярность системы Google связана с качественными результатами так называемого Page Rank ранжирования, т.е. определения позиции ресурса в результирующей выборке поискового запроса в зависимости от количества внешних ссылок на ресурс. Иными словами, чем полезнее ресурс, тем больше ресурсов на него ссылаются, тем ценнее содержащаяся на нем информация. В научном мире это известно как *индекс цитирования*.

Для поисков в русском сегменте сети Интернет особенно ценна способность поисковой системы Яндекс, искать не только слова запроса на русском языке, но и все его словоформы (поиск, поиска, поиске и т. д.).

Также для улучшения результатов используют следующие технологии:

(i) предыдущие запросы пользователя (сужая результаты согласно «интересам» пользователя (Google Personalized Search));
(ii) рекомендации других пользователей (Digg.Com, Del.icio.us);
(iii) сужение поиска согласно актуальности (News, Rss агрегаторы);
(iv) отсылки на аналогично описанные ресурсы (тэги в сервисе Flikr, навигация по ключевым словам).

Однако и эти способы не являются достаточными, а поэтому исследования в этой области продолжаются. Определим некоторые термины, которые не являются общепринятыми, однако понадобятся при дальнейшем изложении.

Для определения «нужности» информации при поиске обычно используют два термина:

(a) *релевантная информация* — информация, отвечающая формальному критерию поиска, например, ключевому слову или принадлежности определенной категории;
(b) *пертинентная информация* — информация, удовлетворяющая информационной потребности ищущего ее человека.

Рассмотрим пример поиска во всей сети Интернет поисковой службой общего назначения. Пусть поиск осуществляется по английскому слову object — объект. Список сайтов выдаваемых в ответом на подобный запрос будут ссылки на документы содержащие данные по программированию (object oriented programming), базам данных (object oriented databases), философии (object and subject relationship), лингвистике (theta role or θ-role). Какая информация будет пертинентной — заранее неизвестно. Основные отличия в результативности поиска достигаются путем предложения пользователю некоторого списка релевантных документов, однако достигнуть высокой степени пертинентности можно лишь используя разные стратегии выбора релевантных документов.

Обычно для оценки качества поиска меряют два параметра: точность (precision) — доля релевантного материала в ответе, полнота (recall) — доля найденных релевантных документов в общем числе релевантных документов коллекции.

Такие стратегии называют ранжированием информацию по принципу релевантности (в некотором смысле) содержащейся в них информации. При этом пользователь, изучающий информационный блок, может быть заинтересован также и другими блоками, если они содержат информацию, релевантную в некотором смысле. В связи с этим встает вопрос сравнения близости информационного содержимого в некоторой понятийной плоскости.

В приведенном выше примере в качестве близкого информационного блока можно рассматривать определение слова object, как и перевод его на другие языки.

Однако, очевидной проблемой является тот факт, что два блока, близкие в одной плоскости, могут оказаться достаточно далекими в другой, и нет никакой возможности сравнивать (и упорядочивать) содержимое без явного сравнения каждой пары. На данный момент подобная проблема решается поисковыми системами путем построения полного индекса значимых терминов по базе документов и последовательному уточнению запроса с использованием специального языка манипуляции данными. Однако при таком подходе поисковая система должна обладать мощными инструментами работы с большими массивами данных (как правило, распределенной реляционной базой данных, хранящей информацию о документах и их содержимом), и при этом нет гарантии, что какая-либо (возможно, пертинентная) информация не останется за пределами пересечения (упомянутое выше определение слова «object» вместе с его переводом на другие языки).

6. Поиск по ключевым словам и построение индекса релевантности. Поиск по ключевым словам — самый распространенный на настоящий момент способ поиска нужной информации в сети Интернет при помощи поисковых машин.

Организация поиска по ключевым словам предполагает введение в поисковой строке определяющего запрос слова или группы слов, которые являются главными для искомого документа. Можно также использовать сложные запросы, использующие логические операции, шаблоны и т. д. Поиск по одному ключевому слову всегда производится однозначно (ключевое слово либо имеется в рассматриваемом тексте, либо отсутствует). При поиске же по нескольким ключевым словам возможны различные способы их комбинирования:

(i) в искомом тексте присутствуют все заданные ключевые слова («и то, и другое»);
(ii) в тексте имеется хотя бы одно из заданных ключевых слов («то или другое»);
(iii) в тексте должно иметься одно ключевое слово, но обязательно должно отсутствовать другое («то, но не другое»);
(iv) ключевые слова составляют фразу, которая обязательно должна присутствовать в тексте «как есть»; ключевые слова обязательно должны быть взаимосвязаны (т.е. располагаться близко друг от друга, например, во фразе могут быть разделены союзом или прилагательным).

Чтобы «объяснить» поисковой системе, как нужно понимать заданную последовательность ключевых слов, используется особый язык построения запросов. Принципы построения языка запросов, как правило, универсальны для большинства поисковых систем, но для иной системы язык запросов может иметь свои особенности.

Пример языка построения запросов в поисковой системе «Яндекс»:

(a) заяц & кролик — поиск текстов, в которых есть оба заданных ключевых слова (и «заяц», и «кролик»);
(b) заяц | кролик — поиск текстов, в которых имеется хотя бы одно из заданных ключевых слов («заяц» или «кролик»);
(c) заяц − кролик — знак минуса перед словом «кролик» предписывает искать только такие тексты, в которых есть слово «заяц», но нет слова «кролик»;
(d) заяц + кролик — знак плюса перед словом «кролик» предписывает искать только такие тексты, в которых есть слово «заяц» и обязательно есть слово «кролик».

Большинство современных поисковых систем являются «интеллектуальными» — при поиске учитываются все возможные формы (падежи, склонения, спряжения, единственное и множественное число) заданных ключевых слов. Например, при задании ключевого слова «заяц» поисковая система будет искать все варианты ключевых слов — «заяц», «зайца», «зайцами» и т. п.

Нашей задачей является построение такого индекса, который бы позволял сравнивать и упорядочивать информационные блоки без прямого сравнения каждой пары, что может привести к возможности настройки релевантности результата и демонстрации пользователю дополнительных, возможно, пертинентных результатов (см. [9].

Замечание 3. Говоря «индекс», мы не подразумеваем индексы, которые используются в системах управления базами данных для ускорения поиска на физическом уровне организации данных. Технология построения таких индексов широко разработана и базируется на различных методах сортировки и поиска. Предлагаемый индекс позволяет не ускорить поиск, а добавить некоторый предопределенный (и возможно, полезный) смысл в результаты поиска.

Для построения такого индекса применимо обобщение *правила Паскаля*, используемое при построении V-пирамид с весовыми коэффициентами (1): элемент, стоящий ниже, является суммой трех элементов непосредственно над ним с некоторыми весами.

В вершине V-пирамиды разместим элемент, соответствующий ключевому понятию, на следующем, первом уровне — элементы, напрямую связанные с ключевым понятием, а в качестве весовых коэффициентов зададим релевантность по ключу. На втором уровне размещаются элементы, «связанные» с понятиями, которые сравниваются уже не с ключевым, а с понятиями первого уровня и т. д.

Для наглядности проиллюстрируем построение такого индекса на примере V-треугольника (2). Будем сопоставлять каждое ключевое слово с одним родительским и двумя дочерними. Оговоримся, что в данном случае количество дочерних слов выбрано лишь для примера, в реальной ситуации количество ветвей может быть от 0 (тупиковая ветвь) до некоторого конечного N. Полученному треугольнику, составленному из ключевых терминов $K_{i,j}$, сопоставим треугольник, составленный из коэффициентов $A_{i,j}$.

Коэффициенты $A_{i,j}$ определяются на этапе построения пирамиды для конечного множества ключевых слов $K_{i,j}$, коэффициенты релевантности определяются по некоторому, вообще говоря, произвольному алгоритму, который может быть как автоматизирован, так и исполняться с помощью человека-классификатора.

Рассмотрим примеры таких треугольников для заранее выбранного набора терминов. Множество ключевых слов K состоит из следующих элементов:

$$
\begin{aligned}
&K_{0,0} = \text{«object»}, && K_{1,0} = \text{«object oriented programming»},\\
&K_{0,1} = \text{«object oriented databases»}, && K_{2,0} = \text{«programming»},\\
&K_{1,1} = \text{«DB4Objects»}, && K_{0,2} = \text{«relational databases>},\\
&K_{3,0} = \text{«Plain ANSI C language»}, && K_{2,1} = \text{«C++ language»},\\
&K_{1,2} = \text{«JDBC»}, && K_{0,3} = \text{«Oracle»}.
\end{aligned}
$$

Множество коэффициентов A состоит из следующих элементов:

$$
\begin{aligned}
&A_{0,0} = 1, \quad A_{1,0} = 1, \quad A_{0,1} = 1, \quad A_{2,0} = 0{,}5, \quad A_{1,1} = 1,\\
&A_{0,2} = 0{,}5, \quad A_{3,0} = 1, \quad A_{2,1} = 1, \quad A_{1,2} = 0{,}5, \quad A_{0,3} = 1.
\end{aligned}
$$

Имея подобную структуру для заданного набора терминов, мы можем при отображении документов, содержащих тот или иной термин, используя весовые коэффициенты, отображать релевантные термины и документы с ними связанные. Связанность терминов можно оценивать согласно значениям, достигнутым при суммировании весовых коэффициентов при прохождении от вершины пирамиды коэффициентов.

Введем следующие обозначения: $V(n,k)$ — коэффициент релевантности; n — степень (уровень) удаленности от определяемого понятия (запроса); k — степень (частота, ранжирование) выраженности определяемого понятия (запроса); $\alpha_{n,k}$ — «стоимость» удовлетворительного (предполагаемого) ответа; $\beta_{n,k}$ — «стоимость» неудовлетворительного (не предполагаемого) ответа.

Замечание 4. Траектория $V(0,0) \to V(n,k)$ иллюстрирует процесс получения ответа $V(n,k)$ на уточнение запроса $V(0,0)$.

7. Частные случаи. Рассмотрим частные случаи обобщенного треугольника Паскаля. При этом будем предполагать, что $V(0,0)=1$.

Пусть $\alpha_{n,k}=1$, $\beta_{n,k}=\mu_{n-1}$, $n=\overline{1,\infty}$, $k=\overline{0,n}$. В этом случае соотношение (2) примет вид

$$B_0^0=1; \qquad B_k^n=B_{k-1}^{n-1}+\mu_{n-1}B_k^{n-1}, \quad n=\overline{1,\infty}, \quad k=\overline{0,n}. \tag{6}$$

Если же $\alpha_{n,k}=1$, $\beta_{n,k}=\mu_k$, $n=\overline{1,\infty}$, $k=\overline{0,n}$, то из (2) имеем

$$A_0^0=1; \qquad A_k^n=A_{k-1}^{n-1}+\mu_k A_k^{n-1}, \quad n=\overline{1,\infty}, \quad k=\overline{0,n}. \tag{7}$$

Числа B_k^n и A_k^n, удовлетворяющие рекуррентным соотношениям (6) и (7), называются *обобщенными числами Стирлинга* первого и второго рода соответственно.

Перейдем к рассмотрению частных случаев формирования индекса релевантности, допускающих описание при помощи обобщенных чисел Стирлинга, построенных на специальным образом нормированных базах (см. [6, 13]). Это те случаи, когда весовые коэффициенты зависят только от одного из двух своих индексов — параметров. Далее считаем $V(0,0)=1$.

7.1. Пусть условия, в которых осуществляется поиск, изменяются от запроса к запросу, но различия между итогами запросов таковы, что не влияют на весовые коэффициенты. Полагаем $\alpha_{n,k}=\alpha_n$, $\beta_{n,k}=\beta_n$, $n=\overline{1,\infty}$, $k=\overline{0,n}$. В этом случае из формулы (2) имеем:

$$V(n,k)=\alpha_n V(n-1,k-1)+\beta_n V(n-1,k), \quad n=\overline{1,\infty}, \quad k=\overline{0,n}.$$

Считая, что в последовательности $\{\alpha_n\}_{n=1}^{\infty}$ все члены отличны от нуля, введем величины

$$B_0^0=1; \qquad B_k^n=V(n,k)\prod_{i=1}^{n}\alpha_i^{-1}, \quad n=\overline{1,\infty}, \quad k=\overline{0,n}.$$

Тогда

$$B_0^0=1; \qquad B_k^n=B_{k-1}^{n-1}+\beta_n\alpha_n^{-1}B_k^{n-1}, \quad n=\overline{1,\infty}, \quad k=\overline{0,n}.$$

Сравнивая последнее соотношение с (6), видим, что после вышеуказанной нормировки значения индекса релевантности выражаются при помощи обобщенных чисел Стирлинга первого рода, построенных на базе $\{\beta_n\alpha_n^{-1}\}_{n=1}^{\infty}$.

Может оказаться, что в последовательности $\{\alpha_n\}_{n=1}^{\infty}$ встречаются члены, равные нулю, но существует бесконечное множество членов, отличных от нуля. Тогда следует произвести объединение некоторых, расположенных подряд, этапов информационного поиска с целью получения эквивалентной схемы поиска, приводящей к тем же распределениям, но уже с коэффициентами $\widetilde{\alpha}_m$, среди которых нет равных нулю. Заранее отметим, что при $\alpha_n=0$ будем считать $\beta_n\neq 0$, так как в противном случае n-й этап привел бы к окончанию информационного поиска, и вопрос изучения индекса релевантности был бы исчерпан.

Переход к эквивалентной схеме поиска реализуется следующим образом. Из последовательности $\{\alpha_n\}_{n=1}^{\infty}$ выбираем, с сохранением порядка, подпоследовательность $\{\alpha_{n_m}\}_{m=1}^{\infty}$ членов, отличных от нуля. К каждому новому этапу с номером m относим старые этапы с номерами от n_m до $n_{m+1}-1$ включительно. На основании формул (2) заключаем, что коэффициенты «стоимости» на объединенном этапе с номером m выразятся через коэффициенты, соответствующие вошедшим в него старым этапам, следующим образом:

$$\widetilde{\beta}_m=\beta_{n_m}\pi_m, \quad \widetilde{\alpha}_m=\alpha_{n_m}\pi_m, \quad \pi_m=\prod_{i=n_m+1}^{n_{m+1}-1}\alpha_i.$$

Таким образом, новая последовательность $\{\widetilde{\alpha}_m\}_{m=1}^{\infty}$ уже не содержит элементов, равных нулю.

Наконец, может оказаться, что в последовательности $\{\alpha_n\}_{n=1}^{\infty}$, начиная с некоторого $n = n_0$, все последующие члены равны нулю. Это означает, что, начиная с $n = n_0$, дальнейшая история поиска упрощенно описывается с учетом только коэффициентов β_n.

7.2. Пусть условия, в которых осуществляется поиск, при чередовании этапов остаются неизменным, но различия между итогами запросов влияют на весовые коэффициенты. Положим $\alpha_{n,k} = \alpha_k$, $\beta_{n,k} = \beta_k$, $n = \overline{1,\infty}$, $k = \overline{0,n}$. В этом случае из соотношения (2) имеем:

$$V(n,k) = \alpha_{k-1}V(n-1,k-1) + \beta_k V(n-1,k), \quad n = \overline{1,\infty}, \quad k = \overline{0,\infty}.$$

Не нарушая общности рассмотрения, считаем, что в последовательности $\{\alpha_k\}_{k=0}^{\infty}$ все члены отличны от нуля. Введем обозначения

$$A_0^n = V(n,0), \quad n = \overline{0,\infty}; \qquad A_k^n = V(n,k)\prod_{i=0}^{k-1}\alpha_i^{-1}, \quad n = \overline{1,\infty}, \quad k = \overline{1,n}. \tag{8}$$

Тогда

$$A_0^0 = 1; \qquad A_k^n = A_{k-1}^{n-1} + \beta_k A_k^{n-1}, \quad n = \overline{1,\infty}, \quad k = \overline{0,n}.$$

Сравнивая последнее соотношение с (8), видим, что величины (8) являются обобщенными числами Стирлинга второго рода, построенными на базе $\{\beta_k\}_{k=0}^{\infty}$.

7.3. Рассмотрим случай, когда $\alpha_{n,k} = \alpha_k$, $\beta_{n,k} = \beta_n$, $n = \overline{1,\infty}$, $k = \overline{0,n-1}$. Тогда из выражения (2) имеем:

$$V(n,k) = \alpha_{k-1}V(n-1,k-1) + \beta_n V(n-1,k), \quad n = \overline{1,\infty}, \quad k = \overline{0,n}. \tag{9}$$

Как и в предыдущем случае, полагаем, что в последовательности $\{\alpha_k\}_{k=0}^{\infty}$ все члены отличны от нуля. Введем обозначения

$$B_0^n = 1, \qquad B = V(n,0), \quad n = \overline{1,\infty}; \qquad B_k^n = V(n,k)\prod_{i=0}^{k-1}\alpha_i^{-1}, \quad n = \overline{1,\infty}, \quad k = \overline{1,n},$$

С учетом этих обозначений соотношение (9) можно представить в виде

$$B_0^0 = 1; \qquad B_k^n = B_{k-1}^{n-1} + \beta_n B_k^{n-1}, \quad n = \overline{1,\infty}, \quad k = \overline{0,n}.$$

Сравнивая последнее соотношение с (6), видим, что величины (9) являются обобщенными числами Стирлинга первого рода, построенными на базе $\{\beta_n\}_{n=1}^{\infty}$.

Таким образом, мы сконструировали индекс, который отображает долю релевантного материала и позволяет производить сравнения на множестве терминов, исходя из весовых коэффициентов терминов и путей.

СПИСОК ЛИТЕРАТУРЫ

1. *Альсведе В., Вегенер И.* Задачи поиска. — М.: Мир, 1982.
2. *Балагура А. А., Кузьмин О. В.* Обобщенная пирамида Паскаля и частично упорядоченные множества// Обозр. прикл. пром. мат. — 2007. — *14*, № 1. — С. 88–91.
3. *Бондаренко Б. А.* Обобщенные треугольники и пирамиды Паскаля, их фракталы, графы и приложения. — Ташкент: Фан, 1990.
4. *Гретцер Г.* Общая теория решеток. — М.: Мир, 1982.
5. *Дейт К. Дж.* Введение в системы баз данных. — М.: Вильямс, 2001.
6. *Докин В. Н.* О треугольной схеме развития популяций// Исслед. геомагнет. аэроном. физ. Солнца. — 1975. — *41*. — С. 104–106.
7. *Кристофидес Н.* Теория графов. Алгоритмический подход. — М.: Мир, 1978.
8. *Кузьмин О. В.* Обобщенные пирамиды Паскаля и их приложения. — Новосибирск: Наука, 2000.
9. *Кузьмин О. В., Логинов Т. А.* Построение индекса релевантности с помощью обобщенных пирамид Паскаля// Вестн. Бурят. ун-та. Сер. 13. Мат. Информ. — 2006. — № 3. — С. 40–45.
10. *Кузьмин О. В., Серёгина М. В.* Верхние отсечения обобщенной пирамиды Паскаля и их интерпретации// Ж. Сиб. фед. ун-та. Сер. мат. физ. — 2010. — *3*, № 4. — С. 533–543.

11. *Лебедев В. Б. , Федотов Е. А.* Моделирование данных информационных систем методами теории решеток// Изв. вузов. Поволж. рег. Сер. тех. науки. — 2015. — *3*. — С. 104–110.
12. *Месарович М., Мако Д., Такахара И.* Теория иерархических многоуровневых систем. — М.: Мир, 1973.
13. *Платонов М. Л., Докин В. Н.* Треугольная схема развития популяций// Исслед. геомагнет. аэроном. физ. Солнца. — 1975. — *35*. — С. 26–31.
14. *Саати Т.* Принятие решений. Метод анализа иерархий. — М.: Радио и связь, 1993.
15. *Стенли Р.* Перечислительная комбинаторика. — М.: Мир, 1990.
16. *Тиори Т., Фрай Дж.* Проектирование структур баз данных. Кн. 1. — М.: Мир, 1985.
17. *Balagura A. A., Kuzmin O. V.* Generalized Pascal pyramids and their reciprocals// Discr. Math. Appl. — 2007. — *17*. — P. 619-628.
18. *Breiman L., Friedma J., Olshen R., Stone C.* Classification and Regression Trees. — New York: Wadsworth Books, 1984.
19. *Hovland C. I.* Computer simulation of thinking// Am. Psychologist. — 1960. — *15*, № 11. — P. 687–693.
20. *Hunt E. B., Janet M., J. S. Philip J. S.* Experiments in Induction. — New York: Academic Press, 1966.
21. *Kuzmin O. V.* Generalized Pascal's pyramids and decision trees// Adv. Appl. Discr. Math. — 2022. — *34*. — P. 1–15.
22. *Kuzmin O. V., Balagura A. A., Kuzmina V. V., Khudonogov I. A.* Partially ordered sets and combinatory objects of the pyramidal structure// Adv. Appl. Discr. Math. — 2019. — *20*, № 2. — P. 219–236.
23. *Kuzmin O. V., Khomenko A. P., Artyunin A. I.* Discrete model of static loads distribution management on lattice structures// Adv. Appl. Discr. Math. — 2018. — *19*, № 3. — P. 183–193.
24. *Kuzmin O. V., Khomenko A. P., Artyunin A. I.* Development of special mathematical software using combinatorial numbers and lattice structure analysis// Adv. Appl. Discr. Math. — 2018. — *19*, № 3. — P. 229–242.
25. *Kuzmin O. V., Seregina M. V.* Plane sections of the generalized Pascal pyramid and their interpretations// Discr. Math. Appl. — 2010. — *20*, № 4. — P. 377–389.
26. *Lovász L.* Combinatorial Problems and Exercises. — Budapest: Akadémiai Kiadó, 1979.
27. *Murthy S. K.* Automatic construction of decision trees from data: A multidisciplinary survey// Data Mining and Knowledge Discovery. — 1998. — *2*. — C. 345–389.
28. *Quinlan J. R.* Induction of decision trees// Machine Learning. — 1986. — *1*. — P. 81–106.
29. *Quinlan J. R.* C4.5: Programs for Machine learning. — San Mate: Morgan Kaufmann Publ., 1993.

Кузьмин Олег Викторович
Иркутский государственный университет
E-mail: quzminov@mail.ru