Nom-11-2024.qxd 05.12.2024 16:58 Page 38

ИНФОРМАТИКА

ПОСТРОЕНИЕ БИНАРНОГО ДЕРЕВА МИНИМАЛЬНОЙ ЦЕНЫ

DOI: 10.36724/2072-8735-2024-18-11-38-44

Manuscript received 30 September 2024;
Accepted 24 October 2024

Гадасин Денис Вадимович,
Московский технический университет связи и информатики (МТУСИ), Москва, Россия,
dengadiplom@mail.ru

Ключевые слова: *дерево поиска, алгоритм Гарсиа-Вока, сложность алгоритма, оптимальность, распределение нагрузки между узлами*

Один из способов, применяемый при формализация предметной области, состоит в постепенной ее дефрагментации по принципу от большего к меньшему и установления связи полученных частей с числовыми значениями. За минимальную, неделимую часть предметной области принимается атрибут. Атрибут может принимать допустимые значения. Время реакции на события, происходящие в предметной области зависит от времени поиска состояния атрибута, которое зависит от вида структуры в которую объединены атрибуты. В работе рассматривается бинарная древовидная структура организации атрибутов. Времени поиска в такой структуре в соответствие ставится показатель стоимости дерева. Стоимость дерева определяется как сумма произведений весов вершин и уровня, на котором эта вершина расположена. Таким образом необходимо так расположить вершины по уровням, что бы полученная стоимость была как можно меньше, а дерево при этом оставалось бы бинарным. Для построения дерева в работе рассматривается алгоритм Гарсиа-Вока. Данный алгоритм позволяет строить дерево минимальной стоимости для жестко заданной последовательности атрибутов, имеющих соответствующие веса. Рассматривается способ снижения алгоритмической сложности. Для проведения эксперимента берутся атрибуты, буквы, алфавита русского языка плюс символ "пробел". В качестве веса атрибута принимается средняя частота появления символа в языке. Эксперимент проводится для четырех способов первоначального упорядочивания символов исходя из их веса: алфавитный порядок, обратный алфавитный порядок, по убыванию, по возрастанию, возрастанию до максимума и убыванию, по убыванию до максимума и возрастанию. Построенное дерево позволяет получить неравномерный двоичный код для каждого символа алфавита и вычислить количество двоичных символов, приходящихся на один символ алфавита. Данная величина полностью соответствует стоимости дерева. Исходя из этого определяется наиболее предпочтительная конфигурация атрибутов для построения бинарного дерева минимальной стоимости.

Для цитирования:
Гадасин Д.В. Построение бинарного дерева минимальной цены // T-Comm: Телекоммуникации и транспорт. 2024. Том 18. №11. С. 38-44.

For citation:
Gadasin D.V. Building a binary tree of the minimum prices. *T-Comm*, vol. 18, no. 11, pp. 38-44. (*in Russian*)

Nom-11-2024.qxd 05.12.2024 16:58 Page 39

Введение

Один из способов, применяемый при формализация предметной области, состоит в постепенной ее дефрагментации по принципу от большего к меньшему и установления связи полученных частей с числовыми значениями. На начальном этапе происходит выделение сущностей, для сущностей определяют атрибуты, а в дальнейшем для каждого атрибута определяется множество значений, которые может принимать отдельный атрибут. Как пример применения такой последовательности дефрагментации используется в системах поддержки принятия решений и при работе с большими данными [1-5].

Значения атрибутов определенной предметной области можно упорядочить по вероятности их появления, и на их основе возможно построить дерево, на котором реализовать алгоритм поиска и определить – являются ли данные поиска допустимыми для выбранного атрибута в предметной области. Дерево представляет собой один из видов направленных графов, на котором необходимо осуществить поиск существования или не существования маршрута, что рассматривается в [6-8].

Условия поиска определяются критериями, которые, в свою очередь, могут быть заданы как в явном виде, так и в неявном, но результат поиска является основанием для принятия решений: о распределении нагрузки между узлами, применения алгоритма обработки данных, оперативного влияния на критическую ситуацию [9-15]. Одним из основных критериев качества поиска является критерий времени затраченного на осуществление поиска. Результатом поиска является ответ – присутствует/не присутствует данное значение в предметной области, следовательно, основным параметром, на основе которого определяется качество поиска является количество сравнений, чем больше сравнений, тем больше времени надо для осуществления поиска по заданному значению. Логично предположить, что в упорядоченной структуре поиск должен осуществляться быстрее, чем в неупорядоченной.

Построение дерево поиска

Возможной структурой, в которую преобразуются исходные данные для поиска, является дерево, построенное с применением алгоритмов [16-19]. Форма дерева влияет на скорость поиска, так как по мере продвижения от корня дерева до листа (внешняя вершина) сравнения и выбор направления происходят во внутренних вершинах, а конечный результат – успешен/не успешен поиск во внешней вершине. Определим, что оптимальным деревом поиска является то, в котором среднее количество сравнений минимально.

Пусть количество вершин N=4, ключи, которым соответствуют данные вершины представляют собой множество ключей K={K_1,K_2,K_3,K_4}, данное множество упорядочено по K, $K_1<K_2<K_3<K_4$, пусть ключи появляются с вероятностями $p_{K_1}, p_{K_2}, p_{K_3}, p_{K_4}$ соответственно. В соответствии с тем, что критерием, по которому производится поиск, является вероятность, то она может:

1. совпадать с вероятностью $p_i = p_{K_i}$

2. не совпадать $p_i \neq p_{K_i}$

В таком случае общее количество вероятностей может быть P=2N+1, где:

1. p_i – вероятность события при котором критерий поиска совпадает с вероятностью появления ключа K_i.

2. q_i – вероятность события при котором критерий поиска не совпадает с вероятностью появления ключа K_i, т.е. принадлежит в интервалу [K_i, K_{i+1}], q_0 поставим в соответствие событие когда значение критерия поиска меньше вероятности появления ключа K_1, а q_n – вероятность события что значение критерия поиска больше вероятности появления ключа K_N.

В случае введения данных ограничений $p_1 + p_2 + ... p_n + q_0 + q_1 + ... q_n = 1$. Минимальное число сравнений при осуществлении поиска в бинарном дереве определяется формулой

$$C_{min} = \sum_{i=1}^{n} p_i \cdot (a_{i-1} + 1) + \sum_{i=0}^{n} q_i \cdot b_{i+1} \quad (1)$$

где n – число, соответствующее максимальному номеру уровня, на котором находятся внутренние узлы дерева; a_i – номер уровня, на котором находится внутренний узел; b_i – номер уровня, на котором находится внешний узел; корень дерева всегда находится на нулевом уровне.

В бинарном дереве поиска любая внутренняя вершина связана двумя связями либо с внутренними вершинами, либо с внешними вершинами, либо с внутренней и внешней. Тогда количество деревьев, которые могут быть образованы из N внутренних вершин равно:

$$COT = \frac{C_{2N}^{N}}{(N+1)} \quad (2)$$

отношения числа сочетаний из удвоенного количества вершин по количеству вершину к общему количеству вершин плюс один [20]. Для случая из четырех вершин общее количество деревьев может быть 14, на рисунке 1 представлены два из них.

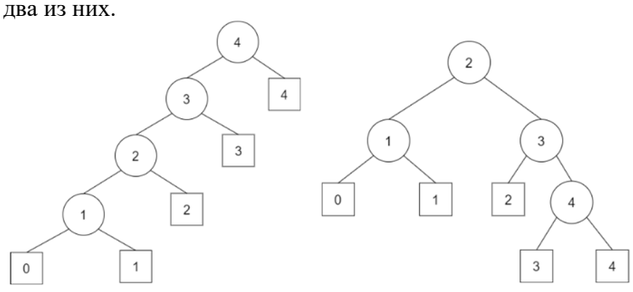

Рис. 1. а) Топология 1; б) Топология 2

В том случае, когда цену дерева принимается количество сравнений, то тогда цена дерева, а есть:

$$C_a = 4 \cdot (p_1 + q_0 + q_1) + 3 \cdot (p_2 + q_2) + 2 \cdot (p_3 + q_3) + (p_4 + q_4)$$

цена дерева б составляет

$$C_б = 3 \cdot (p_4 + q_3 + q\ ) + 2 \cdot (p_3 + q_2) + 2 \cdot (p_1 + q_0 + q_1) + p_2$$

Если при подсчете цен всех оставшихся деревьев цена второго дерева окажется минимальной, то для данных значений вероятности появления ключей данная топология дерева для осуществления поиска будет оптимальной исходя из числа сравнений. В общем случае ограничение на то что бы $p_1 + p_2 + ... p_n + q_0 + q_1 + ... q_n = 1$ является чрезмерным, так как

Nom-11-2024.qxd 05.12.2024 16:58 Page 40

возможно осуществлять поиск деревьев с минимальной ценой для любых значений весов $(p_1, p_2, ...p_n; q_0, q_1 + ...q_n)$, так как в данной последовательности всегда можно провести операцию нормирования весов и получить $p_1 + p_2 + ...p_n + q_0 + q_1 + ...q_n = 1$.

В том случае, если N слишком велико, то число сочетаний $COT \approx \frac{4^N}{\sqrt{\pi} \cdot N^{3/2}}$ и для того, чтобы определить какое дерево является оптимальным необходимо произвести прямой перебор всех деревьев, что является сложной вычислительной задачей. Известно, что если в оптимальном дереве возможно выделить какое-либо поддерево, то оно так же должно быть оптимальным, в противном случае исходное дерево не является оптимальным. Если на рисунке 1б представлено оптимальное дерево, состоящее из весов $T=(p_1,p_2,p_3,p_4,q_0,q_1,q_2, q_3,q_4)$, то его левое и правое поддерево, состоящее из весов $T_L=(p_1,q_0,q_1)$ и $T_R=(p_3,p_2,q_2,q_2,q_3,q_4)$ соответственно так же являются оптимальными. В таком случае построение оптимального дерева возможно начать с минимально возможного оптимального поддерева и итерационно проводить его наращивание по принципу от меньшего к большему.

Введем обозначения, стоимость оптимального дерева с весами $(p_{i+1}, p_{i+2}, ...p_j; q_i, q_{i+1} + ...q_j)$ обозначим как $c(i;j)$, а сумму этих весов как $w(i;j)$, тогда $c(i;j)$ и $w(i;j)$ определены для $\forall$ i и j где $0 \le i \le j \le n$. Определим минимально возможную стоимость дерева с корнем K, который располагается на нулевом уровне как C^K_{min}, где

$$C^K_{min} = w(i;j) + c(i,k-1) + c(k,j) \qquad (3)$$

Тогда исходя из данного уравнения

$$K = \begin{cases} c(i;i) = 0 \\ c(i;j) = w(i,j) + \min\limits_{i<k\le j}\left(c(i,k-1) + c(k,j)\right) \quad \text{для } i < j \end{cases} \qquad (4)$$

Решением уравнения является множество корней деревьев, топология которых будет соответствовать критерию оптимального дерева. Количество значений стоимости c(i;j) этих деревьев для i<j и j-i=1,2,…,n оценивается порядка ½n^2. Помимо оценки стоимости необходимо так же провести операцию минимизации, которая выполняется в приблизительно $\frac{1}{6}n^3$ раз. Исходя из полученных значений алгоритмическая сложность построения оптимального дерева соответствует $O(n^3)$.

Уменьшить алгоритмическую сложность возможно в следствие того, что функция, в результате которой получается множество корней деревьев обладает свойством монотонности. Пусть $K(i,j)=\{k_{ij}\}$ для любых i<j. Вследствие того что k_{ij} принадлежит множеству корней и в случае если веса корней не отрицательны, то не обязательно определять все множество K(i,j), а достаточно найти один элемент данного множества, все остальные элементы определяются из соотношения:

$$k(i,j-1) \le k(i,j) \le k(i+1,j) \qquad (5)$$

Неравенство 5 позволяет снизить количество необходимых проверок для определения оптимальности до $k(i+1,j) - k(i,j-1) + 1$ значений, а приняв j-i=u можно оценить общую сложность алгоритма как

$$\sum_{\substack{u \le j \le n \\ i=j-u}} \left(k(i+1,j) - k(i,j-1) + 1\right) = k(n-u+1,n) - k(0,u-1) + n - u + 1 \qquad (6)$$

При любых допустимых значениях переменных, входящих в формулу 6 итоговая сумм всегда меньше 2n, поэтому алгоритмическая сложность снижается до $O(n^2)$.

Алгоритм Гарсиа-Воча

Для многих задач поиска критическое значение имеет только вероятность появления внешних вершин, а во внутренних вершинах производится только сравнения значения со стоимостью вершины и определение направления движения вверх по дереву (предполагается что движение осуществляется от корня до какой-либо внешней вершины), т.е. $p_1 = p_2 = ... = p_n = 0$. В данном случае возможно доказать, что стоимость дерева снизу ограничена энтропией распределения вероятностей листьев, а сверху тем же значением, увеличенным на два $C(q_0,q_1,...,q_n) = \left[H(q_0,q_1,...,q_n); H(q_0,q_1,...,q_n)\right[$, тогда функция цены дерева определяется простым суммированием произведений вероятности появления внешней вершины и номера уровня этой вершины g_k:

$$C_T = \sum_{k=0}^{n} q_k \cdot g_k$$

Сложность изначального алгоритма построения оптимального дерева сокращается, в соответствии с тем, что деревья, в которых $p_1 = p_2 = ... = p_n = 0$ обладают свойством того, что если по отношению к текущему внешнему узлу вероятность предыдущего больше вероятности последующего, то данная вершина не может находиться более чем на один уровень ниже, если же по отношению к текущему внешнему узлу вероятность предыдущего равна вероятности последующего, то данная вершина в редких случаях находится более чем на один уровень ниже, т.е. если $q_{k-1} > q_{k+1}$ то $g_k \le g_{k+1}$, если $q_{k-1} = q_{k+1}$ то $P(g_k \le g_{k+1}) \to 0$.

Алгоритм построения оптимального дерева выглядит следующим образом:

1. На первом этапе реализации алгоритма формируется множество весов $W=\{w_1,w_2,\ldots,w_n\}$, которые будут соответствовать внешним вершинам (листьям) дерева.

2. В соответствии с тем что сравниваются значения вероятностей то формируется множество вероятностей $Q=\{q_0,q_1,\ldots,q_{k+1}\}$ появления вершины путем проведения операции нормирования $q_k = \frac{w_i}{\sum_{i=1}^{n} w_i}$, где k=i=1,2,…,n и $q_0=q_{k+1}=0$.

3. В качестве начального элемента выбирается элемент с индексом 1 из множества Q и производится операция сравнения веса элемента $q_{k-1} > q_{k+1}$.

Nom-11-2024.qxd 05.12.2024 16:58 Page 41

4. Если результат операции «истина», то необходимо провести слияние элементов q_{k-1} и q_k. В результате операции слияния получается внутренняя вершина $q_{k-1,k}$ дерева уровня $g = \max(g_{q_{k-1}}, g_{q_k}) + 1$, т.е выбирается максимальный уровень на котором находятся вершины q_{k-1} и q_k и добавляется новый уровень. Вес вершины нового уровня определяется простым сложением весов вершин q_{k-1} и q_k, $q_{k-1,k}=q_{k-1}+q_k$ данные вершины удаляются из множества Q, но остаются в формируемом дереве.

5. После проведения операции слияния необходимо определить место куда вставляется вершина $q_{k-1,k}$ и переупорядочить множество Q исходя от веса вновь появившейся вершины. Для выполнения такой операции производится последовательное сравнение веса $q_{k-1,k}$ с весами вершин начиная с q_{k-2}. Если $q_{k-i} \geq q_{k-1,k}$, где i=2,3,..,k то вершине $q_{k-1,k}$ присваивается индекс r=k-i+1, если i=k то присваивается индекс r=1. После этого производится переиндексация элементов множества Q начиная с элемента с индексом q_{r+j} которому присваивается значение элемента $q_{k-i+1+j}$ j=1,2,…,n-1. Новая мощность множества Q становится n=n-1.

6. Если результат операции сравнения «ложно», то выбирается элемент со следующим индексом.

7. Работа алгоритма прекращается когда множество Q={0,W,0}, множество содержит три элемента $q_0=q_{k+1}=0$ и единственную вершину, которая будет являться корневой вершиной дерева и иметь вес суммы всех листьев.

8. Анализ шагов, в обратном порядке, которые позволили получить корневую вершину, позволяет построить дерево, т.к. известно с каким вершинами следующего уровня имеют связи вершины текущего уровня.

Листья в образовавшемся дереве будут находится в порядке, в отличном от порядка, в котором они находились во множестве весов, полученном на начальном этапе. Порядок представления изменяется в следствие того, что в любом случае вершина $q_{k-1,k}$ сдвигается влево, а с этой вершиной всегда связаны внешние вершины.

Для удобства представления желательно преобразовать полученное дерево, в дерево, в котором листья будут иметь изначальный порядок и находиться на том же уровне, что и в исходном. Для этого необходимо доказать, что стоимость дерева, которое получается после проведения операции слияния есть стоимость дерева до операции слияния плюс стоимость вновь полученной внутренней врешины:

$$C(q_{oj}, q_{1j}, \ldots, q_{n-1j}) = (q_{k-1} + q_k) + C(q_o, q_1, \ldots, q_n)$$

где $q_{oj}, q_{1j}, \ldots, q_{n-1j}$ вероятности вершин полученные на j-ой итерации выполнения шага 5 алгоритма.

В любом вновь получаемом дереве, вероятности $q_{oj}, q_{1j}, \ldots, q_{n-1j}$ соответствуют дереву, в котором вершина с весом $q_{k-1,k}=q_{k-1}+q_k$ включает в себя дочерние узлы одного и того же родительского узла. В том случае если r=k-1, т.е. вновь образуемый узел никуда не перемещается то утверждение очевидно.

Если же узел перемещается то $q_{i-1,j} > q_{i+1,j}$ для i<k-1, т.к $q_{k-i} \geq q_{k-1,k} > q_{k-i+1}$, таким образом уровень вершины $q_{k-1,k}$ $g_{q_{k-1,k}} \leq g_{q_{k-i+1}} \leq \ldots \leq g_{q_{k-2}}$ и если в данном случае соблюдается равенство, то узел $q_{k-1,k}$ смещается вправо тем самым определяя правильный порядок листьев. В противном случае, вершина q_{k-2} находится на уровень выше вершины $q_{k-1,k}$, что так же не ведет к изменению цены, а следовательно с помощью циклической перестановки возможно добиться правильного расположения вершин. Сложность алгоритма в таком случае снижается до O(nlogn)

Эксперимент по построению деревьев

Для эксперимента по построению дерева будем использовать русский алфавит, который помимо тридцати трех букв включает в себя и символ пробела. Частоту появления символов возьмем из [21].

Буква	Относ. частота	Буква	Относ. частота	Буква	Относ. частота	Буква	Относ. частота
Пробел	17,49%	З	1,36%	Р	3,90%	Щ	0,30%
А	6,61%	И	6,06%	С	4,51%	Ъ	0,03%
Б	1,31%	Й	1,00%	Т	5,16%	Ы	1,57%
В	3,75%	К	2,88%	У	2,16%	Ь	1,44%
Г	1,40%	Л	3,63%	Ф	0,21%	Э	0,26%
Д	2,46%	М	2,65%	Х	0,80%	Ю	0,53%
Е	6,97%	Н	5,53%	Ц	0,40%	Я	1,66%
Ё	0,03%	О	9,05%	Ч	1,19%		
Ж	0,78%	П	2,32%	Ш	0,60%		

Упорядочим данную таблицу в соответствии с порядком букв в русском языке, а знак, который разделяет слова (пробел) поставим перед буквой «А». При сложении всех относительных частот получилось 100,01%, поэтому для символа «пробел» относительная частота была уменьшена на 0,01% и стала 17,49%. Для удобства подсчета умножим все относительные частоты на 100 для того, чтобы проводить вычисления с целыми числами.

Перед построением дерева необходимо было сформировать множество весов, но в следствие того, что для каждого символа изначально определен вес, то данное множество сформировано, а вес этого множества W=10000. В русском алфавите содержится 33 буквы, в эксперименте принимают участие тридцать три буквы и символ «пробел», поэтому мощность множества Q составляет 36 символов, т.к. для реализации алгоритма в него включены два фиктивных элемента, веса которых соответственно равны нулю, $q_0 = q_{35}=0$. Таким образом значение множества весов не изменяется.

При построении дерева первый элемент который рассматривается q_1=«Пробел» и его вес сравнивается с весом элемента q_3=«Б», вес элемента q_1 больше веса элемента q_3 поэтому происходит переход к элементу q_2=«А» и происходит сравнение с элементом q_4=«В». Результат такой же и происходит переход к элементу q_3=«Б»=131 и его вес сравнивается с элементом q_5= «Г»=140, т.к. 131<140, то необходимо объединить вершины q_3 и q_4 в одну с общим весом q_3+q_4. При объединении получилась вершина «БВ» с общим весом $w_{БВ}$=131+375=506. После того, как была получена новая вершина и определен ее вес, то вес данной вершины сравнивается с весом вершины которая находится на одну вершину левее, а именно весом элемента $q_1=w_A$=661, 506<661 поэтому вершина «БВ» остается на месте и ей присваивается элемент q_3, а для всех элементов множества Q, начиная с элемента q_4 происходит переопределение индекса на единицу меньше.

Работа алгоритма продолжается с элемента q_3= «БВ», вес которого 506. При попадании в элемент q_9= «Л» его вес 363 сравнивается с весом элемента q_{11}=wH=553. Сравнение показало, что 363<553, происходит объединение вершин q_9 и q_{10} в вершину «ЛМ» с весом $w_{ЛМ}$=363+265=628.

Nom-11-2024.qxd 05.12.2024 16:58 Page 42

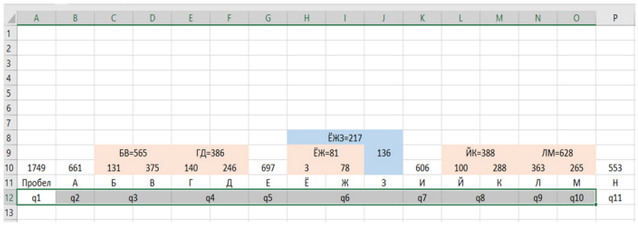

Рис. 2. Объединение вершин

Вес вершины «ЛМ» сравнивается с весом вершины q_8 и исходя из того что ее вес больше происходит сравнение далее последовательно с вершинами q_7, q_6, q_5. Сравнение вершины с q_5 показал что 697 больше 628, поэтому вершина «ЛМ» передвигается влево и получает индекс q_6, а элементы которые имели индексы q_6-q_8 получают соответственно индексы q_7-q_9, а у всех оставшихся элементов множества q индексы уменьшаются на единицу.

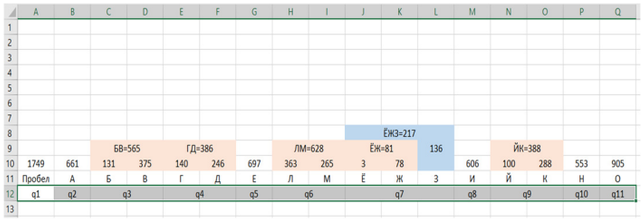

Рис. 3. Сдвиг объединенной вершины

Дерево считается построенным когда в множестве Q остаются три элемента при этом элементы q_0 и q_2 фиктивные, а вес элемента $q_{\text{Пробел,А,…,Я}}$= $w_{\text{Пробел,А,…,Я}}$=10000.

На следующем шаге необходимо определить стоимость полученного дерева для этого необходимо определить уровень вершины, которая располагается в листе дерева, относительно ее корня. Из рисунка 3 видно, что на одной из итерации алгоритма вершина «З» находится на уровень выше всех остальных вершин, при этом изначальный вес этой вершины не изменился. Производится последовательное сложение произведений веса вершины на тот уровень, на котором располагается данная вершина, расчет стоимости показал, что стоимость дерева, для русского алфавита, построенного исходя из порядка букв $C_{\text{Пробел,А,…,Я}}$=4,464.

Представленный алгоритм строит оптимальное двоичное дерево исходя из предопределенного порядка, в рассматриваемом примере алфавит был упорядочен прямым способом. Если порядок изначального множества элементов Q поменяется, то должно получится другое дерево, с другой стоимостью. Для подтверждения (опровержения) данного предположения было предложено 5 вариантов построения дерева.

1. Обратный порядок алфавита, символ «Пробел» находится после буквы А Q={Я,Ю,…,А,Пробел}, $C_{\text{обр.Алф.}}$.

2. Последовательно убывание частоты появления, от самого частого к самому редкому, что соответствует от элемента который весит больше всех, к элементу, который весит меньше всех, при совпадении веса более тяжелым считается элемент, который находится раньше в алфавитном порядке, $C_{\text{убыв.}}$.

3. Последовательно возрастание частоты появления, от самого редкого к самому частому, при совпадении веса более легким считается элемент, который находится позже в алфавитном порядке, $C_{\text{возр.}}$.

4. Порядок, который схож с графиком функции обратной параболы, т.е. все множество весов упорядочивается по убыванию веса, количество элементов делится пополам, если количество элементов четное, то элементу $q_{n/2}$ присваивается значение q_1, $q_{n/2-1}$=q_2, $q_{n/2+1}$=q_3 и т.д., если количество элементов нечетное то элементу $\lceil q_{n/2} \rceil$ присваивается значение q_1, остальное аналогично, $C_{\text{«обр.П»}}$.

5. Порядок, который схож с графиком функции параболы, аналогично 4, только изначально происходит упорядочивание весов по возрастанию и веса с меньшим весом находятся ближе к «вершине параболы», $C_{\text{«пр.П»}}$.

Результаты построений деревьев сведены в таблицу.

$C_{\text{Пробел,А,…,Я}}$	$C_{\text{обр.Алф.}}$	$C_{\text{«обр.П»}}$	$C_{\text{убыв.}}$	$C_{\text{возр.}}$	$C_{\text{«пр.П»}}$
4,464	4,509	4,510	4,447	5,390	4,390

Из таблицы видно, что четыре значения из шести находятся очень близко к друг другу, максимальное отклонение от среднего арифметического таких значений составляет ≈0,036 и два явно выделяющихся значения минимальное и максимальное. Работа алгоритма базируется на анализе весов, при этом задан первоначальный порядок этих весов. Реализация первых трех деревьев, соответствующих первым трем стоимостям таблицы, можно рассматривать как случайный порядок распределения вероятности, даже в случае с «обратной параболой» веса располагаются в менее строгом порядке чем в случае с убыванием.

Случай с «убыванием» и «возрастанием» можно было бы считать двумя крайними случаями если бы не было примера с «прямой параболой». Дерево, полученное исходя из начального возрастания весов имеет самую большую стоимость только потому, что в алгоритме предусматривается сдвиг вершины, полученной путем слияния влево, а это в свою очередь приводит к том, что на более низких уровнях оказываются вершины с относительно высокими весами.

Дерево, полученное исходя первоначального порядка организации вершин в виде «параболы» имеет самую меньшую стоимость в следствие того, что первая операция слияния проводится в вершине, которая находится в центре множества Q, и процесс сдвига происходит влево и на более низких уровнях оказываются вершины с более низкими весами.

Путь к листьям дерева определяется последовательностью прохождения по уровням, в бинарном дереве с вышележащим уровнем связано только два нижележащих, поэтому, поставив строгое соответствие, что продвижение по левой ветви дерева есть 0, а по правой ветви дерева есть 1 то маршрут продвижения от корня дерева к его листьям образует последовательность нулей и единиц. Можно говорить, что листья имеют уникальную последовательность, а длина данной последовательности равна тому уровню, на каком находится лист, т.е. получается двоичное кодирование листа дерева, при котором коды листьев могут отличаться по длине, что соответствует неравномерному кодированию.

Полученные коды так же отвечают условию Фано, т.е. один код не является префиксной комбинацией любого другого кода или не является суффиксной комбинацией любого другого кода.

Nom-11-2024.qxd 05.12.2024 16:58 Page 43

Для полученного кода возможно определить среднее число двоичных символов, которые приходятся на один кодируемый символ.

$$k_{cpl} = \sum_{i=1}^{n} l_{кода_i} \cdot p_i$$

где $l_{кода_i}$ – длина i-го кода; p_i – вероятность его появления.

В рассматриваемом случае длина кода соответствует уровню, на котором находится лист, а вероятность появления – частоте встречаемости, тогда данная формула есть не что иное, как расчет цены дерева.

Зная среднее число символов, приходящихся на один символ возможно определить количество информации, которое приходится на двоичный символ $I_{двс} = \frac{H(N)}{k_{cpl}}$, где H(N) – энтропия алфавита. Из формулы Шеннона-Фано определим энтропию исходного алфавита и рассчитаем количество информации, которое приходится на один двоичный символ для вариантов, которые принимали участие в эксперименте.

$$H(N) = \sum_{i=1}^{n} p_i \cdot \log p_i \approx 4{,}351$$

$I_{Пробел,А,…,Я}$	$I_{обр.Алф.}$	$I_{«обр.П»}$	$I_{убыв.}$	$I_{возр.}$	$I_{«пр.П»}$
0,975	0,965	0,965	0,978	0,807	0,991

В том случае если провести кодирование исходных данных с помощью алгоритма Шеннона-Фано, то среднее количество информации на один двоичный символ составляет 0,974. Из таблицы видно, что при организации исходных данных в виде «обратной параболы» количество информации, приходящейся на один двоичный символ максимально, а, следовательно, и код, построенный исходя из такой последовательности, наиболее информативен.

Заключение

Проведение эксперимента показало, что прежде чем проводить построение бинарного дерева минимальной цены, необходимо провести предобработку данных, которые будут в этом принимать участие, провести их предварительную сортировку, а также привести их к виду «обратной параболы». Тогда сложность алгоритма увеличивается на сложность применяемого метода сортировки, которая в свою очередь зависит от выбранного метода сортировки.

Литература

1. *Павлов С.В., Докучаев В.А., Маклачкова В.В., Гадасин Д.Д.* Автоматизация поддержки принятия решений при подборе специалистов по управлению сложными инфокоммуникационными системами // T-Comm: Телекоммуникации и транспорт. 2023. Т. 17, № 12. С. 36-43. DOI 10.36724/2072-8735-2023-17-12-36-43. EDN DWHCMB.

2. *Статьев В.Ю., Докучаев В.А., Маклачкова В.В.* Информационная безопасность на пространстве "Больших данных" // T-Comm: Телекоммуникации и транспорт. 2022. Т. 16, № 4. С. 21-28. DOI 10.36724/2072-8735-2022-16-4-21-28. EDN IXUYWS.

3. *Pavlov S.V., Dokuchaev V.A., Maklachkova V.V., Mytenkov S.S.* Features of supporting decision making in modern enterprise infocommunication systems // T-Comm: Телекоммуникации и транспорт. 2019. Т. 13. . С. 71-74.

4. *Буренин А.Н., Легков К.Е.* Вопросы безопасности инфокоммуникационных систем и сетей специального назначения: основные угрозы, способы и средства обеспечения комплексной безопасности сетей // Наукоемкие технологии в космических исследованиях Земли. 2015. Т. 7. № 3. С. 46-61.

5. *Dokuchaev V.A., Maklachkova V.V., Statev V.Yu.* Classification of personal data security threats in information systems // T-Comm: Телекоммуникации и транспорт. 2020. Т. 14. № 1. С. 56-60.

6. *Кристофидес Н.* Теория графов: алгоритмический подход. 2-е изд., испр. М.: Мир, 1978. 430 с.

7. *Харари Ф.* Теория графов. 4-е изд. М.: Едиториал УРСС, 2009. 296 с.

8. *Миниэка Э.* Алгоритмы оптимизации на сетях и графах. М.: Мир, 1981.

9. *Гадасин Д.В., Шведов А.В.* Применение транспортной задачи для балансировки нагрузки в условиях нечеткости исходных данных // T-Comm: Телекоммуникации и транспорт. 2024. Т. 18, № 1. С. 13-20. DOI 10.36724/2072-8735-2024-18-1-13-20. EDN WKNPIX.

10. *Гадасин Д.В., Тремасова Л.А., Гадасин Д.Д.* Распределение поступающей нагрузки с применением SS-метода // I-Methods. 2023. Т. 15, № 3. EDN HQEYTW.

11. *Гадасин Д.В., Андриянова А.К., Тремасова Л.А.* Определение нечеткости поискового запроса через множество аксиом объекта // Технологии информационного общества : Сборник трудов XVII Международной отраслевой научно-технической конференции, Москва, 02-03 марта 2023 года. М.: Издательский дом Медиа Паблишер, 2023. С. 132-134. EDN DWJUTE.

12. *Гадасин Д.В., Комкова М.Г., Пантелеева К.А., Гадасин Д.Д.* Связь между величиной энтропии и количеством информации при представлении предметной области разными лингвистическими единицами // DSPA: Вопросы применения цифровой обработки сигналов. 2023. Т. 13, № 2. С. 12-21. EDN WRHCSW.

13 *Буслаев А.П., Кучелев Д.А., Яшина М.В.* Динамические системы и математические модели трафика информации // T-Comm: Телекоммуникации и транспорт. 2018. Т. 12. № 3. С. 22-38.

14 *Бугаев А.С., Таташев А.Г., Яшина М.В., Лавров О.С., Носов Е.А.* Восстановление динамики транспортного потока на основе детерминированно-стохастической модели и данных с интеллектуально транспортных систем // T-Comm: Телекоммуникации и транспорт. 2019. Т. 13. № 10. С. 35-44.

15 *Козлов С.В., Кубанков А.Н.* Процессные основы интеграции и комплексного развития информационных, управляющих, роботизированных, телекоммуникационных систем // Наукоемкие технологии в космических исследованиях Земли. 2020. Т. 12. № 1. С. 23-31.

16. *Гадасин Д.В., Смальков Н.А., Кузин И.А.* Использование метода роя частиц для балансировки нагрузки в сетях Интернета вещей // Системы синхронизации, формирования и обработки сигналов. 2022. Т. 13, № 2. С. 17-23. EDN LIUWNT.

17. *Белов В.В., Чистякова В.И.* Алгоритмы и структуры данных: учебник. М.: КУРС: ИНФРА-М, 2023. 240 с. ISBN 978-5-906818-25-6.

18. *Dokuchaev V.A., Petukhov D.* Cloud Data Formats Transformation Algorithms // Systems of Signals Generating and Processing in the Field of on Board Communications. 2023. Vol. 6, No. 1, pp. 92-96. DOI 10.1109/IEEECONF56737.2023.10092056. EDN AXTQBW.

19. *Гадасин Д.В., Вакурин И.С., Тремасова Л.А.* Алгоритм распределения данных между системами хранения на основе свойства самоподобия // Электросвязь. 2024. № 4. С. 44-50. DOI 10.34832/ELSV.2024.53.4.007. EDN BRSLCL.

20. *Стенли Р.* Перечислительная комбинаторика. Деревья, производящие функции и симметрические функции. М.: Мир, 2005.

21. *Исаак Яглом, Акива Яглом.* Вероятность и информация. М.: Наука, 1973. 512 с.

Nom-11-2024.qxd 05.12.2024 16:58 Page 44

BUILDING A BINARY TREE OF THE MINIMUM PRICE

Denis V. Gadasin, *Moscow Technical University of Communications and Informatics (MTUCI), Moscow, Russia,*
dengadiplom@mail.ru

Abstract

One of the methods used in the formalization of the subject area is to gradually defragment it according to the principle from larger to smaller and establish the connection of the obtained parts with numerical values. An attribute is taken as a minimal, indivisible part of the subject area. The attribute can take valid values. The reaction time to events occurring in the domain depends on the time to search for the attribute state, which depends on the type of structure in which the attributes are combined. The paper considers the binary tree-like structure of the organization of attributes. The search time in such a structure is matched by the value of the tree. The cost of a tree is defined as the sum of the products of the weights of the vertices and the level at which this vertex is located. Thus, it is necessary to arrange the vertices by levels in such a way that the resulting cost would be as small as possible, while the tree would remain binary. To build a tree, the Garcia-Voch algorithm is considered in the work. This algorithm allows you to build a minimum cost tree for a hard-coded sequence of attributes with appropriate weights. A method of reducing algorithmic complexity is considered. To conduct the experiment, the attributes that make up the alphabet of the Russian language plus the "space" symbol are considered. The average frequency of occurrence of the symbol in the language is taken as the weight of the attribute. The experiment is conducted for four ways of initial ordering of characters based on their weight: alphabetical order, reverse alphabetical order, descending, ascending, ascending to maximum and descending, descending to maximum and ascending. The constructed tree allows you to get an uneven binary code for each character of the alphabet and calculate the number of binary characters per one character of the alphabet. This value fully corresponds to the cost of the tree. Based on this, the most preferred attribute configuration for building a binary tree of minimum cost is determined.

Keywords: *search tree, Garcia-Wacha algorithm, algorithm complexity, optimality, load distribution between nodes.*

References

[1] S. V. Pavlov, V. A. Dokuchaev, V. V. Maklachkova, D. D. Gadasin, “Automation of decision support in the selection of specialists in the management of complex information and communication systems,” *T-Comm*. 2023. Vol. 17, No. 12, pp. 36-43. DOI 10.36724/2072-8735-2023-17-12-36-43.
[2] V. Yu. Statiev, V. A. Dokuchaev, V. V. Maklachkova, “Information security in the space of "Big Data",” *T-Comm*. 2022. Vol. 16. No. 4. pp. 21-28. DOI 10.36724/2072-8735-2022-16-4-21-28
[3] S. V. Pavlov, V. A. Dokuchaev, V. V. Maklachkova, S. S. Mytenkov, “Features of supporting decision making in modern enterprise infocommunication systems,” *T-Comm*. 2019. Vol. 13. No. 3, pp. 71-74.
[4] A. N. Burenin, K. E. Legkov, “Security issues of infocommunication systems and special-purpose networks: main threats, methods and means of ensuring comprehensive network security,” *H&ES Research*. 2015. Vol. 7. No. 3, pp. 46-61.
[5] V. A. Dokuchaev, V. V. Maklachkova, V. Yu. Statev, “Classification of personal data security threats in information systems,” *T-Comm*. 2020. Vol. 14. No. 1, pp. 56-60.
[6] N. Christophides, “Graph theory: an algorithmic approach,” 2nd ed., ispr. Moscow: Mir, 1978. 430 p.
[7] F. Harari, “Graph Theory,” 4th ed. Moscow: Unified URSS, 2009. 296 p.
[8] E. Minieka, “Optimization algorithms on networks and graphs,” Moscow: Mir, 1981.
[9] D. V. Gadasin, A.V. Shvedov, “Application of the transport task for load balancing in conditions of source data fuzziness,” *T-Comm*. 2024. Vol. 18, No. 1, pp. 13-20. DOI 10.36724/2072-8735-2024-18-1-13-20.
[10] D. V. Gadasin, L. A. Tremasova, D. D. Gadasin, “Distribution of incoming load using the SS method,” *I-Methods*. 2023. Vol. 15, No. 3.
[11] D. V. Gadasin, A. K. Andrianova, L. A. Tremasova, “Definition of the fuzziness of a search query through a set of axioms of an object,” *Information Society Technologies : Proceedings of the XVII International Industrial Scientific and Technical Conference*, Moscow, 02-03 March 2023. Moscow: Media Publisher, 2023, pp. 132-134.
[12] D. V. Gadasin, M. G. Komkova, K. A. Panteleeva, D. D. Gadasin, “The relationship between the amount of entropy and the amount of information when representing a subject area by different linguistic units,” *DSPA: Issues of the application of digital signal processing*. 2023. Vol. 13, No. 2, pp. 12-21.
[13] A. P. Buslaev, D. A. Kuchelev, M. V. Yashina, “Dynamic systems and mathematical models of information traffic,” *T-Com*m. 2018. Vol. 12. No. 3, pp. 22-38.
[14] A. S. Bugaev, A. G. Tatashev, M. V. Yashina, O. S. Lavrov, E. A. Nosov, “Reconstruction of the dynamics of traffic flow based on a deterministic-stochastic model and data from intelligent transport systems,” *T-Comm*. 2019. Vol. 13. No. 10, pp. 35-44.
[15] S. V. Kozlov, A. N. Kubankov, “Process foundations of integration and comprehensive development of information, control, robotic, telecommunication systems,” *H&ES Research*.. 2020. Vol. 12. No. 1, pp. 23-31.
[16] D. V. Gadasin, N. A. Smalkov, I. A. Kuzin, “Using the particle swarm method for load balancing in Internet of Things networks,” *Systems of signal synchronization, generating and processing*. 2022. Vol. 13, No. 2, pp. 17-23.
[17.] V. V. Belov, V. I. Chistyakova, “Algorithms and data structures,” Moscow : COURSE : INFRA-M, 2023. 240 p. ISBN 978-5-906818-25-6.
[18] V. A. Dokuchaev, D. Petukhov, “Cloud Data Formats Transformation Algorithms,” *Systems of Signals Generating and Processing in the Field of on Board Communications*. 2023. Vol. 6, No. 1, pp. 92-96. DOI 10.1109/IEEECONF56737.2023.10092056.
[19] D. V. Gadasin, I. S. Vakurin, L. A. Tremasova, “Algorithm of data distribution between storage systems based on the property of self-similarity,” *Telecommunications*. 2024. No. 4, pp. 44-50. DOI 10.34832/ELSV.2024.53.4.007
[20] R. Stanley, “Enumerative combinatorics. Trees producing functions and symmetric functions,” Moscow: Mir, 2005.
[21] Isaac Yaglom, Akiva Yaglom, “Probability and information,” Moscow: Nauka, 1973. 512 p.