УДК 004.074.32

МНАЦАКАНОВ А.В.,
КУЗНЕЦОВ А.А.

К ПРОБЛЕМЕ ХЕШ-АДРЕСАЦИИ БЕЗ КОЛЛИЗИЙ ПОСТОЯННОГО МАССИВА КЛЮЧЕЙ

В статье исследуется проблема получения хеш-адресации без коллизий заданного постоянного массива ключей. Разработан алгоритм построения совершенной хеш-адресации в виде иерархической системы разделяющих булевых функций, позволяющий значительно сократить объем вычислительных ресурсов, необходимых для формирования хеш-преобразования, не порождающего коллизий при заданных ограничениях на сложность хеш-функции. Проведенный анализ показал преимущества предложенного алгоритма получения совершенной хеш-адресации, по сравнению с известными алгоритмами решения этой проблемы.

In article the problem of designing of the perfect hash-addressing for a constant set of keys is investigated. The algorithm of designing of the perfect hash-addressing as hierarchical system of Boolean functions is developed. The algorithm allows to significant reduce of volume of computing resources which is necessary for design of the perfect hash-transformation with the prescribed limit of hash-function complexity. The analysis has shown the advantages of proposed perfect hash-addressing designing algorithm in comparison to known algorithms of this problem decision.

Введение

Операции поиска по ключу и сортировки данных являются одними из базовых в информационных технологиях. Существует значительное число важных практических применений, в которых удельный вес этих операций составляет до 80% [2].

Прогрессирующие возможности вычислительных систем и информационная интеграция обуславливают динамичный рост объемов массивов ключей, на которых выполняется поиск [2]. Непрерывное расширение сферы использования информационных технологий имеет следствием ужесточение требований к оперативности поиска. В частности, большая часть систем распознавания зрительных и звуковых образов, в которых активно используется поиск по ключу, работают в режиме реального времени. В близком режиме работает значительная часть современных баз данных. Расширение информационной интеграции привело к росту числа информационных систем коллективного доступа, специфика которых предъявляет повышенные требования к скорости поиска.

Анализ показывает, что в современных условиях роста объемов поисковых массивов и ужесточения требований к оперативности доступа, эффективность использования двоичного поиска и B-деревьев сущест-

венно снижается в силу зависимости времени поиска от объема массива ключей.

Это требует расширения использования альтернативных технологий поиска по ключу, таких как хеш-адресация и генетические алгоритмы [5].

Самым быстрым методом поиска по ключу является ассоциативный поиск, который может быть реализован как аппаратно (ассоциативные запоминающие устройства - АЗУ), так и программно (хеш-память). Важным достоинством ассоциативного поиска является то, что время поиска не зависит от объема массива ключей. Ввиду значительной стоимости выпускаемых микросхем АЗУ их практическое использование ограничено. Поэтому в современных условиях для большинства применений более эффективным представляется использование хеш-адресации.

Существует класс практических применений поиска по ключу, для которых массив ключей является постоянным или квазипостоянным (интенсивность операций поиска по ключу на несколько порядков превосходит интенсивность операций изменения поискового массива). К их числу относится большинство систем распознавания образов, электронного перевода и аутентификации удаленных пользователей, заметная часть баз данных.

При хеш-поиске в постоянных массивах ключей возможно получение однозначного хеш-преобразования, исключающего возникновение коллизий. В литературе такой метод хеш-поиска получил название совершенной хеш-адресации (perfect hash-addressing)[1].

Основным достоинством совершенной хеш-адресации является отсутствие коллизий, то есть, время поиска по ключу определяется временем однократного обращения к памяти. Это позволяет достичь предельно-высокой скорости поиска, независимо от объема поискового массива [1].

Задача получения хеш-преобразования, не порождающего коллизий для заданного массива ключей допускает множество решений, однако, при имеющих место на практике ограничениях на объем вычислительных ресурсов, используемых при совершенной хеш-адресации, эта задача имеет экспоненциальную сложность [1].

В связи с этим, актуальной является задача построения алгоритма, позволяющего значительно сократить объем вычислительных ресурсов, необходимых для формирования хеш-преобразования, не порождающего коллизий и удовлетворяющего заданным ограничениям на сложность функциональных преобразований.

Основные определения и постановка задачи

Хеш-преобразованием, не порождающем коллизий для заданного постоянного набора ключей, называется однозначная функциональная зависимость $F(\Omega) = \Theta$ (где – Ω множество состоящее из m n-разрядных клю-

чей и Θ – множество r-разрядных адресов хеш-памяти), причем $F(k_i) \neq F(k_j)$ при любых $k_{i,}k_j \notin \Omega$ и $i \neq j$ [2].

При имеющих место на практике ограничениях на сложность функциональных преобразований и объем используемой памяти формирование хеш-преобразования без коллизий (ХБК) требует значительных затрат вычислительных ресурсов. В связи с этим, основным критерием эффективности ХБК является временная T_c сложность его получения при заданном постоянном или наборе Ω ключей.

Основным недостатком хеш-адресации является избыточность используемой памяти, что значительно ограничивает сферу использования хеш-памяти. Избыточность использования памяти характеризуется коэффициентом α загрузки, который определяется отношением количества m заданных ключей к числу адресов хеш-памяти:

$$\alpha = \frac{m}{2^r} \qquad (1)$$

К настоящему времени предложено ряд методов, позволяющих, для заданного набора ключей, получить хеш-преобразование не порождающее коллизий [4, 5, 6]. Большинство из них основаны на подборе хеш-преобразования и ориентированы на программную реализацию хеш-адресации.

Недостатками этих методов являются высокая вычислительная сложность и возможность возникновения тупиковых ситуаций. Большая вычислительная сложность существующих методов ограничивает область их использования сравнительно небольшими массивами ключей [6]. Поэтому актуальной является проблема разработки подходов, требующих меньших затрат вычислительных ресурсов.

Решение этой проблемы может быть найдено в рамках использования многоуровневого направленного формирования хеш-преобразования без коллизий (ХБК).

Методика построения хеш-преобразования без коллизий формированием системы разделяющих булевых функций

Решение задачи формирования ХБК, путем сокращения разрядности ключей, можно разделить на два этапа. При этом, первом этапе производится сокращение разрядности ключей за счет удаления малоинформативных разрядов ключей, которые не влияют на их различение, а на втором, производится выделение групп разрядов по которым осуществляется перекодирование с использованием табличных преобразователей.

Предположим, что вероятность появления нуля и единицы в коде ключа одинакова, тогда вероятность P_p того, что будет выполняться условие $k_i \neq k_j, i \neq j$ при любом $k_i, k_j \in \Omega$, будет равна:

$$P_p = \frac{1}{\exp(m)} \bullet \left(\frac{2^n}{2^n - m} \right)^{2^n - m} \bullet \sqrt{\frac{2^n}{2^n - m}} \quad (2)$$

Из выражения (2) можно определить минимальную разрядность n', которая может быть достигнута на первом этапе формирования ХБК, при заданном уровне вероятности P_p различимости ключей.

Предлагаемый алгоритм сокращения разрядности ключей, путем исключения малозначащих разрядов включает в себя две стадии:

На первой стадии производится ранжирования разрядов по невозрастанию их значимости для различимости ключей. При этом используется интегральная оценка $W(j)$ значимости j-го разряда ключа, которая вычисляется по формуле:

$$W(j) = h(j) + \frac{1}{m} \cdot \sum_{l=j+1}^{m} K(j,l) \quad (3)$$

где $h(j)$ – частота равенства единицы j-го разряда ключа и $K(j,l)$ – коэффициент корреляции j-го и l-го разрядов ключевого слова, которые в свою очередь могут выть определены по следующим формулам [6]:

$$h(j) = \frac{1}{m} \cdot \sum_{i=1}^{m} x_{ij},\ j \in \{1,...,n\} \quad (4)$$

$$K(j,l) = \frac{1}{m} \cdot \sum_{i=1}^{m} (1 \oplus x_{ij} \oplus x_{lj}),\ j,l \in \{1,...,n\},\ j \neq l \quad (5)$$

Вторая стадия состоит в непосредственном исключении малозначащих разрядов ключей:

1. Текущим разрядом t принимается разряд, имеющий минимальное значение интегральной оценки значимости: $t = n$.
2. Разряд t исключается из всех кодов ключей и выполняется проверка условия различимости ключей. Если условие выполняется и $t > 1$, то при $t = t - 1$ осуществляется переход на п. 5. В случае невыполнения условия различимости разрядов осуществляется переход на п. 3.
3. Из кодов всех ключей исключается разряд d, который имеет максимальную корреляцию с разрядом t: $d = \arg\max K(t, j)$.
4. Если выполняется условие различимости ключей, то разряд d окончательно исключается из отсортированного списка разрядов ключа и при $t = t - 2$ осуществляется переход на п. 5. В случае невыполнения условия различимости ключей, выполняется переход к п. 5, при $t = t - 1$.
5. Если t ≥ 1 то осуществляется возврат на п. 2, иначе конец.

После выполнения первого этапа формирования ХБК будет получено упорядоченное по $W(j)$ множество Ξ номеров разрядов ключей, которые не могут быть исключены без нарушения условия различимости.

На втором этапе формирования ХБК, осуществляется построение систем разделяющих булевых функций, которые выполняют перекодирование множества Ω' сокращенных ключей в множество Θ, содержащее ключи (хеш-адреса) меньшей разрядности. В большинстве случаев системы разделяющих булевых функций реализуется таблично, так, что ограничением на число разрядов, составляющих множество Θ является объем памяти – V, выделяемый под табличный преобразователь.

Исходя из полученного на первом этапе формирования ХБК множества Ξ состоящего из q разрядов (численное значение q может быть определено из (1)), формирование системы разделяющих булевых функций состоит в следующем:

1. Из множества Ξ выбирается разряд d такой, что h(d) наиболее близка к 0.5. Разряд d, в свою очередь, делит множество Ω ключей на два подмножества ϑ_0 (множество, состоящее из ключей в которых разряд d равен 0) и ϑ_1 (множество, состоящее из ключей в которых разряд d равен 1). Множество, содержащее наибольшее число ключей выбирается текущим G: $G = \max(\vartheta_0, \vartheta_1)$. Далее разряд d исключается из множества Ξ и добавляется к множеству Φ. При этом счетчик t подмножеств устанавливается равным 2-м.
2. На множестве Ξ выбирается разряд d, который делит G на два подмножества, с минимальной разностью количества ключей.
3. Проверяется условие:

$$\forall \vartheta_j,\ j = 1,...,2t,\ \vartheta_j \not\subset G : \left|\vartheta_j\right| \le \frac{n}{2 \cdot (t+1)}, \qquad \textbf{(6)}$$

4. где $\vartheta_1, \vartheta_2, ..., \vartheta_{2t} \subset G$ – подмножества на которые множество Ω ключей делится разрядами множества Φ и разрядом d. Если условие (6) выполняются, то разряд d исключается из множества Ξ и добавляется к множеству Φ. При этом $t = t \cdot 2$ и в качестве G выбирается одно из подмножеств $\vartheta_0, \vartheta_1, ..., \vartheta_t$ содержащее максимальное число элементов.
5. Если $t < V$, то осуществляется переход на п.2, иначе находится такое наибольшее число g, для которого количество ключей в каждом из подмножеств $\vartheta_0, \vartheta_1, ..., \vartheta_t$ меньше 2^g.
6. Осуществляется построение $\log_2 t$ -разрядной таблицы, состоящей из t строк, выбор которых осуществляется в соответствии со значениями разрядов множества Φ. Каждая строка полученной таблицы содержит g-разрядные коды замещения $\log_2 t$ разрядов ключа, являющиеся фактически старшими разрядами хеш-адреса. Таким образом, исполь-

зование этой таблицы позволяет осуществить перекодирование $\log_2 t$ разрядов ключа в g разрядов. Кроме того, каждая из строк таблицы содержит данные для проведения дальнейшего сжатия каждого из подмножеств $\vartheta_0, \vartheta_1, ..., \vartheta_t$.

Приведенную методику можно повторно применять к каждому из подмножеств $\vartheta_0, \vartheta_1, ..., \vartheta_t$ в отдельности, пока в каждом из разделяемых подмножеств не останется один ключ.

Для ускорения нахождения табличного представления разделяющих функций первые две операции предложенной методики можно модифицировать следующим образом:

1. Из множества Ξ выбирается любой q-тый разряд, при этом механизм выбора множества G не изменяется.
2. В качестве текущего d, выбирается разряд $d \in \Xi, d \neq q$ с наименьшим коэффициентом корреляции, то есть $K(d, q)$.

В остальном модифицированная методика построения разделяющих функций полностью повторяет вышеописанную.

Полученный набор таблиц перекодирования представляет собой иерархическое хеш-преобразование.

Построенная, в результате использования предложенного алгоритма, система разделяющих булевых функций может быть реализована средствами FPGA, что позволяющего значительно повысить быстродействие хеш-преобразователя в сравнении с программной реализацией.

Оценка основных характеристик и сравнительный анализ

Операция исключения разрядов требует многократного формирования подмножеств $\vartheta_0, \vartheta_1, ..., \vartheta_t$. Если максимально допустимая разрядность таблицы перекодирования равна u, то общее число N_o операций сравнения, требующихся для формирования системы разделяющих функций [6]:

$$N_o = \sum_{j=u}^{n} j^2 \qquad (7)$$

Таблица перекодирования будет выполнять разбиение множества Ω на 2^u подмножеств, каждое из которых содержит $n/2^u$ ключей. В связи с этим, основной является задача выбора множества Φ, включающего u из n разрядов и формирования на выбранном множестве булевой функции $\varphi(x_1,\ldots,x_u)$ от u переменных. В [3] показано, что существует C_n^u способов выбора подмножества Φ.

В общем случае, число функций от u переменных равно 2^{2^u}, однако ввиду того, что эти функции должны разделять множество m ключей на

2^u равновеликих подмножества, то общее число функций, удовлетворяющих этому условию составит $2^u!$. Таким образом, на первом уровне иерархии ХБК, верхняя граница общего числа вариантов выбора разделяющих функции составит $C_n^u \cdot 2^u!$.

Технологически формирование системы разделяющих функций в общем случае состоит в выборе множество из u разрядов такого, что число неразличимых ключей каждого вида было бы меньше $n/2^u$.

В данном случае важной является задача выбора разрядности функций u. С одной стороны, численное значение u определяется ограничением на объемом памяти, с другой – достаточно, с большой вероятностью гарантировать, что приемлемое множество может быть найдено.

Поэтому одним из основных критериев эффективности предложенной методики построения ХБК является оценка вероятности P_e того, что на множестве из u разрядов ключа может быть построена система булевых разделяющих функций. Вероятности P_c того, что совпадут h ключей из m, при использовании u – разрядного подмножества [3]:

$$P_c = C_n^h \cdot (p_e{}^2 + (1-p_e)^2)^{h \cdot u} (1 - p_e{}^2 - (1-p_t)^2)^{u \cdot (n-h)} \quad (8)$$

где p_e – вероятность того, что разряд ключа равен единице (на практике вместо p_e целесообразным представляется использование среднего значения частоты единиц в массиве ключей).

Тогда, среднее число ξ совпавших ключей на множестве из выделенных разрядов u составит:

$$\xi = n \cdot (p_e^2 + (1-p_e)^2)^u \quad (9)$$

Для достаточно быстрого поиска подходящего множества из u разрядов ключа, необходимо и достаточно чтобы выполнялось условие $\xi << m/2^u$. Согласно [5] это условие трансформируется к виду:

$$u > \frac{\ln(n)}{\ln\left(\frac{2}{p_e{}^2 + (1-p_e)^2}\right)}. \quad (10)$$

Например, при $p_e = 0.6$, для того, чтобы выполнялось условие разделимости массива разрядности $n = 1000$ по u разрядам на 2^u подмножества необходимо, чтобы $u > 5.1$.

Согласно [5], среднее число T_e проб, необходимых для разделения Ω на 2^u подмножеств так, чтобы не существовало более $n/2^u$ одинаковых ключей составляет:

$$T_e = \frac{1}{\Phi\left(\dfrac{n/2^u - n\cdot(p_e^2 + (1-p_e)^2)^u}{\sqrt{n\cdot(p_e^2 + (1-p_e)^2)^u \cdot (1-(p_e^2 + (1-p_e)^2)^u)}}\right)} \quad \textbf{(11)}$$

где Ф() – функция Лапласа. Анализ (11) показывает, что для практически значимых значений n и p_e, значение u всегда меньше границы, определяемой техническими возможностями.

Пусть $\{u_1, u_1, ..., u_k\}$ – разрядность таблиц перекодирования на каждом из k уровней иерархии хеш-преобразования тогда, при максимальном значении коэффициента α загрузки памяти, должно выполняться условие [5]: $\sum_{i=1}^{k} u_i = \log_2 n$. Учитывая, что общий объем памяти V необходимый для построения табличного преобразователя равен $V = \sum_{i=1}^{k}\prod_{j=1}^{i} 2^{u_j}$, получим, что, при использовании предложенной методики, максимально достижимое значение коэффициента α загрузки составит:

$$\alpha = \frac{n\cdot m}{V\cdot \log_2 n} \quad \textbf{(12)}$$

Проведенный анализ характеристик предложенной методики формирования ХБК показал, что при большом числе ключей m ($m > 300$), скорость построения ХБК существенно превосходит известные алгоритмы Чанга и Сичели [3], основанные на случайном формировании таблиц перекодирования. При увеличении числа m ключей преимущество предложенной методики в скорости становится более ощутимой. Вместе с тем, значительное повышение скорости достигается за счет увеличенного расхода памяти на хранение таблиц перекодирования.

В отличии от известных методов, которые не гарантируют нахождения ХБК для любого заданного массива ключей, предложенный алгоритм позволяет получить решение для любого заданного массива ключей за конечное число шагов.

Выводы

В результате проведенных исследований, направленных на повышение эффективности хеш-адресации без коллизий постоянных массивов ключей был предложен алгоритм формирования многоуровневого хеш-преобразования в виде иерархической системы разделяющих булевых функций. Алгоритм предполагает формирования хеш-преобразования в виде системы булевых функций и позволяет существенно сократить (на 50%) время их получения по сравнению с известными подходами [4,6].

В отличие от известных методов формирования хеш-преобразования без коллизий предложенный алгоритм позволяет получить решение для любого заданного массива ключей за конечное число шагов.

Использование предложенного алгоритма позволяет сократить сроки разработки специализированных систем обработки информации с большим удельным весом операций поиска в постоянных массивах ключей или время перенастройки для хеш-поиска в квазипостоянных массивах ключей.

*Ключевые слова***:** perfect hash-addressing, perfect hash-memory, database management systems, Boolean functions

Список использованной литературы

1. Кохонен Т. Ассоциативная память. М. :Мир, 1980. – 198 с.
2. Марковский А.П., Гаваагийн Улзисайхан, Бардис Николас, Об одном подходе к повышению эффективности и уровня защищенности систем хранения информации на основе хеш-памяти // Вісник Національного технічного університету України „КПИ”. Інформатика, управління та обчислювальна техніка. К., ТОО „ВЕК+” 1998, - №31. С.94-108.
3. Berman F., Bock M.E., Dittert E., O`Donnel M.J., Plank D. Collections of function of perfect hashing // SIAM Journal Computers. – 1986, - Vol. 15, №2, - P. 604-618.
4. Jagannathan R. Optimal partial-match hashing design // ORSA Journal of Computing. – 1991, - Vol.3, №2, - P.86-91.
5. Ningping Sun, Ryozo Nakamura, Nonbing Zhu, Akiro Tada, Wenling Sun. An analysis of average search cost of external hashing with separate chain.// Processing of 7-th WSEAS International Conference on Circuits, Systems, Communications and Computers (CSCC-2003).- 2003,-P. 315-324.
6. Ramakrishna M.V., Bannai Y. Direct perfect hashing function of external files. // Journal of Database Administration. – 1991, - Vol.2, №1, - P.19-28.