Д.А. ЕЛИУСИЗОВ

ПОСТРОЕНИЕ МИНИМАЛЬНОЙ СОВЕРШЕННОЙ ХЕШ-ФУНКЦИИ

(Представлена академиком НАН РК Н.К. Блиевым)

Описывается построение минимальной совершенной хеш-функции, используя методы универсального хеширования. Также рассматриваются различные методы построения совершенных хеш-функций.

1 Введение

При решении задачи информационного поиска широко используются хеш-функции [1]. Смысл хеширования заключается в том, что для нахождения информации по ключу вычисляется некоторая функция, значение которой указывает на адрес в таблице, где и хранится искомая информация.

В данной статье описываются новые подходы к построению совершенных хеш-функций для большого множества ключей, используя методы универсального хеширования.

2 Обозначения и определения

В последующем будут использованы следующие обозначения и определения:

U – множество всех возможных ключей. S – текущее статичное множество рассматриваемых ключей $S \subset U, |S| = n << |U|$. $M = \{0,1,...,m-1\}$ – множество компьютерных ячеек для хранения информации.

Функцией хеширования (хеш-функцией) называется отображение $h: U \to M$. Основной проблемой при использовании хеш-функций является возможное появление *коллизий*, т.е. та-

ких значений $x \neq y$, при которых $h(x) = h(y)$. Для решения этой проблемы существует несколько подходов, к примеру, построение с помощью цепочек [2]. Хеш-функция, которая не имеет коллизий (т.е. является инъективной) называется *совершенной*. Если $m = n$, то функция называется *минимальной совершенной хеш-функцией* (МСХФ).

$P(A)$ – вероятность события A. $M(X)$ – математическое ожидание дискретной случайной величины X.

3 Универсальное хеширование

При универсальном хешировании выбор хеш-функции во время исполнения программы осуществляется случайным образом из некоторого множества, независимо от входных данных.

Пусть H – конечное семейство функций $h: U \to M$. Это семейство называется *универсальным*, если для любых двух ключей $x, y \in U$ число функций $h \in H$, для которых $h(x) = h(y)$, равно $|H|/m$. Из этого следует, что при случайном выборе хеш-функции вероятность коллизии между двумя данными ключами равна $1/m$.

4 Построение МСХФ

Начнем с построения минимальной совершенной хеш-функции, предложенной Яешке в [4].

Пусть f – функция такая, что все числа $f(k_1), f(k_2), ..., f(k_n)$ являются попарно взаимно простыми.

Теорема. Если множество $K = \{k_1, k_2, ..., k_n\}$ содержит различные натуральные ключи, то существует целое число A такое, что функция $h(x) \equiv \left[\frac{A}{f(x)}\right] (\mathrm{mod}\ n)$ $x \in K$ является минимальной совершенной функцией хеширования.

Подробное доказательство есть в [3].

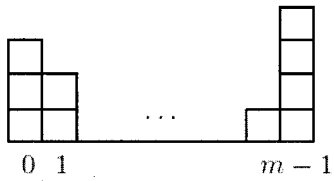
Интересным также является и то, как подобрать функцию f. Один из способов – взять в качестве f линейную функцию $ax + b$. Но при подобном подходе используемые числа могут быть достаточно большими. И основной проблемой при использовании такой хеш-функции является сравнительно небольшие размерность и количество входных данных. [8]

5 Смешанное построение

Основной результат статьи излагается в этом разделе.

Сегментация

Если использовать произвольную функцию хеширования для ключей множества S (при $m = n$), скорее всего, возникнут коллизии. Предположим тогда, что все n ключей "попали" в n ячеек памяти, причем в некоторые ячейки попало несколько ключей. На рисунке это выглядит следующим образом:



Таким образом, каждая ячейка теперь представляет собой некоторый сегмент, содержащий ключи. Это разбиение назовем процедурой *сегментации*. Также пусть i-ый сегмент $(0 \le i \le m-1)$ содержит n_i ключей.

Теорема. Если выбирать функцию h из универсального семейства H, то

$$P\left(\sum_{i=0}^{m-1} n_i^2 < 4n\right) > \frac{1}{2}.$$

Теперь можно выбирать произвольную функцию h из универсального множества, пока не получим требуемого условия $\sum_{i=0}^{m-1} n_i^2 < 4n$.

Следствие. Пусть $n_{\max} = \max\{n_1, ..., n_{m-1}\}$. Тогда $n_{\max} < 2\sqrt{n}$.

Согласно предыдущей теореме имеем $n_{\max}^2 < \sum_i n_i^2 < 4n \Rightarrow n_{\max} < 2\sqrt{n}$.

Выявление или поиск

Если же теперь для i-го сегмента $(0 \le i \le m-1)$ применить свою $МСХФ_i$ для распределения, то финальная МСХФ h_f будет выглядеть следующим образом: $h_f = МСХФ_i(x) + shift[i]$, где $shift[i]$ – это сдвиг от i-го сегмента, определяемый по формуле $shift[i] = \sum_{j=0}^{i-1} n_j$. Такую процедуру назовем процедурой *выявления*. Подобные методы также изложены в [5] и [6].

Описание алгоритма

Теперь изложим основной алгоритм. Предлагаемый подход будет отличаться от существу-

ющих тем, что процедуры сегментации и выявления будут выполняться не один раз, а несколько. Заметим, что после однократной сегментации в каждом сегменте возникает та же самая исходная задача. А это означает то, что можно проводить неоднократную сегментацию, а затем и последовательное выявление до тех пор, пока в новых "подсегментах" количество ключей будет наиболее приемлемое для реализации описанной ранее МСХФ.

Определение количества сегментаций. Предположим, что для успешной работы реализуемой МСХФ необходимо не более 20 ключей. Также будем считать, что общее количество ключей достигает 10^6. Тогда согласно следствию доказанной теоремы о максимальном элементе после:

- первой сегментации количество ключей в наибольшем сегменте не будет превышать $2\sqrt{10^6} = 2000$
- второй сегментации не будет превышать $2\sqrt{2000} < 90$
- третьей сегментации не будет превышать $2\sqrt{90} < 20$

Таким образом, достаточно трех процедур сегментации (а соответственно и выявления) для успешной работы МСХФ при использовании такого количества ключей.

Сложность алгоритма. Память $O(n)$. Построение $O(n)$. Запрос $O(1)$.

6 Применение

Описанный подход применяется для статического множества ключей. Для динамических систем с добавлением, удалением или обновлением данных также существуют различные способы для эффективного хранения информации, к примеру, B+ деревья. Хеш-функции также применяются и при оптимизации операционных систем для преобразования виртуального адреса [7] в физический адрес.

ЛИТЕРАТУРА

1. *D. Knuth.* The art of computer programming, Vol. III. Addison-Wesley

2. *Thomas H. Cormen, Charles E. Leiserson.* Introduction to algorithms – 2-nd edition. USA : MIT Press, 2001. 1180p.

3. *Андерсон Д. А.* Дискретная математика и комбинаторика. М. : Вильямс, 2004. 960с.

4. *G. Jaeshke.* Reciprocal Hashing: A Method for Generating Minimal Perfect Hashing Functions.// Communications of the ACM, vol. 24, no.12, p. 829-833, 1981

5. *F. C. Botelho, Y. Kohayakawa, N. Ziviani.* An Approach for Minimal Perfect Hash Functions for Very Large Databases.// Universidade Federal de Minas Gerais, Belo Horizonte, Brazil, Tech. Rep., 2006.

6. *R. Sprugnoli.* Perfect Hashing Functions: A Single Probe Retrieving Method for Static Sets.// Communications of the ACM, vol. 20, no. 11, pages 841-850.

7. *Дюйсембаев А.Е.* Математические модели сегментации программ. М.: Физматлит, 2001. 207с.

8. *Елиусизов Д.А.* Обоснование схем хеширования для решения задачи информационного поиска.// Сборник тезисов 13 международной научной конференции молодых ученых «Ломоносов 2006», Апрель 2006. МоиН РФ. МГУ им. Ломоносова, изд. Факультета ВмиК., 21-22-с.

Резюме

Универсалды ХЭШ әдісін қолдана отырып, минималды кәміл, ХЭШ функциясының құрылымы баяндалған. Сондай-ақ кәміл функциясы құрылымының әртүрлі әдістері қарастырылған.

Summary

The construction of Minimal Perfect Hash Function using approaches of Universal hashing is given. Different methods of perfect hashing are also considered.

Казахстанско-Британский технический университет, г. Алматы

Поступила 25.10.2010 г.