5. Функція розстановки (функція хешування)

Вибираючи хеш-функцію, ми, зазвичай, не знаємо, які саме будуть ключі. Але варто створювати хеш-функцію у певному сенсі «випадковою», яка добре «розсіває» значення по елементах таблиці. Зрозуміло, що «випадкова» функція повинна бути все ж детермінованою у тому сенсі, що під час повторних викликів з одним і тим самим аргументом вона повинна повертати одне і те ж хеш-значення.

Функцію розстановки вважають **«хорошою»**, якщо вона забезпечує рівномірний розподіл записів у таблиці: для кожного ключа всі N хеш-значень повинні бути рівноймовірні. Щоб це припущення мало сенс, зафіксуємо розподіл ймовірностей P на множині можливих ключів U. Припустимо, що ключі з множини $n = \{0, 1, \ldots, N - 1\}$ вибирають незалежно один від одного і кожний розподілений з ймовірністю P. Тоді рівномірне хешування означає, що

$$\sum_{k:h(k) = j} h(k) = \frac{1}{N},$$

де $j = 0, 1, \ldots, N - 1$

На жаль, розподіл P переважно невідомий і перевірити цю формулу неможливо. І ключі не завжди варто вважати незалежними.

На практиці треба врахувати, що час обчислення функції розстановки визначає середній час пошуку в таблиці, тому її треба задавати доволі простою обчислювальною процедурою.

Відомо декілька методів побудови функції розстановки. Вони зазвичай ґрунтуються на тому, що ключ задається в комп'ютері деяким цифровим кодом, який допускає прості арифметичні та логічні перетворення.

Передбачають, що область визначення хеш-функції — множина цілих невід'ємних чисел. Якщо ключі не є такими, їм можна надати такого вигляду. Наприклад, послідовності символів можна інтерпретувати як числа, записані у системі числення з основою, яка підходить для зображення числа. Наприклад, ідентифікатор pt — це пара чисел (112, 116), які є ASCII-кодами. Цю пару можна розглянути як зображення числа в системі з основою 128, і тоді цьому ідентифікаторові буде відповідати числовий код 112*128+116=14452. Якщо символ S, який є аргументом хешування, займає більше одного машинного слова, то **на першому кроці** хешування з S можна сформувати одне машинне слово S'. Здебільшого S' обчислюють сумуванням усіх слів за допомогою звичайного додавання або за допомогою порозрядного додавання за модулем 2.

На другому кроці з S' обчислюють результуючий індекс, причому це можна зробити кількома способами, наприклад, помножити S' на себе і використати q середніх бітів за значення функції хешування (якщо таблиця має 2^q елементів). Оскільки q середніх бітів залежать від кожного біта S', цей метод дає добрі результати.

Напевно найпоширенішою є функція хешування, що ґрунтується на **методі ділення.** За цією функцією кожному ключу k ставлять у відповідність залишок від ділення k на N, де N — кількість можливих хеш-значень (розмір таблиці): $h(x) = x \bmod N$. Наприклад, при N=12, k=100, хеш-функція буде h(k)=4.

При відображенні ключів у адреси методом ділення зберігається рівномірність розподілу, яка є на множині ключів. Ключі з близькими значеннями відображаються в унікальні адреси. Якщо два чи більше згромадження ключів відображаються в ті самі адреси, то збереження рівномірності буде недоліком. Наприклад, при дільнику 101 ключі 2000, 2001, …, 2017 відображаються в адреси 79, 80, 82, …, 99. А ключі 3310, 3311, …, 3324 — у адреси 79, 80, 82, …, 92, 93. Тому відбудеться багато колізій. Причина у тому, що ключі цих двох груп однакові за модулем 101.

Звідси видно, що деяких значень N треба уникати. Наприклад, якщо $N = 2^p$, то $h(k)$ — це просто p молодших розрядів ключа k. Якщо немає впевненості, що всі комбінації молодших бітів ключа будуть перетинатись з однаковою частотою, то степінь двійки за N не вибирають. Так само недобре вибирати за розмір таблиці степінь 10, якщо ключі є десятковими значеннями.

є числа в системі з основою 2^p, то погано визначати N як значення $2^p - 1$, оскільки при цьому однакове хеш-значення мають ключі, які відрізняються лише перестановкою цифр з системи «2^p».

Якщо за модулем d збігається багато ключів, а N і d не є взаємно простими числами, то використання N як дільника може призвести до низької ефективності хешування діленням. Якщо ж N і d є взаємно простими числами, то, звичайно, ключі не збігаються за модулем N, і тому за дільник треба вибирати просте число. Особливо треба уникати парних дільників. Як показують дослідження, добрі результати отримують при простому дільнику, далекому від степеня двійки.

Нехай, наприклад, треба помістити приблизно 2000 записів у хеш-таблицю з ланцюжками для усунення колізії. Якщо враховувати, що при пошуку елемента можна перебрати три елементи, то для ділення виберемо значення 701, бо 701≈2000/3, число просте і до степеня двійки далеко. Отже, у даному випадку хеш-функцію обчислюють за формулою h(k) = k mod 701.

Краще проекспериментувати з реальними даними, щоб з’ясувати наскільки рівномірно будуть розподілені їхні хеш-значення.

Побудова хеш-функції **методом множення** полягає у такому. Нехай кількість хеш-значень дорівнює N. Зафіксуємо константу A в інтервалі $0 < A < 1$ і визначимо $h(k)$ за формулою $h(k) = [N * (k * A \bmod 1)]$, де $k * A \bmod 1$ позначає дробову частину добутку $k * A$.

Перевага методу множення в тому, що якість хеш-функції мало залежить від вибору N. Переважно у цьому випадку за N вибирають степінь числа 2, оскільки множення на таке значення реалізується дуже ефективно як зсув.

Метод множення працює при довільному виборі константи A, але деякі значення є найкращими. В праці Д.Кнута показано, що вибір за A значення рівного $\frac{\sqrt{5} - 1}{2} \approx 0.618033$ є доволі вдалим.

Розглянемо приклад. Нехай значення ключа k=123456, N=10000 і значення A визначено попередньою формулою. Тоді значення хеш-функції буде таким:

$$h(k) = [10000 * (123456 * 0.618033 \bmod 1)] =$$
$$= [10000 * (76300.004115 \bmod 1)] =$$
$$= [10000 * 0.004115] = [41.15] = 41$$

При хешуванні **методом середини квадрата** ключ множиться сам на себе, а адресу одержують відтинанням бітів або цифр від обох кінців добутку, яке виконують доти, доки число бітів чи цифр, які залишились, не стане рівним потрібній довжині адреси. У всіх отримуваних добутках повинні використовуватись ті самі позиції.

Наприклад, нехай ключ тризначний — 738, його квадрат — 544644. Якщо потрібна двоцифрова адреса, то за її значення можна взяти 3—4 розряди результату, тобто 46.

У **методі згортання** ключ розбивають на частини, кожна з яких має довжину, яка дорівнює довжині потрібної адреси (крім останньої). Щоб сформувати адресу, частини додають, при цьому ігнорують перенесення у старшому розряді. Якщо ключі зображені у двійковому коді, то замість додавання можна використати операцію виключного АБО (XOR).

Є різні варіації цього методу. Згортання є функцією хешування, зручною для стискання багатослівних ключів і наступного переходу до інших функцій хешування.

Наприклад, нехай ключ є 187249653, а довжина адреси 3 цифри. Найпростіше згортання передбачає таке підсумовування 187+249+653 і дає адресу 89.

У **методі граничного згортання** інвертують числа у крайніх частинах ключа, і отже, у нашому прикладі будуть додані числа 781+249+356 і отримана адреса 386.

Перетворення системи числення є методом хешування, в якому робиться спроба одержати випадковий розподіл ключів за адресами в таблиці. Ключ, поданий у системі числення q (q переважно 2 або 10), розглядають як число в системі числення p ($q < p$), причому p і q — взаємно прості. Це число з системи числення p перетворюють у систему q і адресу формують шляхом вибору правих цифр (чи бітів) нового числа, або застосовують метод ділення.

Наприклад, ключ $(5403)_{10}$ розглядають як $(5403)_{11}$ і перетворюють у десяткову систему $(5403)_{11} = 5 * 11^3 + 4 * 11^2 + 0 * 11 + 3 = 7142$. Якщо нас цікавить двоцифрова адреса, то це дві праві цифри, тобто 42.