

Алгоритми та складність

Звіт

Лабораторний проект №1

*Виконав студент 2-го курсу*

*факультету кібернетики*

*КНУ ім. Тараса Шевченка*

***Кожановський Олександр***

*К-29*

*2018 р.*

***Теорія.***

Хешування - перетворення масиву вхідних даних довільної довжини в бітовий рядок фіксованої довжини за певним алгоритмом.

Ідеальна хеш-функція (англ. Perfect hash function) - хеш-функція, яка без колізій відображає різні елементи з множини об'єктів на множину ключів за O (1) часу в гіршому випадку.

Для реалізації ідеального хешування будемо будувати дворівневу хеш-таблицю, в якій перший рівень звичайний, але в якості даних буде виступати ще одна хеш-таблиця в якій і будуть зберігатись ключі. Оскільки параметри таблиці другого рівня залежать від кількості в ній ключів, то можлива реалізація лише статичної хеш-таблиці (додавання і вилучення ключів неможливі).

**Перший рівень:**

Використовується функція . Тобто n ключів кодуються в m рядків. При цьому p має бути простим і перевищувати m та будь-який з ключів.

**Другий рівень:**

На даному рівні замість створення списку ключів будемо використовувати вторинну хеш-таблицю S j, що зберігає всі ключі.

Використовується така ж функція , але зі своїми коефіцієнтами. Функція вибирається з множини універсальних хеш-функцій Hp,mj.

Для забезпечення відсутності колізій значення m встановлюється як квадрат кількості ключів в даній хеш-таблиці.

***Реалізація.***

Алгоритм був реалізований на мові с++ в середовищі розробки Visual Studio 2017.

Функція хешування має вигляд

*int Hash(int k, int a, int b, int p, int m)*

*{*

*return ((a\*k + b) % p) % m;*

*};*

Користувачеві надається доступ до вхідних параметрів змінних першого рівня та кількості ключів, які генеруються рандомно від 0 до 100 і не повторюються.

Для спрощення алгоритму і полегшення побудови другого рівня (відсутність знаходження точки повернення при виявленні колізії) з початкового масиву ключів *int \*key* створюється двовимірна таблиця, в якій ключі за їх хешем розкидаються до відповідних рядків.

Вірогідність виникнення колізій на другому рівні при не перевищує ½. А саме .

Експериментально було встановлено, що при випадкових значеннях та та n >> m колізій майже не виникає, якщо кількість ключів в таблиці 2го рівня була в межах від 2 до 4 колізії виникали значно частіше. Це суперечило умові ідеального хешування.

Були побудовані графіки функцій хешування при певних значеннях змінних (Див. **Graph.xlsx**). Було визначено, що колізій можна уникнути при певному виборі змінних a та b. Оскільки явної залежності між ключами та вибором змінних не було знайдено, то при виявленні колізії весь рядок створювався заново з іншими випадково вибраними коефіцієнтами.

З теорії, для зменшення часу пошуку необхідно щоб на першому рівні m == n (задається користувачем).

Як результат – хеш-таблиця без колізій та з часом пошуку О(1).