ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC KHOA CÔNG NGHỆ THÔNG TIN
——————oOo——————

IT003 - Cấu trúc dữ liệu và giải thuật

Hash Table and Hash Method

Sinh viên : *Trần Nhật Khoa - 22520591*

THÀNH PHỐ HỒ CHÍ MINH

3 - 2023

Mục lục

1 Đề bài

Hãy xây dựng hàm băm áp dụng trong bảng băm kích thước M để lưu trữ dữ liệu có khóa là dãy nhị phân độ dài không vượt quá 100.

Giải thích lý do xây dựng hàm băm như trên.[0.5em] Đầu tiên em xin phép bỏ qua các bước về tạo dữ liệu trong bài báo cáo này, tất cả sẽ có chi tiết trên **Github**
`https://github.com/trannhatkhoacm1612/Report-Hash-method`

2 Cách giải quyết

2.1 Lời giải 1

Không khó để ta nghĩ tới việc truy xuất thông thường bằng một hàm băm đơn giản.

```
Node table[M];

// Hàm băm đơn giản
int hashFunction(char* key) {
    int sum = 0;
    int len = strlen(key);
    for (int i = 0; i < len; i++) {
        sum += (int)key[i];
    }
    return sum % M;
}

// Thêm một khóa vào bảng băm
void put(char* key, int value) {
    int index = hashFunction(key);
    strcpy(table[index].key, key);
    table[index].value = value;
}

// Tìm kiếm giá trị tương ứng với khóa
int get(char* key) {
    int index = hashFunction(key);
    if (strcmp(table[index].key, key) == 0) {
        return table[index].value;
    } else {
        return -1;
    }
}
```

2.2 Lời giải 2

Tuy nhiên cách trên rất dễ xảy ra đụng độ, khi ta thêm một khóa mới vào bảng băm thì hàm sẽ không biết lấy ở giá trị nào. Do đó ta sẽ nghĩ tới phương pháp **Chaining**:

```
int hash_function(string key) {
    int sum = 0;
    for (int i = 0; i < key.size(); i++) {
        sum += key[i];
    }
    return sum % M;
}

void insert(string key, string value) {
    int hash_value = hash_function(key);
    hash_table[hash_value].push_back(value);
}

vector<string> search(string key) {
    int hash_value = hash_function(key);
    return hash_table[hash_value];
}
```

Phương pháp chaining là một trong những phương pháp phổ biến để giải quyết xung đột (collision) trong bảng băm (hash table). Khi áp dụng phương pháp chaining, mỗi vị trí của bảng băm sẽ chứa một danh sách liên kết, trong đó các phần tử có cùng giá trị băm (cùng ánh xạ vào vị trí đó) sẽ được lưu trữ trong cùng một danh sách.
Hay hiểu đơn giản hơn, nó sẽ kéo dài ô chứa tại một vị trí đó và thêm vào những khóa có cùng giá trị truy xuất.

Tuy nhiên, việc sử dụng phương pháp này đòi hỏi **độ phức tạp truy suất** khi bạn phải **truy suất nhiều lớp** và cần phải có lượng dung lượng lưu trữ đủ lớn vì nó sẽ kéo dài ra ô nhớ tại vị trí đó.

2.3 Lời giải 3

Và sau khi tìm hiểu em đã tìm ra một hàm băm thích hợp nhất, đó là sử dụng phương pháp **Modular Hashing**.

Phương pháp modular hashing sử dụng phép toán modulo để đảm bảo rằng giá trị băm của khóa nằm trong giới hạn của bảng băm. Ví dụ, nếu bảng băm có kích thước là M, giá trị băm của một khóa K được tính bằng cách lấy tổng của các giá trị mã ASCII của các ký tự trong khóa K, sau đó chia tổng đó cho M và lấy phần dư. Điều này đảm bảo rằng giá trị băm của khóa nằm trong khoảng từ 0 đến M-1:

```
unsigned int hashFunction(const std::string& key, int tableSize) {
    unsigned int hashVal = 0;
    for (char ch : key) {
        hashVal = 37 * hashVal + ch;
    }
    return hashVal % tableSize;
}
```

Và ta có thể sử dụng thêm một phương pháp khác để cải tiến phương pháp trên đó là **Polynomial hashing**.

```
unsigned int hashFunction(const std::string& key, int tableSize) {
    unsigned int hashVal = 0;
    unsigned int a = 33;
    for (char ch : key) {
        hashVal = (a * hashVal) + ch;
        a = a * 33;
    }
    return hashVal % tableSize;
}
```

Phương pháp polynomial hashing cũng sử dụng các giá trị mã ASCII của các ký tự trong khóa để tính giá trị băm. Tuy nhiên, phương pháp này sử dụng phép toán nhân và cộng để tạo ra một giá trị băm duy nhất. Cụ thể, giá trị băm của khóa K được tính bằng cách sử dụng công thức như ở trên.[1em] Trong đó, a là một số nguyên tố và hashVal là giá trị băm trước đó. Công thức trên có thể được hiểu là sử dụng giá trị băm trước đó nhân với số nguyên tố a, sau đó cộng với mã ASCII của ký tự tiếp theo trong khóa K.

- Hết -