Cấu trúc dữ liệu và giải thuật

TS. Phạm Tuấn Minh
Khoa Công nghệ Thông tin, Đại học Phenikaa
minh.phamtuan@phenikaa-uni.edu.vn
https://sites.google.com/site/phamtuanminh/

Chương 3: Cây và bảng băm

- Các khái niệm cây
- Cây nhị phân tìm kiếm
- Cây AVL
- **Bảng băm**

Ý tưởng

- Cấu trúc dữ liệu bảng băm là một mảng có kích thước cố định chứa các phần tử. Kích thước bảng là TableSize, bảng chạy từ 0 tới TableSize-1
- Tìm kiếm thực hiện trên một thành phần dữ liệu của phần tử, gọi là khóa (key/pivot)
- Hàm băm (Hash Function): Mỗi key được ánh xạ tới một số trong khoảng 0 tới TableSize-1 và đặt trong ô tương ứng

0	
1	
2	
3	john 25000
4	phil 31250
5	
6	dave 27500
7	
8	mary 28200
9	

1-3

Hàm băm

- Nếu các khóa đầu vào là số nguyên, thì chỉ cần trả về Key mod TableSize thường là một cách hợp lý, trừ khi Key có đặc điểm đặc biệt.
- Ví dụ, nếu kích thước bảng là 10 và các khóa đều kết thúc bằng 0, thì đó là một lựa chọn tồi => Thường chọn kích thước bảng là số nguyên tố
- Khi key là số nguyên ngẫu nhiên, thì hàm không chỉ tính toán mà cần phần bố khóa đều
- Thông thường key là chuỗi, khi đó cần chọn hàm băm hợp lý
  - Ví dụ: Cộng các giá ASCII của các kí tự trong chuỗi

```
int hash( const string & key, int tableSize ) {
    int hashVal = 0;
    for( char ch : key )
        hashVal += ch;
    return hashVal % tableSize;
}
```

1-4

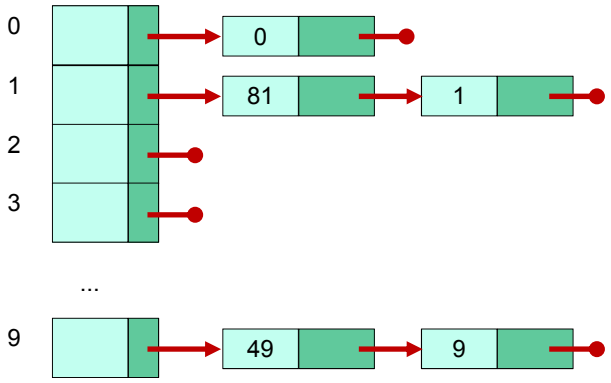
Đụng độ

- Khi thêm một phần tử, nếu giá trị hàm băm của key trùng với giá trị ô đã có phần tử thì xảy ra đụng độ
- Một số phương pháp giải quyết đụng độ: separate chaining và open addressing

1-5

Separate chaining

- Để chứa danh sách các phần tử có cùng giá trị hàm băm
- Tìm kiếm phần tử: Sử dụng hàm băm để xác định danh sách, sau đó tìm trên danh sách đó
- Chèn: Kiểm tra danh sách xem có phần tử trong đó không, nếu không có thể chèn vào đầu danh sách



1-6

Open Addressing

- Seperate chaining hashing có hạn chế là sử dụng danh sách liên kết -> chậm
- Open addressing:
  - Cố gắng tìm ô chưa sử dụng
  - Nếu tất cả các ô đều sử dụng, thì cần bảng lớn hơn
- Ví dụ:
  - f là hàm giải quyết xung đột, $h_i(x)$ = hash(x) + f(i)) mod TableSize
  - f(i) = i
  - Chèn key {89, 18, 49, 58, 69} vào bảng, đụng độ đầu tiên xảy ra khi chèn 49: Nó sẽ được chèn vào ô trống đầu tiên tiếp theo là ô 0

1-7

Cấu trúc dữ liệu và giải thuật

- Nội dung bài giảng được biên soạn bởi TS. Phạm Tuấn Minh.

1-8