TRƯỜNG ĐẠI HỌC AN GIANG
KHOA CÔNG NGHỆ THÔNG TIN

PHÂN TÍCH THIẾT KẾ GIẢI THUẬT

Giảng viên phụ trách:
NGUYỄN THÁI DƯ

Chương 4 BẢNG BĂM

- Giới thiệu bài toán
- Hàm băm
- Các phương pháp xử lý đụng độ

ĐẶT VẤN ĐỀ

- Cho S là tập hợp n phần tử trong 1 cấu trúc dữ liệu được đặc trưng bởi 1 giá trị khóa
- Tìm 1 phần tử có hay không trong S
  - Tìm tuyến tính (O(n)), chưa được sắp xếp
  - Tìm nhị phân ($O(\log_2 n)$), đã được sắp xếp
- Có hay chăng 1 thuật toán tìm kiếm với O(1)
  - Có, song ta phải tổ chức lại dữ liệu
  - Dữ liệu được tổ chức lại là Bảng băm

Giới thiệu về Bảng Băm

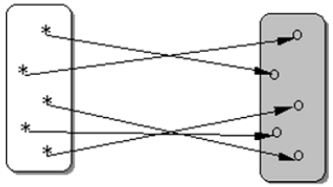
- Là CTDL trong đó các phần tử của nó được lưu trữ sao cho việc tìm kiếm sẽ được thực hiện bằng cách truy xuất trực tiếp thông qua từ khóa.
- Bảng băm có M vị trí được đánh chỉ mục từ 0 đến M-1, M là kích thước của bảng băm.
- **Các phương pháp băm:**
  - PP kết nối trực tiếp
  - PP kết nối hợp nhất
  - PP dò tuyến tính
  - PP dò bậc 2
  - PP băm kép

Hàm băm (Hash functions)

- **Hàm băm**: **biến đổi khóa thành chỉ mục trên bảng băm**
  - Khóa có thể là dạng số hay dạng chuỗi
  - Chỉ mục được tính từ 0..M-1, với M là số chỉ mục của bảng băm
  - Hàm băm thường dùng: **key % M**, với M là độ lớn của bảng băm
- **Hàm băm tốt phải thoả yêu cầu**
  - Giảm thiểu xung đột
  - Phân bố đều trên M địa chỉ khác nhau của bảng băm

Mô tả dữ liệu

- K: tập các khoá (set of keys)
- M: tập các địa chỉ (set of addresses).
- HF(k): hàm băm dùng để ánh xạ một khoá k từ tập các khoá K thành một địa chỉ tương ứng trong tập M. Thông thường **HF(k) = k mod M**



Tập khóa K Hàm băm Tập địa chỉ M

Ưu điểm bảng băm

- Dung hòa tốt giữa thời gian truy xuất và dung lượng bộ nhớ
  - Nếu không giới hạn bộ nhớ: one-to-one, truy xuất tức thì
  - Nếu dung lượng bộ nhớ có giới hạn thì tổ chức một số khóa có cùng địa chỉ, lúc này thời gian truy xuất có bị suy giảm đôi chút.
- Bảng băm ứng dụng nhiều trong thực tế, thích hợp tổ chức dữ liệu có kích thước lớn và lưu trữ ngoài

Cách xây dựng bảng băm

- Dùng hàm băm để ánh xạ khóa K vào 1 vị trí trong bảng băm. Vị trí này như là 1 địa chỉ khi tìm kiếm.
- Bảng băm thường là mảng, danh sách liên kết, file(danh sách đặc)

Ví dụ một bảng băm đơn giản

- Khóa k sẽ được lưu trữ tại vị trí **k mod M**

(M kích thước mảng)

- Ví dụ: Thêm phần tử x = 95 vào mảng M=10

95 mod 10 = 5

0	1	2	3	4	5	6	7	8	9
					95				

Ví dụ một bảng băm đơn giản

- **Với các giá trị: 31, 10, 14, 93, 82, 95,79,18, 27, 46**

0	1	2	3	4	5	6	7	8	9
10	31	82	93	14	95	46	27	18	79

Tìm kiếm trên bảng băm

- Thao tác cơ bản nhất được cung cấp bởi Hashtable là "tìm kiếm"
- Chi phí tìm kiếm trung bình là O(1), không phụ thuộc vào số lượng phần tử của mảng (Bảng).
- Chi phí tìm kiếm xấu nhất (ít gặp) có thể là O(n)

Các phép toán trên hàm băm

- Khởi tạo (Initialize)
- Kiểm tra rỗng (Empty)
- Lấy kích thước bảng băm (size)
- Tìm kiếm một phần tử trong bảng băm (Search)
- Thêm 1 phần tử vào bảng băm (Insert)
- Xóa 1 phần tử khỏi bảng băm (Remove)
- Duyệt (Traverse)

Vấn đề nảy sinh

- Giả sử **thêm 55** vào bảng băm sau:

0	1	2	3	4	5	6	7	8	9
		82			**95**		27		

  - **55** phải lưu vào vị trí **5**. Tuy nhiên vị trí này đã có chứa **95**
  - **k1 ≠ k2 mà f(k1) = f(k2) => Đụng độ**

=> Cần giải quyết *đụng độ (xung đột)*

Vấn đề xung đột khi xử lý bảng băm

- Trong thực tế có nhiều trường hợp có nhiều hơn 2 phần tử sẽ được "băm" vào cùng 1 vị trí
- Hiển nhiên phần tử được "băm" đầu tiên sẽ chiếm lĩnh vị trí đó, các phần tử sau cần phải được lưu vào các vị trí trống khác sao cho vấn đề truy xuất và tìm kiếm phải dễ dàng

Làm giảm xung đột

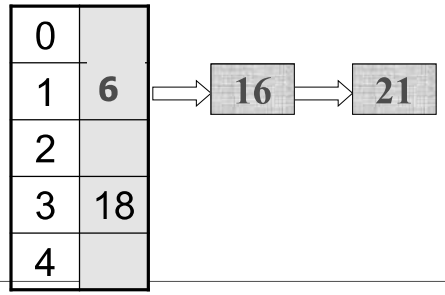
- Hàm băm cần thỏa mãn các điều kiện:
  - Xác xuất phân bố khoá là đều nhau
  - Dễ dàng tính toán thao tác
  - Ít xảy ra đụng độ

Giải quyết xung đột

- **Các phương pháp băm:**
  - PP nối kết trực tiếp
  - PP nối kết hợp nhất
  - PP dò tuyến tính
  - PP dò bậc hai
  - PP băm kép

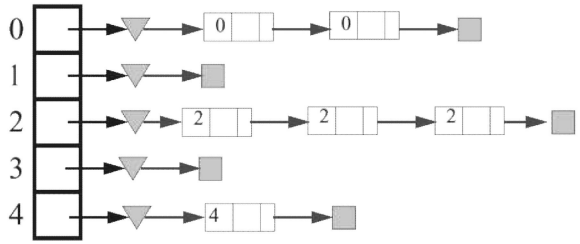
Sử dụng DS liên kết (nối kết trực tiếp)

- Ý tưởng: "**Các phần tử băm vào trùng vị trí k được nối vào danh sách nối kết**" tại vị trí đó.
- Ví dụ: cho hàm băm F(k) = k mod 5,
  - **Thêm 6, 16** vào bảng băm sau:



Sử dụng DS liên kết (nối kết trực tiếp)

- Các nút bị băm cùng địa chỉ (các nút bị xung đột) được gom thành một danh sách liên kết
- Các nút trên bảng băm được *băm* thành các danh sách liên kết. Các nút bị xung đột tại địa chỉ i được nối kết trực tiếp với nhau qua danh sach liên kết i.



*Nhận xét

PP DSLK có nhiều khuyết điểm:

- Khi có quá nhiều khoá vào cùng vị trí, DSLK thì tại vị trí đó sẽ rất dài

  => Tăng chi phí tìm kiếm
- Các ô trống còn dư nhiều => lãng phí về thời gian tìm kiếm và không gian lưu trữ

Sử dụng PP "nối kết hợp nhất"

- Ý tưởng: "Nếu có 1 khóa bị băm vào vị trí đã có phần tử thì nó sẽ được chèn vào ô trống phía cuối mảng". (Dùng mảng có M phần tử)

Sử dụng PP "nối kết hợp nhất"

- Bảng băm trong trường hợp này được cài đặt bằng danh sách liên kết dùng mảng, có M nút. Các nút bị xung đột địa chỉ được nối kết nhau qua một danh sách liên kết.
- Mỗi nút của bảng băm là một mẫu tin có 2 trường:
  - Trường key: chứa các khóa node
  - Trường next: con trỏ chỉ node kế tiếp nếu có xung đột.
- Khi khởi động bảng băm thì tất cả trường key được gán NULL, tất cả trường next được gán –1.

Key	next
NULL	-1
…	…
NULL	-1

Sử dụng PP "nối kết hợp nhất"

- Khi thêm một nút có khóa key vào bảng băm,hàm băm f(key) sẽ xác định địa chỉ i trong khoảng từ 0 đến M-1.
  - Nếu chưa bị xung đột thì thêm nút mới vào địa chỉ này .
  - Nếu bị xung đột thì nút mới được cấp phát là nút trống phía cuối mảng. Cập nhật liên kết next sao cho các nút bị xung đột hình thành một danh sách liên kết.
- Khi tìm một nút có khóa key trong bảng băm,hàm băm f(key) sẽ xác định địa chỉ i trong khoảng từ 0 đến M-1, tìm nút khóa key trong danh sách liên kết xuất phát từ địa chỉ i.

Ví dụ: nối kết hợp nhất

- Minh họa cho bảng băm có tập khóa là tập số tự nhiên, tập địa chỉ có 10 địa chỉ (M=10) (từ địa chỉ 0 đến 9), chọn hàm băm

  f(key) = key % 10.
- Thêm Key=10, 42, 20,109

	Key	Next
0	10	9
1		-1
2	42	-1
….	null	-1
8	109	-1
9	20	8

Sử dụng PP "Dò tuyến tính"

- Ý tưởng: "Nếu có 1 khóa bị băm vào vị trí đã có phần tử thì nó sẽ được chèn vào ô trống gần nhất" theo phía bên phải.

Sử dụng PP "Dò tuyến tính"

- Bảng băm trong trường hợp này được cài đặt bằng danh sách kề có M nút, mỗi nút của bảng băm là một mẫu tin có một trường key để chứa khoá của nút.
- Khi khởi động bảng băm thì tất cả trường key được gán NULL.
- Khi thêm nút có khoá key vào bảng băm, hàm băm f(key) sẽ xác định địa chỉ i trong khoảng từ 0 đến M-1:

0	Null
1	Null
2	Null
3	Null
…	Null
M-1	Null

Sử dụng PP "Dò tuyến tính"

- **Thêm các nút 32, 53, 22, 92, 17, 34, 24, 37, 56 vào bảng băm.**

0	NULL	0	NULL	0	NULL	0	NULL	0	56
1	NULL	1	NULL	1	NULL	1	NULL	1	NULL
2	32	2	32	2	32	2	32	2	32
3	53	3	53	3	53	3	53	3	53
4	NULL	4	22	4	22	4	22	4	22
5	NULL	5	92	5	92	5	92	5	92
6	NULL	6	NULL	6	34	6	34	6	34
7	NULL	7	NULL	7	17	7	17	7	17
8	NULL	8	NULL	8	NULL	8	24	8	24
9	NULL	9	NULL	9	NULL	9	37	9	37

Sử dụng PP "Dò tuyến tính"

- Hàm băm lại của phương pháp dò tuyến tính là truy xuất địa chỉ kế tiếp. Hàm băm lại lần i được biểu diên bằng công thức sau:
- **f(key)=(f(key)+i) %M** với f(key) là hàm băm chính của bảng băm.
- Lưu ý địa chỉ dò tìm kế tiếp là địa chỉ 0 nếu đã dò đến cuối bảng

Nhận xét:

- Chúng ta thấy bảng băm này chỉ tối ưu khi băm đều, tốc độ truy xuất lúc này có bậc 0(1).
- Trường hợp xấu nhất là băm không đều hoặc bảng băm đầy, lúc này hình thành một khối đặc có n nút trên tốc độ truy xuất lúc này có bậc 0(n).

Sử dụng PP "Dò bậc hai"

- Bảng băm dùng phương pháp dò tuyến tính bị hạn chế do rải các nút không đều,bảng băm với phương pháp dò bậc hai rải các nút đều hơn.
- Hàm băm lại của phương pháp dò bậc hai là truy xuất các địa chỉ cách bậc 2.
- Hàm băm lại hàm f_i được biểu diễn bằng công thức sau:
  - $f_i(key)=(f(key) + i^2) \% M$
  - với f(key) là hàm băm chính của bảng băm.
- Nếu đã dò đến cuối bảng thì trở về dò lại từ đầu bảng.
- Bảng băm với phương pháp dò bậc hai nên chọn số địa chỉ M là số nguyên tố.

Ví dụ:

- Cho hàm băm chính: F(k) = k mod 5, sử dụng PP "dò bậc hai" $f_i(key)=(f(key) + i^2) \% M$
- Thêm 6, 16 vào bảng băm sau:

0	NULL	0	16
1	21	1	21
2	NULL	2	6
3	18	3	18
4	NULL	4	NULL

Thêm vào các khóa **10, 15, 16, 20, 30, 25, ,26, 36**

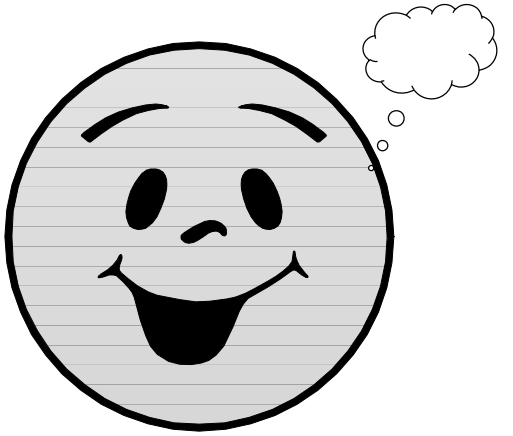
0	10	0	10	0	10	0	10	0	10
1	NULL	1	20	1	20	1	20	1	20
2	NULL	2	NULL	2	NULL	2	NULL	2	36
3	NULL	3	NULL	3	NULL	3	NULL	3	NULL
4	NULL	4	NULL	4	30	4	30	4	30
5	15	5	15	5	15	5	15	5	15
6	16	6	16	6	16	6	16	6	16
7	NULL	7	NULL	7	NULL	7	26	7	26
8	NULL	8	NULL	8	NULL	8	NULL	8	NULL
9	NULL	9	NULL	9	25	9	25	9	25

Sử dụng PP "Băm kép"

- Ta sử dụng 2 hàm băm:

  f1(key)= key % M

  f2(key)= (M–2) – key % (M-2)



Cảm ơn !