Algoritmos e Estruturas de Dados

tabelas de dispersão

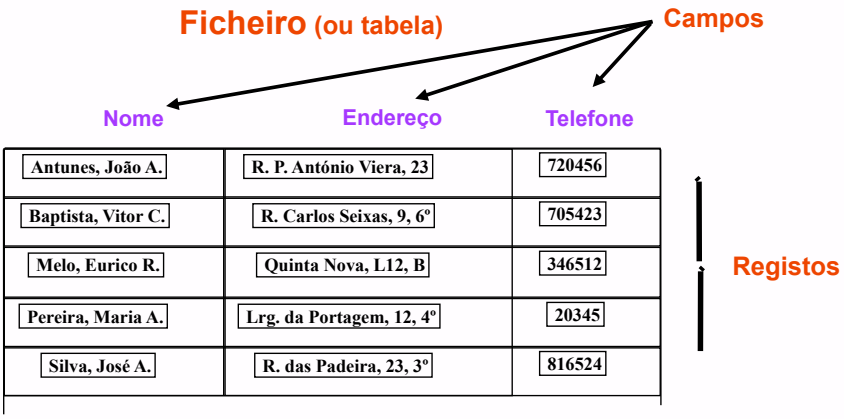
2010-2011

Carlos Lisboa Bento

Tabelas de dispersão

conceitos

Pesquisa: localização de um *Registo* num *Ficheiro*
(aceder ao registo: ler/alterar informação)



- Ficheiro de tamanho n: sequência de n items r(1), r(2), r(3) .... r(n) designados por registos
- Registos: divididos em campos (no ex: nome, endereço, telefone)

Tabelas de dispersão
conceitos

Pesquisa binária

Método

• A chave da pesquisa é comparada com a chave do registo a meio da tabela

• Se forem iguais -> fim da pesquisa;

• Senão, continua-se a pesquisa na metade superior ou inferior da tabela, consoante a chave de pesquisa for maior ou menor do que a chave do registo a meio da tabela;

A pesquisa binária pode ser efectuada na tabela principal ou então utilizada conjuntamente com a tabela sequencial indexada.

Eficiência:

- O número máximo de comparações é $\log_2 n$.

Tabelas de dispersão
conceitos

Em todos os métodos de pesquisa estudados até agora a **pesquisa é feita à custa de uma dada sequência de comparações de chaves** até encontrar a chave pesquisada ou se concluir que essa chave não existe.

Situação ideal

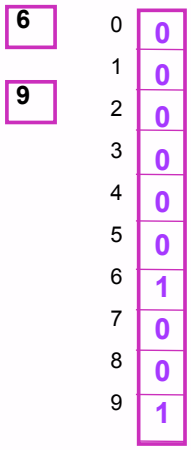
Não ter comparações desnecessárias;
a chave ser acedida num único acesso

Tabelas de dispersão
conceitos

Pesquisa directa

Uma forma de obter acesso directo a qualquer chave (e, consequentemente, ao registo correspondente) é organizar a tabela sob a forma de um *array*.
Se as chaves forem inteiras, elas próprias podem ser usadas como índices do *array*.

Exemplo:



insert(i) a[i]++

find(i) a[i] ≠ 0

remove(i) find(i); a[i]--

Operações em tempo constante

Tabelas de dispersão
conceitos

Pesquisa directa

Problemas:

1.Suponhamos que tinhamos chaves inteiras de 32 bits em vez de 16 bits

2^{32} = 4.295 x 10^9 posições !!

•Suponhamos que as chaves são cadeias de caracteres e que representamos um caracter por 7 bits (0 – 127)

ex.: junk = 'j' . 128^3 + 'u' . 128^2 + 'n' . 128^1 + 'k' . 128^0 = 224 229 227

Questão central: como evitar ter uma tabela incomportavelmente grande ?

Tabelas de dispersão
conceitos

Dispersão (*Hashing*)

Solução:

Números Grandes —mapeados, funções de dispersão→ **Números Pequenos**

ex.: x % tamanhoTabela gera pequenos valores entre 0 e tamanhoTabela - 1

c/ tamanhoTabela = 10 000 junk → 224 229 227 → 9 227

Problema:

Vários números grandes com o mesmo correspondente número pequeno.

⇨ **Colisão**

Solução:

Resolução de colisões: **linear probing** **quadratic probing** **separate chaining**

Tabelas de dispersão
conceitos

Dispersão (*Hashing*)

Em resumo:

- Escolher uma função de dispersão
  - divisão
  - entrelaçamento
  - centro do quadrado
  - mudança de base
- Escolher um método de resolução de colisões
  - linear probing
  - quadratic probing
  - separate chaining hashing
  - bucket addressing
  - bucket addressing with overflow area

Tabelas de dispersão
conceitos

Funções de Dispersão:: divisão

$$N = dim(tabela)$$
$$h(K) = K \bmod N$$

- Vantajoso escolher para **N um número primo**.
- **Não sendo N número primo, mas não tendo factores primos inferiores a 20** obtemos também bons resultados.
- Quando pouco se sabe sobre as chaves a divisão é muitas das vezes a função de dispersão escolhida

Tabelas de dispersão
conceitos

Funções de Dispersão:: entrelaçamento (folding)

- A chave é dividida em vários segmentos.
- Os segmentos são combinados e transformados criando-se o endereço de dispersão.
- Ex.:
NC = 123456789 ➔ 123 + 456 + 789 = 1368 ➔ 1368 % 100 = 68

Tabelas de dispersão
conceitos

Funções de Dispersão:: centro do quadrado

- A chave é elevada ao quadrado.
- Os algarismos centrais do resultado são usados como índices na tabela de dispersão.
- **Neste método (tal como no anterior) todos os elementos da chave inicial contribuem para a formação da chave de dispersão o que contribui para obter bons resultados de dispersão.**
- Em termos práticos é interessante usar uma **potência de 2** como tamanho da tabela de dispersão e extrair o centro da representação binária.
- Ex.:
  3121 ➔ 3121^2 = 9740641 ➔ 10010100101000001011100001 ➔
  $010100010_2 = 322_{10}$

Tabelas de dispersão
conceitos

Funções de Dispersão:: mudança de base

- Mudar a base da chave.
- Ex.:
  $345_{10} = 423_9$ ➔ 423 % 100 = 23 ➔ $23_9 = 21_{10}$

Tabelas de dispersão
conceitos

Funções de Dispersão:: problemas de overflow intermédio

Problema:

Cadeia de Caracteres —(**mudança de base**)→ **Números Grandes** —(**mapeados**)→ **Números Pequenos**

funções de hashing

Esta conversão gera em geral números maiores que os suportados pelo computador
ex.: $128^4 = 2^{28}$ (apenas a um factor de 8 da representação de um inteiro numa máquina de 32 bits!!)

Solução:

temos que para um polinómio $A_3 X^3 + A_2 X^2 + A_1 X^1 + A_0 X^0$ este pode ser reescrito em
$(((A_3) X + A_2) X + A_1) X + A_0$

3 multiplicações e 3 adições, generalizando para ordem n, n multiplicações e n adições

não temos de calcular valores intermédios elevados do tipo X^i

Solução: aplicar o operador % após cada multiplicação ou adição

Tabelas de dispersão
conceitos

Colisões

Problema:

Pode acontecer que h(chave1) = h(chave2) ← **Colisão de *hash***

É necessário

- minimizar as colisões de dispersão — **p. ex., aumentando a gama de valores da função de *hash***
- resolver as colisões de dispersão, quando elas surgirem

Tabelas de dispersão
conceitos

Linear Probing

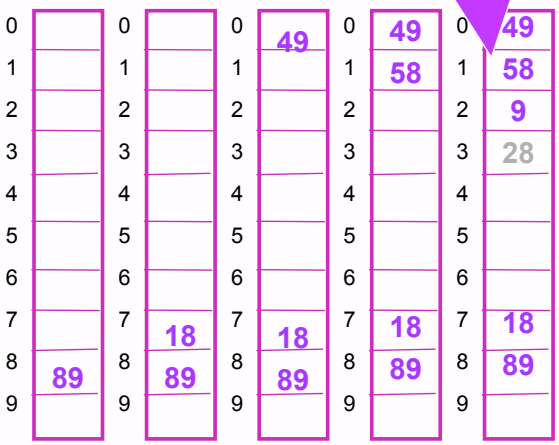
Linear probing: procura nas células seguintes à ocupada uma posição livre

Formação de agrupamentos primários

No extremo se só houver uma célula livre vai percorrer toda a tabela (longe do tempo constante para inserção / procura

Ex.: inserir 89 18 49 58 9

hash (89, 10) = 9
hash (18, 10) = 8
hash (49, 10) = 9 → 9+1
hash (58, 10) = 8 → 8+1→ 8+2 → 8+3
hash (9, 10) = 9 → 9+1→ 9+2 → 9+3
hash (28, 10) = 8 → 9+1→ 9+2 → 9+3 → 9+4 → 9+5



Tabelas de dispersão
demos na Web

http://www.engin.umd.umich.edu/CIS/course.des/cis350/hashing/WEB/HashApplet.htm

Tabelas de dispersão
conceitos

Quadratic Probing

Objectivo: eliminação de agrupamentos primários

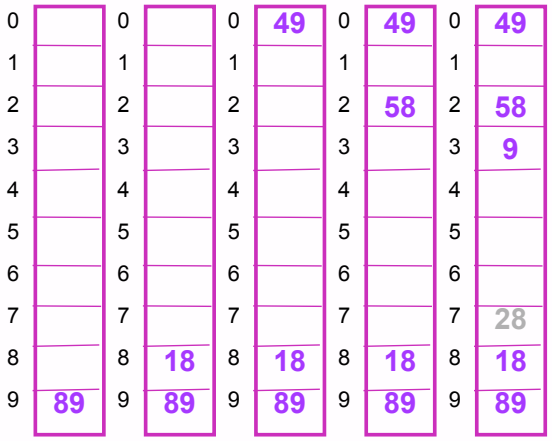
$F(i) = i^2$

ex.: H $H + 1^2$ $H + 2^2$ $H + 3^2$ … … $H + i^2$

comparar com linear probing H H + 1 H + 2 H + 3 … … H + i

Ex.: inserir 89 18 49 58 9

hash (89, 10) = 9
hash (18, 10) = 8
hash (49, 10) = 9 → $9+1^2$
hash (58, 10) = 8 → $8+1^2$ → $8+2^2$
hash (9, 10) = 9 → $9+1^2$ → $9+2^2$
hash (28, 10) = 8 → $8+1^2$ → $8+2^2$ → $8+3^2$



Tabelas de dispersão
demos na Web

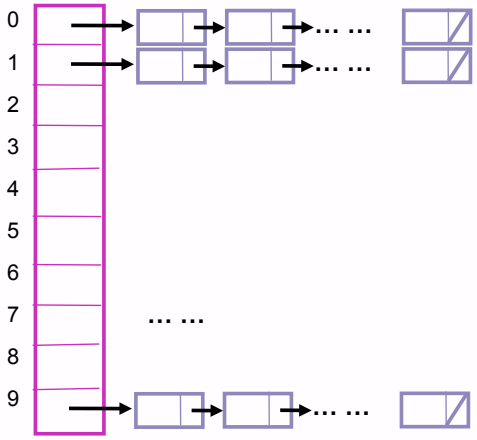
http://www.engin.umd.umich.edu/CIS/course.des/cis350/hashing/WEB/HashApplet.htm

Tabelas de dispersão
conceitos

Separate Chaining Hashing

Para cada posição uma “chain” de registos.

Cada célula da hashtable contém um ponteiro para a lista respectiva (pode guardar ou não também uma chave)



Só fará sentido com factores de carga significativos

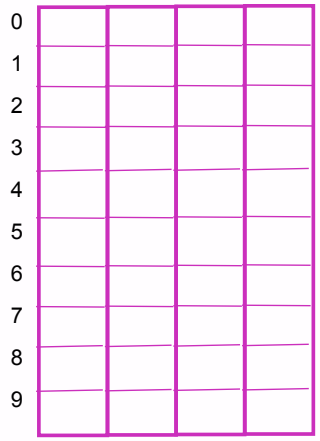
Tabelas de dispersão
demos na Web

http://www.engin.umd.umich.edu/CIS/course.des/cis350/hashing/WEB/HashApplet.htm

Tabelas de dispersão
conceitos

Bucket Addressing

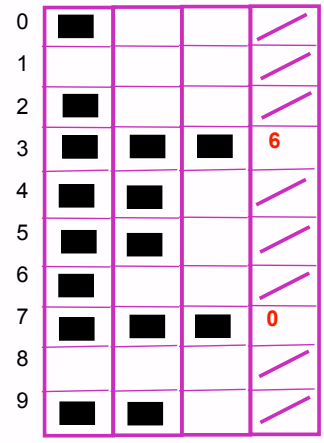
Cada endereço tem associadas várias posições para resolução de colisões. Ou seja, existe um “balde” (bucket) para onde “despejar” as chaves



Tabelas de dispersão
conceitos

Bucket Chaining

Cada endereço tem associadas várias posições para resolução de colisões, se não forem suficientes recorre a uma função de resolução de colisões, mas coloca no bucket corrente informação da posição onde vai ser colocada a próxima chave.



Tabelas de dispersão
demos na Web

http://www.engin.umd.umich.edu/CIS/course.des/cis350/hashing/WEB/HashApplet.htm

Tabelas de dispersão
conceitos

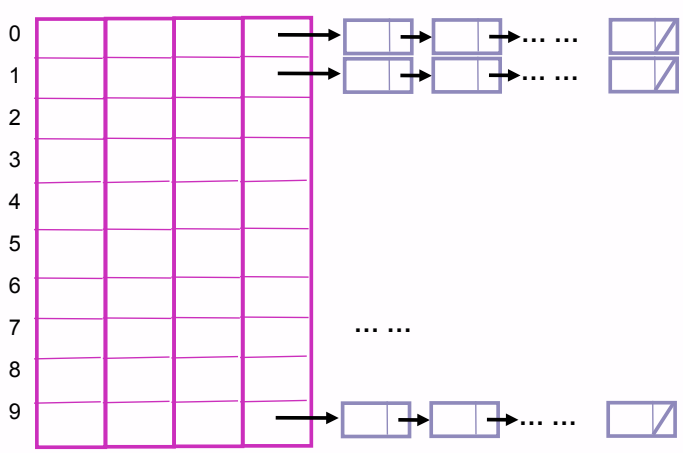
Bucket Addressing wth Overflow Area

Cada endereço tem associadas várias posições para resolução de colisões

Tem disponível também um espaço para casos de "*overflow*"

Este espaço pode ser uma lista ligada...

…ou simplesmente uma tabela auxiliar



Tabelas de dispersão
conceitos

Funções de redispersão:: minimizar o *clustering*

- **Por vezes o que é feito é criar duas tabelas de dispersão:**
  - Tabela de dispersão primária
  - Tabela de dispersão de overflow
  - Cada tabela tem a sua própria função de dispersão.

Tabelas de dispersão
demos na Web

http://www.engin.umd.umich.edu/CIS/course.des/cis350/hashing/WEB/HashApplet.htm

Tabelas de dispersão
conceitos

Escolha de uma função de dispersão

- • Deve dispersar as chaves uniformente;
- • Eficiência no cálculo dos índices;
- • O método do resto da divisão inteira nem sempre é bom, pois produz colisões com frequência (várias chaves terem o mesmo índice);
- • Outros métodos (já apresentados anteriormente):
  - Centro do quadrado: neste método a chave é multiplicada por ela própria e os dígitos do meio são usados como índice;
  - Entrelaçamento: a chave é dividida em vários segmentos com os quais se obtém um novo valor e é aplicada uma função (de resto, por expo.)

Nota: se as chaves não forem inteiros (por exemplo forem uma string alfanumérica), é necessário fazer a sua conversão para inteiros antes de aplicar qualquer dos métodos estudados.

Tabelas de dispersão
conceitos

Medida de qualidade de uma função de redispersão

Propriedade:

- Uma boa função de redispersão deve ser a que, para qualquer índice i, as sucessivas redispersões rh(i), rh(rh(i)), etc, cobrem tantos inteiros entre 0 e Max_tabela quantos possível

Exemplo:

rh(i) = (i + c) mod m

m - número de elementos da tabela

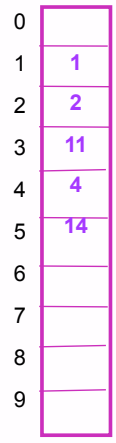
c - valor constante

Esta função cobre todos os valores da tabela desde que c e m sejam primos relativos (i.e., não tenham nenhum divisor comum superior a 1)

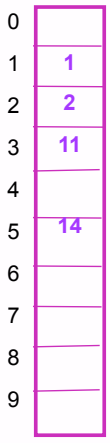
Tabelas de dispersão
conceitos

Eliminações numa tabela de dispersão (com hash(k,10))

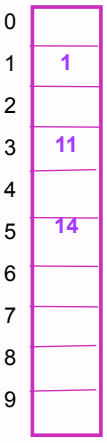
Inserir 1, 4, 2,14,11:



Eliminar 4:



Eliminar 2:



Aceder 11:

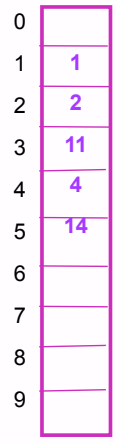
Tabelas de dispersão
conceitos

Eliminações numa tabela de dispersão (com hash(k,10))

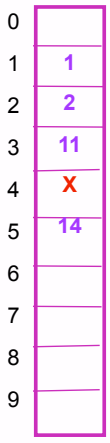
Solução

- Marcar as células apagadas com uma flag – mais tarde podem ser reocupadas.
- De tempos a tempos purgar a tabela de dispersão de células marcadas apagadas.

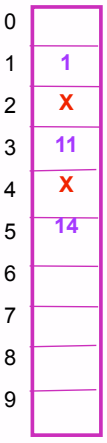
Inserir 1, 4, 2, 14, 11:



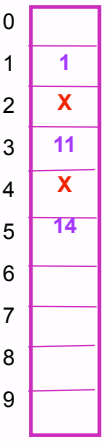
Eliminar 4:



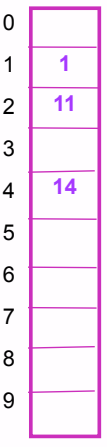
Eliminar 2:



Aceder 11:



Purgar:



Tabelas de dispersão

… end ;-)