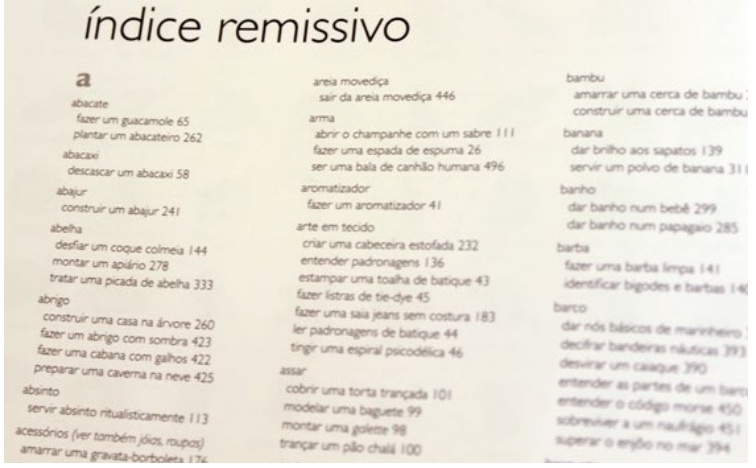
Tabela de Espalhamento

Hash

Prof. Leandro Colevati

Introdução

- O uso de listas ou árvores para organizar informações é interessante e produz resultados bastante bons
- Entretanto, em nenhuma dessa estruturas se obtém o acesso direto a alguma informação, a partir do conhecimento de sua chave
- O acesso direto ao conteúdo de alguma informação, a partir de sua chave, é possível através do uso de tabelas de indexação, ou de espalhamento
- Essas são as tabelas hash
    - Imagine a criação de um índice remissivo de um manual técnico, por exemplo.


índice remissivo

a

abacate
fazer um guacamole 65
plantar um abacateiro 262
abacaxi
descascar um abacaxi 58
abajur
construir um abajur 241
abelha
desfiar um coque colmeia 144
montar um apiário 278
tratar uma picada de abelha 333
abrigo
construir uma casa na árvore 260
fazer um abrigo com sombra 423
fazer uma cabana com galhos 422
preparar uma caverna na neve 425
absinto
servir absinto ritualisticamente 113
acessórios (ver também jóias, roupas)
amarrar uma gravata-borboleta 176

areia movediça
sair da areia movediça 446
arma
abrir o champanhe com um sabre 111
fazer uma espada de espuma 26
ser uma bala de canhão humana 496
aromatizador
fazer um aromatizador 41
arte em tecido
criar uma cabeceira estofada 232
entender padronagens 136
estampar uma toalha de batique 43
fazer listras de tie-dye 45
fazer uma saia jeans sem costura 183
ler padronagens de batique 44
tingir uma espiral psicodélica 46
assar
cobrir uma torta trançada 101
modelar uma baguete 99
montar uma golette 98
trançar um pão chalá 100

bambu
amarrar uma cerca de bambu
construir uma cerca de bambu
banana
dar brilho aos sapatos 139
servir um polvo de banana 311
banho
dar banho num bebê 299
dar banho num papagaio 285
barba
fazer uma barba limpa 141
identificar bigodes e barbas 14
barco
dar nós básicos de marinheiro
decifrar bandeiras náuticas 393
desvirar um caiaque 390
entender as partes de um barc
entender o código morse 450
sobreviver a um naufrágio 451
superar o enjôo no mar 394

Definição

- Uma função de hashing associa um elemento de um certo conjunto (strings, números, arquivos, etc.) a um número inteiro de tamanho conhecido
- Uma tabela de espalhamento é um tipo abstrato de dados para busca em conjuntos dinâmicos cuja implementação tem certas propriedades:
  - os dados são acessado por meio de um vetor de tamanho conhecido
  - a posição do vetor é calculada por uma função de hashing

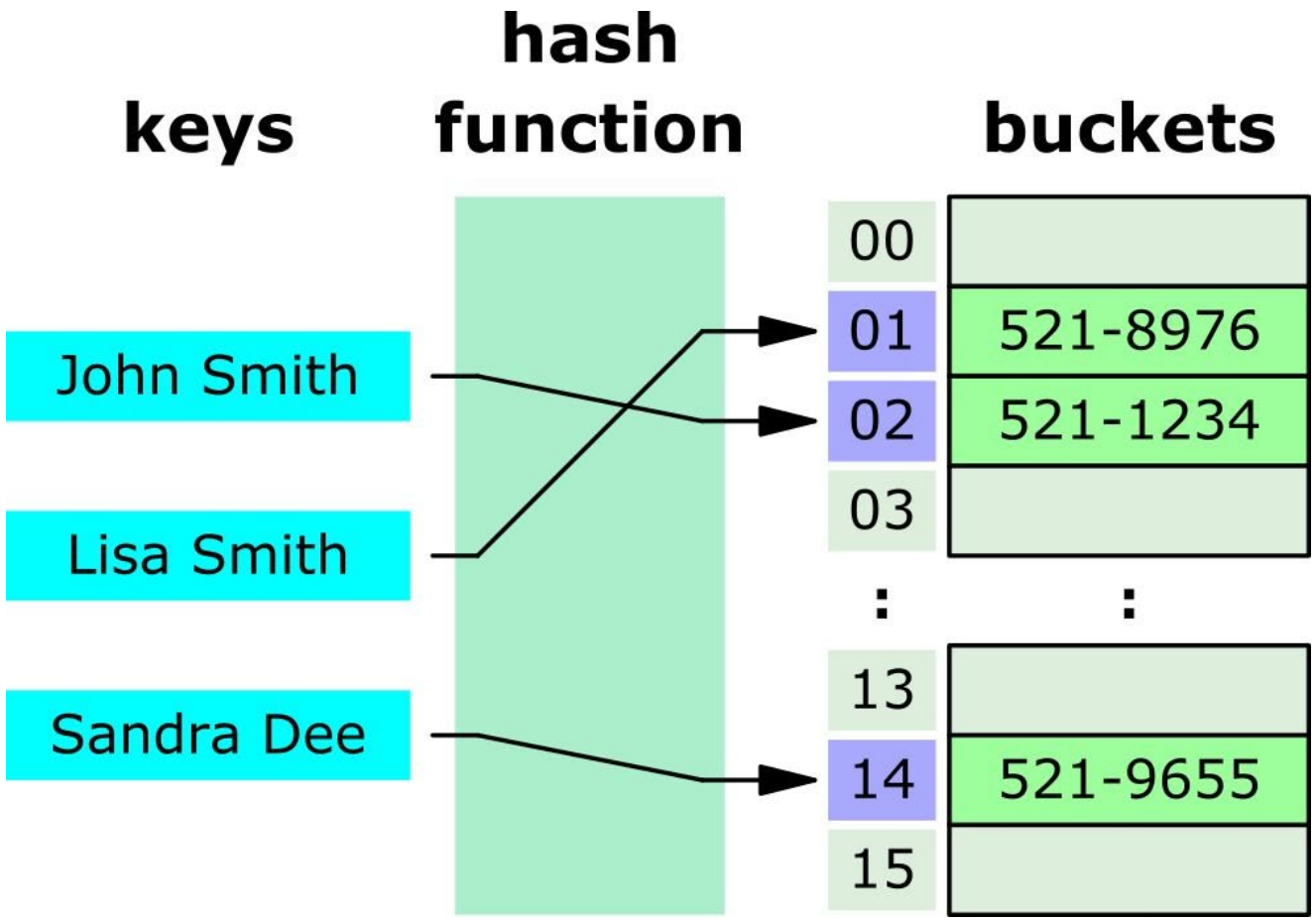
Funcionamento

- O valor retornado pela função hash é usado para localizar o elemento na tabela. Temos então:

Pos(elemento) = H(elemento,n)

- Em que H(elemento,n) é a função hash aplicada ao elemento, considerando uma tabela de tamanho n

Funcionamento



Características

- Embora permita o acesso direto ao conteúdo das informações, o mecanismo das tabelas hash possui uma desvantagem fundamental em relação a listas e árvores
- Numa tabela hash é virtualmente impossível estabelecer uma ordem para os elementos, ou seja, a função hash faz indexação mas não preserva ordem

Características

- Embora sejam projetadas para permitir o acesso direto ao conteúdo de uma informação, a partir de sua chave, isso nem sempre ocorre
- Em muitas situações, dois elementos possuem a mesma chave e, teoricamente, levariam à mesma posição na tabela
- Esse é o fenômeno da **colisão**

Exemplo

- Criar uma tabela de espalhamento de n = 365 (dias do ano) para armazenar 50 pessoas aleatórias
  - Caímos no paradoxo do aniversário:
    - Nesse cenário proposto, apesar de tomarmos muito menos pessoa que dias do ano, ao menos 2 pessoas farão aniversário no mesmo dia
  - Isso nos traz ao menos uma colisão

Colisões

- Uma vez que colisões são praticamente inevitáveis, dependendo do tamanho da fonte de dados, devemos ter técnicas para reduzi-las ou trata-las
- Para reduzir as colisões a saída é a escolha de uma boa função hash
- Para tratar as colisões temos que saber como proceder na ocorrência de alguma colisão

Escolha da função hash

- Considerando uma tabela de espalhamento de tamanho M, uma boa função de hashing espalha bem:
  - A probabilidade de uma chave ter um hash específico é (aproximadamente) 1/M
  - Ou seja, esperamos que cada lista tenha n/M elementos
- Métodos genéricos (funcionam bem na prática):
  - Método da divisão
  - Método da multiplicação
- Hashing perfeito: Se conhecermos todos as chaves, a priori, é possível encontrar uma função de hashing injetora
  - isto é, não temos colisões
  - tais funções podem ser difíceis de encontrar

Método de Divisão

- Obtemos o resto da divisão pelo tamanho M do hashing

$$h(x) = x \textbf{ mod } M$$

- Exemplo:
  - Elemento “bala”
    - ASCII ➔ 01100010 01100001 01101100 01100001
    - Para decimal ➔ 1.650.551.905
  - Tabela de tamanho (M) = 1783 posições

$$h(\text{“bala”}) = 1.650.551.905 \textbf{ mod } 1783 = 277$$

Método da Multiplicação

- Multiplicamos por um certo valor real A e obtemos a parte fracionária
  - Escolhemos A conveniente, por exemplo A = (√5 − 1)/2
  - Posição relativa no vetor não depende de M (pode ser M = 1024)
    - mod 1 ➔ Remover a parte inteira do número

$$h(x) = \lfloor M\ (A \cdot x \ \textbf{mod}\ 1) \rfloor$$

- Exemplo:

$$h(\text{“bala”}) = \lfloor 1024 \cdot [((\sqrt{5} - 1)/2 \cdot 1.650.551.905)\ \textbf{mod}\ 1] \rfloor$$

$$h(\text{“bala”}) = \lfloor 1024 \cdot [1020097177{,}4858876\ \textbf{mod}\ 1] \rfloor$$

$$h(\text{“bala”}) = \lfloor 1024 \cdot 0{,}4858876 \rfloor$$

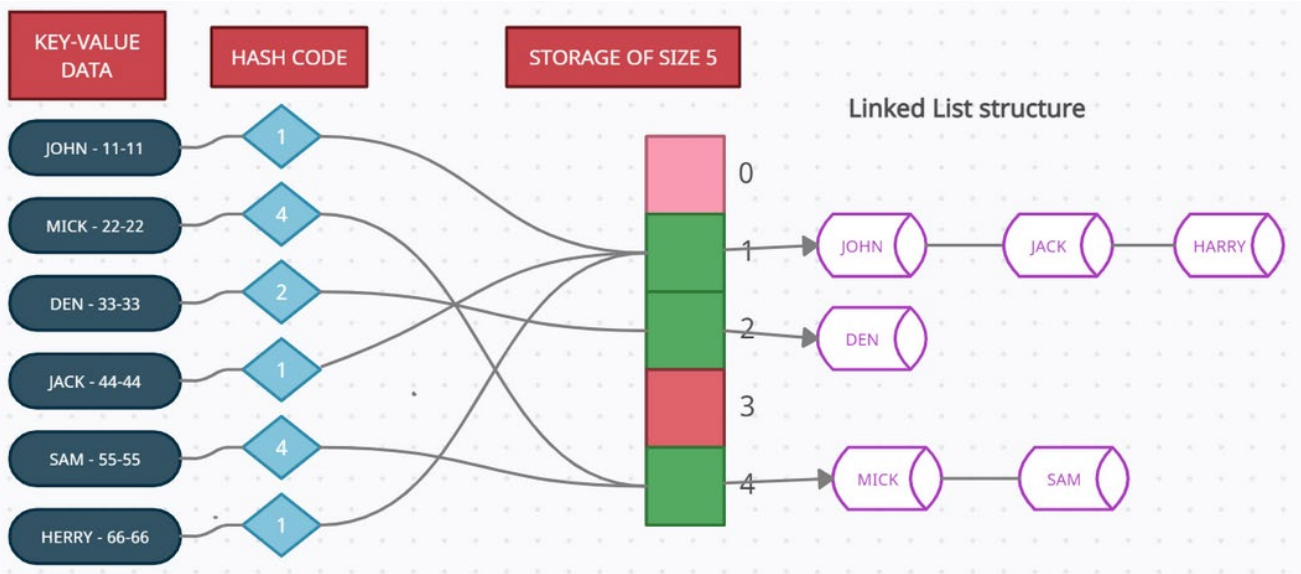
$$h(\text{“bala”}) = \lfloor 497{,}5489024 \rfloor = 497$$

Tratamento das colisões

- Tratamento feito por 3 técnicas diferentes:
  - Encadeamento
  - Sondagem
  - Rehashing

Técnica de Encadeamento

- Trata se de criar listas para os elementos com mesmo valor hash
- Assim, ao se fazer a inserção de um elemento para uma posição já ocupada, cria se uma lista ordenada com os elementos daquela posição da tabela
- Isso impede o acesso direto mas mantém o custo de acesso bastante baixo



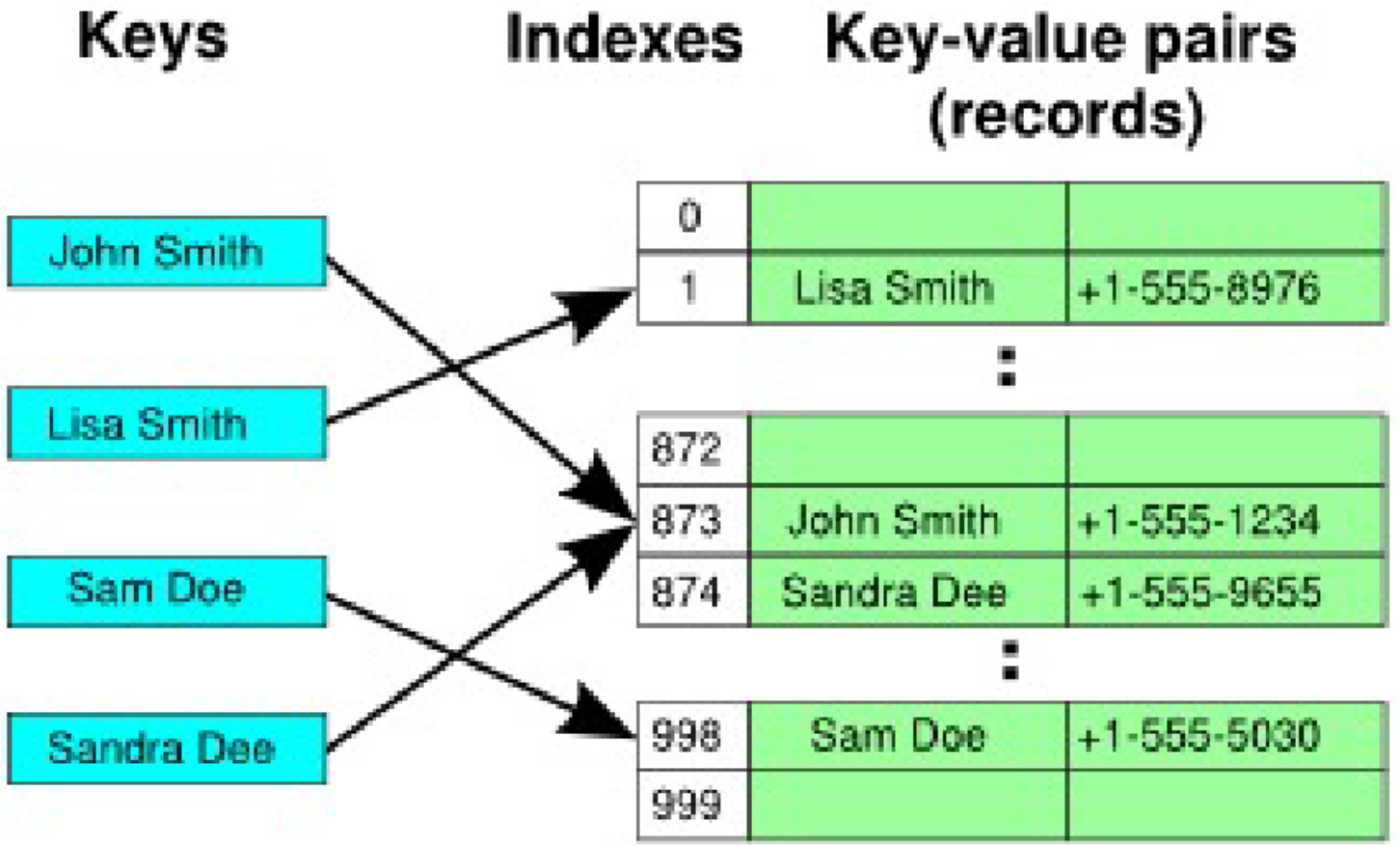
Técnicas de Sondagem

- A técnica de encadeamento é simples e relativamente eficiente
- Entretanto, apresenta a desvantagem de criar uma segunda estrutura de dados para manipular as colisões
- As técnicas de sondagem tratam as colisões de forma a não necessitar dessa segunda estrutura quando o tamanho da tabela de espalhamento (M) é maior que o número de elementos (n)
- São as técnicas de sondagem:
  - Sondagem Linear
  - Sondagem Quadrática
  - Hash Duplo

Técnica de Sondagem Linear

- Em caso de colisão o novo elemento ocupará a próxima posição livre da tabela
- Em tabelas com grande ocupação isso implica em tempos elevados de busca
- Causa também o problema conhecido como clustering (aglomeramento) primário, ou seja, mais provável que o cluster formado por esse um par de registros contíguos cresça pela adição de ainda mais registros em colisão, independentemente de os novos registros terem hash para o mesmo local dos dois primeiros.

Técnica de Sondagem Linear



Técnica da Sondagem Quadrática

- No caso de colisão, baseado em uma função de 2º grau, uma nova posição para o elemento é gerado
- Reduz problemas de clusters primários
- Pode gerar clusters secundários
  - Nesse fenômeno, uma função de hash de baixa qualidade pode fazer com que muitas chaves sejam hash para o mesmo local, uma vez que a função pode levar a que todos sejam colocadas na mesma cadeia de hash, fazendo com que tenham tempos de acesso lentos.
- Clusters primários e secundários podem ser minimizados com uma boa função hash ou usando hash duplo

Técnica de Sondagem de Hash Duplo

- É como a sondagem linear:
  - Quando detectamos conflito, ao invés de dar um pulo de 1, damos um pulo h(k, i) calculado a partir de uma segunda função de hashing
- Isto é:

$$\mathbf{h(k, i) = (hash_1(k) + i \cdot hash_2(k)) \bmod M}$$

- Cuidados:
  - $hash_2(k)$ nunca pode ser zero
  - $hash_2(k)$ precisa ser co-primo com M
    - garante que as sequências são longas
- Exemplos:
  - Escolha M como uma potência de 2 e faça que $hash_2(k)$ seja sempre ímpar
  - Escolha M como um número primo e faça que $hash_2(k) < M$

Técnica de Rehashing

- A operação de rehashing é aplicada, normalmente, quando a tabela se torna excessivamente ocupada
- Nela o que se faz é aumentar a capacidade da tabela, para um primo próximo do dobro da atual capacidade
- A função hash é ajustada para o novo limite e os elementos são realocados na tabela

Remoção

- Os problemas de colisões não aparecem apenas no momento de inserir ou buscar um elemento na tabela
- A remoção de um elemento também deve tratar a possibilidade de colisão
- No caso de encadeamento, o processo se resume à remoção de um elemento de uma lista linear

Remoção em Técnicas de Sondagem

- Para as técnicas de sondagem o processo é mais complexo
  - Uma simples exclusão pode levar a uma nova definição de hash de todos que vem na sequência, em função dos clusters primários e secundários
  - Uma técnica simples, que cria bastante desperdício, é a de marcar a posição do elemento removido como apagada (mas não livre)
    - Isso evita a necessidade de movimentar elementos na tabela mas cria muito lixo (posições sem valor na tabela)
  - Uma melhoria nessa técnica é reaproveitar as posições marcadas como removidas no caso de novas inserções
    - Isso funciona se a freqüência de inserções for equivalente à de remoções
  - Em casos extremos é necessário refazer o hashing completo dos elementos

Exemplo em Teste de Mesa

Tabela de Espalhamento com Técnica de Encadeamento:

Entrada: 3, 69, 96, 73, 53, 42, 20, 51, 83, 50

Função hash:

```
int hash(int numero) {
     int posicao = numero mod 5;
     return posicao;
}
```

Tabela de Espalhamento	
0	20 -> 50 -> NULL
1	96 -> 51 -> NULL
2	42 -> NULL
3	3 -> 73 -> 53 -> 83 -> NULL
4	69 -> NULL

Exemplo

- Considerando um projeto Java, com a importação de uma biblioteca de lista encadeada de objetos, criar um dicionário de palavras, que contenham o verbete e seu significado. O projeto deve partir de uma tabela de espalhamento de 26 posições e deve estar otimizado para tratar colisões com Técnica de Encadeamento.
- O projeto deve:
  - Inicializar a tabela de espalhamento com as características da Técnica de Encadeamento
  - Criar uma função hash baseada no primeiro caractere do verbete (Tratar para minimizar diferenças entre maiúsculos e minúsculos)
  - Adicionar palavras no dicionário
  - Buscar palavras no dicionário
  - Remover palavras do dicionário
  - Listar palavras cadastradas por caractere inicial