Tabelas Hash

Aleardo Manacero Jr.

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

Introdução

- 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

Como funciona

- Uma tabela hash é construída através de um vetor de tamanho n, no qual se armazenam as informações

- Nele, a localização de cada informação é dada a partir do cálculo de um índice através de uma função de indexação, a função hash

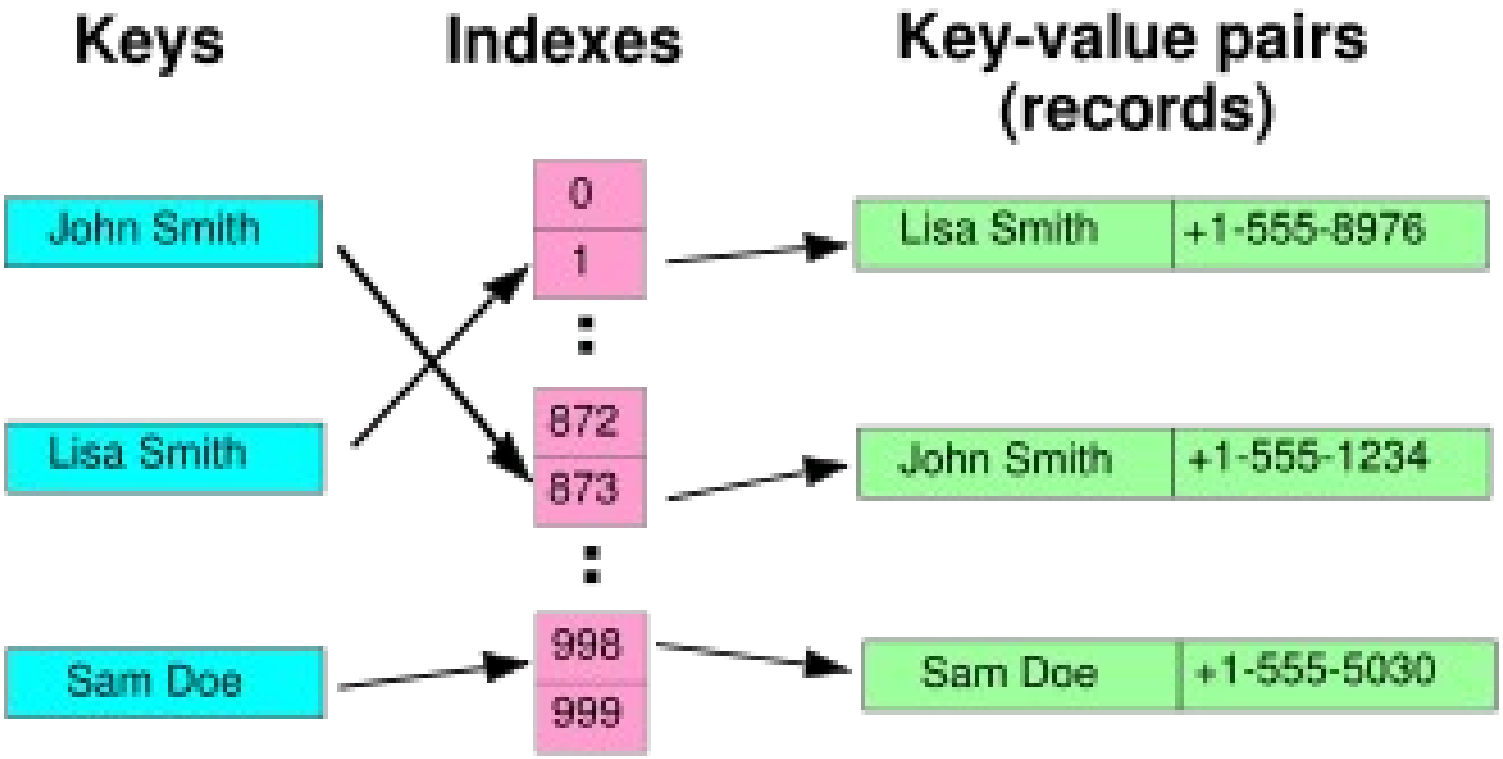
Como funciona

- 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 hahs aplicada ao elemento, considerando uma tabela de tamanho n

Como funciona



Uma desvantagem

- 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

Uma inconveniência

- 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

Colisões

- Apenas como um dado estatístico sobre a probabilidade de ocorrência de colisões, podemos considerar o paradoxo do aniversário

- Segundo ele, numa tabela de 365 posições (os dias do ano) e tomados aleatoriamente 50 pessoas (bem menos que as 365 posições disponíveis), teríamos pelo menos duas pessoas com mesma data de aniversário (uma colisão)

Evitando colisões

- Uma vez que colisões são teoricamente inevitáveis, 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 de hash

- Uma boa função hash é aquela que garanta o máximo espalhamento pela tabela

- Um hashing perfeito é aquele que garanta que dada uma tabela de n posições e n elementos a serem inseridos, então não haverá colisão

- Existem estratégias conhecidas para escolha da função hash

Escolha da função de hash

- Uma delas é dada pelo método de Cichelli:

$$h(pal) = (l(pal) + g(pal[0]) + g(pal[l(pal)])) \bmod T$$

- Outras funções podem ser estabelecidas de forma semelhante

Escolha da função de hash

- A eficiência da estrutura de hash depende, fundamentalmente, da função de hash

- A verificação da qualidade de uma função hash é feita através de testes estatísticos

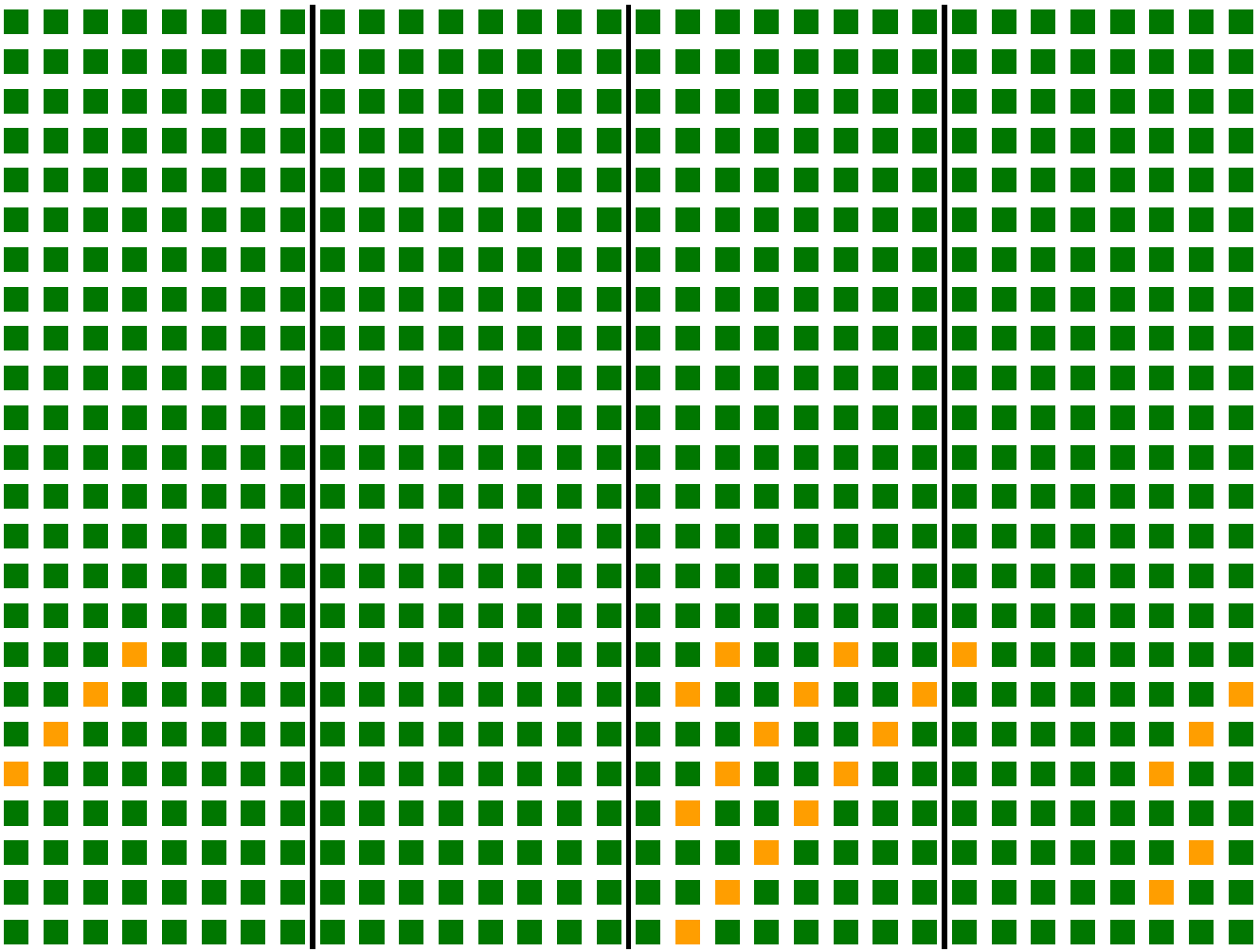
- Um destes testes é o de avalanche, em que se verifica o impacto da alteração de um bit no parâmetro de entrada da função hash sobre os bits de seu resultado

Escolha da função de hash

```
uint32 joaat_hash(uchar *key, size_t key_len)
{uint32 hash = 0;
  size_t i;

        for (i = 0; i < key_len; i++)
        {   hash += key[i];
            hash += (hash << 10); // adiciona-se 10 bits
zero e
            hash ^= (hash >> 6);  // faz ou-exclusivo
bitwise
        }
        hash += (hash << 3);
        hash ^= (hash >> 11);
        hash += (hash << 15);
        return hash;
}
```

Exemplo de avalanche



Tratamento de colisões

- O tratamento de colisões é feito por três técnicas diferentes:
  - Encadeamento
  - Sondagem
  - Rehashing

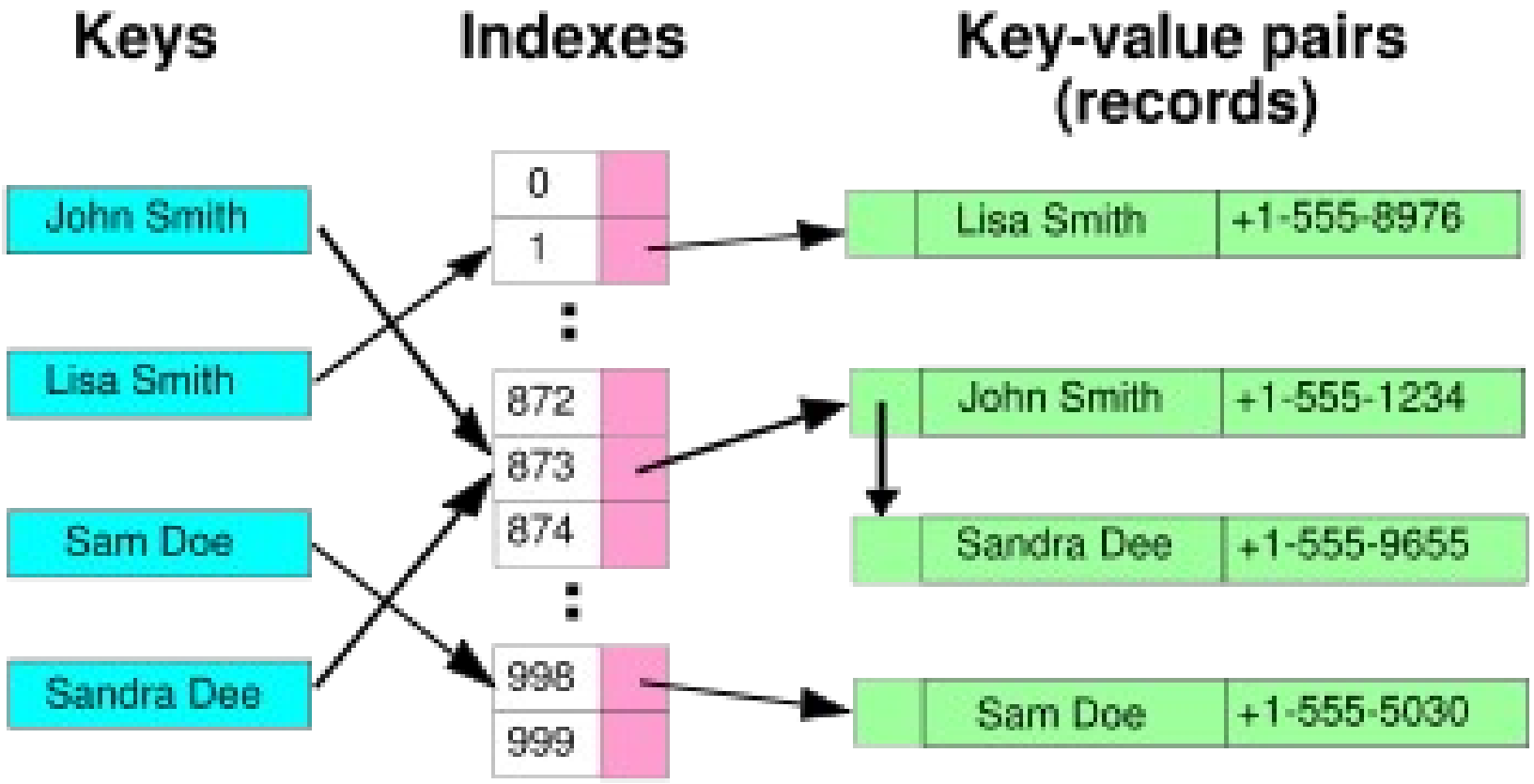
Encadeamento

- 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

Encadeamento



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 (se n > número de elementos)

Técnicas de sondagem

- A sondagem é feita por três técnicas básicas:
  - Sondagem linear
  - Sondagem quadrática
  - Hash duplo

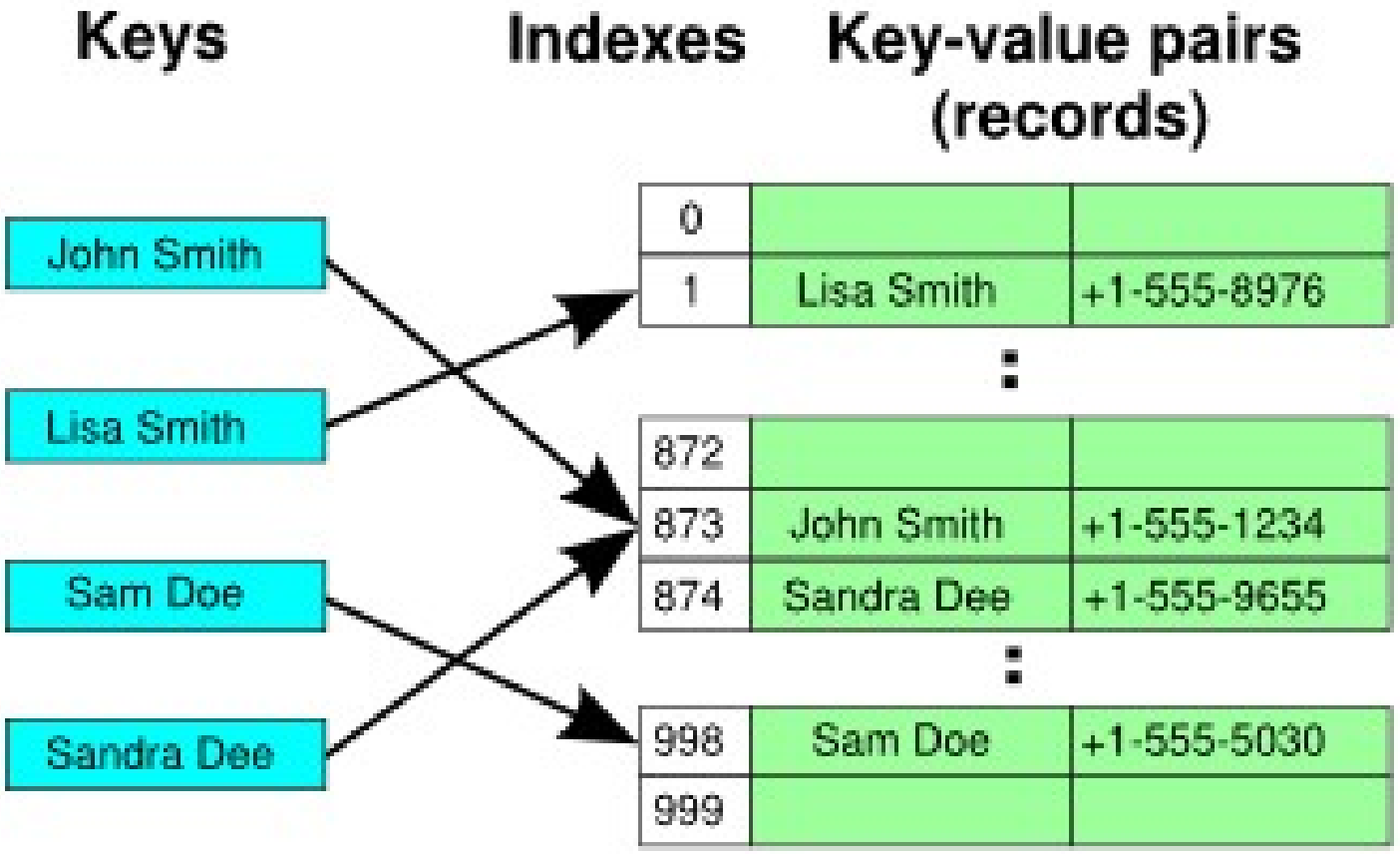
Sondagem linear

- 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

Sondagem linear



Sondagem quadrática

- No caso de colisão, a posição do novo elemento é determinada por uma relação quadrática, na forma:

$$p(I) = h(k) + (-1)^{i-1} (((i+1) /2)^2 \text{ para } i=1,\ldots,n$$

- Isso reduz os problemas da sondagem linear, embora crie clusters secundários

Hashing duplo

- Nesse caso, na ocorrência de colisões, usa-se uma segunda função hash, gerando a seguinte seqüência de sondagem:

$$h(K),\ h(k) + h_p(k),\ h(k)+2h_p(k),\ \ldots,\ h(k)+i*h_p(k)$$

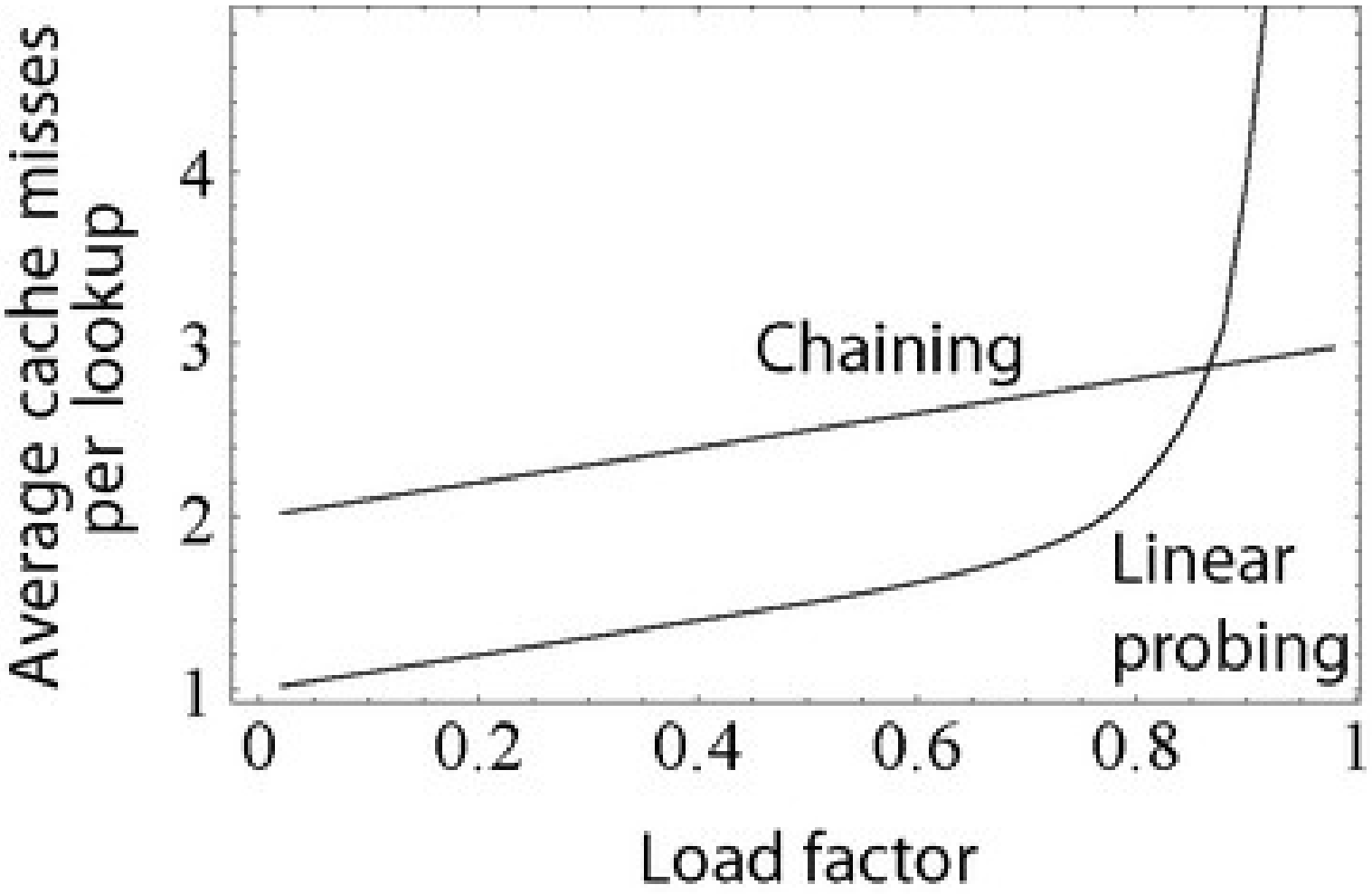
Rehashing

- 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

Desempenho das técnicas



Remoção de elementos

- 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 de elementos

- Já para as técnicas de sondagem o processo é mais complexo

- 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

Remoção de elementos

- 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

Hashing extensível

- É uma técnica de hashing aplicada ao armazenamento em dispositivo secundário

- Nela a tabela contém endereços das páginas de disco que conteriam as informações (arranjadas numa lista)

- Colisões são tratadas de forma simplificada, uma vez que vários elementos estão na mesma página (posição da tabela)