2
Atualidade de uma base de dados

Manter a atualidade de uma base de dados é um problema que pode ser abordado de diferentes maneiras. Cho e Garcia-Molina [CHO] definem esse problema da seguinte forma: uma instância de uma fonte de informações pode ser replicada; ao ocorrerem alterações ou atualizações na instância original, a cópia ficará desatualizada, com conteúdo diferente daquele presente na fonte. O problema consiste em definir uma política que faça com que a cópia possa representar o mais fielmente possível a versão original.

Podemos visualizar o ambiente da Internet como um grande repositório de informações, que pode ser acessado por uma vasta quantidade de usuários. Todos esses usuários desejam obter as informações o mais rapidamente possível, e é importante que essas informações cheguem ao usuário em suas versões mais atualizadas.

Para resolver o problema de entregar rapidamente as páginas ao usuário, uma estratégia comumente utilizada é a da manutenção de cópias das páginas em um local onde o acesso pode ser feito mais rapidamente - um cache, ou repositório. Quando o usuário deseja acessar uma página de um servidor web, primeiramente verifica-se se esta já possui uma cópia armazenada no cache; nesse caso, a cópia é imediatamente entregue ao solicitante, economizando assim o tempo de acesso ao servidor web. Caso contrário, é feito um acesso ao servidor, a página é então entregue ao solicitante e uma cópia é feita no cache, para ser utilizada nas próximas solicitações a esse mesmo endereço. Dessa forma é feita a montagem e manutenção do cache.

Um cache pode ser construído baseado nos princípios de localidade temporal e espacial. A localidade temporal sugere que uma página que foi acessada em um instante provavelmente será acessada novamente dentro de pouco

tempo; localidade espacial refere-se ao fato de que máquinas em uma mesma região (p.ex., pertencentes à rede interna de uma corporação) geralmente acessam as mesmas páginas. Nesse último caso, utiliza-se um cache comum para as máquinas que pertencem à região. [WNG]

Pode ocorrer de, após uma página ter sido armazenada em cache, a versão original sofrer alterações. Nesse caso, a cópia em cache ficaria desatualizada, e sua entrega a um usuário que deseje acessá-la pode ser um problema, visto que as informações nela contidas podem já estar erradas. Uma solução para isso seria programar os servidores web para que notifiquem todos os repositórios que já os tenham acessado caso a página que eles tenham copiado sofra alguma alteração. Podemos ver que esse mecanismo é altamente consumidor de recursos, tanto de processamento quanto de rede, e que não é facilmente escalável. Além disso, pode acontecer das atualizações ocorrerem com mais frequência do que o servidor tenha capacidade de informar aos clientes.

Nesse trabalho abordamos a manutenção da atualidade das cópias em cache como sendo de responsabilidade de quem armazena a cópia. Periodicamente, é necessário que o sistema de cache realize uma consulta aos servidores web para verificar se as páginas foram alteradas. As que tiverem sido alteradas devem ser então substituídas.

Não é possível, no entanto, que todas as páginas armazenadas no cache estejam atualizadas a todo instante. Sempre que ocorrer uma edição na página original, a cópia do cache estará desatualizada, a partir desse instante, até o momento em que o sistema visite o servidor web, perceba a alteração, e atualize sua cópia local. É desejável que a maior quantidade possível de páginas do cache estejam iguais à versão do servidor web durante o máximo de tempo, de modo a permitir que as consultas ao cache resultem na entrega de cópias corretas aos solicitantes.

2.1.
Particularidades de página Web - Generalizações

No ambiente da Internet, as páginas estão distribuídas por uma enorme quantidade de servidores, espalhados por todo o mundo. Assim, o tempo de acesso a uma página pode ser bastante diferente do tempo de acesso a outra, dependendo da distância física, da quantidade e da qualidade dos enlaces de rede entre elas, da capacidade de armazenamento e processamento dos equipamentos de rede utilizados na transmissão dos pacotes de dados. Também existe a diferença de tamanho entre as páginas: diferentes páginas armazenadas em um mesmo servidor podem consumir intervalos de tempo diferentes para serem copiadas do servidor web para o cache, se uma tiver, por exemplo, 1kB e outra 10MB.

Outro fator bastante específico do ambiente web é que uma página pode estar armazenada em diferentes servidores (espelhamento). Sistemas de servidores web podem realizar balanço de carga, e direcionar duas requisições a uma mesma página a dois endereços web completamente diferentes.

Nesse estudo, no entanto, considera-se como hipótese simplificadora que todas as páginas estariam armazenadas em servidores sem espelhamento, aos quais o tempo de acesso é constante, e que todas as páginas possuem o mesmo tamanho, e demorariam, portanto, o mesmo tempo para serem copiadas do servidor web para o cache. Para tanto, será considerado que todas as páginas possuem um valor médio, de tamanho e tempo de acesso.

2.2.
Particularidades de página Web – Especificidades

As edições que páginas web recebem no decorrer do tempo são interpretadas como eventos que ocorrem em uma determinada taxa, que pode ou não variar com o tempo. Em uma primeira hipótese simplificadora, assumimos que as páginas web são editadas em seus servidores de acordo com um Processo de Poisson. Essa modelagem é usada em sequências de eventos que ocorrem de maneira aleatória e independente, com uma taxa constante no tempo. [CHO]

Seja um Processo de Poisson homogêneo P onde eventos ocorrem a uma taxa λ. Se T é o tempo decorrido até a ocorrência do próximo evento, então a função densidade de probabilidade para T é dada por:

$$f_T(t) = \begin{cases} \lambda e^{-\lambda t}, para\ t > 0 \\ 0, \quad para\ t \leq 0 \end{cases} \quad \text{(Eq 1)}$$

Integrando-se a função densidade de probabilidade obtém-se a probabilidade de que uma página altere-se no intervalo $(0, t]$:

$$Pt\{T \leq t\} = \int_0^t f_T(t)dt = 1 - e^{-\lambda t} \quad \text{(Eq 2)}$$

2.3. Métricas para atualidade de uma base de dados

Como desejamos que a base de dados (páginas) armazenada no cache represente corretamente as páginas originais armazenadas nos servidores web, durante o máximo de tempo possível, apresentamos aqui o conceito de "freshness", de acordo com o definido por Cho e Garcia-Molina [CHO]:

Seja **p** uma página pertencente ao conjunto de cópias *C*. Se **p** apresenta o mesmo conteúdo da versão do servidor no instante *t*, então dizemos que o *freshness* de **p** no instante t, *F(p,t)*, é igual a **1**. Caso contrário, o *freshness* de **p** é **0**.

$$F(p,t), p \in C = \begin{cases} 1, p\ atualizada\ no\ instante\ t \\ 0, c.c. \end{cases} \quad \text{(Eq 3)}$$

Sobre a base de dados, definimos seu *freshness* no instante *t* como sendo a média dos *freshness* das páginas:

$$F(C,t) = \frac{1}{|C|}\sum_{p \in C} F(p,t) \quad \text{(Eq 4)}$$

2.4. Conceito de freshness médio

Considerando-se somente uma página, define-se seu freshness médio (no tempo) como sendo a média dos valores que este assumiu durante um intervalo de tempo.

$$\bar{F}(p) = \lim_{t\to\infty} \frac{1}{t} \int_0^t F(p,t)dt \qquad \text{(Eq 5)}$$

Para um conjunto de páginas *C*, o freshness médio (no tempo) é calculado sobre os valores de freshness do conjunto, definido anteriormente.

$$\bar{F}(C) = \lim_{t\to\infty} \frac{1}{t} \int_0^t F(C,t)dt \qquad \text{(Eq 6)}$$

ou

$$\bar{F}(C) = \lim_{t\to\infty} \frac{1}{t} \int_0^t \left(\frac{1}{|C|} \sum_{p\in C} F(p,t) \right) dt \qquad \text{(Eq 7)}$$

2.5. Estratégias clássicas de manutenção da atualidade (revisitação)

O estudo de Cho e Garcia-Molina [CHO] apresenta quatro aspectos que devem ser considerados no processo de sincronização. Consideramos, neste trabalho, uma sincronização sendo realizada no instante de revisitação.

1) Frequência de revisitação: representa a quantidade de oportunidades que se tem de realizar a revisitação das páginas. Quanto mais vezes for possível realizar atualizações, maior será a atualidade da base como um todo. Assumimos que é possível atualizar N páginas a cada intervalo de I unidades de tempo.
2) Alocação de recursos: Depois de decidido quantas oportunidades de revisitação o sistema terá, ainda é necessário definir quantas vezes cada uma das cópias das páginas será revisitada. Pode-se, por

exemplo, dividir as oportunidades de revisitação igualitariamente entre todas as páginas do conjunto, ou realizar essas revisitações de maneira proporcional à taxa de edição de cada uma delas. O primeiro caso é denominado **política de alocação uniforme**, e o segundo, **política de alocação não-uniforme**.

3) Ordem de revisitação: Faz diferença no resultado do processo a sequência na qual as páginas são revisitadas. Dentre as possibilidades, citamos as seguintes:
    - **Ordem fixa**, na qual, definida a sequência na qual as páginas são revisitadas, essa sequência é realizada repetidamente. Assim, o intervalo entre revisitações de uma mesma página é constante.
    - **Ordem aleatória**, onde a página a ser revisitada a cada oportunidade é sorteada dentre aquelas que ainda não foram revisitadas. Após todas as páginas terem sido sorteadas, o algoritmo permite que qualquer uma seja novamente escolhida. O intervalo entre revisitações é uma variável aleatória, mas a quantidade média de revisitações por unidade de tempo é respeitada.
    - **Ordem puramente aleatória**, onde a cada oportunidade de revisitação é sorteada uma página qualquer. Assim, o intervalo entre revisitações é aleatório, e podemos ter páginas que nunca venham a ser revisitadas.

A Tabela 1 apresenta o valor esperado para o freshness de um conjunto de páginas atualizado utilizando cada uma dessas políticas [CHO]. O valor r é definido como a taxa λ/f, sendo λ a taxa em que uma página é editada na origem e f (=N/I) a frequência de revisitação, frequência com que o elemento da cópia é atualizado. Então $r = \lambda I/N$.

Política	Freshness $\overline{F}(C)$
Ordem fixa	$\frac{1-e^{-r}}{r}$
Ordem aleatória	$\frac{1}{r}\left(1-\left(\frac{1-e^{-r}}{r}\right)^2\right)$
Ordem puramente aleatória	$\frac{1}{1+r}$

Tabela 1 – Freshness esperado para diferentes políticas de ordem de revisitação [CHO]

4) Pontos de revisitação: podemos distribuir as oportunidades de revisitação dentro de um intervalo de tempo de diversas maneiras, desde concentrando-as em um único ponto como as distanciando homogeneamente no intervalo.

As estratégias clássicas de manutenção de atualidade são definidas utilizando-se diferentes combinações desses aspectos. Escolhida a frequência de revisitação, define-se se a política de alocação será uniforme ou não-uniforme. Em seguida, a sequência de revisitação das páginas deve ser programada (sequencial, aleatória ou puramente aleatória), e enfim a concentração dos pontos de revisitação.

Essas estratégias clássicas serão utilizadas como baseline para as políticas desenvolvidas nesse trabalho.

2.6. Trabalhos relacionados

O artigo de [ECK] apresenta uma abordagem do problema de revisitação de páginas buscando escalonar uma quantidade de revisitações a uma base de dados, limitada por critérios de controle (quantidade de visitas por unidade de tempo, intervalo entre visitas), visando obter a menor obsolescência possível das cópias armazenadas. A implementação proposta no artigo utiliza programação dinâmica, e os algoritmos desenvolvidos no artigo ajustam os instantes de revisitação para

alcançar esse objetivo. Além disso, a revisitação proposta em [ECK] é feita sobre uma única fonte de informações. A relação do referido artigo com o presente trabalho é que aqui também encontramo-nos limitados por critérios de controle, e visamos à menor obsolescência das páginas copiadas. Entretanto, utilizamos aqui uma estratégia diferente, que é o aprendizado por reforço, e as revisitações ocorrem sobre qualquer uma das páginas armazenadas no repositório, não somente a uma única fonte de informação.

[CHO] e [WNG], em seus artigos, definem os conceitos que são utilizados neste trabalho, como *freshness* e obsolescência. Em [CHO] encontramos ainda os critérios utilizados para implementação das heurísticas utilizadas como baseline neste trabalho e que foram apresentados na seção anterior.

Em [KAE] e [SUT] são apresentados os conceitos de aprendizado por reforço, a técnica utilizada neste trabalho para implementação do algoritmo que busca atingir o objetivo proposto. [TES] fornece um exemplo de utilização de aprendizado por reforço que sugere que o emprego desta técnica pode ser bastante eficiente.

[SZA] apresenta uma abordagem ao problema de revisitação de páginas, limitado por critérios de controle, também sobre o conjunto de páginas da Wikipédia, utilizando políticas simples para agendamento das revisitações.