2
Fundamentação Teórica e Trabalhos Relacionados

Este capítulo descreve os principais aspectos de trabalhos encontrados na literatura para o problema de revisitação de páginas *Web*, e relaciona estes trabalhos com a pesquisa realizada nesta tese. Alguns destes trabalhos fornecem a fundamentação teórica para este tese, e por esta razão recomendamos especial atenção às Seções 2.1.1, 2.1.3, 2.3.1 e 2.3.2. A Seção 2.1 apresenta várias abordagens para medir o nível de atualização do repositório, que consistem em formas alternativas de avaliar as políticas de revisitação. As estratégias para evitar sobrecarga dos servidores *Web* são discutidas na Seção 2.2. Algumas políticas de revisitação propostas na literatura são apresentadas na Seção 2.3. Para uma visão geral sobre máquinas de busca e *crawlers Web*, indicamos (Bae99, Cha03, Ara01). Como a evolução da Internet e o processo de modificação das páginas *Web* têm grande impacto nas políticas de revisitação, indicamos os estudos (Fet04, Cho00, Nto04).

2.1
Medidas para o Nível de Atualização do Repositório

Esta seção apresenta propostas encontradas na literatura para medidas do nível de atualização/desatualização de um repositório de páginas *Web*. Um aspecto comum nestas medidas é a suposição de que as durações dos intervalos entre modificações de uma página são variáveis aleatórias independentes e identicamente distribuídas, ou seja, as modificações ocorrem segundo um processo de renovação.

2.1.1
Freshness

A métrica *freshness*, proposta em (Cho03a), captura a média no tempo do número de páginas atualizadas no repositório. Podemos entender o *freshness* de uma página como a proporção do tempo que a página permanece atualizada em uma execução com duração infinita de uma política de revisitação. Como o nível de atualização de algumas páginas podem ter maior impacto que outras

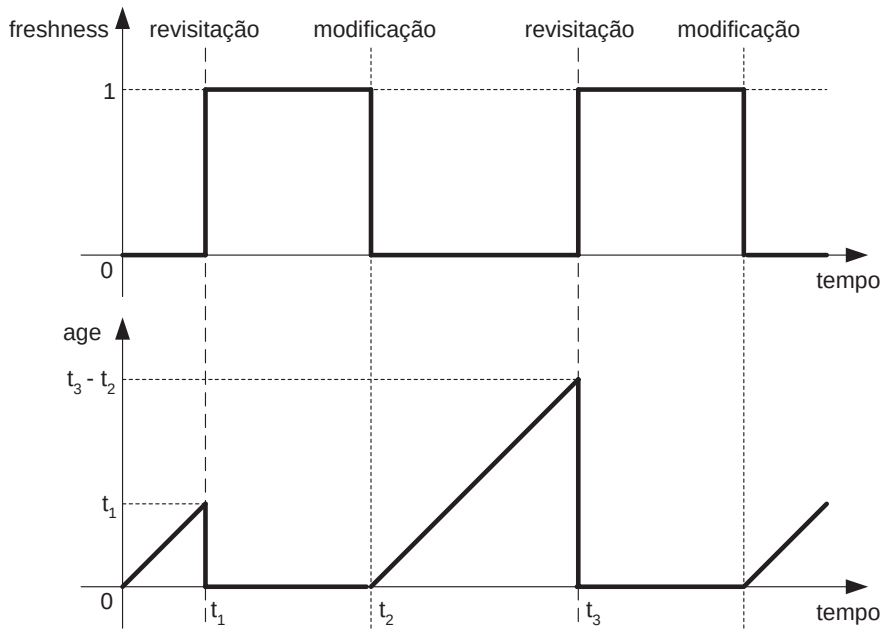

Figura 2.1: *Freshness* e *age* de uma página. O *freshness* vale 1 quando a página está atualizada, ou 0 caso contrário. O *age* vale 0 quando a página está atualizada, ou vale o tempo deste a última modificação quando está desatualizada.

na qualidade do serviço fornecido pelas máquinas de busca[1], o *freshness* de um repositório é definido como a média do *freshness* das páginas, ponderada pelas importâncias das páginas. Ou seja, o *freshness* de um repositório é dado por

$$A = \sum_i w_i A_i,$$

onde A_i e w_i são o *freshness* e a importância da página i, respectivamente. Para facilitar a interpretação do *freshness* assumimos que $\sum_i w_i = 1$. Por exemplo, considere um repositório com n páginas onde 40% das páginas estão atualizadas 20% do tempo, e as 60% restantes estão atualizadas 80% do tempo. Adotando importâncias uniformes $w_i = 1/n$, temos que $A = 0,4n(1/n)0,2 + 0,6n(1/n)0,8 = 0,54$.

Para definir formalmente o *freshness* A_i de uma página i, considere o *freshness* $A_i(t)$ de uma página i em um instante t como sendo

$$A_i(t) = \begin{cases} 1, & \text{se a página } i \text{ está atualizada no instante } t, \\ 0, & \text{caso contrário.} \end{cases}$$

[1]Por exemplo, se páginas com maior *page rank* (Pag98) têm maior chance de serem retornadas em um consulta, então faz sentido estabelecer que páginas com maior *page rank* tenham maior peso no cálculo do *freshness* do repositório.

O *freshness* $A_i(t)$ de uma página i ao longo do tempo t está ilustrado na Figura 2.1. Desta forma, o *freshness* A_i de uma página i é definido como (Cho03a)

$$A_i = \lim_{t\to\infty} \frac{1}{t} \int_0^t A_i(t)dt.$$

Vamos considerar a classe das políticas de revisitação tal que a frequência de ocorrência de cada duração dos intervalos entre revisitações à página i converge para uma distribuição $h_i(.)$ com média finita, quando o tempo de execução da política tende ao infinito. Vamos assumir também que as páginas se modificam segundo um processo de Poisson, o que é razoável para a maioria das páginas na Internet (Cho03a). Portanto, a probabilidade da duração de um intervalo entre modificações da página i ter pelo menos $t \geq 0$ unidades de tempo é dada por $F_i(t) = 1 - \exp(-\lambda_i t)$, onde $\lambda_i > 0$ é a taxa de modificação da página i.

Como as durações dos intervalos entre modificações da página i são sem memória (Mon03), o *freshness* da página i é definido em (Cho03a) como a razão entre o tempo esperado que a página i permanece atualizada em um intervalo entre revisitações, e o tempo esperado entre revisitações. Ou seja, se $B_i(t)$ é o tempo esperado que a página i permanece atualizada em um intervalo entre revisitações com duração t, então

$$A_i = \frac{\int_0^\infty B_i(t)h_i(t)dt}{\int_0^\infty t \times h_i(t)dt}. \tag{2-1}$$

Note que a distribuição $h_i(.)$ é determinada pela política de revisitação adotada.

Intuitivamente, a chance da página i estar atualizada x unidades de tempo desde a última revisitação pode ser obtida pela expressão

$$p_i(x) = \mathrm{E}[A_i(x)] = 0 \times (1 - \exp(-\lambda_i x)) + 1 \times \exp(-\lambda_i x) = \exp(-\lambda_i x).$$

Portanto, considerando um intervalo de tempo com duração Δ e um ponto qualquer x neste intervalo, temos que o tempo esperado que a página i permanece atualizada neste intervalo é aproximadamente $p_i(x)\Delta$. Logo, dividindo o intervalo entre revisitações com duração t em n partes de tamanho $\Delta = t/n$, temos que

$$B_i(t) = \lim_{n\to\infty} \sum_{j=1}^{n} p_i\left(j \times t/n\right)\left(\frac{t}{n}\right) = \int_0^t p_i(x)dx = \frac{1 - \exp(-\lambda_i t)}{\lambda_i}. \tag{2-2}$$

Portanto, substituindo a Equação (2-2) na Equação (2-1), podemos obter o *freshness* de uma página i através da distribuição $h_i(.)$ das durações

dos intervalos entre revisitações consecutivas. Desta forma, se uma variável aleatória U_i tem densidade de probabilidade $h_i(.)$, então podemos calcular A_i utilizando a expressão

$$A_i = \frac{\mathrm{E}[B_i(U_i)]}{\mathrm{E}[U_i]} = f_i \times \mathrm{E}[B_i(U_i)] = \frac{1 - \mathrm{E}[\exp(-\lambda_i U_i)]}{\lambda_i / f_i}, \qquad (2\text{-}3)$$

onde $f_i = \mathrm{E}[U_i]^{-1}$ é a frequência de revisitação da página i.

2.1.2 Age

A métrica *age* captura a idade média das páginas de um repositório. Temos a definição formal a seguir, fornecida em (Cho03a), para o *age* no instante t de um repositório com N páginas:

$$I(t) = \sum_{i=1}^{N} w_i I_i(t),$$

onde $I_i(t)$ é o *age* da página i no instante t, definido como

$$I_i(t) = \begin{cases} 0, & \text{se a página } i \text{ está atualizada no instante } t \\ t - t_m(i), & \text{caso contrário,} \end{cases}$$

onde $t_m(i)$ é o instante da primeira modificação depois da última revisitação à página i. Temos uma ilustração de $I_i(t)$ na Figura 2.1. Quando uma página i se modifica segundo um processo de Poisson com taxa λ_i, Cho e Garcia-Molina (Cho03a) mostram que o *age* esperado de uma página após t unidades de tempo desde a última revisitação vale

$$\mathrm{E}[I_i(t)] = \int_0^t (t - x)\lambda_i \exp(-\lambda_i x)dx = t\left(1 - \frac{1 - \exp(-\lambda_i t)}{\lambda_i t}\right).$$

Quando aplicamos uma política com distribuição $h_i(.)$ para a duração dos intervalos entre revisitações de uma página i, a expressão para o *age* da página i pode ser obtida de forma similar à apresentada para o *freshness*.

2.1.3 Staleness

A medida *staleness*, proposta em (Wol02), captura a proporção esperada do tempo que uma página permanece desatualizada em um intervalo de tempo finito de operação do *crawler*. De forma similar ao *freshness* do repositório (Seção 2.1.1), o *staleness* do repositório é definido como a média do *staleness* das páginas ponderada pelas importâncias das páginas.

O *staleness* de uma página i em um intervalo finito de operação do *crawler*, com duração T, depende dos instantes $0 \leq t_{i,1} < t_{i,2} < \ldots < t_{i,x_i} \leq T$ dentro deste intervalo onde ocorrem as x_i revisitações à página i. O *staleness* depende também da distribuição de probabilidades $F(.)$ (acumulada), com média $\lambda_i^{-1} > 0$, do intervalo entre modificações consecutivas da página i. Wolf et al (Wol02) demonstram que o *staleness* da página i vale

$$S_i(t_{i,1}, \ldots, t_{i,x_i}) = \frac{1}{T} \sum_{j=0}^{x_i} \int_{t_{i,j}}^{t_{i,j+1}} \left(1 - \lambda_i \int_0^{\infty} (1 - F(t - t_{i,j} + v))dv\right) dt.$$

Embora o *staleness* esteja definido em um intervalo de tempo finito, ele permite obter o nível de atualização do repositório para o caso em que as páginas não se modificam segundo um processo de Poisson. Note que as métricas *freshness* e *age* estão definidas para operação infinita do *crawler*, mas temos expressões para o *freshness/age* do repositório apenas para o caso de modificações das páginas segundo um processo de Poisson. Observando os intervalos entre modificações das páginas dos Jogos Olímpicos de Nagano (1998), Wolf et al (Wol02) concluíram que algumas delas não apresentavam bom ajuste à distribuição exponencial, sendo melhor modeladas pela distribuição Weibull ou Pareto. Entretanto, na prática não é uma tarefa simples ajustar a distribuição de cada página em repositórios tão grandes quanto das máquinas de busca comerciais.

A restrição de *politeness* (Definição 1.2) pode ser mais facilmente inserida no modelo proposto em (Wol02), pois os instantes de revisitação aparecem na definição do *staleness*. Ou seja, podemos definir o problema com a restrição de *politeness* como

$$\begin{aligned}
\text{minimize} \quad & \sum_{i \in R} w_i S_i(t_{i,1}, \ldots, t_{i,x_i}) \\
\text{sujeito à} \quad & \sum_{i \in R} x_i = \lfloor C \times T \rfloor \\
& |t_{p,j} - t_{q,k}| \geq P, \quad \text{para } (p,q) \in S, \quad j = 1, \ldots, x_p \quad k = 1, \ldots, x_q \\
& x_i \in \mathbb{N}, \quad \text{para toda página } i \in R \\
& 0 \leq t_{i,j} \leq T, \quad \text{para toda página } i \in R, \quad j = 1, \ldots, x_i
\end{aligned}$$

onde R é o conjunto de páginas do repositório, w_i é a importância da página i, C é a frequência total de revisitações do *crawler*, P é o tempo mínimo entre requisições consecutivas a um servidor, S contém todos os pares de páginas que pertencem a um mesmo servidor, T é a duração da operação do *crawler*, e x_i é o número de revisitações à página i no intervalo de operação do *crawler*.

2.1.4 Obsolescência de Consultas a um Servidor

No modelo proposto em (Eck08), cada requisição ao servidor permite obter as informações necessárias para atualizar as páginas deste servidor. As modificações de uma página do servidor são modeladas como um processo de Poisson não homogêneo (Ros07) com taxa $\lambda(.)$. A chegada de consultas à máquina de busca que provocam o retorno desta página também é modelada como um processo de Poisson não homogêneo com taxa $a(.)$. O modelo considera um intervalo finito de operação do *crawler* com duração T, e um conjunto de requisições a um servidor neste intervalo ocorrendo nos instantes $0 \leq u_1 \leq \cdots \leq u_{n-1} \leq T$.

A *obsolescência de uma consulta* que ocorre no instante q, entre os instantes de revisitação u_{i-1} e u_i, é definida como o número esperado de modificações ocorridas nas páginas do servidor no intervalo $(u_{i-1}, q]$, que é dado por

$$\Lambda(u_{i-1}, q) = \int_{u_{i-1}}^{q} \lambda(t)\ dt.$$

A probabilidade de uma consulta ocorrer no instante $u_{i-1} < q \leq u_i$ é dada por $a(q)/A(u_{i-1}, u_i)$, onde $A(u_{i-1}, u_i) = \int_{u_{i-1}}^{u_i} a(t)\ dt$ é o número esperado de consultas no intervalo $(u_{i-1}, u_i]$. Assim, a obsolescência esperada de uma consulta que ocorre neste intervalo vale

$$c(u_{i-1}, u_i) = \int_{u_{i-1}}^{u_i} \Lambda(u_{i-1}, q) \frac{a(q)}{A(u_{i-1}, u_i)}\ dq.$$

Denotando por $Q(s, f)$ a variável aleatória que representa o número de consultas ao servidor que ocorrem no intervalo $(s, f]$, e utilizando o fato de que as consultas são independentes, temos a seguinte obsolescência esperada para as consultas que ocorrem entre as requisições u_{i-1} e u_i:

$$\begin{aligned} C(u_{i-1}, u_i) &= \sum_{k=0}^{\infty} \Pr[Q(u_{i-1}, u_i) = k]\ k\ c(u_{i-1}, u_i) = A(u_{i-1}, u_i)\ c(u_{i-1}, u_i) = \\ &= \int_{u_{i-1}}^{u_i} \Lambda(u_{i-1}, q)\ a(q)\ dq = \int_{u_{i-1}}^{u_i} a(q) \int_{u_{i-1}}^{q} \lambda(t)\ dt\ dq = \\ &= \int_{u_{i-1}}^{u_i} \left(\lambda(t) \int_{t}^{u_i} a(q)\ dq \right) dt. \end{aligned}$$

Este modelo permite considerar variações no tempo da quantidade de modificações sofridas pelas páginas de cada servidor, bem como variações no interesse por consultas aos servidores. Para servidores com clientes espalhados em vários continentes, a variação no número de consultas tende a ser menos

significativa. Por outro lado, a manutenção das páginas de um servidor tendem a ocorrer com mais intensidade no horário comercial de um determinado país. Neste sentido, este modelo parece mais flexível que simplesmente assumir uma taxa fixa de modificação. Infelizmente este modelo não pode ser diretamente aplicado ao problema de revisitação de páginas *Web* com servidores hospedando mais de uma página, pois não é possível atualizar todas as páginas do servidor com uma única requisição.

2.1.5 Outras Medidas de Atualização do Repositório

Algumas medidas de desatualização do repositório buscam estabelecer com mais detalhes o impacto da desatualização das páginas sobre as consultas à máquina de busca. Wolf et al (Wol02) propõem que no cálculo do *staleness* do repositório (Seção 2.1.3) pesos sejam atribuídos às páginas em função da probabilidade da página ser retornada, receber *click*, e não estar de acordo com os termos utilizados na consulta. O cálculo do *staleness* utilizando estes pesos é chamado *embarrassment*. Wolf et al (Wol02) argumentam que a chance de *click* é função da posição (*rank*) da página no resultado retornado pela máquina de busca, e do número de *links* em cada página de resposta da máquina de busca. Olston e Pandey (Ols08) propõem separar nas páginas as partes que se modificam muito das que se modificam pouco, visto que as partes que se modificam muito já estarão obsoletas quando a página for indexada pela máquina de busca. Portanto, a política de revisitação deveria levar em conta o padrão de modificação das partes que se modificam pouco. Esta estratégia é chamada *longevity*.

2.2 Restrição de Politeness

Embora os *crawlers* sejam importantes para diversas tarefas, como a manutenção das máquinas de busca, sua utilização acarreta alguns problemas para a Internet (Kos95). As pessoas/empresas que mantêm sites na Internet desejam o acesso de clientes reais, portanto um tráfego excessivo produzido por *crawlers* é considerado indesejado quando reduz a eficiência das respostas à estes clientes. Os dois principais prejuízos provocados pelos *crawlers* são (i) aumento do tráfego na rede, aumentando assim o tempo para transmitir as mensagens (requisições dos clientes e respostas dos servidores), e (ii) sobrecarga de servidores *Web* devido à capacidade limitada (CPU, memória) de atender as requisições.

O protocolo *robot exclusion* (Kos96) é uma tentativa de diminuir a

sobrecarga de servidores. Este protocolo permite que o mantenedor do site determine quais páginas podem ser acessadas por *crawlers*. Entretanto, ele não evita que os *crawlers* acessem com uma taxa elevada as páginas permitidas.

Uma taxa elevada de requisições a um mesmo servidor pode ser interpretada como um ataque do tipo *denial of service* (DOS10). Este tipo de ataque pode prejudicar a capacidade de resposta às requisições dos clientes, afetando o servidor *Web*, o *firewall* e o roteador/*switch*. Como o roteador é o ponto de entrada da rede local do mantenedor, geralmente o tratamento deste tipo de ataque é feito neste ponto (roteador) com a utilização de uma lista de clientes (endereços IP) bloqueados devido à alta taxa de requisições. Como a duração do intervalo de tempo utilizado para medir a taxa de requisições de cada cliente é variável, a estratégia mais segura para evitar o bloqueio do *crawler* é manter um tempo mínimo entre requisições a um mesmo servidor (restrição de *politeness*, Definição 1.2).

A primeira proposta para o tempo mínimo entre requisições a um servidor foi de 60 segundos (Kos93). Neste caso, levaria mais de 2 meses para revisitar todas as páginas de um servidor com 100,000 páginas. Algumas outras definições deste tempo encontradas na literatura são 10 segundos (Cho03a) e 15 segundos (*crawler* WIRE, (Bae02)). O *crawler* Mercator (Hey99) define o tempo mínimo entre requisições a um servidor como 10 vezes o tempo de *download* da última página revisitada neste servidor. Castilho reporta em (Cas04a) que de acordo com o *log de acesso* de alguns servidores *Web*, o tempo entre requisições adotado pelo *crawlers* conhecidos varia entre 20 segundos e 3-4 minutos. Nesta pesquisa adotamos um tempo mínimo de 15 segundos entre requisições a um mesmo servidor, que também é o tempo mínimo adotado pelo *crawler* WIRE.

2.3 Políticas de Revisitação

2.3.1 Cho e Garcia-Molina

Cho e Garcia-Molina (Cho03a) investigaram políticas de revisitação para o caso em que temos uma requisição por página em cada ciclo com duração I, ou seja, todas as páginas são revisitadas com a mesma taxa. As requisições são igualmente espaçadas dentro do ciclo, e estes ciclos se repetem indefinidamente. A ordem com que as páginas são revisitadas em cada ciclo é de acordo com uma das políticas abaixo:

Fixed-Order As páginas são revisitadas sempre na mesma ordem. Ou seja,

os intervalos entre revisitações de uma página têm a mesma duração I. Neste caso, a taxa de revisitações de qualquer página vale $f = 1/I$.

Random-Order A ordem é determinada por uma permutação aleatória em cada ciclo. Portanto, as durações dos intervalos entre revisitações de uma página pertencem ao intervalo $[0, 2I]$. Como todas as página recebem uma revisitação em cada ciclo, a taxa de revisitação de qualquer página vale $f = 1/I$.

Purely-Random Em cada ponto de revisitação uma página é escolhida aleatoriamente. Neste caso, algumas páginas podem não ser revisitadas em um ciclo, permitindo que a duração do intervalo entre revisitação seja arbitrariamente longo. A chance de uma página ser escolhida é mantida de tal forma que a taxa de revisitação de cada página seja $f = 1/I$.

Cho e Garcia-Molina (Cho03a) provaram expressões para o *freshness* e o *age* de uma página quando aplicamos cada uma destas políticas. Destacamos a seguir duas destas expressões que são utilizadas ao longo do texto.

Teorema 2.1 *(Cho03a) Uma página que se modifica de acordo com um processo de Poisson com taxa λ, e é revisitada com taxa f de acordo com a política* fixed-order*, tem* freshness $(1 - \exp(-r))/r$*, onde $r = \lambda/f$.*

Teorema 2.2 *(Cho03a) Uma página que se modifica de acordo com um processo de Poisson com taxa λ, e é revisitada com taxa f de acordo com a política* purely-random*, tem* freshness $1/(1 + r)$*, onde $r = \lambda/f$.*

As expressões para o *freshness* e o *age* de uma página obtidos com as políticas *fixed-order* e *purely-random* podem ser generalizadas para taxas de revisitação não uniformes. Podemos obter o mesmo nível de atualização que a política *fixed-order* desde que as revisitações a uma página permaneçam igualmente espaçadas. Para a política *purely-random*, basta que a chance de uma página ser escolhida seja proporcional à sua frequência de revisitação.

Com esta generalização para taxas não uniformes de revisitação, Cho e Garcia-Molina (Cho03a) apresentam uma alocação de recursos ótima e eficiente utilizando o método dos *multiplicadores de Lagrange* (Bor02). Para o caso da política *fixed-order*, o problema consiste em encontrar a frequência ótima f_i de revisitação de cada página i com base na formulação

$$\begin{aligned} \text{maximize} \quad & \sum_{i \in R} w_i (1 - \exp(-\lambda_i / f_i)) f_i / \lambda_i \\ \text{sujeito à} \quad & \sum_{i \in R} f_i = C, \\ & f_i \geq 0, \text{ para toda página } i, \end{aligned}$$

onde R é o conjunto de páginas no repositório, w_i é a importância da página i, C é a frequência total de revisitações realizadas pela *crawler*, λ_i é a taxa de modificação da página i. Realizando a relaxação lagrangiana da restrição $\sum_{i \in R} f_i = C$ utilizando o multiplicador μ, obtemos

$$f_i = -\frac{\lambda_i}{1 + W_{-1}(\mu \lambda_i e^{-1}/w_i - 1)},$$

onde $W_{-1}(.)$ é a função Lambert W no *branch* -1 (Cor93). Como f_i é obtido por uma função monótona decrescente de μ, temos várias opções de métodos numéricos para determinar o valor de μ tal que $\sum_{i \in R} f_i = C$. Se esta busca do valor de μ é feita em $k << n$ iterações, onde n é o número de páginas no repositório, então a alocação de recursos pode ser obtida em tempo $O(n)$. O número de iterações k depende da precisão desejada.

Quando as taxas de revisitação são uniformes, as políticas *fixed-order*, *random-order* e *purely-random* respeitam a restrição de *politeness* (Definição 1.2) desde que I/n seja maior que o tempo mínimo permitido entre requisições a um servidor, onde n é o número de páginas hospedadas neste servidor. Para taxas de revisitação não uniformes, a política *purely-random* respeita a restrição de *politeness*, mas a política *fixed-order* pode violar, conforme discutido na Seção 4.1.1.

2.3.2 Wolf et al

Quando dispomos de x_i revisitações para a página i em um intervalo de tempo com duração T, Wolf et al (Wol02) provam que a maneira de distribuir estas revisitações de modo a minimizar o *staleness* é mantendo-as igualmente espaçadas, ou seja, $t_{i,j+1} - t_{i,j} = T/(x_i + 1)$. Porém, conforme discutido na Seção 4.1.1, uma política com revisitações igualmente espaçadas por página pode violar a restrição de *politeness* (Definição 1.2).

A alocação de recurso para este modelo com intervalo finito de operação consiste em determinar o número de revisitações x_i de cada página i de modo a minimizar o *staleness* do repositório, onde a soma dos x_i's deve ser no máximo o total de revisitações que o *crawler* é capaz de fazer neste intervalo. Utilizando o fato de que a forma ótima de distribuir as revisitações de uma página é mantendo-as igualmente espaçadas, Wolf et al (Wol02) mostram que esta alocação recursos pode ser obtida em tempo $O(\max\{N, N \log(R/N)\})$, onde N é o número de páginas e R é o total de revisitações no intervalo de operação do *crawler*.

Quando as revisitações de uma página são igualmente espaçadas, e a

página se modifica segundo um processo de Poisson, temos que o *staleness* vale 1 menos o *freshness* da página. Neste caso, quando a página i se modifica com taxa λ_i, e dispõe de x_i revisitações no intervalo T de operação do *crawler*, então o staleness da página i é dado por (Wol02)

$$S_i = 1 + \frac{x_i + 1}{\lambda_i T}\left(\exp\left(-\frac{\lambda_i T}{x_i + 1}\right) - 1\right).$$

Definindo a frequência de revisitação f_i da página i como $(x_i + 1)/T$, temos

$$1 - S_i = \frac{1 - \exp(-\lambda_i/f_i)}{\lambda_i/f_i},$$

que é a expressão para o *freshness* da política *fixed-order* (igualmente espaçada por página) proposta em (Cho03a).

Wolf et al (Wol02) consideram também o problema de alocar cada revisitação programada a um *crawler* livre. Ou seja, eles assumem que a máquina de busca tem vários *crawlers* executando em paralelo, onde cada um deles pode realizar apenas uma revisitação por vez. Desta forma, as revisitações programadas competem por *crawlers* livres, e um preço deve ser pago quando algumas revisitações não podem ser realizadas no instante programado. A restrição de *politeness* pode ser tratada com esta formulação, se assumirmos um *crawler* por servidor, e cada *crawler* está livre apenas após o tempo mínimo permitido entre requisições a um servidor. A formulação proposta em (Wol02) pode ser vista como um *problema de transporte* (Ahu93). Os nós fonte representam os instantes de cada revisitação definidos pela política igualmente espaçada por página e pela alocação de recursos, e cada nó de demanda corresponde a um instante de revisitação que não viola a restrição de *politeness* (ou seja, os instantes representados pelos nós de demanda são separados por uma duração mínima). O custo de cada aresta é a diferença entre os tempos representados pelos nós fonte e de demanda ligados pela aresta. Observe que a solução desta formulação é aproximada, pois o custo da aresta não corresponde ao ganho de *staleness* quando ajustamos a revisitação representada pelo nó fonte para que ocorra no instante representado pelo nó de demanda. A dificuldade em utilizar o *staleness* como custo nas arestas é o fato desta métrica depender dos instantes das revisitações adjacentes da página. Outro problema desta abordagem é o fato de que para resolver uma instância real teríamos um grafo com um número elevado de nós e arestas, fazendo com que os algoritmos disponíveis para o problema de transporte tenham um tempo de execução muito alto.

2.3.3 Eckstein et al

Eckstein et al (Eck08) desenvolveram uma política de revisitação que respeita a restrição de *politeness* (Definição 1.2), para o modelo de *obsolescência* de consultas a um servidor (Seção 2.1.4). Esta política determina $N > n$ instantes dentro do intervalo de operação onde requisições podem ser feitas ao servidor, onde n é o número de requisições ao servidor programadas para este intervalo. Ou seja, a política realiza uma discretização do tempo indicando os possíveis instantes de revisitação. Desta forma, o problema pode ser modelado como

$$\begin{aligned} \text{minimize} \quad & \sum_{i=1}^{n} C(u_{i-1}, u_i) \\ \text{sujeito à} \quad & u_i - u_{i-1} \geq P, \quad i = 1, \ldots, n \\ & u_0 = 0, \quad u_n = T \\ & u_1, \ldots, u_{n-1} \in \mathcal{U}, \end{aligned}$$

onde n é o número de requisições ao servidor no intervalo de operação com duração T, P é o tempo mínimo entre requisições consecutivas ao servidor, e $\mathcal{U}$ é o conjunto de instantes de revisitação permitidos no intervalo $[0, T]$ (discretização).

A solução do problema discretizado é obtida em (Eck08) através de uma programação dinâmica com tempo $O(nN^2)$, cuja solução se afasta no máximo $O(1/N)$ da solução ótima (quando a discretização utiliza instantes igualmente espaçados). Eckstein et al (Eck08) propõem também a aplicação de uma busca local partindo da solução fornecida pela programação dinâmica, para melhorar esta solução. Quando aplicando esta busca local após a execução das políticas, os experimentos realizados em (Eck08) mostram que outras políticas mais simples (como igualmente espaçada por servidor) fornecem praticamente a mesma *obsolescência* obtida pela programação dinâmica, embora a programação dinâmica tenha garantia teórica de qualidade ($O(1/N)$).

Uma desvantagem da aplicação desta abordagem para o problema de revisitação de páginas *Web* é o elevado tempo de execução, que é em média $O(n^2)$ por requisição no melhor caso (quando $C(u_{i-1}, u_i)$ é computado em $O(1)$ e N é da ordem do número de requisições). Além disso, este modelo assume que toda informação sobre o servidor é obtida em uma requisição, e portanto precisa ser adaptado para o problema de revisitação de páginas *Web*. Algumas idéias de como adaptar este modelo são apresentadas em linhas gerais em

(Eck08).

2.3.4 Coffman et al

Coffman et al (Cof98) investigam políticas onde a duração entre pontos de revisitação é determinada por observações independentes de uma variável aleatória X, e uma página é escolhida em cada ponto de revisitação. Como consideramos nesta pesquisa apenas políticas onde as revisitações são igualmente espaçadas por página ou por servidor, neste sentido o modelo de (Cof98) é mais geral. Entretanto, Coffman et al (Cof98) assumem que a importância de cada página é proporcional à sua taxa de modificação, o que não é assumido em nossa análise. Além disso, se a variável aleatória X puder assumir valores menores que o tempo mínimo permitido entre requisições a um servidor, então a escolha consecutiva de duas páginas de um mesmo servidor pode violar a restrição de *politeness* (Definição 1.2). Nenhuma análise é feita em (Cof98) para o impacto da restrição de *politeness*.

De forma similar ao *freshness* (Seção 2.1.1), no modelo de (Cof98) as políticas são avaliadas de acordo com a média da proporção do tempo que cada página permanece desatualizada, ponderada pelas importâncias das páginas. Para isso, assume-se que as páginas se modificam segundo um processo de Poisson. Coffman et al (Cof98) provam que para minimizar o tempo que uma página permanece desatualizada, devemos manter suas revisitações o mais igualmente espaçadas possível. Uma demonstração deste fato, simples e aplicada ao nosso modelo, também é apresentada na Seção 4.1.1.

Coffman et al (Cof98) avaliam uma política randomizada que sorteia em cada ponto de revisitação uma página com probabilidade proporcional à sua frequência de revisitação. Eles fornecem solução analítica para as frequências ótimas de revisitação das páginas. Embora esta política seja uma generalização da política RANDOM estudada na Seção 5.2, estamos interessados em como esta política se comporta quando temos a restrição de *politeness*. Além disso, a política RANDOM serve como *baseline* para as políticas igualmente espaçadas por servidor, devido à sua simplicidade.

2.3.5 Edwards et al

Edwards et al (Edw01) propõem uma formulação baseada em programação não linear para a construção do escalonamento de revisitações em um intervalo finito de operação do *crawler*. De forma similar ao *freshness*, o objetivo é minimizar a média ponderada (no tempo) do número de páginas

obsoletas. Infelizmente, Edwards et al (Edw01) não apresentam uma forma eficiente para obter a solução desta formulação.

2.4 Modelo Adotado nesta Tese

Adotamos o *freshness* como medida de atualização do repositório, conforme definido na Seção 2.1.1. Portanto, as políticas são avaliadas considerando operação do *crawler* com duração infinita, e assumimos que as páginas são modificadas de acordo com um processo de Poisson. Ou seja, desejamos saber o comportamento assintótico do nível de atualização do repositório quando aplicamos estas políticas. Desta forma, não podemos representar explicitamente no modelo os instantes exatos de revisitação de cada página, e sim definir políticas de tempo que determinem o instante da próxima revisitação de cada página. Como a restrição de *politeness* é apresentada na Definição 1.2 em função dos instantes de requisições aos servidores, para evitar representar estes instantes explicitamente fixamos políticas de tempo que respeitam a restrição de *politeness*. Além da restrição de *politeness*, temos também a restrição de taxa de revisitação apresentada na Definição 1.1. Portanto, a alocação de recursos pode ser formulada como

$$\begin{aligned} \text{maximize} \quad & \sum_{i \in R} w_i A_i(f_i) \\ \text{sujeito à} \quad & \sum_{i \in R} f_i \leq C, \\ & f_i \geq 0, \text{ para toda página } i, \end{aligned}$$

onde $A_i(f_i)$ é o *freshness* da página i quando esta dispõe de uma frequência de revisitação f_i, R é o conjunto de páginas no repositório, w_i é a importância da página i, e C é a frequência total de revisitações realizadas pelo *crawler*. A restrição de *politeness* está implícita na política de tempo adotada.