大規模 Web アーカイブ更新クローラにおける Web サーバアクセススケジューリング手法

田村　孝之[†,††]　　喜連川　優[††]

† 三菱電機株式会社 情報技術総合研究所
〒 247–8501 鎌倉市大船 5–1–1
†† 東京大学生産技術研究所
〒 153–8505 東京都目黒区駒場 4–6–1
E-mail: ttamura@acm.org, kitsure@tkl.iis.u-tokyo.ac.jp

あらまし　筆者らは刻々と変化する Web 情報からの社会知の抽出を目指し，日本語 Web ページを中心とする大規模 Web アーカイブの構築を行っている．Web の大域的構造や時間変化の分析に求められる高い網羅性と時間分解能を達成するため，各 Web ページの収集を独立のタイミングで繰り返す更新クローラの開発を進めてきた．網羅的なクローリングにおいては，性能向上のため多数の Web サーバに並行してアクセスする必要があり，有限の通信リソースの配分が問題となる．更新クローラではさらに，各 Web ページのダウンロードをそれぞれに固有の周期で繰り返すため，タイミングの制約を考慮する必要が生じる．本稿では，Web サーバへの通信リソース割り当てをリアルタイムスケジューリングの問題と捉え，過去の実績から算出した Web サーバ毎の平均ダウンロード時間に基づく実現方法を提案するとともに，実際のクロールログを用いてその効果を示す．

キーワード　Web アーカイブ，Web クローラ，更新クローリング，通信リソース割り当て，リアルタイムスケジューリング

A method for scheduling Web server accesses in an incremental crawler for large scale Web archives

Takayuki TAMURA[†,††] and Masaru KITSUREGAWA[††]

† Information Technology R&D Center, Mitsubishi Electric Corporation
5–1–1 Ofuna, Kamakura-shi, 247–8501 Japan
†† Institute of Industrial Science, The University of Tokyo
4–6–1 Komaba, Meguro-ku, Tokyo, 153-8505 Japan
E-mail: ttamura@acm.org, kitsure@tkl.iis.u-tokyo.ac.jp

Abstract　We are building a large scale archive of Japanese-centric Web pages for exploiting social knowledge from continuously changing Web. To meet the demands for both exhaustiveness and high time-resolution from structural and temporal analyses of the Web, we have developed an incremental crawler, which revisits Web pages adaptively to their estimated change frequencies. In exhaustive crawling, we need to solve the problem of allocating finite communication resources to concurrent connections to a lot of Web servers. Incremental crawling, moreover, requires each Web page to be downloaded according to its change interval, we have to take into account timing constraints. This paper proposes a method, as a real time scheduling technique, for allocating communication resources to Web servers using their average download times derived from the past behavior. Its effectiveness is convinced using actural crawl logs.

Key words　Web archiving, Web crawler, Incremental crawling, Communication resource allocation, Real-time scheduling

1. はじめに

筆者らは刻々と変化する膨大な Web 情報を実社会の鏡像と見做し，その大域的な構造や時間変化の分析に基づく社会知の抽出に取り組んでいる [1], [2]．Web 分析の基盤となるのは日本の Web 情報を網羅的に収集・蓄積した大規模な Web アーカイブである．Web の大域的構造と時間変化の分析を主眼とする Web アーカイブに要求される網羅性と時間分解能の両立を図るには，Web ページを 1 回ずつ収集する一括クローラ (batch crawler) ではなく，連続的に動作し Web ページ毎に独立したタイミングで再アクセスを行う更新クローラ (incremental crawler) が不可欠である．更新クローラは Web ページ毎の更新頻度に適応したスケジューリングを行い，低更新頻度 Web ページのアクセスに割かれていたリソースを高更新頻度 Web ページのアクセスに振り向けることで，最短 1 日程度の高い時間分解能を得ることを可能にする．

一方，Web ページ再アクセスをスケジュール通り実施するには，Web サーバとの通信に必要なリソースを確保できることが前提となる．リソース不足は，一括クローラにスループット低下をもたらすが，更新クローラでは Web ページアクセスにタイミング制約が伴うため，その影響はより甚大となる．リアルタイムスケジューリングはタイミング制約を考慮したリソース割り当て手法であり，タスクへのリソース（CPU）割り当てをタスクの実行周期と実行時間（CPU 占有時間）に基づいて行う．ここで，タスクを更新クローラによる周期的 Web ページ（Web サーバ）アクセスと読み換えると，アクセス周期とダウンロード経過時間（通信リソース占有時間）に基づくタスクへの通信リソース割り当てに応用することができる．ただし，ダウンロード経過時間は過去の実績に基づいて推定するものとする．

以下，2. で関連研究について述べる．3. では更新クローラの動作概要を説明し，4. で Web ページ周期アクセスのための Web サーバアクセススケジューリング手法について述べる．5. では実際のクローリングから得られた統計値を元に本手法適用の効果を見積もり，6. で全体のまとめを行う．

2. 関連研究

本稿では，4. で述べるように，Web サーバ単位でのスケジューリングを重視し，アクセスマナーの遵守や過負荷状態の検出を可能にしているが，クローラに関する文献でこれらに言及したものは少ない．WebFountain クローラ [3] は各 Web ページの更新確率やページ収集後の経過時間などを変数とする非線形連立方程式を一定期間毎に解いて収集対象 Web ページを決定する，更新クローラである．クローラリソースの制約は考慮されているものの，Web サーバ単位での制御は行っていない．また，恒常的なリソース不足を検出し，アクセス戦略（時間分解能，収集範囲等）を定量的に修正することもできない．Cho ら [4], [5] はクローラリソースが不足する場合に，Web ページの更新頻度に基づいてアクセスの優先度を決定する方法を論じており，特に，更新頻度が非常に高い Web ページは取得した内容が最新である期間も短いため，他の Web ページの収集を優先した方がスナップショット全体の新鮮度が向上するという結論を導いている．しかし，この議論は最新情報の検索が目的であることを前提としたものであり，過去の任意の時点における分析を目的とするアーカイブ用途には必ずしも当てはまらない．User-centric クローリング [6] はリソース負荷を低減するために検索ニーズに応じて Web ページに重み付けするものであるが，収集対象に偏りを生じることになるためやはりアーカイブ用途には適さない．

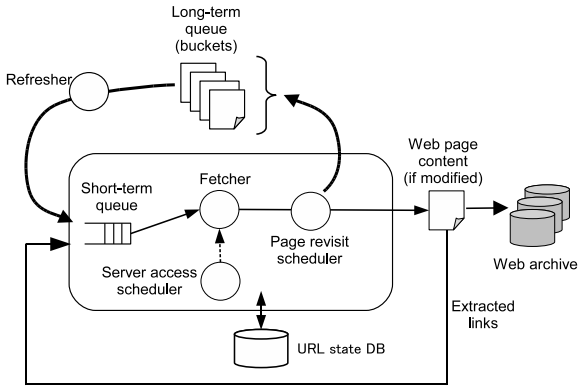

図 1　更新クローラの構成.
Fig. 1　Architecture of incremental crawler.

リアルタイムスケジューリングにおいては，多くの研究がプリエンプティブな環境を想定してなされているが [7]，更新クローラはあるページのダウンロード中に別のページのダウンロードを割り込ませることはないため，非プリエンプティブな手法が重要である．動的なスケジューリングアルゴリズムとして非プリエンプティブ版の EDF (Earliest Deadline First) が最適であることが知られている [8]．これは，リソースに空きが生じた時点で最も近いデッドラインを持つタスクを選択するというものである．

3. 更新クローラの動作モデル

図 1 に本稿で扱う大規模 Web アーカイブ構築用更新クローラの構成を示す．筆者らが実装した更新クローラはクラスタ上で並列動作するが，既知 Web ページの再アクセスは各ノードが独立に行うため他ノードとの通信を意識する必要はない．

更新クローラは以下のように動作する．

（1） Server access scheduler は Web サーバ毎に一定の時間間隔で short-term queue から URL を取得し，通信リソース（ソケット）を割り当てた後に fetcher を起動する．

（2） Fetcher は受け取った URL から HTTP 要求メッセージを作成し，Web サーバに送信する．

（3） Fetcher は Web サーバから応答を受け取ると，そのコンテンツを Web アーカイブに格納し，さらに，コンテンツが HTML であればリンク抽出処理を起動する．

（4） Fetcher は HTTP 応答ヘッダから抽出した情報やコンテンツハッシュ値，現在時刻等を用い，URL state DB の当該 URL に関するレコードを更新する．

（5） Page revisit scheduler は URL state DB の情報に基づき，当該 URL の次回アクセス時刻を決定する．さらに，URL を short-term queue から取り除き，long-term queue に挿入する．

（6） Refresher は時間分解能毎に起動され，long-term queue の最も早い時刻に対応するバケットファイルを取り出し，その中の各 URL を short-term queue に挿入する．

4. Web サーバアクセススケジューリング

4.1 Web サーバアクセス間隔の導出

一般にクローリングにおいては同一 Web サーバから同時に複数の Web ページを取得することはマナーに反するとされる．また，Web サーバへのアクセスを逐次に行ったとしても，連続するアクセスの間隔が短いと Web サーバが高負荷となるので好ましくない．ここで，Web サーバ i に属する全ての Web ページの集合を $\{P_{ij}\}$ とし，P_{ij} の再アクセス間隔を I_{ij}（P_{ij} の更新傾向から推定），十分長い期間 τ における P_{ij} のアクセス回数を N_{ij} とする．Web サーバ i へのアクセスを時間 I_i 毎に逐次的に行うという制約の下で各ページの取得が可能となるには，

$$I_i \sum_j N_{ij} \leqq \tau \tag{1}$$

が成り立つ必要がある．$N_{ij} = \lfloor \tau / I_{ij} \rfloor \simeq \tau / I_{ij}$ により，式 (1) から以下が得られる．

$$I_i \leqq \frac{\tau}{\sum_j N_{ij}} \simeq \left(\sum_j \frac{1}{I_{ij}} \right)^{-1} \equiv {L_i}^{-1} \tag{2}$$

L_i は Web サーバ i に属する Web ページのアクセス頻度の総和であり，Web サーバ i の負荷を表す指標と考えられる．その逆数 ${L_i}^{-1}$ は Web ページ再アクセススケジューリングに従ったアクセスに必要な Web サーバアクセス間隔の理論的な上限となっている．Web サーバに属する Web ページ数の増加や各 Web ページの再アクセス間隔の短縮により Web サーバの負荷指標が大きくなると，より小さな間隔で Web サーバにアクセスする必要が生じる．

式 (2) を満たすことができないのは以下の 2 つの場合である．

（1） マナーへの配慮から設けた下限値 $I_{\min}$ に対し，${L_i}^{-1} < I_{\min}$ となる場合．

（2） Web ページの取得に実際に要した経過時間 e_i に対し，${L_i}^{-1} < d \cdot e_i (d > 0)$ となる場合．(Najork ら [9] は，Web サーバ管理者からのクレームを避けるための経験則として，ページ取得時間の 10 倍の間隔を空けて Web サーバをアクセスするというルールを述べている．)

これらを考慮した Web サーバアクセス間隔は次式で与えられる．

$$I_i = \max({L_i}^{-1}, I_{\min}, d \cdot e_i) \tag{3}$$

$I_i > {L_i}^{-1}$ となる Web サーバ i は過負荷状態にある．

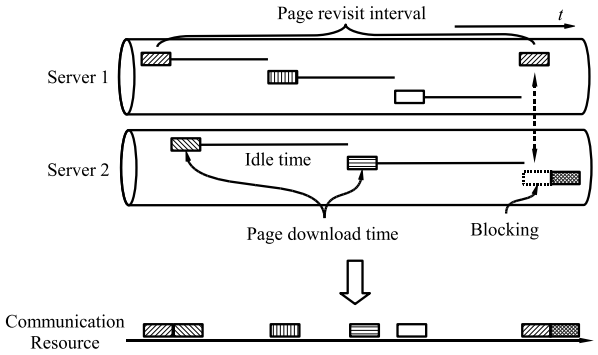

図 2 Web ページアクセス，Web サーバアクセス，および通信リソースの関係．

Fig. 2 Relationship among Web page access, Web server access, and Communication Resource.

4.2 通信リソース割り当て方式

図 2 に Web ページアクセス，Web サーバアクセス，および通信リソースの関係を示す．Web ページへのアクセスは Web サーバ毎に直列化されるが，ほとんどの場合ダウンロード時間はアクセス間隔と比べて小さいので，複数 Web サーバを同一通信リソース上に束ねることができる．Web サーバの通信リソースへの割り当ては，EDF により動的に行うこともできるが，スケジューリング動作が頻繁に発生するのを避けるため，ここでは静的な割り当てを行うことにする．

Web サーバ i へのアクセス間隔を I_i，Web サーバ i から Web ページを取得する際の経過時間を e_i とすると，$I_i - e_i$ の期間は通信リソース数を増加させることなく他の Web サーバへのアクセスを行うことができる．すなわち，同一のアクセス間隔 I_k を持つ Web サーバの集合 S_k において，

$$\sum_{i \in S_k} e_i \leqq I_k \tag{4}$$

が成り立つ場合，S_k に必要な通信リソース数は 1 である．

任意の Web サーバの集合 S_k においては，各 Web サーバへのアクセス間隔を同一の値 $I_k = \min_{i \in S_k} I_i$ で置き換えることにする．ここで，S_k が $I_i = I_{\min}$ となる Web サーバ i を含む場合，任意の Web サーバ j に対して $I_{\min} \leqq I_j$ であるから I_i がさらに短縮されることはない．また，S_k が $I_i = e_i$ となる Web サーバ i を含む場合，$|S_k| = 1$ であり I_i は変化しない．従って，アクセス間隔が短縮されるのは過負荷状態にない Web サーバのみであり，マナー上の制約や物理的制約に違反することはない．

この時，S_k に必要な通信リソース数が 1 である条件は次式で与えられる．

$$\sum_{i \in S_k} e_i \leqq \min_{i \in S_k} I_i \tag{5}$$

ただし，Web サーバへのアクセス間隔を大幅に短縮するのは負荷制御の点で好ましくないため，S_k の要素に対して以下の条件を追加する．

$$\frac{\max_{i \in S_k} I_i}{\min_{i \in S_k} I_i} \leqq 1 + \delta \quad (\delta > 0) \tag{6}$$

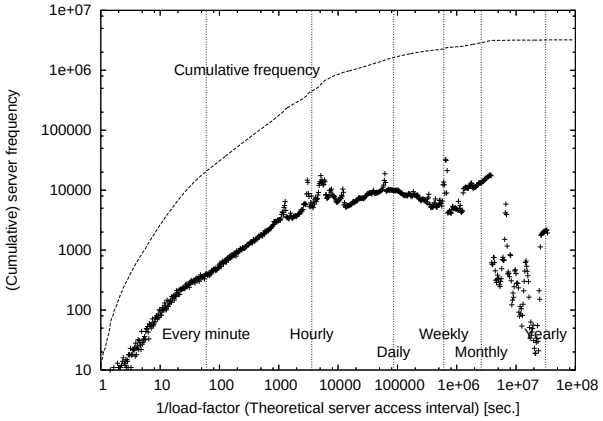

図 3　Web サーバアクセス間隔理論値の分布.
Fig. 3　Frequency of theoretical web server access intervals.

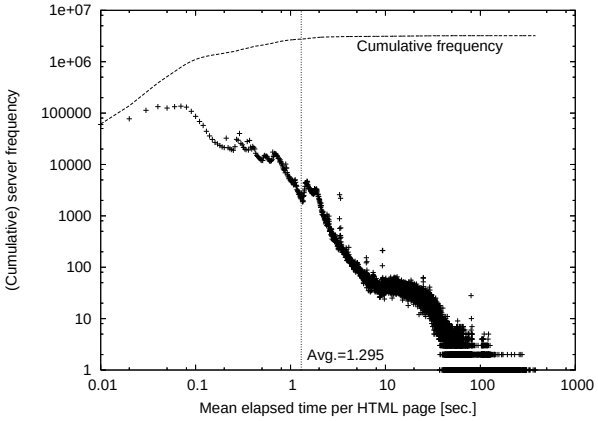

図 4　Web サーバ毎の平均ページ取得時間の分布.
Fig. 4　Frequency of mean page-download time per server.

すなわち，S_k 内を均質な性能を持つ Web サーバで構成することで，動的なスケジューリングなしにラウンドロビンによる各 Web サーバへのアクセスを可能にする．

以上により，次式

$$\bigcup_{k=1}^{C} S_k = S, \quad S_k \cap S_{k'} = \emptyset \ \ (k \neq k') \tag{7}$$

で与えられるアクセス対象 Web サーバの集合 S の分割 S_k $(1 \leqq k \leqq C)$ がそれぞれ式（5）および式（6）を満たすとき，必要な通信リソース数は C となる．

一方，プリエンプティブなスケジューリングに対応する必要通信リソース数は，Web サーバ i の通信リソース占有率 e_i/I_i の合計であり，C の粗い見積もりを与える．

$$C \geqq \sum_{i \in S} \frac{e_i}{I_i} \tag{8}$$

5. 大規模クロールログによる効果の見積もり

ここでは更新クローラの過去の動作ログから得た統計値に基づいて，本 Web サーバアクセススケジューリング手法の適用効果を見積もる．本手法の更新クローラへの適用は，現在実装を進めている段階である．

図 3 は Web サーバ負荷指標（Web サーバに属するページの再アクセス間隔の逆数の総和）の逆数として与えられる Web サーバアクセス間隔理論値（I）の分布を示したものである．ただし，対象としたサーバは 2006 年 12 月の 12 日間にアクセスしたものとした．このようなサーバの数は 3,212,253，サーバ当たりの平均既知ページ数（12 日間にアクセスしていないものも含む）は 585.2 であった．図 3 によると非常に緩やかなアクセスで十分なサーバが多いことが分かる．1 分毎にアクセスする必要があるのは全体の 1% に満たない．しかしながら，数秒毎にアクセスする必要があるサーバも 100 程度存在している．これらのサーバには 10 万単位のページが属しており，全てのページを定期的に再アクセスしようとすると容易に過負荷となってしまう．そのため，実際のクローリングにおいては Web サーバアクセス間隔の下限は通常 1 分とし，1 万以上のページが属しているサーバについてのみ 5 秒とした．これは，多数のページを有する Web サーバは十分なリソースを持っているはずとの見込みに基づいている．

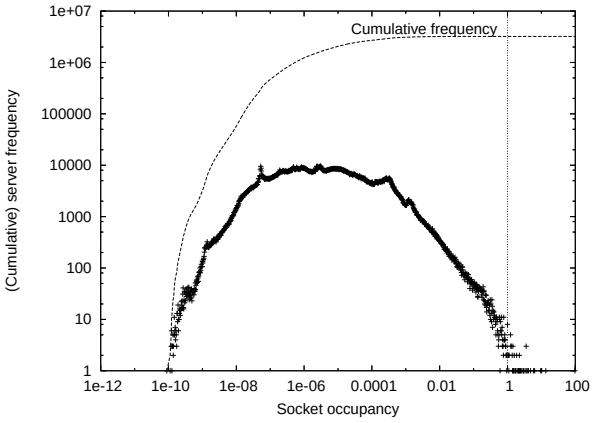

図 5　Web サーバ毎の通信リソース占有率の分布.
Fig. 5　Frequency of socket occupancy per server.

図 4 は Web サーバ毎に求めた平均ページ取得時間（e）の分布を示す．平均ページ取得時間が 1 秒未満のサーバが大半を占めていることが分かる．なお，単純平均ではページ取得時間は 0.845 [s]，Web ページサイズ（HTML のみ）は 21806 [byte] であった．

図 5 は，図 4 の平均ページ取得時間（e）と図 3 のサーバアクセス間隔理論値（I）の比（e/I）で定まる通信リソース占有率の分布を示したものである．ほとんどの Web サーバで通信リソース占有率は非常に小さな値を取っていることが分かる．また，わずかながら占有率が 1 を超えているものが存在している．これらのサーバからは，間断なくダウンロードを繰り返したとしてもコンテンツを収集しきれないことを示唆している．

図 6 は各 Web サーバの通信リソース占有率に基づき，通信リソースの消費量を概算した曲線（Estimated），および 4. の手法に基づいて実際に Web サーバ集合の分割を行った結果（Simulated）をそれぞれ示している．概算曲線については通信リソース占有率とその頻度の積を通信リソース占有率が小さい方から累積したものである．通信リソース占有率が 1 を超えている領域については，値は不正確であるが約 1,300 の通信リ

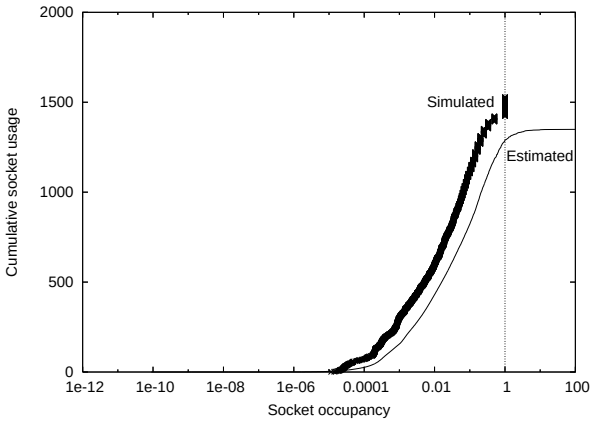

図 6　通信リソース消費量の見積もり.
Fig. 6　Estimation of socket usage.

ソースが必要という結果が得られている．一方，Simulated については，式（6）の制約により，200 ほど多い 1,500 の通信リソースを消費するという結果になっている．占有率 1 の部分で y 軸方向に増加しているのは，占有率が 1 より大きい Web サーバに通信リソースを 1 つずつ割り当てているからである．1,500 程度の通信リソースは数十のクロールノードを用いれば容易に実現でき，これによりごく一部の Web サーバを除いて通信リソース待ちのオーバヘッドなしに周期的なダウンロードを実施することが可能である．

6. ま と め

本稿では，大規模 Web アーカイブの構築のための更新クローラにおいて，適応的に決定した Web ページ周期アクセスのスケジューリングを確実に実施するための，Web サーバ単位のアクセススケジューリング手法について述べた．本手法では，Web サーバへのアクセス履歴から導出したアクセス間隔理論値と平均ダウンロード経過時間に基づいて特性の類似した Web サーバをグループ化し，各グループ内でラウンドロビンにアクセスを行うことによりリソース待ちの発生を防ぐ．

実クローリングのログを用いて本手法の効果を確認した結果，ごく一部の Web サーバにおいてリソース占有率が 1 を超えており，本質的に有効な情報収集手段がないことが明らかになったが，その他の 300 万以上の Web サーバについては，1,500 程度の通信リソース（同時接続数）で対応できることが分かった．

今後は実装を進めると共に，Web サーバの状態の変化に対応するための動的再スケジューリングの実現方法や効果について検討を進めて行きたい．

謝辞　本研究の一部は文部科学省リーディングプロジェクト e-Society 基盤ソフトウェアの総合開発「先進的なストレージ技術および Web 解析技術」による．

文　献

[1] 豊田, 喜連川：“日本におけるウェブコミュニティの発展過程”, 日本データベース学会 Letters, **2**, 1, pp. 35–38 (2003).
[2] M. Toyoda and M. Kitsuregawa: “What’s really new on the web?: identifying new pages from a series of unstable web snapshots”, WWW ’06: Proceedings of the 15th international conference on World Wide Web, New York, NY, USA, ACM Press, pp. 233–241 (2006).
[3] J. Edwards, K. S. McCurley and J. A. Tomlin: “An adaptive model for optimizing performance of an incremental web crawler.”, WWW, pp. 106–113 (2001).
[4] J. Cho and H. Garcia-Molina: “The evolution of the web and implications for an incremental crawler”, Proc. of VLDB, pp. 200–209 (2000).
[5] J. Cho and H. Garcia-Molina: “Effective page refresh policies for web crawlers.”, ACM Transactions on Database Systems (TODS), **28**, 4, pp. 390–426 (2003).
[6] S. Pandey and C. Olston: “User-centric Web crawling”, WWW ’05: Proceedings of the 14th international conference on World Wide Web, New York, NY, USA, ACM Press, pp. 401–411 (2005).
[7] C. L. Liu and J. W. Layland: “Scheduling algorithms for multiprogramming in a hard-real-time environment”, Journal of the ACM, **20**, 1, pp. 46–61 (1973).
[8] K. Jeffay, D. F. Stanat and C. U. Martel: “On non-preemptive scheduling of periodic and sporadic tasks”, Proceedings of IEEE Real-Time Systems Symposium (RTSS), pp. 129–139 (1991).
[9] M. Najork and J. L. Wiener: “Breadth-first crawling yields high-quality pages.”, WWW, pp. 114–118 (2001).