Web データの共起関係に基づくキャッシュ制御の研究

Cache Control of Web Data Based on Co-Occurrence

上山　憲昭（Noriaki KAMIYAMA, Ph. D.）
立命館大学情報理工学部　教授
(Professor, College of Information Science and Engineering, Ritsumeikan Universit)
IEEE　ACM　電子情報通信学会
受賞：ICOIN 2020 Best Paper Award (2020 年)　電子情報通信学会 論文賞 (2019 年)　WTC 2014 Best Paper Award (2014 年)　IFIP/IEEE IM 2013 Best Paper Award 賞（2013 年）　他
研究専門分野：情報ネットワーク　ネットワークセキュリティ

あらまし

Web ページの表示待ち時間を低減する技術として、Web ページを構成する複数のオブジェクトを同時並列に取得する HTTP/2 が標準化され広く普及している。しかし HTTP/2 の Web 表示待ち時間の低減効果を高めるには、少数の配信サーバから多数のオブジェクトを取得する必要がある。そこでオブジェクトの組に対し、その組が出現する Web ページの数を共起度と定義すると、共起度の高いオブジェクト組が優先的にキャッシュに残るよう、キャッシュ制御を行うことが望ましい。そこで本研究では共起度に基づくキャッシュ置換制御の可能性を明らかにするため、実際の多数の Web ページを測定し、Web ページの共起現象の程度を分析する。さらに共起を考慮したキャッシュ挿入・置換制御を行うことで、共起するオブジェクト組を構成するオブジェクトを、単一のキャッシュサーバに可能な限り集約する手法を提案する。

１．研究の目的

近年の Web ページは多数のデータオブジェクトから構成され、多数の配信サーバからオブジェクトを取得する。Web ページのリッチ化・複雑化が Web 応答時間増大の１つの要因となっている。Web 応答時間を低減するため HTTP/2 の標準化が進められ、正式な仕様として承認された。HTTP/2 は複数のオブジェクトを単一の TCP セッション上で並列に取得することで遅延時間を短縮するが、並列配信は同一の配信サーバから取得するオブジェクト集合に対してのみ可能である。そのため多数の配信サーバから、各々からは少数のオブジェクトのみを分散して取得する場合には、HTTP/2 の並列配信の効果は小さい。

また Web 応答時間を低減する技術として CDN（Content Delivery Network）が広く普及している。CDN はネットワーク上の多数の場所に設置されたキャッシュサーバ（CS）にオブジェクトのコピーをキャッシュし、ユーザの近くに存在する CS から配信することで Web 応答時間を低減する。CS は配信要求に対し、キャッシュに要求オブジェクトが存在しない場合、オリジンサーバからオブジェクトを取得してキャッシュしたのちユーザに配信する。キャッシュの空き容量が不足する場合はキャッシュ済みオブジェクトの一部を削除するが、その選択方法（キャッシュ置換法）やキャッシュにどのオブジェクトを挿入するかを選択するキャッシュ挿入法が、キャッシュの Web 応答時間の低減効果に影響する。本研究では、任意の２つのオブジェクトが同一の Web ページ内で出現することを「共起」と呼び、同一の Web ページ内で共起する他のオブジェクトの数を「共起度」と呼ぶ。HTTP/2 の並列配信による Web 応答時間の低減効果は、共起度の高いオブジェクトほど大きくなる。そのため共起度の高いオブジェクトが優先して残るようキャッシュ置換を行うことが望ましい。本研究の目的は共起度を考慮したキャッシュ挿入・置換法を確立することである。

２．研究の背景

Web 閲覧サービスはインターネット上で最も普及したサービスの１つであり、世界中の多くの人々が毎日 Web 閲覧サービスを利用している。しかし毎週 67%のユーザがブラウジング時に長い待ち時間を経験している。本稿では、Web 応答時間を Web ページのハイパーリンクをクリックしてから Web ブラウザに Web ページ全体が表示されるまでの待機時間として定義する。ユーザはページが 2 秒以内にロードされることを期待しており、その 40%が 3 秒以内にページを開くまで待機

すると言われている[2]。また 400 ミリ秒の遅延によりGoogle検索エンジンでの検索が0.74%減少することや[3]、Web 応答時間が 0.1 秒低減するごとに Amazon の収益が 1%増加することが報告されている[4]。また高速に表示されるWebページはユーザが購買を完了する回数が 15%も多く、1 ページだけ閲覧した後にページから離脱する回数が 9%も少ないことが報告されている[5]。したがって、多くのインターネットサービスプロバイダ(ISP: Internet Service Provider)やコンテンツプロバイダにとって、ウェブ応答時間を短縮することは、ユーザの体感品質およびコンテンツプロバイダの利益を改善するために解決する必要がある重要な課題である。

従来のWebページは静的なテキストや画像といったオブジェクトがサーバに用意され、Web ブラウザは HTTP を用いてこれら静的オブジェクトを単にダウンロードして表示していた。しかし近年、クライアント端末からの要求受信時に、サーブレットや JSP (Java server pages)のプログラムをサーバ側で実行するか、JavaScript で書かれた Ajax や DOM (document object model)によるプログラムを HTML に埋め込みユーザ端末側で実行することで生成される動的オブジェクトの割合が増加している[1]。また広告を専用のサーバから取得するなど、各オブジェクトの配信元が多元化している。このように一つのWebページを構成するオブジェクトは複雑性を増している。

Web オブジェクトの取得には TCP セッション上で HTTP (hypertext transfer protocol)が用いられる。従来、広く用いられていたHTTP/1.1では、オブジェクトの取得に要する遅延時間を低減するため、同一配信ホストから取得する複数のオブジェクトを１つの TCP セッション上で取得するHTTP persistent connectionと、さらに同一ホストから複数のオブジェクトを並列に取得するHTTP pipeliningが実装されている。しかし同一のTCPコネクション上で転送されるパケットが属するオブジェクトをユーザ端末が識別できないため、配信サーバは HTTP request を受信した順番でオブジェクトを返信する必要があり、送信準備ができたオブジェクトの返信開始が、他のオブジェクトの返信完了まで待たされるHead of Line (HOL)問題が生じる。

このようなHOL問題を解決するためGoogleは、パケットに「SPDY stream」と呼ばれる所属HTTPセッションの識別子を付加することで、ユーザ端末が各パケットの所属オブジェクトを識別可能とし、配信サーバが任意の順番でオブジェクトを配信可能とする SPDY を開発した[6]。SPDYはHTTPの機能として標準化され、現在、SPDYが組み込まれたHTTPがHTTP/2として普及しつつある。新バージョンである HTTP/2 にはいくつかの機能が追加され、その中でも特に主要な機能である SPDY による多重化やサーバープッシュ機能によりHTTP/1.1までの問題点であったHOL問題が回避され、Web応答時間の低減改善が期待されている。

しかし HTTP/2 のオブジェクト並列配信機能は同一の配信サーバから取得するオブジェクトに対してのみ有効である。すなわち異なる配信サーバから取得するオブジェクトを同一TCPセッション上で取得することができず、HTTP/2の並列配信による遅延時間削減効果も期待できない。HTTP/2の効果は同一の配信サーバから取得するオブジェクト数が増えるほど大きくなる。そのため HTTP/2 の効果は期待したほど得られないという報告もなされている[7]。Web応答時間の低減効果を高めるには、少数の配信サーバから多数のオブジェクトを取得する必要がある。

３.研究の方法

3.1 HTTP/2 の効果向上のための共起度に基づくキャッシュ制御

Web応答時間を低減する技術としてCDNが広く普及している。HTTP/2を利用するには配信サーバとユーザ端末の両方が対応する必要があるが、オブジェクトのオリジナルを提供するオリジンサーバが HTTP/2 に未対応であっても、CDN 事業者がキャッシュサーバを HTTP/2 に対応させることでキャッシュサーバとユーザ端末間で HTTP/2 を利用することができる。既にAkamai、Amazon CloudFront、Microsoft Azure、Google Cloud Platformなどの主要CDNサービスがHTTP/2に対応している。

キャッシュサーバは配信要求に対しキャッシュに要求オブジェクトが存在しない場合、オリジンサーバからオブジェクトを取得してキャッシュしたのちユーザ

に配信する。キャッシュの空き容量が不足する場合はキャッシュ済みオブジェクトの一部を削除するが、その選択方法（キャッシュ置換法）がキャッシュの Web 応答時間の低減効果に影響する。本研究では任意の個数のオブジェクト組が複数の Web ページ内で出現することを「共起」と呼び、あるオブジェクト組が出現する Web ページの数を「共起度」と呼ぶ。例えばあるオブジェクト組を含む 20 の Web ページが存在する場合、これらオブジェクト組の共起度は 20 となる。図 1 に青、オレンジ、緑の 3 つのオブジェクトから構成される 3 つの Web ページが存在する状況における、オブジェクトの 3 つの組の共起度を例示する。例えば青とオレンジのオブジェクト組は、2 つの Web ページに出現しているため共起度は 2 となる。

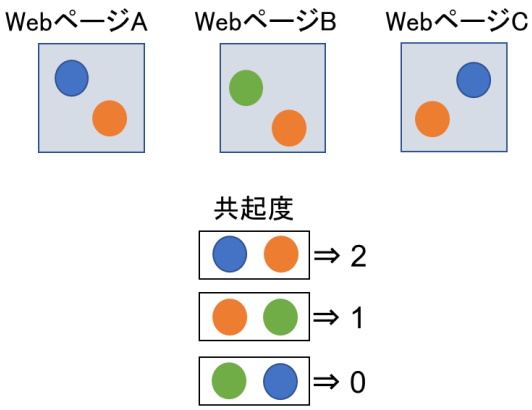

図 1 共起度の例

従来のキャッシュ制御方式は各オブジェクトを独立に扱う。しかし HTTP/2 の並列配信による Web 応答時間の低減効果は、共起度の高いオブジェクトほど大きくなる。そのため共起度の高いオブジェクトの集合が優先して残るよう、キャッシュ置換を行うことが望ましい。例えば共起度の高い 10 個のオブジェクトの組が存在するとき、これら 10 個のオブジェクトの組を一塊として、キャッシュへの挿入と削除を行うことが考えられる。その結果、1 つの Web ページを閲覧したときに同一のキャッシュサーバから配信されるオブジェクト数が増加し、HTTP/2 の効果の向上が期待される。

3.2 共起度の測定分析

前節で述べた共起度に基づくキャッシュ置換制御の効果は、現実の Web ページにおいてどの程度、共起現象が見られるかに大きく依存する。そこで本節では、実際の多数の Web ページに含まれるオブジェクトの共起度を測定分析することで、共起度に基づくキャッシュ制御の可能性を明らかにする。

3.2.1 オブジェクトデータの測定法

本節では、Web ページに含まれるオブジェクトデータの測定手順を述べる。アクセス数の多い高人気の Web ページに含まれるオブジェクトを共起度分析の対象とすることが望ましい。そこで adult、arts、business、computers、games、health、home、kids&teens、news、recreation、reference、regional、science、shopping、society、sports の 16 のカテゴリごとに、Web ページのアクセス数のランキングを公開している Alexa の Web ページから[8]、各カテゴリに対して上位 500 の Web ページの URL を測定対象としてリスト化した。次に生成した評価 URL リストの各 URL に対して、測定用 PC から GET の HTTP リクエストを送信した際に発生する通信特性を、HAR (HTTP Archive)ファイルとして取得した。HAR ファイルは、測定用 PC とサーバ間で転送される HTTP データのヘッダ情報から、測定用 PC において、各オブジェクトの URL、サイズ、取得に要した遅延時間等の各種通信特性を算出し、JSON (JavaScript Object Notation)形式で出力したものである。なお 16 の各カテゴリから 500 の合計で 8,000 の Web ページを測定対象としたが、正常に HAR ファイルを取得できた 7,604 ページを対象に分析を行う。

3.2.2 オブジェクトの識別法

共起度分析の前にまずは各オブジェクトの分析を行う。取得した JSON ファイルを解析することで、各 Web ページを構成する各オブジェクトの URL 名を取得し、URL 名で全てのオブジェクトを識別(URL 識別)したが、異なる URL でもアクセス結果が同じである事象が多数確認された。そこでオブジェクトの Hash 値、オブジェクトのサイズ、mime タイプの 3 要素が同一のオブジェクトを同一のオブジェクトとして判別する方法(Hash 識別)についても分析を行った。7,604 ページを構成するオブジェクトの総数は 707,690 で、その異なり数は URL 識別の場合は 489,100、Hash 識別の場合は 373,860 であった。

Web データの共起関係に基づくキャッシュ制御の研究

Cache Control of Web Data Based on Co-Occurrence

3.2.3 オブジェクトの重複度

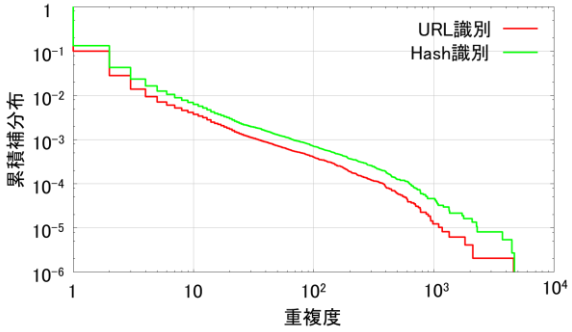

図 2 オブジェクトの重複度の累積補分布

各オブジェクトに対し、7,604 の Web ページの中で出現する Web ページの数を重複度と定義する。図 2 に、URL 識別、Hash 識別それぞれについての各オブジェクトの重複度の累積補分布を示す。重複度 1 のオブジェクトが全体の 9 割程度を占めるが、重複度の分布の裾野は広く、冪乗則が観測される。URL 識別では 0.05%程度のオブジェクトは 100 以上の Web ページに、0.001%程度のオブジェクトは 1,000 以上の Web ページに出現している。また Hash 識別では 0.08%程度のオブジェクトは 100 以上の Web ページに、0.004%程度のオブジェクトは 1,000 以上の Web ページに出現している。

動的オブジェクトであるためクエリパラメータは全て異なっているが、Hash 値は全て同一であるため、実際には同じオブジェクトであると考えられる。しかしながら、URL 識別では URL のみで識別するため、これらは異なるオブジェクトとして認識される。このことから Hash 識別の方がより正確にオブジェクトを識別できることが確認できる。

3.2.4 オブジェクトの 2 個組の共起度

まず 707,690 個のオブジェクトの中で重複度が 2 以上かつ同一 Web ページに出現しているオブジェクトから任意の 2 個組を作成する。1 つ以上の Web ページで出現した組に対し、7,604 の Web ページにおける共起度（出現 Web ページ数）の累積補分布を図 3 に示す。

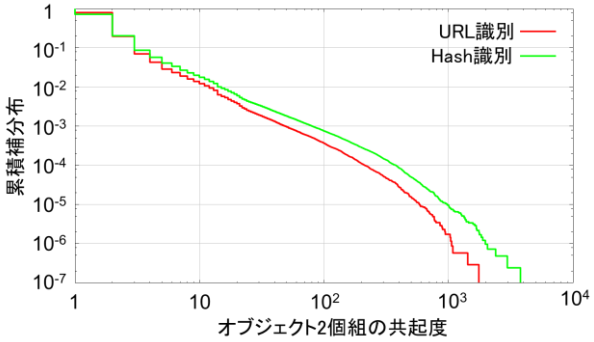

図 3 オブジェクト 2 個組の共起度の累積補分布

URL 識別では 3,304,528 個の 2 個組を作成し、そのうち 20%程度の 2 個組が 2 つ以上の Web ページに出現していた。Hash 識別では 4,186,537 個の 2 個組を作成し、そのうち 20%程度の 2 個組が 2 つ以上の Web ページに出現していた。オブジェクト 2 個組の共起度の分布にも冪乗則が観測され、裾野は広く、URL 識別では 0.05%程度の 2 個組は 100 以上の、0.005%程度の 2 個組は 500 以上の共起度を有していた。また Hash 識別では 0.1%程度の 2 個組は 100 以上の、0.01%程度の 2 個組は 500 以上の共起度を有していた。このことから少数のオブジェクト 2 個組は極めて高い共起度を有していることが確認できる。

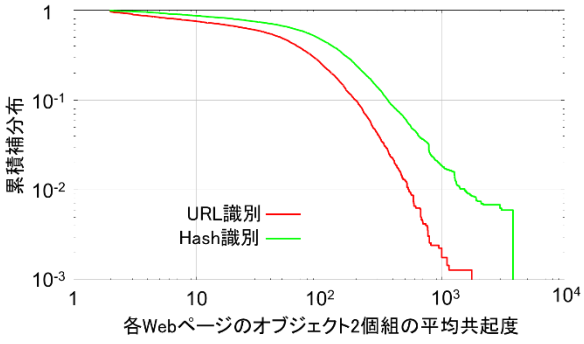

図 4 各 Web ページのオブジェクト 2 個組の平均共起度の累積補分布

次に各ページを構成する 2 個組の集合の共起度の平均値を求めた。図 4 に各 Web ページに対する平均共起度をプロットする。URL 識別で約 30%、Hash 識別で約 60%程度のページで、平均共起度が 100 を超えていた。少数の高重複度オブジェクトが多くのページで平均値

を引き上げているため、このような全体的に高い平均共起度となっていると考えられる。また、少数の Web ページは極めて高い共起度を有していることが確認できる。

3.3 共起を考慮した Web キャッシュ挿入・置換方式

Web 応答時間を低減する技術として複数のキャッシュサーバが存在するとき、共起を考慮したキャッシュ制御を行うことで、同じキャッシュサーバに単一の Web ページにアクセスした際に要求されるオブジェクトを可能な限り集約させる。キャッシュ制御は挿入時と置換時の 2 つがあり、挿入時に共起を考慮することで同じサーバに共起組のオブジェクトを集約させ、置換時に共起を考慮することで共起組のオブジェクトが排出されにくくなる。本節では、共起を考慮したキャッシュ制御、つまり共起するオブジェクト組を構成するオブジェクトを単一のキャッシュサーバに可能な限り集約する手法を提案する。

3.3.1 挿入ポリシィ

複数のキャッシュサーバが存在するとき、オブジェクトを挿入するキャッシュサーバを選択する必要があるが、HTTP/2 の並列取得効果を得るためには、共起オブジェクト組は同一のキャッシュサーバに収納することが望ましい。

そこでオブジェクトをキャッシュサーバに新たにキャッシュするキャッシュサーバを選択するとき、既にキャッシュされているオブジェクトに、新たにキャッシュするオブジェクトが加わることによって、新たに生じる共起オブジェクト組の数をキャッシュサーバ毎に求め、その値が最も大きいキャッシュサーバを新たなオブジェクトをキャッシュするキャッシュサーバとして選択する。

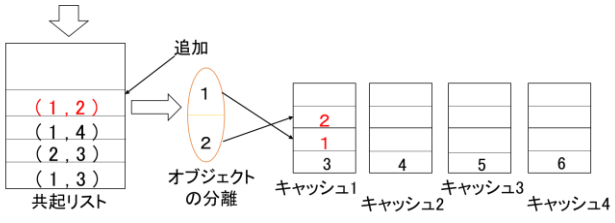

図 5 共起組を考慮したキャッシュ挿入の例

図 5 は共起を考慮した際のオブジェクトの挿入の流れを示す。ここでの共起リストとは共起オブジェクト組の情報を管理する場所である。オブジェクトの挿入はオブジェクトごとに行われるため、それぞれのオブジェクトが共起組のオブジェクトであったか単一のオブジェクトであったかを判断できない。そこで共起オブジェクト組がキャッシュに挿入される場合、共起リストに共起ペアを保存してからオブジェクト組を分解する。次に各オブジェクトを挿入するキャッシュサーバを決定するが、共起リストの情報を参考に、新しくキャッシュするオブジェクトを追加することによって、各キャッシュサーバで新たに作成可能な共起オブジェクト組の数を算出する。そして新たに作成可能な共起オブジェクト組が最も多く存在するキャッシュサーバを、そのオブジェクトを挿入するキャッシュサーバに決定する。新たに作成可能な共起オブジェクト数が同数の場合は、それらキャッシュサーバの中で最初にチェックを行ったものを選択する。

図 5 ではオブジェクト 1 と 2 の組を挿入している。共起リストを確認すると、オブジェクト 1 の共起ペアとしてオブジェクト 3 と 4 があげられる。キャッシュに挿入するオブジェクトのペアが存在するか探索すると、キャッシュ 1 及びキャッシュ 2 にそれぞれ一つずつ存在する。この場合、キャッシュ 1 を先に探索したため、オブジェクト 1 はキャッシュ 1 に挿入される。次にオブジェクト 2 の共起ペアとしてはオブジェクト 1 と 3 があげられる。同様に探索を行うと、キャッシュ 1 に最も多く共起ペアが存在するため、オブジェクト 2 はキャッシュ 1 に挿入される。

3.3.2 置換ポリシィ

キャッシュサーバのキャッシュがあふれたとき、キャッシュ済みオブジェクトのいくつかを排出する必要がある。提案手法ではオブジェクトを排出する際に、排出オブジェクトの共起ペアがキャッシュ内に存在しているかを確認する。存在していない場合は Least Recently Used (LRU)方式に基づいて排出するが、存在している場合は排出予定だったオブジェクトを残し、LRU に基づき次の候補オブジェクトを排出する。

また共起を考慮したキャッシュ置換制御では共起オブジェクト組がリクエストされた場合、共起オブジェ

クト組それぞれの単一のオブジェクトに加えて共起オブジェクト組単位でアクセス順のリストを管理する。リストに従って LRU 方式を用いて排出するオブジェクトを決定する。このとき、LRU のアクセス順リストを共起オブジェクト組で管理することで、LRU のアクセス順リスト内には、同じオブジェクトが異なるオブジェクト組を構成するオブジェクトとして複数回出現する。LRU リスト内に複数回出現するオブジェクトであったとしても、キャッシュとしてはオブジェクトひとつについては一つしかキャッシュを行わない。そこで、あるオブジェクトや共起オブジェクト組がリクエストされた際にオブジェクトごとに参照数をカウントする。キャッシュ内のオブジェクトが LRU リストから参照されている回数を用いてキャッシュにおけるオブジェクトの保持の要否を判断する。

あるオブジェクト、あるいは共起オブジェクト組のリクエストが発生したとき、他の共起オブジェクト組によってそれらのオブジェクトは既にキャッシュ内には存在しているものの、LRU のリストにはリクエストのあったオブジェクトあるいは共起オブジェクト組が存在しない状況が発生する。そのようなとき、LRU のリストにリクエストされたオブジェクトあるいは共起オブジェクト組を追加し、キャッシュされているオブジェクトの参照数をインクリメントする。

排出するオブジェクトを決定するとき、LRU に従い最も使われていないオブジェクトまたは共起オブジェクト組を取り出し、そこに含まれるオブジェクトについて、キャッシュの参照数をデクリメントする。参照数が 0 になったオブジェクトは、キャッシュから削除する。これを、必要な数の空き領域が得られるまで繰り返す。

提案方式

		Input	
	Initial	1	5
Cache	1	1	1
	2	2	2
	3	3	5
	4	4	4
LRU list	(1,2),1, 2,3,4	2,3,4, (1,2),1	4,(1,2), 1,5

⇒オブジェクト3を削除

LRU

		Input	
	Initial	1	5
Cache	1	1	1
	2	2	5
	3	3	3
	4	4	4
LRU list	1,2,3,4	2,3,4,1	3,4,1,5

⇒オブジェクト2を削除

図 6 共起組を考慮したキャッシュ挿入の例

図 6 に、置換ポリシィによる排出制御と単純な LRU を用いた排出制御の比較を例示する。キャッシュサーバの中身をオブジェクト 1、2、3、4 があらかじめ格納された状態とし、オブジェクト 1 とオブジェクト 5 が順番に参照されるときの動作を示す。その時のオブジェクト情報を LRU リストに示す。LRU リストに存在する一番左のオブジェクトが最も過去の時間に参照されたものを示し、(1,2)のような表記は共起オブジェクト組を示す。まずオブジェクト 1 が参照されるとき、単純な LRU はオブジェクト 1 の情報を更新するが、提案方式だとオブジェクト 1 の情報に加えて共起組である(1,2)の情報も更新する。次にオブジェクト 5 が参照されるとき、キャッシュサーバの中身があふれるため置換を行う必要がある。単純な LRU だと排出されるオブジェクトが 2 となるが、提案方式だとオブジェクト 2 が排出されても LRU リストに(1,2)が残っているためオブジェクト 2 を排出できない。そのため排出されるオブジェクトは 3 となる。

4. 将来展望

これまでに単一の拠点に複数のキャッシュサーバが存在する場合に、それらキャッシュへの Web オブジェクトの挿入・置換を、オブジェクトの共起度を考慮して行うことを検討した。ところでコンテンツ事業者が複数の CDN 事業者を利用したり、複数のコンテンツ事業者が提供するオブジェクトを 1 つの Web ページに併せて表示したりするなど、1 つの Web ページが複数の CDN 事業者の提供オブジェクトで構成される場合も多い。しかし CDN 事業者は独立して CS を構築・運用するため、異なるコンテンツ事業者の提供オブジェクトを同一の CS からは配信できない。しかし仮想マシンを用いて CDN を構築する仮想 CDN を用いることで、複数 CDN 事業者間で物理マシンを共有できる。

さらに同一物理マシンを使用する仮想 CS 間で HTTP/2 の TCP セッションを共有できる仕組みを確立すれば、異なる CDN 事業者のオブジェクトの並列配信が可能となる。この際、クラウド事業者が仮想 CS 間のオブジェクトの共起度を計算し、共起度の高いオブジェクトが同一の物理ホストにキャッシュされるよう、仮

想マシンを物理マシンに割り当てることが有効である。そこで共起度を考慮したキャッシュ制御方式を、仮想マシンでキャッシュを構成している場合に拡張することが期待される。

おわりに

本研究では、Web 応答時間を低減する仕組みとして普及している HTTP/2 の効果の向上のためには、同一 Web ページを構成するオブジェクトをできるだけ同一のキャッシュサーバに集約させる必要がある。そこで本研究では、同一の Web ページを構成する複数のオブジェクト組が、他の Web ページでも出現する現象を共起と呼び､共起するページの数を共起度として定義し、実際の Web ページを測定して共起現象がどの程度、出現するかを分析した。さらに複数のキャッシュサーバで共起を考慮したキャッシュ挿入･置換法を提案した。

参考文献

[1] M. Butkiewicz, H. V. Madhyastha, and V. Sekar, Understanding Website Complexity: Measurements, Metrics, and Implications, ACM IMC 2011.

[2] J. Mickens, Silo: Exploiting JavaScript and DOM Storage for Faster Page Loads, USENIX WebApps 2010.

[3] S. Sundaresan, N. Feamster, R. Teixeira, and N. Magharei, Characterizing and Mitigating Web Performance Bottlenecks in Broadband Access Networks, ACM IMC 2013.

[4]X. S. Wang, A. Balasubramanianm A. Krishnamurthy, and D. Wetherall, How speedy is SPDY?, NSDI 2013

[5] E. Nygren, R. Sitaraman, and J. Sun, The Akamai Network: A Platform for High-Performance Internet Applications, ACM SIGOPS 2010

[6] M. Jiang, X. Luo, T. Miu, S. Hu, and W. Rao, Are HTTP/2 Servers Ready Yet?, IEEE ICDCS 2017

[7] Not as SPDY as You Thought, Guy's Pod, Thoughts and research on Web Performance ¥& Secirity, Jun. 2012

[8] Alexa, https://www.alexa.com/siteinfo

この研究は、平成２９年度ＳＣＡＴ研究費助成の対象として採用され、平成３０年度～令和２年度に実施されたものです。