4P-4

ブラウザ間通信を用いたアクセス集中時のコンテンツ配信法

Contents Delivery by Communication between Browsers

知識 友紀江† 千葉 大紀‡ 後藤 滋樹‡

† 早稲田大学 基幹理工学部 情報理工学科
‡ 早稲田大学 基幹理工学研究科 情報理工学専攻

概要

大量のアクセスが特定のサーバに集中すると，レスポンスが低下したり，サーバが異常停止することがある．本論文はサーバの性能や台数を増強せずに，大量のアクセスが集中するコンテンツを多数のユーザに迅速に配信可能な方法を提案する．具体的には，アクセス集中時にブラウザ間通信を活用して，ユーザ同士でコンテンツの配布を行い，サーバの負荷を分散する．本論文では実際にアクセスが集中する環境を用いて提案手法を評価し，従来のサーバ・クライアントモデルによるユニキャスト通信と比較した．その結果，提案手法がサーバの負荷を軽減し，コンテンツのダウンロード時間を大幅に短縮できることを確認した．

1 提案手法

本研究では，通常時とアクセス集中時で配信法を切り替え，アクセス集中時にはブラウザ間通信 [1] を活用してユーザ同士でコンテンツの配信を行う．これにより，コンテンツ配信サーバが配信するファイルを削減し，負荷を軽減することで，安定したコンテンツ配信を実現する．具体的な手順を以下に示す．

通常時の配信法 アクセス集中が起きていない場合には，一般的なサーバ・クライアント型のユニキャスト通信と同様に，コンテンツ配信サーバがすべてのユーザに対してコンテンツを送信する．このときのコンテンツ配信手順を図 1に示す．ユーザがコンテンツ配信サーバへリクエストを送ると，コンテンツ配信サーバはそのユーザへコンテンツを送る．

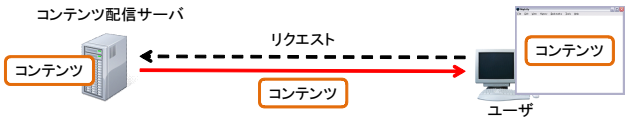

図 1: 通常時の配信法

アクセス集中時の配信法 アクセス集中が発生している場合には，コンテンツ配信サーバは一部のユーザに対してのみコンテンツの送信を行う．コンテンツ配信サーバからコンテンツを受信したユーザは WebSocket サーバとなり，他のユーザ (クライアント) へコンテンツを配信する．ここで言うクライアントとは，コンテンツ配信サーバからコンテンツを受信しないユーザのことであり，WebSocket サーバへ接続するためのファイルだけを受信する．なお，今回は WebSocket の実装上 JavaScript ファイルを受信する．

アクセス集中時のコンテンツ配信手順を図 2を用いて説明する．ユーザ A がコンテンツ配信サーバへリクエストを送ると，コンテンツ配信サーバはコンテンツと JavaScript ファイルをユーザ A へ送る．ユーザ A はコンテンツをブラウザ上に表示すると共に，自身がコンテンツを配信する WebSocket サーバになる．この時，コンテンツ配信サーバはユーザ A の IP アドレスを WebSocket サーバリストに追加する．その後，ユーザ B がコンテンツ配信サーバへリクエストを送ると，コンテンツ配信サーバは WebSocket サーバの IP アドレスを含む JavaScript ファイルをユーザ B へ送る．その JavaScript ファイルを元に，ユーザ B は WebSocket サーバ (ユーザ A) へリクエストを送る．そして，ユーザ A (WebSocket サーバ) はユーザ B へコンテンツを送る．

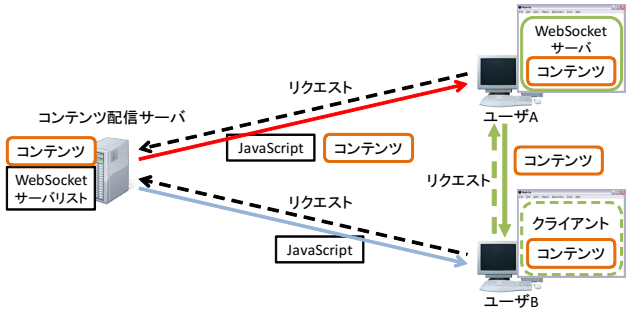

図 2: アクセス集中時の配信法

2 性能評価

2.1 評価方法

提案手法を適用したコンテンツ配信サーバの性能を評価する．今回はコンテンツ配信サーバへ大量のリクエストが集中している環境を想定する．また，既存手法としてサーバ・クライアントモデルによるユニキャスト通信を行うコンテンツ配信サーバに対して同様の評価を行い，結果を比較する．

コンテンツ配信サーバには5000KBのテキストデータをコンテンツファイルとして用意した．そのファイルに対するリクエストを発生させ，各リクエストに対するコンテンツファイルのダウンロード時間とコンテンツ配信サーバのCPU使用率を測定する．

なお，本評価では提案手法における配信法の切り替えは行わず，始めからアクセス集中時の配信法をとることとする．また，全リクエストに対するWebSocketサーバとクライアントの比率を1:2とする．この値は，予備実験において様々な比率 (1:1 〜 1:9) で行った結果を比較して決定した．

2.2 評価結果

1分間に大量のリクエストをコンテンツ配信サーバへ送り，各リクエストに対するコンテンツファイルのダウンロード時間とコンテンツ配信サーバのCPU使用率を測定した．平均ダウンロード時間を図3に示す．また，50件のリクエストを送った場合のCPU使用率を図4に示す．

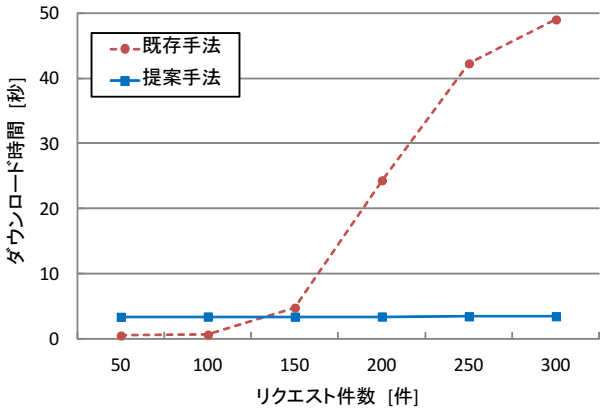

図 3: 平均ダウンロード時間

図3を見ると，既存手法ではリクエスト件数が増加してアクセスが集中するほど，ダウンロード時間が増加していく．一方で，提案手法はリクエスト件数が増加してもほぼ一定のダウンロード時間を維持している．このことから，アクセス集中が発生している際に短時間でコンテンツを配信する方法として提案手法が有効であることが分かった．しかし，リクエスト件数が100件以下の場合は，既存手法よりも提案手法の方がダウンロード時間が長くなることが分かった．これは，提案手法におけるWebSocketサーバからクライアントへのコンテンツ送信にかかる時間が既存手法のダウンロード時間よりも長いためである．したがって，提案手法では通常時とアクセス集中時の判定を行い，配信法の切り替えを行うことで，どの状態でも短時間でコンテンツをダウンロードすることができる．

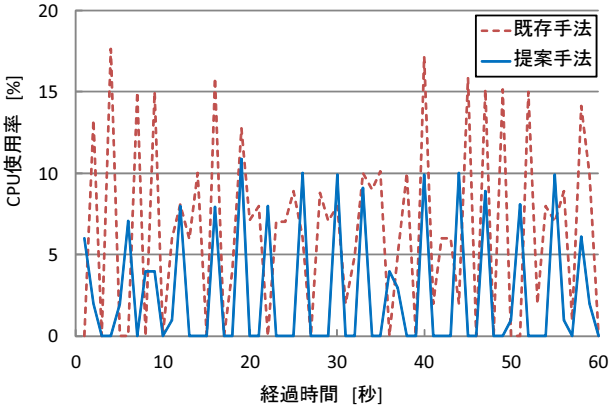

図 4: CPU 使用率 (リクエスト件数：50 件)

リクエスト数が50件の場合のコンテンツ配信サーバのCPU使用率 (図4) は，既存手法では平均6.5%であり，提案手法では平均2.6%であった．この結果より，提案手法によりダウンロード時間が短縮できない場合 (e.g. 図3のリクエスト件数が100件以下) でも，サーバの負荷分散の効果があることが分かった．

3 まとめ

本論文では，アクセス集中時にブラウザ間通信を活用し，ユーザ同士でコンテンツの配布を行う方法を提案した．さらに，実際にアクセスが集中する環境を用いて提案手法の評価を行い，従来のサーバ・クライアントモデルによるユニキャスト通信と比較した．その結果，提案手法はコンテンツ配信サーバの負荷を分散し，アクセス数の増加に伴うコンテンツダウンロード時間の増加を抑えるという点で非常に有効であることを示した．今後の課題は，WebSocketサーバの条件設定と接続先WebSocketサーバの選択法である．

参考文献

[1] 與儀那広, 城間政司, 長田智和, 谷口祐治, 玉城史朗, WebSocketを用いたWebブラウザ間P2P通信の実現とその応用に関する研究，電子情報通信学会技術研究報告．NS, ネットワークシステム 110 (372), pp.59–62, 2011-01-13.

[2] I. Fette, The WebSocket Protocol, RFC 6455, December 2011, http://www.ietf.org/rfc/rfc6455.txt