7S-04

WebRTCを用いたP2Pネットワークの構築と効率化についての検討と評価

成田 裕司 † 坪川 宏 †
† 東京工科大学 大学院 バイオ・情報メディア研究科

1 はじめに

近年、ES6の策定やさまざまなJavaScript API，フレームワークが登場し，そのAPIの中にあるWebRTCを用いることでブラウザ間P2P通信が行える．また，現在では，DataChannel機能を用いたストリーミング動画の配信（CDN）にも使用されている．

しかしながら，WebRTCを用いたシグナリングサーバのみで行う会議ビデオチャット，CDNにおいてはノード数を多く扱う大規模な構成で扱うことは難しい．特に会議チャットなどWebカメラと音声を使用するものはGetUserMedia APIにより，実現しているが動画音声の多段中継などの問題により，クライアントサイドのみで大規模なネットワークを実現することは難しい．

そこで本研究では，クライアンサイドを主とし，WebRTCのDataChannel機能を用いて複数ブラウザでのJavaScriptによるネットワーク構築を行う．また，大規模なP2Pシステムを支えるDHTアルゴリズムのひとつであるKademiliaに着目し，WebRTC，JavaScriptで実装を行い，クライアントサイドでのネットワーク構築を行う．これにより，クライアントサイドで適切な制御を行うことで効率的なブラウザ上ネットワークの実現を目指す．

2 システム概要

本システムはJavaScriptとWebRTCおよびSkyWay-PeerJSを用いて，Kademliaを実装している．また，シグナリングサーバに関しては，クライアントサイドに着目しているため，WebRTCプラットフォームとして提供しているSkyWayを現段階では用いて実装している．

Kademliaを選択した理由として，JavaScriptおよびブラウザに対して多くのネットワーク維持制御による処理を行わず，送受信の中で自身の2分木構造のルーティングテーブルの更新が行えるためである．ここで，本システムのネットワーク概念図を図1に示す．

構築されたP2Pネットワークは画像や動画などを主としたバイナリ型のデータを想定しているおり，GetUserMediaなどで扱われるStream型のデータは想定しない．

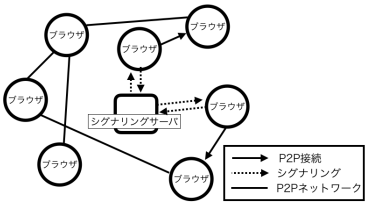

図 1: WebRTCネットワーク

3 システム構成

本システムを実装するにあたり，ウェブブラウザのUI操作がネットワーク制御処理および接続，送受信処理によって，阻害されないことが重要である．そのため，複数ブラウザに対して，接続・送信時受信を行う際にイベントハンドラによる処理などJavaScriptに適した記述が必要である．以下にシステム構成を述べる．

3.1 ネットワーク参加

WebRTCでは，接続するために他のPeerIDを使用し，シグナリングサーバを経由した後にブラウザ同士の接続を行う．そこで，個々のブラウザに与えられるオリジナルなIDをなんらかの方法で入手し，ネットワークに参加しなけらばならない．そこでWebRTCのシグナリングサーバを用い，参加に対する処理を行うことで対応する．サーバへ接続した際に，自身のNodeIDと近いIDを受け取ることでこの問題を解決する．

3.2 Kademlia

KademliaのNodeIDのハッシュサイズはSHA1を使用し，160bitとなっている，PeerJSとは別に独自にIDの生成を行い，NodeID，PeerIDとして使用する．これにより，シグナリングサーバから得られるランダムなIDを制御可能な形にすることで，ID生成にルール適用が行える．

各ノード情報として格納する値はNodeID，LifeTimeを各ノード情報として扱い，このノード情報を

Implementation and Evaluation of CDN on the browser using WebRTC
† Yuji Narita
† Hiroshi Tubokawa
Graduate School of Bionics, Computer and Media Scienses, Tokyo University of Technology (†)
1404-1 Katakuracho, Hachioji, Tokyo 192-0914, Japan

Copyright ©2016 Information Processing Society of Japan.
All Rights Reserved.

ノードリストに格納する．ノードリストにはインデックス情報として自身のノード ID からの距離となる XOR の値を持たせる．

今回実装した Kademlia.js には PING，FindNode，FindValue，Store の問い合わせ方法があり本システムでもこの 4 種類の問い合わせ方法を実装している．

FindNode には自身が複数のノードに対して，接続および問い合わせを行う．この時，ループ関数内で接続および問い合わせ結果を待つ場合，接続待ちと受信待ちによりスレッド専有状態となってしまう，このため，図 2 の様なループ外で受信を行い，適当なタイミングで受信後の処理を行う仕組みとした．ループ外でイベントを待つことにより，効率的に送受信を行うことができる．

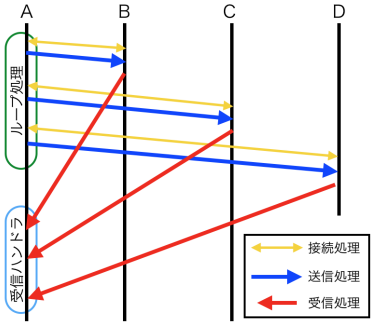

図 2: 送受信時のデータの流れ

3.3 WebRTC 部

WebRTC 部は，送信部と受信部を分離して実装を行っている．送信部の多くは Kademlia からの呼び出しにより，送信メソッドの処理が行われる．受信部に関しては，接続および受信イベント用メソッドで送信後受信処理も一括に受け取り処理を行うことで，送信時のレスポンスを待つことなく次の処理へ移れるように実装が行われている．これにより，ループ時や接続待ち時スレッド専有時間を減らしている．

3.4 シグナリングサーバ

今回はクライアントサイドに着目して実装しているため，SkyWay を代替として用いる．SkyWay の接続可能台数をテストしたところ 50 台程度であった．大規模なネットワークを構築する際には，独自での実装が必要不可欠になる．そこでサーバサイドの構築について述べておく．図 3 がサーバ構成図である．共通のノード ID データベースを複数のシグナリングサーバで管理することで大規模なネットワークを構築する際のシグナリングサーバのセッション数増加による安定性と同一ネットワーク空間に参加させることができる対応する．

機能としては，本来のシグナリング機能に加え，ネットワーク参加時におけるネットワーク上のノード情報収集を FindNode で行うが，この処理はブラウザへの負荷が大きくなってしまうため，サーバ側からコネクション ID を元に XOR 値の近いノードリストを与える機能を追加する．

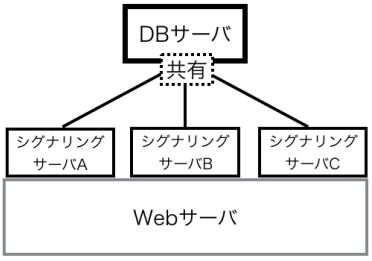

図 3: サーバ構成図

4 動作検証

今回，WebRTC および PeerJS を用い，JavaScript に適した Kademlia を実装した．検証については，本システムで使用する SkyWay の最大接続ノード数が 50 台程度という点から小規模での WebRTC 通信部動作検証を行う．また，JavaScript で実装した Kademlia のノード参加離脱時の動作などの DHT について検証も行う．

5 まとめ

今回，WebRTC を用いた P2P ネットワークの構築時の実装案について述べ，JavaScript での実装を行い，基本的な動作に問題がないことは確認している．PaaS での最大ノード数がボトルネックとなり，大規模なネットワーク検証ができないため，今後は，クライアントサイドと共にサーバサイドを独自で開発を行い，より柔軟なネットワーク構築を目指す．

参考文献

[1] P. MAYMOUNKOV：”Kademlia: A Peer-to-peer Information System Based on the XOR Metric”：Peer-to-Peer Systems，Proc. the 1st International Workshop on Peer-to-Peer Systems，IPTPS’02，53-65，2002-03

[2] 工原 誠，渡邊 寛，福士 将，天野 憲樹，野上 保之：”Web ベースボランティアコンピューティングのためのブラウザ間 P2P 通信機能” 情報処理学会研究報告. [ハイパフォーマンスコンピューティング] 2014-HPC-144(7), 1-7, 2014-05-1

Copyright ©2016 Information Processing Society of Japan.
All Rights Reserved.