TreeVNCの拡張

学籍番号 155702F 氏名 大城由也 指導教員：河野真治

平成 30 年 11 月 22 日

概 要

1 研究目的

講義や発表の場では、用意された資料や PC 画面を見ながら進行することが多い。ゼミなどでは発表者を切り替えながら発表を行う。通常このような場面では、資料やスライドを表示するためにプロジェクタが使用されている。ゼミの際には発表者を切り替えるたびにプロジェクタにケーブルを差し替える必要がある。ケーブルの差し替えの際に発表者の PC によってはアダプターの種類や解像度の設定により、正常に PC 画面を表示できない場合がある。

画面配信システム TreeVNC は発表者の画面を参加者の PC に表示するソフトウェアである。TreeVNC を使用することで、参加者は不自由なく手元の PC を使用しながら講義を受ける事が可能になる。更に発表者の切り替えの際も、ケーブルの差し替えを行わずに共有する画面の切替を可能としている。

TreeVNC は VNC(Virtual Network Computing) を使用した画面配信を行っている。しかし、通常の VNC では配信側に全ての参加者が接続するため、多人数の際の処理性能が落ちてしまう。TreeVNC では有線でネットワークに接続した参加者をバイナリツリー状に接続し、 配信コストをクライアントに分散させる仕組みをとっている。そのため、講義で発表者の画面を表示する際、多人数の生徒が参加しても処理性能が下がらない。また、ツリーのルートが参照している VNC サーバーを変更することで、共有する画面の切替が可能となっている。

しかし、画像配信システムは送信するデータ量が多いため、現在の TreeVNC では無線 LAN 接続の場合、画面の配信に遅延が生じてしまう。そこで本研究では、マルチキャスト対応の実装やデータの分割･圧縮方法の評価を行うことにより、無線 LAN での配信環境の向上を目指し、TreeVNC の有用性を評価することで講義やゼミを円滑に行えることを目標とする。

2 VNC

VNC(Virtual Network Computing) は、 RFB プロトコルを用いて遠隔操作を行うリモートデスクトップソフトウェアである。サーバー側とクライアント (ビューア) 側に分かれており、サーバー起動後クライアントがサーバに接続することで遠隔操作を可能としている。

3 RFB(Remote Frame Buffer) プロトコル

RFB プロトコルは、自身の画面を送信しネットワーク越しに他者の画面に表示するプロトコルである。ユーザがいる側を RFB クライアント、Framebuffer への更新が行われる側を RFB サーバと呼ぶ。Framebuffer とは、メモリ上に置かれた画像データである。

プロトコルを起動した際の動作は以下の順である。

1. プロトコルバージョンの確認や認証を行う。
2. クライアントに向けて Framebuffer の大きさやデスクトップに付けられた名称などが含まれた初期メッセージが送信される。
3. RFB サーバ側は Framebuffer の更新が行われるたびに RFB クライアントに対して Framebuffer の変更部分だけを送信する。
4. RFB クライアントから FramebufferUpdateRequest が来るとそれに返信する。

4 TreeVNC の構造

TreeVNC は Java を用いて作成された TightVNC を元に構成されている。TreeVNC はクライアント同士を接続させ、データを受け取ったクライアントが次のクライアントにそのデータを流す方式を取っている。また、サーバへ接続しにきたクライアントをバイナリツリー状に接続する (図 1)。バイナリツリー状に接続することで、N 台のクライアントが接続しにきた場合、画面配信の画像データをコピーする回数が従来の VNC ではサーバが N 回コピーする必要があるが、TreeVNC では各ノードが 2 回ずつコピーするだけで済み、負荷を分散することができる。

TreeVNC で送受信される画像データ量は莫大であり、大きなネットワークスループットが必要となるため、現在では有線接続が必須となっている。

バイナリツリーのルートのノードを Root Node と呼び、そこに接続されるノードを Node と呼ぶ。Root Node は

1. 子 Node にデータを流す機能

2. 各 Node の管理
3.VNC サーバから流れてきたデータの管理
を担っている。
各 Node は
1. 親 Node から送られてきたデータを自身の子 Node に流す機能
2. 子 Node から送られてきたデータを親 Node に流す機能
を担っている。

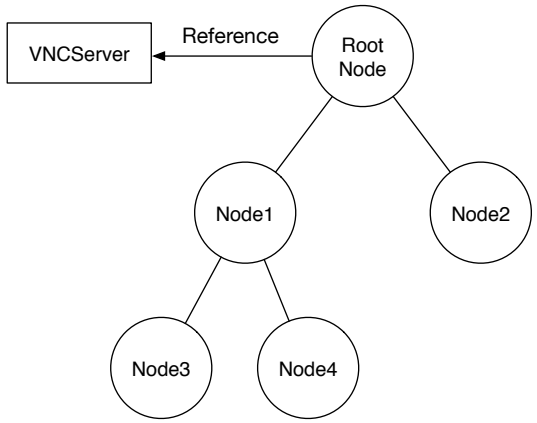

図 1: 構成される木構造

5 TreeVNC の原理

従来の VNC と TreeVNC の構造 (図 2) の比較を行い、TreeVNC の原理と性能を示す。

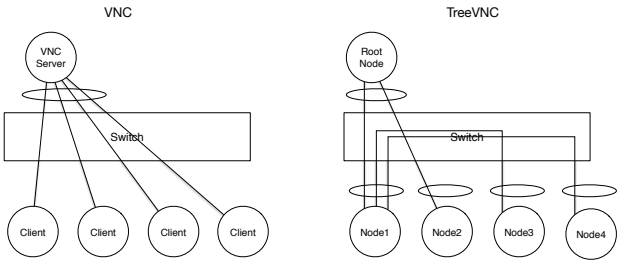

図 2: ポート一本にかかる負荷

表 1 はそれぞれのポート一本あたりの通信量である。通常の VNC の場合、クライアント数に比例してポート一本あたりの負荷が増えている。TreeVNC の場合は 1 つの Node に対して 2 台の Node が接続するため、最大でも親から送信されるデータと 2 つの子に送信するデータ分の負荷になる。

送信するデータ量も通常の VNC の場合 Node 数に比例した量のデータを送信する必要があり、CPU に負荷がかかってしまう。それに対して TreeVNC はクライアントが増えても送信するデータはクライアント毎に分散されているため、1 台の CPU に掛かる負荷は一定となる。そのため、性能が低下せずに画面配信を行うことが出来る。

表 1: ポート一本あたりの通信量 (N はノード数, d はデータ量)

	通常の VNC	TreeVNC
通信量	N * d	(2 + 1) * d

6 圧縮形式

TreeVNC は ZRLEE というエンコードでデータのやり取りを行う。ZRLEE は RFB プロトコルで使えるエンコーディングタイプの ZRLE を元に生成され、ZRLE は Zlib で圧縮されたデータとそのデータのバイト数がヘッダーとして付けて送られてくる。また Zlib は java.util.zip.deflater と java.util.zip.inflater で圧縮と解凍が行える。

しかし、java.util.zip.deflater は解凍に必要な辞書を書き出す (flush) ことが出来ない。そのため図 3 のように、Zlib 圧縮されたデータを途中から受け取ってもデータを正しく解凍することが出来ない。

そこで ZRLEE は 一度 Root Node で受け取った ZRLE のデータを unzip し、 データを update rectangle という画面毎のデータに辞書を付けて zip し直すことで初めからデータを呼んでいなくても解凍を行えるようになっている (図 4)。一度 ZRLEE に変換してしまえば子 Node はそのデータをそのまま流すだけで良い。ただし、deflater と inflater では前回までの通信で得た辞書をクリアしないといけないため、Root Node と Node 側では毎回新しく作る必要がある。

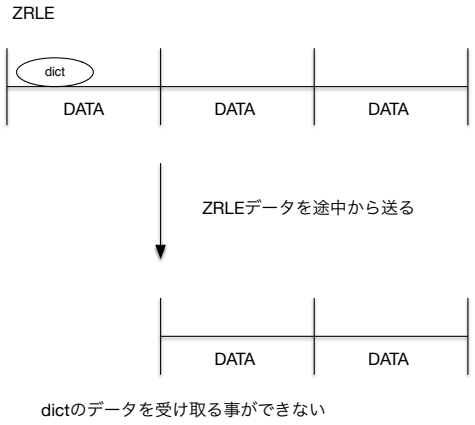

図 3: ZRLE での問題点

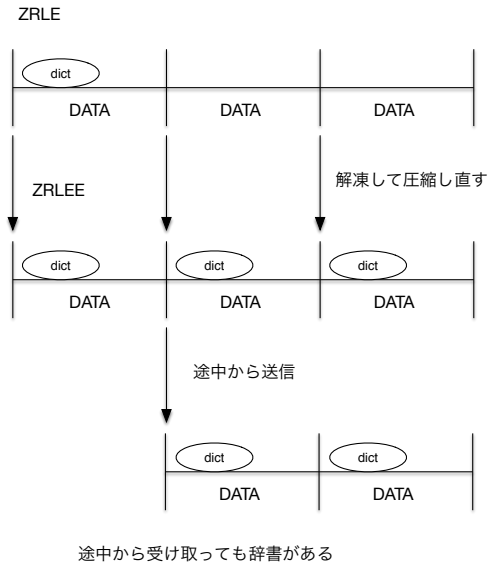

図 4: ZRLEE

7　マルチキャスト配信への対応

TreeVNC は現在有線接続でのみ安定した動作をみせている。しかし、講義の際、毎回コードを持参することは負担であるため、無線接続での安定した動作を確立したい。そこで、Wi-Fi を利用したマルチキャストの対応を行う。

8　まとめ

本研究では TreeVNC によるマルチキャスト配信への対応を行っている。

この研究により今後の TreeVNC の利用時に無線 LAN を使っての接続がより快適なものになると考えられる。

今後の課題としては、 無線 LAN で送信できるサイズへのデータ分割、圧縮などが挙げられる。