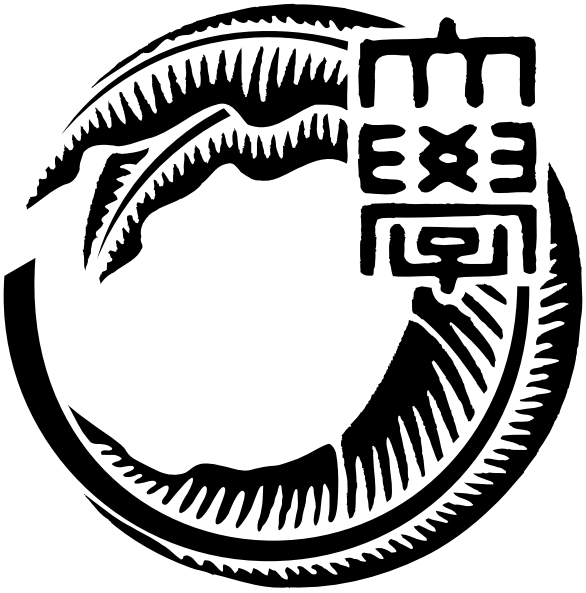
平成30年度 卒業論文

画面配信システム TreeVNC の拡張



琉球大学工学部情報工学科

155702F 大城由也

指導教員 河野 真治

目 次

図 目 次

ソースコード目次

第1章　画面配信ソフトウェア TreeVNC の活用

現代の講義や発表、プレゼンなどは、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場合もある。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。また、参加者もプロジェクタに集中を割く必要があり、手元の PC と交互に参照する場合、負担になる可能性がある。

当研究室で開発してい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。そのため 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。通常の VNC では配信側の PC に全ての参加者が接続するため、多人数が接続した際処理しきれず、最悪の場合ソフトウェアが落ちてしまう。TreeVNC では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、画面共有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場合がある。そこで本研究では、multicast でのデータ通信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。

第2章　TreeVNC の基本概念

TreeVNC は当研究室で開発している画面配信ソフトウェアである。
本章は TreeVNC の基本概念となっている技術について説明する。

2.1　Virtual Network Computing

TreeVNC の名称にもある VNC (Virtual Network Computing) は、RFB プロトコルを用いて PC の遠隔操作を行うことを目的とした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側に分かれており、起動したサーバーにクライアントが接続することで遠隔操作を可能にしている。

2.2　RFB プロトコル

RFB (Remote Frame Buffer) プロトコルは、自身の画面をネットワークを通じて送信し他者の画面に表示するプロトコルである。

ユーザがいる (画面を表示される) 側と FrameBuffer への更新が行われる (自身の画面を送信する) 側に分かれ、それぞれを RFB クライアント、RFB サーバと呼ぶ。FrameBuffer は、メモリ上に置かれた画像データのことである。

RFB プロトコルでは、始めにプロトコルバージョンの確認、認証を行う。その後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前などが含まれている初期メッセージが送信される。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。更に、RFB クライアントの FramebufferUpdateRequest が来るとそれに答え返信する。変更部分だけを送信する理由は、更新がある度に全画面を送信していると、送信するデータ面、更新にかかる時間面において効率が悪いからである。

2.3　TreeStructure

TreeVNC はサーバーに接続してきたクライアントをバイナリツリー状に接続している。また、接続してきたクライアントをノードとし、その下に新たなクライアントを接続して

いくことでサーバーが画面のデータを配信する回数を抑えることで負荷分散している (図 2.1)。バイナリツリー状に接続することで、画像データのコピーを各ノードに負担させることができ、従来の VNC ではクライアントが N 台接続するとサーバー側が N 回コピーを行なって配信していた (図 2.2) が、この接続方法であれば各ノードが 2 回ずつコピーすることで配信を可能にしている。

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、子 Node にデータを流す機能、各 Node の管理、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、親 Node から送られてきたデータを自身の子 Node に流す機能、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。

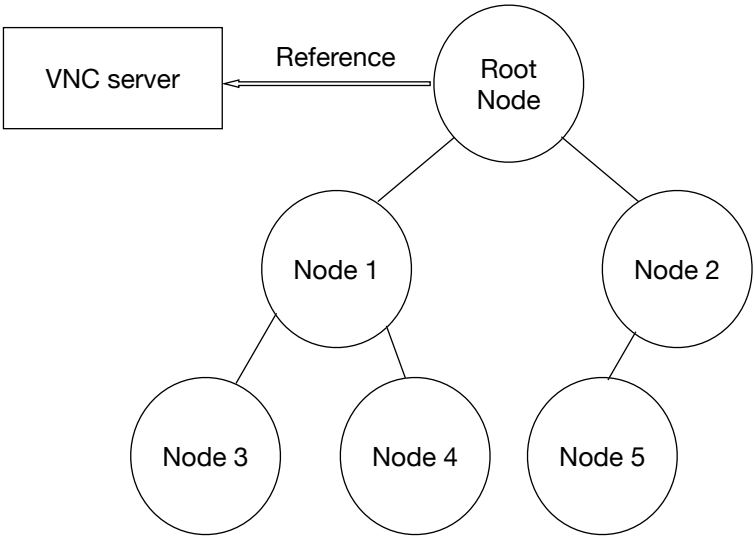


図 2.1: TreeVNC の接続方法

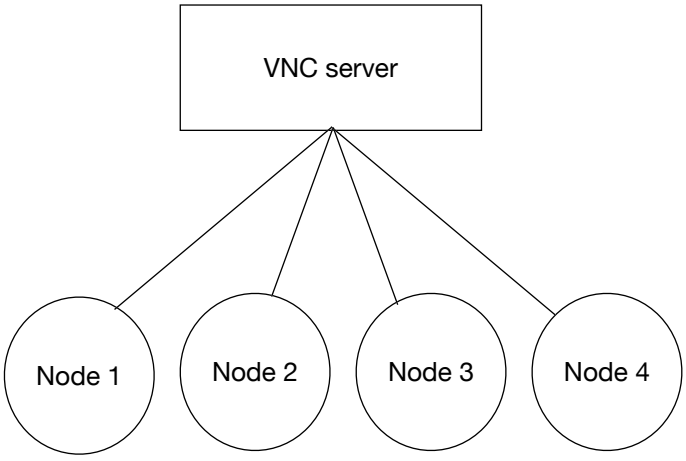


図 2.2: 従来の VNC の接続方法

2.4 通信経路

TreeVNC の通信経路は以下の六つである。

- Root Node から任意の Node に直接通信を行う send direct message (Root to Node)
- 任意の Node から Root Node に直接通信を行う send direct message (Node to Root)
- Root Node から木の末端までの全ての Node に通信を行う message down tree (Root to Node)
- 任意の Node から上に辿って Root Node まで通信を行う message up tree (Node to Root)
- Root Node から配信者の VNC サーバへの通信を行う send message (Root to VNC-Server)
- 配信者の VNC サーバから Root Node への通信を行う send message (VNCServer to Root)

2.5 メッセージ通信

TreeVNC の各 Node と VNCServer 間で通信されるメッセージを表 2.1 に示す。

通信経路	message	説明
send direct message (Node to Root)	FIND_ROOT	TreeVNC 接続時に Root Node を探す。
	WHERE_TO_CONNECT	接続先を Root Node に聞く。
	LOST_CHILD	子 Node の切断を Root Node に知らせる。
send direct message (Root to Node)	FIND_ROOT_REPLY	FIND_ROOT への返信。
	CONNECT_TO_AS_LEADER	左子 Node として接続する。接続先の Node が含まれている。
	CONNECT_TO	右子 Node として接続する。接続先の Node が含まれている。
message down tree (Root to Node)	FRAMEBUFFER_UPDATE	画像データ。EncodingType を持っている。
	CHECK_DELAY	通信の遅延を測定する。
message up tree (Node to Root)	CHECK_DELAY_REPLY	CHECK_DELAY への返信。
	SERVER_CHANGE_REQUEST	画面切り替え要求。
send message (Root to VNCServer)	FRAMEBUFFER_UPDATE_REPLY	画像データの要求。
	SET_PIXEL_FORMAT	pixel 値の設定。
	SET_ENCODINGS	pixel データの encodeType の設定。
	KEY_EVENT	キーボードからのイベント。
	POINTER_EVENT	ポインタからのイベント。
	CLIENT_CUT_TEXT	テキストのカットバッファを持った際の message。
send message (VNCServer to Root)	FRAMEBUFFER_UPDATE	画像データ。EncodingType を持っている。
	SET_COLOR_MAP_ENTRIES	指定されている pixel 値にマップする RGB 値。
	BELL	ビープ音を鳴らす。
	SERVER_CUT_TEXT	サーバがテキストのカットバッファを持った際の message。

表 2.1: 通信経路とメッセージ一覧

2.6 MulticastQueue

配信側の画面が更新されると VNC Server から画像データが FRAME_BUFFER_UPDATE メッセージとして送られる。その際、画像データの更新を複数の Node に同時に伝えるために Multicast Queue というキューに画像データを格納する。

2.7 木構造の再構成

バイナリツリーでの接続のため、 Node が切断されたことを検知できないと構成した木構造が崩れてしまい、新しい Node を適切な場所に接続できなくなってしまう。そこで木構造を崩さずに再構成を行う必要がある。

TreeVNC の木構造のネットワークトポロジーは Root Node が持っている nodeList で管理している。Node の接続が切れた場合、Root Node に切断を知らせる必要がある。この時 LOST_CHILD というメッセージで Node の切断を検知･木の再構成を行う。LOST_CHILD の検出方法には MulticastQueue を使用している (一定期間 MulticastQueue から画像データが取得されない場合、 MemoryOverFlow を回避するために Timeout スレッドが用意されており、これを検知した場合 Node の接続が切れたと判断する)。

2.8 ZRLEE

TreeVNC では、ZRLEE というエンコード方法でデータの圧縮を行う。ZRLEE は RFB プロトコルで使用できる ZRLE というエンコード方法を元に生成されている。

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属して送られて来る。Zlib は java.util.zip.deflater と java.util.zip.inflater でエンコード・デコードが行える。しかし、java.util.zip.deflater はデコードに必要な辞書を書き出すことができない (図 2.3)。従って、エンコードされたデータを途中から呼ぶと正しく解凍できない。

ZRLEE は 1 度 Root Node で受け取った ZRLE のデータを unzip し、データを update rectangle という画面ごと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デコードできるようにしている (図 2.4)。1 度 ZRLEE に変換してしまえば子 Node はそのデータを流すだけで良い。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成する必要がある。

2.9 ShereScreen

従来の VNC では、配信者が切り替わるたびに VNC の再起動、サーバー、クライアント間の再接続を行う必要がある。TreeVNC では、画面上にある ShareScreen ボタンを押すことで配信者の切り替えを実行できるように設定し、この問題に対処している。

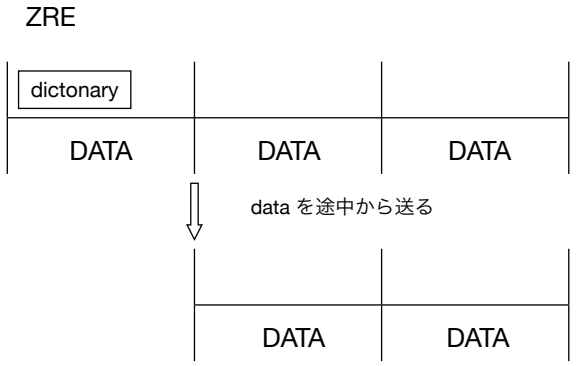


図 2.3: ZRE の問題点

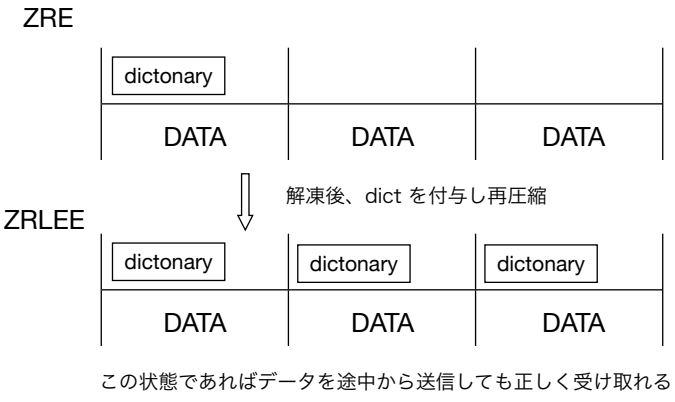


図 2.4: ZRLEE

ShareScreen 実行後、Root Node に対し SERVER_CHANGE_REQUEST というメッセージが送信される。このメッセージには ShereScreen ボタンを押した Node の番号やディスプレイの情報が付加されている。メッセージを受け取った Root Node は配信を希望している Node の VNC サーバーと通信を行い、切り替え作業に入る。

2.10 ネットワーク複数時の接続

TreeVNC は Root Node が複数のネットワークに接続している場合、(図 2.5) の様にネットワーク別に木構造を形成する。

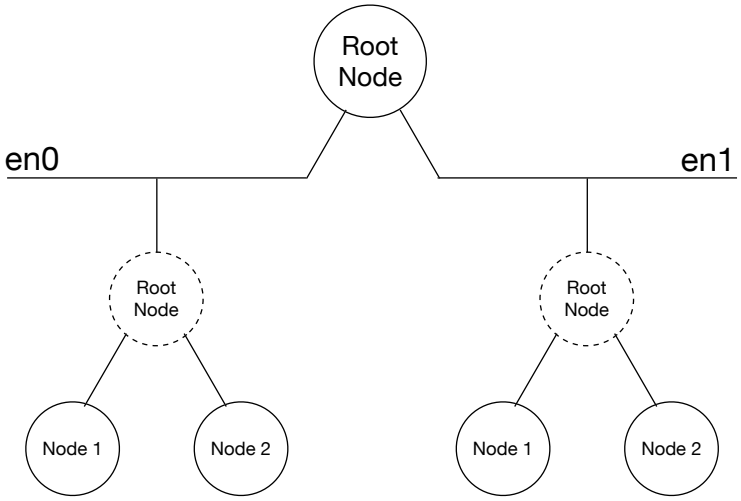


図 2.5: multinetworktree

TreeVNC は Root Node が TreeManager というオブジェクトを持っている。TreeManager は TreeVNC の接続部分を管理している。TreeManager では木構造を管理する nodeList が生成される。この nodeList を元に、新しい Node の接続や、切断検出時の接続の切り替え等を行う。Root Node の保持しているネットワーク毎に TreeManager を生成する。新しい Node が接続してきた際、 interfaces から Node のネットワークと一致する TreeManager を取 得し、 Node 接続の処理を任せる。

第3章　TreeVNC の改良点

3.1　VNCServer の暴走

ReceiverTask 側で正しくデータを受け取れなかったりといった予期しない

3.2　errordialog

TreeVNCServer にクライアントとして接続した際に、接続を許可するかどうか確認する authentication のポップアップが Root 側にも表示されてしまっていた問題を修正した。

第4章　マルチキャスト対応

4.1　有線接続との接続形式の違い

マルチキャスト通信では、サーバー側は一度の送信で接続しているデバイス全てにデータを届けることができる。

TreeVNC に無線 LAN 接続した場合、マルチキャスト通信を行うため、新たな Node を有線接続の際行なっていたバイナリツリー状での接続方式とは異なった方法で管理する (図 4.1)。

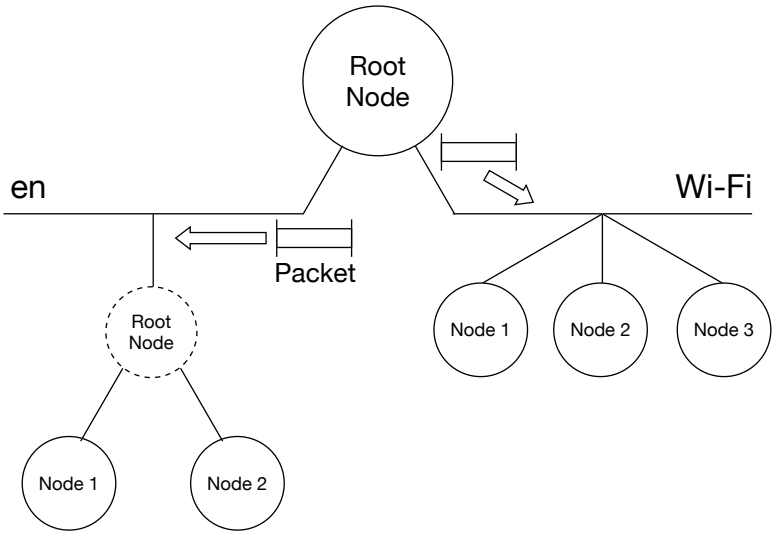


図 4.1: 無線 LAN 接続時の接続方法

4.2　Blocking

無線 LAN 通信は、有線接続と比較して一度に送信できるデータ量が少ない。そのためサーバー側が送信した更新データが正確にクライアントに送られない可能性がある。

対策として、データを分割 (Blocking) して送信する手法を実装した。

従来の TreeVNC では、配信側の画面が更新された場合、サーバーから FRAME_BUFFER_UPDATE メッセージが送信され、更新データは MulticastQueue というキューに格納される。

Blocking は、更新データを長方形 (Rectangle) で分割し、 MulticastQueue に格納する。MulticastQueue に格納されたデータは、

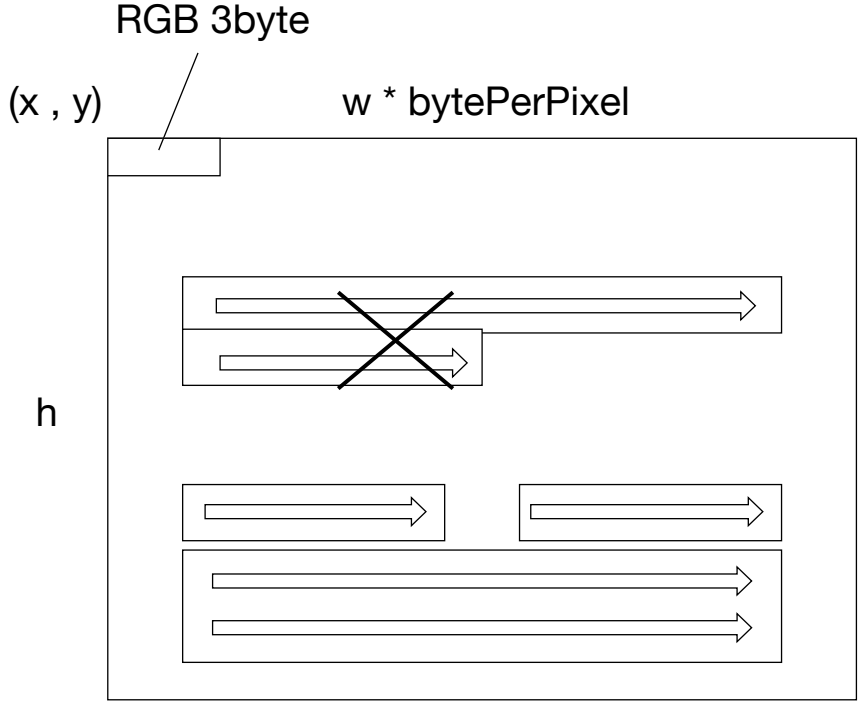


図 4.2: 更新データの分割方法

第5章　まとめ

本研究では TreeVNC の改良と無線 LAN 接続時の Multicast 対応を行った。これにより、有線接続環境がない状況でも無線接続での画面配信による講義等の円滑な進行を行えるようになった。

今後の課題としては、

謝辞

本研究の遂行，また本論文の作成にあたり、御多忙にも関わらず終始懇切なる御指導と御教授を賜わりました河野真治准教授に深く感謝したします。また、一年間共に研究を行い、暖かな気遣いと励ましをもって支えてくれた並列信頼研究室のみなさんに感謝致します。

最後に、有意義な時間を共に過ごした情報工学科の学友、並びに物心両面で支えてくれた両親に深く感謝致します。

2019 年 3 月

大城由也