Mintオペレーティングシステムにおける仮想ネットワークインタフェースの改善

平成30年2月15日

岡山大学 工学部 情報系学科

吉田 修太郎

背景と目的

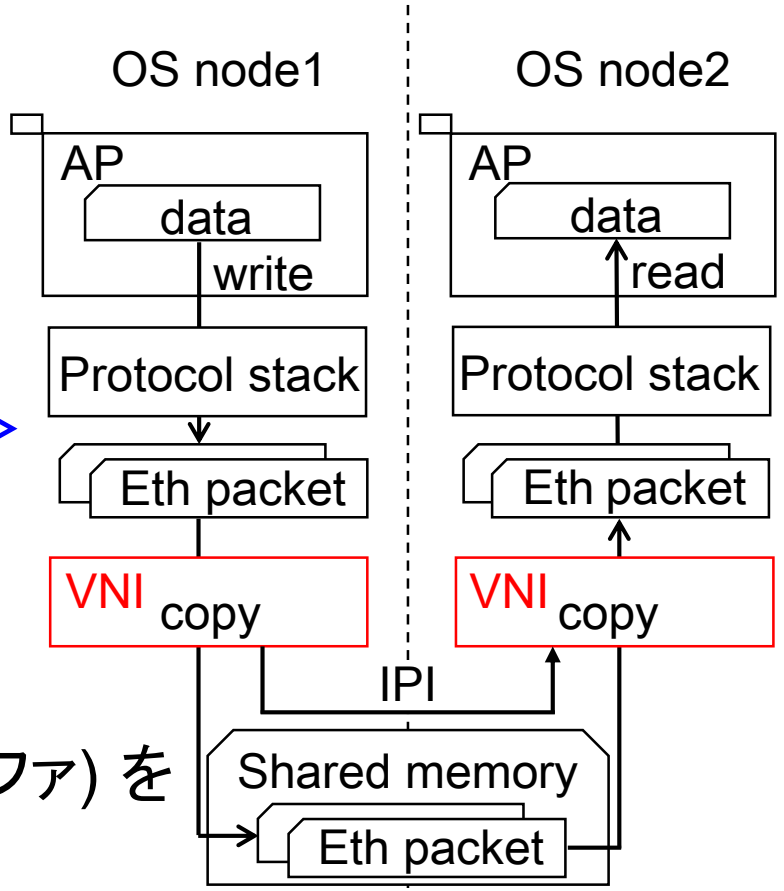
Mint:

1 台の計算機上で
複数の OS(OS ノード) を走行

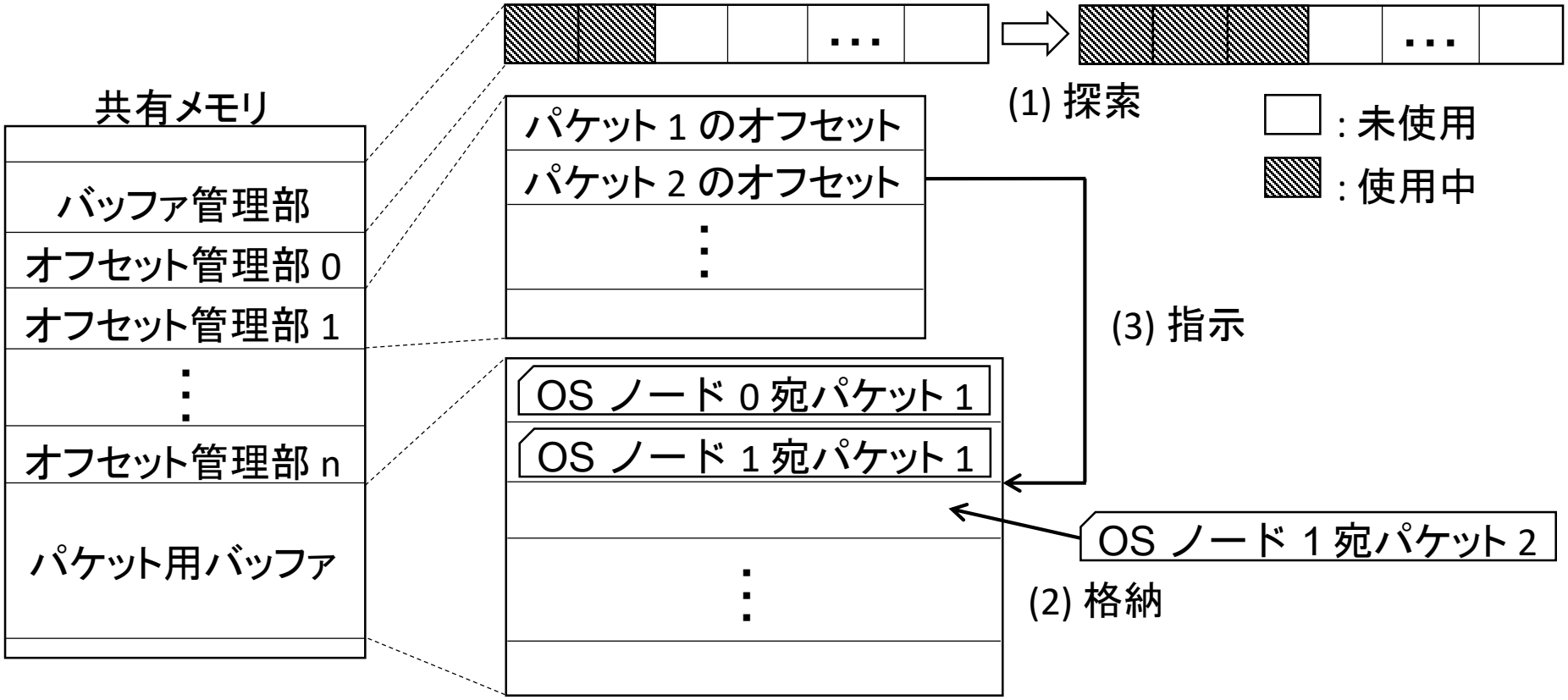
＜仮想ネットワークインタフェース (VNI)＞

(1) OS ノード間で Ethernet 互換の
通信を実現

(3) 共有メモリの特定領域 (送受信バッファ) を
介してパケットを送受信



既存の送受信バッファにおける構成と処理流れ



パケット用バッファ: パケットを格納

オフセット管理部: パケットの格納位置を管理

バッファ管理部: パケット用バッファの各領域が使用中か否か管理

既存の VNI における課題

(1) TCP プロトコルを用いた通信におけるスループットの向上
TCP プロトコルを用いた通信のスループットは約 128Mbps

(2) OS ノード間における排他制御の検討

(A) 既存の VNI における送受信バッファの構成では，
共有メモリの使用は OS ノード間で排他制御を要する

(B) 既存の VNI には OS ノード間における排他制御は未実装であり，共有メモリへのアクセスを時分割で行わない限り通信できない

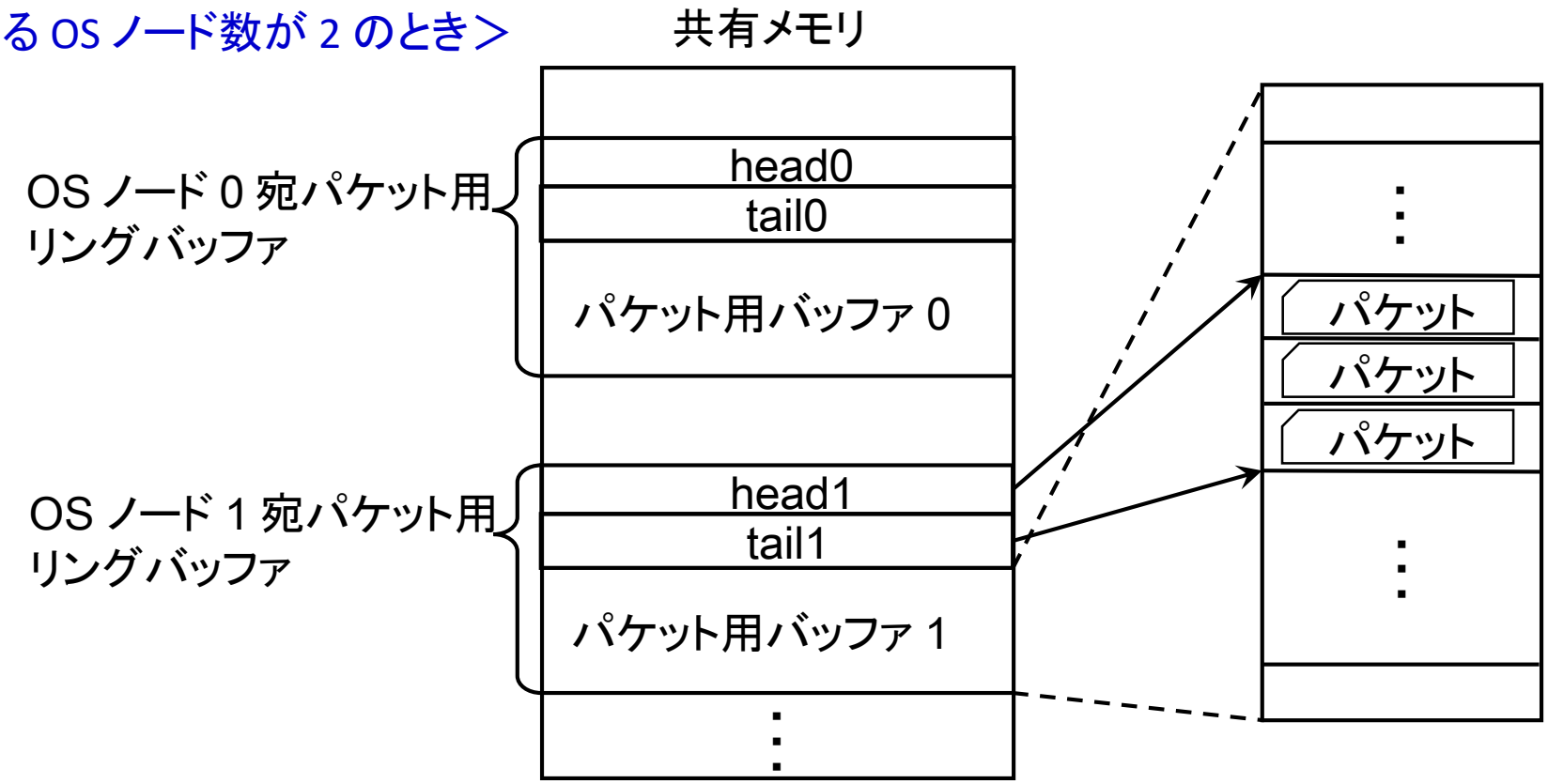
⇒ 通信性能の向上は望めない

⇓

送受信バッファの構成を見直し，共有メモリの使用について
OS ノード間における排他制御の必要性を検討

OS ノード間で競合を生じない送受信バッファ構成

＜通信する OS ノード数が 2 のとき＞



(1) パケットを格納する領域を宛先ごとに分割する
(2) 送受信バッファにおける 1 つの分割した領域に対する書込みは，
それぞれ 1 つの OS ノードのみが行う

➡ 送受信バッファの操作は OS ノード間で競合しない

スループット計測

＜目的＞

(1) 送受信バッファの改変による, VNI を用いた通信におけるスループットの変化を調査する

＜環境＞

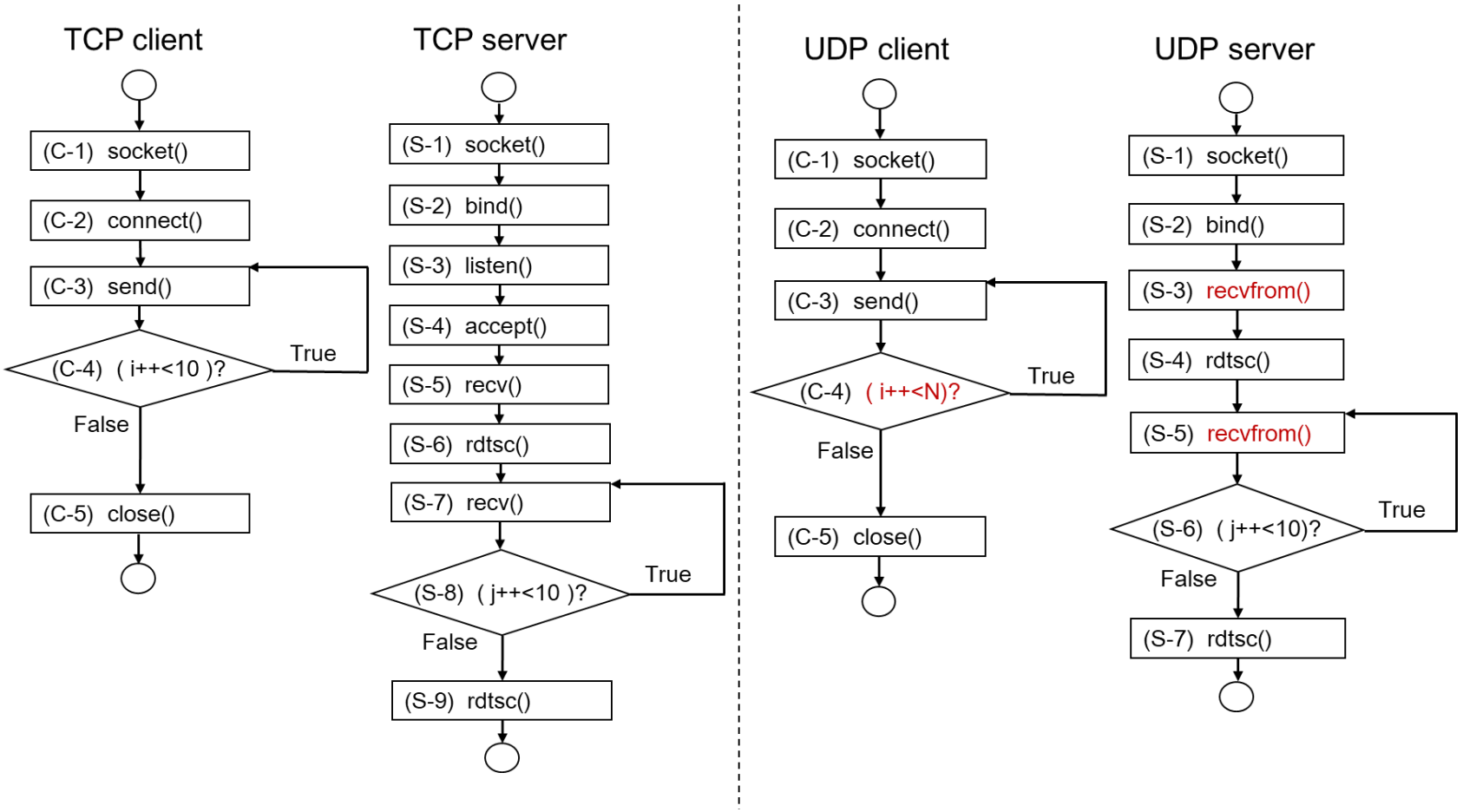
OS	Debian 7.11
カーネル	Mint(Linux kernel v3.15 から改変)
起動する OS ノード数	2
各 OS ノードの持つコア数	1
CPU	Intel Core i7-4770 (3.40GHz)
メモリの容量	16GB
メモリ I/O の帯域幅	25.6GB/sec
共有メモリのサイズ	16MB
送受信バッファのサイズ	392,200B

＜手法＞

(1) 通信 プロトコルとして UDP プロトコルを用いた場合と TCP プロトコルを用いた場合の 2 つの場合について通信のスループットを計測

(2) 計測にはユーザプログラムを用いる

計測用プログラムの処理流れ



(1) 1 回 15,000Byte の send()/recv() を 10 回繰り返す

(2) recv() したデータの量を，recv() に要した時間で割る

スループットの計測結果

＜結果＞

送受信バッファ	TCP	UDP
改変前	128Mbps	65Gbps
改変後	10Gbps	75Gbps

＜改変前と改変後の比較＞

(1) UDP プロトコルを用いた通信と TCP プロトコルを用いた通信のどちらの場合においてもスループットは向上

(2) UDP プロトコルを用いた通信と比較して, TCP プロトコルを用いた通信の方がスループットの上昇率が大きい

まとめ

＜実績＞

(1) 送受信バッファにおける構成の再検討

(2) 送受信バッファの再実装

(3) VNI を用いた通信のスループット計測

(A) 既存の VNI による通信

(B) 送受信バッファを再実装した VNI による通信

＜今後の課題＞

再実装した送受信バッファ構成では，通信する OS ノードの増加に伴い，より多くのパケット用バッファを要する

予備スライド

改変後の送受信バッファにおける問題点

(1) 通信する OS ノードの増加に伴い，より多くのパケット用バッファを要する

(2) n 個の OS ノード間で通信するとき，要するパケット用バッファは ${}_nC_2 \times 2$ 個

例：通信する OS ノード数 100 のとき

${}_{100}C_2 \times 2 = 4950$

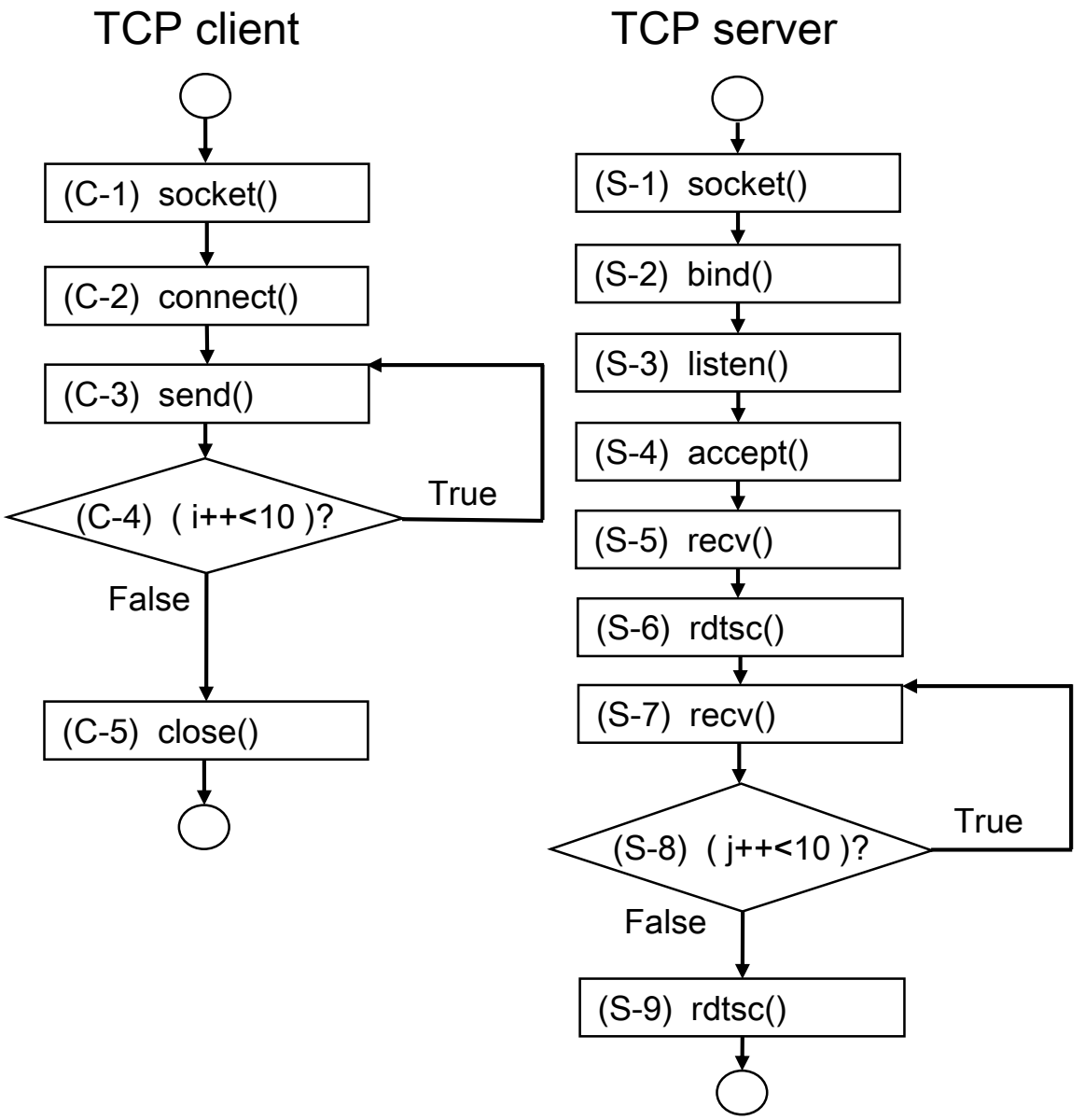
(1) パケット用バッファの大きさの最大値は 2 つの OS ノード間で通信する場合の 1/2475

(2) パケット用バッファの大きさを 392,200B (計測時と同じ大きさ)とした場合，送受信バッファの大きさは約 1.8GB

計測用プログラムの処理流れ (TCP)

(1) 1 回 15,000Byte の send()/recv() を 10 回繰り返す

(2) recv() したデータの量を，recv() に要した時間で割る



計測用プログラムの処理流れ (UDP)

(1) 1 回 15,000Byte の send() を N 回 (N>10)，recvfrom() を 10 回繰り返す

(2) recvfrom() したデータの量を，recvfrom() に要した時間で割る