高遅延ネットワーク環境におけるiSCSIリードアクセス時のTCP輻輳ウィンドウ制御手法の性能評価

豊田　真智子[†]　山口　実靖[††]　小口　正人[†]

ブロードバンドネットワーク技術の発展により，低コストで構築可能なIP-SANが次世代SANとして期待され，その代表的技術であるiSCSIが注目を集めている．しかしiSCSIを用いたストレージアクセス時には，複雑なプロトコル処理などのオーバヘッドによりTCP/IPのみで構築されたネットワークよりも性能が劣化することが確認されている．

本稿では，スループットのばらつきを抑制するために提案した輻輳ウィンドウコントロール手法を用いることにより，性能を向上させる手法について述べる．提案手法をiSCSIストレージアクセスに適用し，性能評価を行った．その結果，高遅延環境においては提案手法を用いない場合よりもスループットが向上し，本手法の有効性が確認された．

Performance Analysis of Controlling TCP Congestion Window on iSCSI Read Access in Long-Latency Environment

Machiko Toyoda,[†] Saneyasu Yamaguchi[††] and Masato Oguchi[†]

As the broadband networks are widely used, IP-SAN is expected as the next generation's SAN because of its low cost. The iSCSI protocol, represented as IP-SAN technology, is becoming increasingly important. However, the performance of iSCSI network is lower than that of network based only on TCP/IP, due to its complex protocol processing overhead in accessing storage with the iSCSI.

In this paper, we present performance improvement using dynamic Congestion Window control method to balance throughput unevenness. We evaluated iSCSI network performance with remote storage access on iSCSI protocol using the proposed model. As a result, iSCSI performance improved compare to the case without the proposed model in long-latency environment, and we confirmed the proposed model is effective.

1. はじめに

計算機システムの性能向上やGigabit Ethernetの普及などにより，大容量のデータを高速に処理できるようになった．それに伴い，マルチメディアコンテンツなどのデータを大量に蓄積するアプリケーションも登場し，計算機システムが処理するデータ量が飛躍的に増大している．ストレージの増加に比例して増大する管理コストの削減を目的としてSAN（Storage Area Network）が登場し，すでに高い評価を得ている．現在普及しているFC-SANは，サーバとストレージ間をファイバチャネルで接続する．ファイバチャネルはプロトコル処理が軽く，サーバのCPUにかかるデータ転送負荷が小さいという利点を持つ反面，異なるメーカ間での相互接続が困難であったり，対応製品が高価であるため導入コスト，管理コストが増加するといった問題点も指摘されている．そのため，安価なEthernetとTCP/IPを用いて構築するIP-SANが近年期待され始めた．現在IP-SANで注目を集めている規格が，2003年2月にIETFにより承認されたiSCSIである[1)2)]．iSCSIでは，サーバ（Initiator）とストレージ（Target）間のデータのやりとりをSCSIコマンドで行う．これにより，遠隔地にあるストレージデバイスが直接接続されているかのごとくシームレスにアクセス可能である．

このiSCSIを用いたストレージアクセスにおける性能を向上させるため，我々は文献3)において，輻輳ウィンドウを動的にコントロールすることによりストレージアクセスを行う手法を提案した．本稿では，iSCSIストレージアクセス時に輻輳ウィンドウコント

† お茶の水女子大学
Ochanomizu University
†† 東京大学 生産技術研究所
Institute of Industrial Science, The University of Tokyo

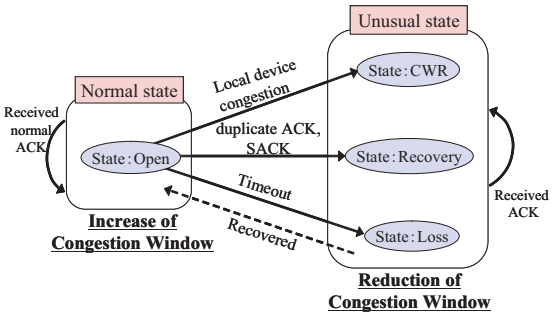

図 1 Linux TCP 実装の状態遷移図

ロール手法を適用し，遅延時間が異なる各環境においてその性能を評価する．高遅延環境において本手法を適用した場合，スループットが最大約 28%向上することが確認された．また，iSCSI プロトコルにおけるパケットの振る舞いについて述べ，本手法の有効性についても議論する．

本稿は以下のように構成される．まず 2 節で研究背景を述べる．3 節では輻輳ウィンドウを動的にコントロールする手法を紹介する．4 節で提案手法の性能評価を行うため，遅延時間が異なる環境において，提案手法を適用した場合と適用しない場合それぞれについて iSCSI シーケンシャルリードアクセスを行い，その結果を示す．5 節で高遅延環境下における iSCSI プロトコルのパケットの振る舞いについて述べ，性能への影響について議論する．6 節で関連研究について触れ，最後に 7 節でまとめを述べる．

2. 研究背景

2.1 Linux TCP 実装

TCP では，輻輳制御において輻輳ウィンドウという概念を用いている．輻輳ウィンドウとは，ネットワークの輻輳制御を目的としてデータ送信側が自主的に制限するためのパラメータであり，受信側からの確認応答パケット (= ACK) なしに連続送信を行う最大パケット数である．通常，輻輳ウィンドウは ACK を 1 つ受信するごとに 1 ずつ増加する．本実験で用いた Linux OS においては，通信時の状態が正常であれば ACK 受信ごとに輻輳ウィンドウは増加するが，エラーが検出されると異常と判断され，輻輳ウィンドウは低下する（図 1）．輻輳ウィンドウが低下する原因としては，送信側デバイスドライバのバッファが溢れることによる Local Congestion エラーを検出した場合（CWR），重複 ACK，SACK を受信した場合（Recovery），タイムアウトを検出した場合（Loss）の 3 つが挙げられる．また，Linux の TCP 実装では，通信中に一度設定された輻輳ウィンドウは，そのウィンドウの値を使い切らない限りは変化しないという特徴を持ち，この時スループットはほぼ一定の値で安定することが確認されている[4]．

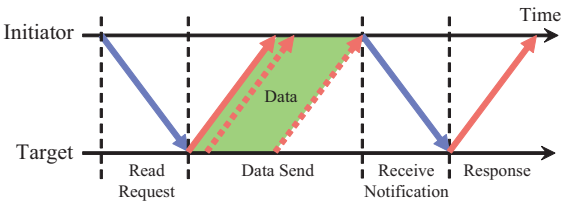

図 2 iSCSI シーケンシャルリードアクセスのシーケンス

2.2 iSCSI ストレージアクセスにおける問題点

iSCSI は，サーバに直接接続する従来のストレージ接続形態である DAS（Direct Attached Storage）において広く用いられている SCSI コマンドを，TCP/IP パケット内にカプセル化してネットワークに転送する技術である[1)2)]．FC-SAN では接続距離が 10km 程度と限定されてしまうのに対し，iSCSI では TCP/IP ネットワークを利用することにより接続距離に制限が存在しないため，ストレージのアウトソーシングやデータセンタへのデータバックアップといった長距離接続での利用に対する期待が大きい．

図 2 は，iSCSI を用いたシーケンシャルリードアクセスを行った場合のパケット転送シーケンスである．アプリケーションにおいて read システムコールが発行されると，iSCSI 層において Read リクエストである SCSI Command PDU が送信される（図 2 における“ Read Request ”）．SCSI Command PDU には転送要求するブロックサイズが記述されているため，これを受信した Target では SCSI Data-In PDU を返信後（図 2 の“ Data Send ”における実線矢印），要求された大きさのデータを連続して送信する（図 2 の“ Data Send ”における点線矢印）．Initiator では，届いたデータに対して TCP レベルで ACK を返信し（図 2 における“ Receive Notification ”），最後のデータを受信したことを示す ACK が Target に到着すると，応答パケットである SCSI Response PDU を Target が送信し（図 2 における“ Response ”），1 回の Read コマンドが終了する．

TCP/IP のみで構成されたネットワークでソケット通信を行っている場合は（以降，ソケット通信と呼ぶ），時間が経過するにつれて TCP のセルフクロッキング機能が動作し，ACK の受信タイミングに合わせてデー

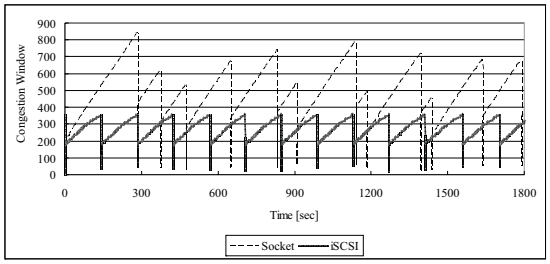

図 3 ソケット通信と iSCSI 通信における輻輳ウィンドウの時間変化

タが少しずつ送信されるため，パケットの送出速度が適切に調節される．一方 iSCSI では，常に Read リクエストを受信した後にデータが一斉送信されるため，いつまでたってもバーストが消えることがない．そのため，ソケット通信に比べ CWR エラーが起こりやすくなり，輻輳ウィンドウの成長も小さい．図 3 は片道遅延時間 16ms におけるソケット通信と iSCSI 通信の輻輳ウィンドウの時間変化を示したものである．この図において，輻輳ウィンドウの低下はすべて CWR エラーによるものである．ソケット通信の輻輳ウィンドウは，変動はあるものの比較的大きな値まで成長するのに対し，iSCSI 通信の輻輳ウィンドウはソケット通信よりも成長が小さいことが確認できる．また，我々は文献 4) において，輻輳ウィンドウとスループットが密接に関連しており，輻輳ウィンドウが増加減少の鋸型の変化を繰り返す場合には，輻輳ウィンドウの振る舞いに伴いスループットも不安定になることを述べた．iSCSI を利用したストレージアクセスを行う場合は，CWR エラーの頻度を減らし，輻輳ウィンドウを大きな値で保つことが性能向上につながる．

3. 輻輳ウィンドウコントロール手法

本節では，iSCSI ストレージアクセスにおいて確認されるスループットのばらつきを抑制し，性能を向上させるために提案した輻輳ウィンドウコントロール手法について述べる．

前節で述べたように，iSCSI を用いたネットワークにおけるパケットの振る舞いは，TCP/IP のみで構成されたネットワークの振る舞いと異なるため，性能を向上させるためには iSCSI の振る舞いを考慮した特別な処理が必要である．そこで我々は，輻輳ウィンドウとスループットが関連して動作することに注目し，輻輳ウィンドウを動的にコントロールし，その増減を収束させる手法を提案した[3]．輻輳ウィンドウの収束は iSCSI アクセスにおけるブロックサイズを調節することで行う．本手法の概要を図 4 に示す．

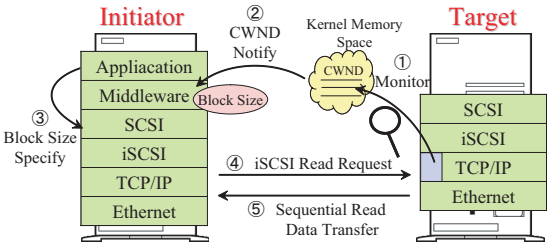

図 4 輻輳ウィンドウコントロール手法の概念図

輻輳ウィンドウはカーネル空間において管理される TCP パラメータであるため，通常のユーザプログラムでその値を知ることはできない．そこで，TCP ソースコードにモニタ用の関数を挿入し，ユーザ空間からもアクセス可能なカーネルメモリ空間に記録する仕組みを作成した．これにより，TCP パラメータをカーネルメモリ空間にアクセスするための特殊ファイルを読み出すことによって確認が可能となる．輻輳ウィンドウコントロール手法は，この仕組みを Target に実装し，Target からの輻輳ウィンドウ通知を受けて Initiator のアプリケーションがストレージアクセスのブロックサイズを調節する仕組みをミドルウェアとして提供する機能を持つ．本手法によるコントロール手順を以下に示す．

(1) Target で輻輳ウィンドウをモニタし，変化を観察する
(2) 観察中に CWR が検出され，輻輳ウィンドウが低下した場合にはその時の輻輳ウィンドウ値を，輻輳ウィンドウが一定値であると判断した場合には輻輳ウィンドウの限界値（CWR エラーが起こらなかった場合の最大値）を Initiator に通知し，Target においても通知した輻輳ウィンドウ値を記録する
(3) 通知を受けた Initiator では，ミドルウェアが輻輳ウィンドウからブロックサイズを決定し，アプリケーションがブロックサイズを再指定する
(4) Initaitor から Target にシーケンシャルリードコマンドを送信し，ストレージアクセスを行う
(5) Target が Initiator に向けて要求されたブロックサイズのデータ転送を実行する
(6) CWR を検出するか，一定値であると判断する度にこの処理を繰り返す

本手法適用後，輻輳ウィンドウは CWR エラーが起こらない限界値で一定に保たれ，その時のブロックサイズが本手法から計算される最適値となる．なお，本手法においてミドルウェアが指定するブロックサイズは以下の式を用いて計算した．

表 1　使用計算機

CPU	Intel Xeon 2.4GHz
Main Memory	512MB DDR SDRAM
OS	Initiator, Target ： Linux2.4.18-3 Dummynet ： FreeBSD 4.9 - RELEASE
NIC	Initiaotr, Target : Intel PRO/1000XT Server Adapter Dummynet : Intel PRO/1000MT Server Adapter

転送ブロックサイズ [*byte*] = 輻輳ウィンドウ値×最大転送単位（*MTU*）

本実験時の MTU（Maximum Transmission Unit）は Ethernet の最大セグメント長（1500Byte）から TCP/IP ヘッダ（オプションを含む）を除いた 1448Byte である．また，Target における輻輳ウィンドウのモニタは数秒間隔で行うため，そのオーバーヘッドが性能に及ぼす影響は十分小さいと考えられる．

4. 輻輳ウィンドウコントロール手法の性能測定実験

本節では，輻輳ウィンドウコントロール手法を実装した環境における遅延時間が異なる場合の性能を測定するために，iSCSI ストレージアクセスにおいて輻輳ウィンドウコントロール手法を用いた場合と用いない場合の実験を行い，提案手法が与える影響について比較を行った．

4.1 実験環境

本実験は以下の環境で行った．Initiator と Target 間は Gigabit Ethernet で接続し，TCP/IP 接続を確立した．遅延がない場合の実験には 1000Base-T スイッチングハブを，遅延がある場合の実験には人工的な遅延装置である FreeBSD Dummynet[5] を Ethernet の接続途中に挿入した．Initiator，Target，Dummynet はすべて PC 上に構築し，Initiator，Target の OS には Linux を，Dummynet の OS には FreeBSD を用いた．実験で使用した計算機の環境を表 1 に示す．

また，本実験で用いた iSCSI 実装において，Target にはニューハンプシャー大学 InterOperability Lab が提供する UNH IOL reference implementation ver.3 on iSCSI Draft 18[6] を用いた．この UNH 実装では，大きなブロックサイズで read コマンドを発行しても，SCSI 層において要求したブロックサイズより小さなブロックサイズに分割されてしまい，これにより iSCSI ストレージアクセスの性能が大きく低下してしまうことが確認されている[7]．本実験ではこの実装による性能測定への影響を避けるため，Initiator に UNH 実装を用いず，UNH 実装の Initiator と同等の機能を持ち，かつ大きなブロックサイズのデータ転送も行える自作 Initiator を用いて実験を行った．この自作 Initiator は通常のユーザ空間のアプリケーションとして動作し，iSCSI Target と TCP/IP コネクションを確立して iSCSI プロトコルで通信を行うものである．

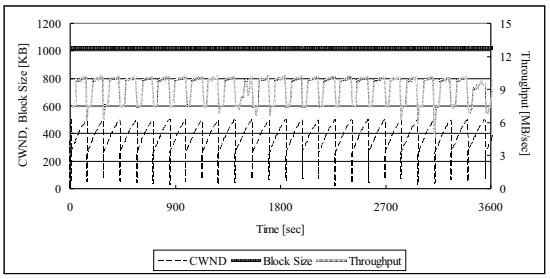

図 5　iSCSI シーケンシャルリードアクセス時の輻輳ウィンドウ，ブロックサイズ，スループットの時間変化：片道遅延時間 16ms

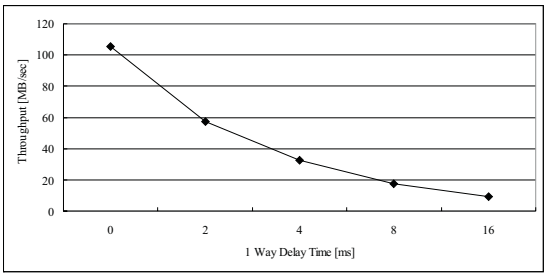

図 6　iSCSI シーケンシャルリードアクセス時の平均スループット

4.2 実験概要

実験は Initiator から Target へシーケンシャルリードアクセスを行い，遅延時間を変化させた場合の性能測定を行う．この時 raw デバイスを用いることによりキャッシュの影響を排除した．Initiator が指定するブロックサイズは，輻輳ウィンドウコントロール手法を用いない場合は 1024KB に，輻輳ウィンドウコントロール手法を用いた場合はその初期値を 1024KB に設定し，Target から読み込むデータサイズの合計を 100GB として実験を行った．また，iSCSI を使用した場合のネットワーク性能に焦点を当てて評価を行うため，Target は UNH 実装が提供するメモリモードで動作させた．

4.3 提案手法を用いない場合の実験結果

図 5 に，提案手法を用いない場合の実験結果として，片道遅延時間 16ms(Round Trip Time：32ms) における輻輳ウィンドウ，ブロックサイズ，スループッ

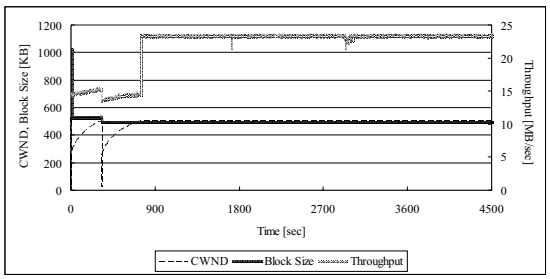

図 7 提案手法を用いた場合の iSCSI シーケンシャルリードアクセス時の輻輳ウィンドウ，ブロックサイズ，スループットの時間変化：片道遅延時間 8ms

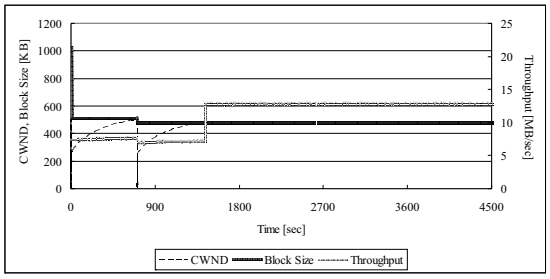

図 8 提案手法を用いた場合の iSCSI シーケンシャルリードアクセス時の輻輳ウィンドウ，ブロックサイズ，スループットの時間変化：片道遅延時間 16ms

トの時間変化を示す．この図における輻輳ウィンドウ低下の原因はすべて CWR エラーによるものである．輻輳ウィンドウの増減に伴いスループットも変化を示し，輻輳ウィンドウが最大値を示す時と減少した直後におけるそれぞれのスループットを比較したところ，約 3MB/sec もの差があることが確認された．平均スループットが 10MB/sec 程度の環境における 3MB の差は，かなり大きな違いである．

また，遅延時間を変化させた場合において 100GB のデータをリードした時の平均的なスループットを図 6 に示す．遅延の増加に伴い，iSCSI の性能が著しく低下している様子が確認できる．本実験では，ディスクアクセスを伴わないメモリモードにおいてリードアクセスを行っているため，この結果はネットワークのみが与える性能であり，高遅延ネットワーク環境において iSCSI プロトコルが提供できる限界の性能であると言える．

4.4 提案手法を用いた場合の実験結果

次に提案手法を用いた場合の実験結果として，図 7 に片道遅延時間 8ms の環境，図 8 に片道遅延時間 16ms の環境における輻輳ウィンドウコントロール手法を適用した時の輻輳ウィンドウ，ブロックサイズ，スループットの時間変化を示す．Target において CWR

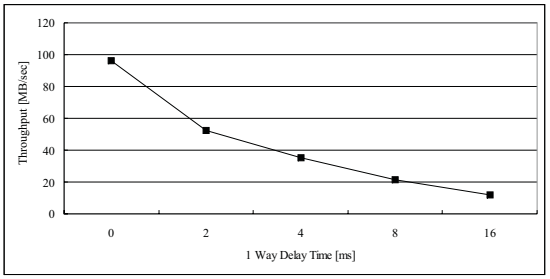

図 9 提案手法を用いた場合の平均スループット

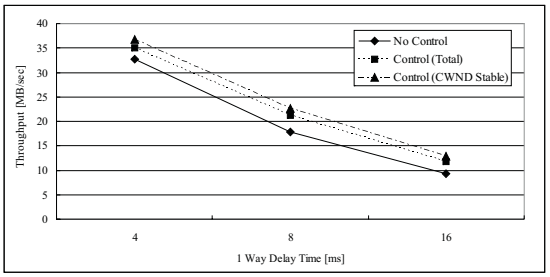

図 10 高遅延時の平均スループットの比較

エラーを検出すると Initiator のミドルウェアが機能し，アクセスブロックサイズを変化させることによって輻輳ウィンドウをコントロールする．提案手法を用いない場合の通信と異なりブロックサイズがやや低い値に設定されるが，輻輳ウィンドウが一定値となった後のスループットは，鋸型の変化をする時に比べ大幅に向上している様子が確認される．

図 9 は，遅延時間を変化させた場合において，100GB のデータをリードした時の平均スループットである．また，高遅延環境において輻輳ウィンドウをコントロールした場合としない場合，さらにコントロールして輻輳ウィンドウが一定値となった後の 3 つの場合の各平均スループットを比較したグラフを図 10 に示す．全体的な性能では，遅延の増加に伴いその性能が大きく低下をしている様子が図 9 から確認される．しかし，提案手法を用いた場合と用いない場合のスループットを比較すると，輻輳ウィンドウをコントロールした場合には性能低下を抑える効果があることが図 10 から確認される．また，輻輳ウィンドウ一定値後のスループットが 3 つの中で最も良い性能を示している．シーケンシャルリードを行う合計データを 100GB からさらに大きくした場合は，すべてのデータをリードし終えるまでの時間がより長くなってしまうため，リードにかかった総時間に対する輻輳ウィンドウが一定値となるまでにかかった時間の割合がより小さなものになることがわかる．このことから，本実験で行っ

た 100GB より大きなデータをリードする場合には，輻輳ウィンドウをコントロールしない場合との性能差がさらに広がることが予想される．遅延が小さな環境では大きなブロックサイズでデータをやりとりする方が高いスループットとなるが，遅延が大きな環境においては輻輳ウィンドウの増減を抑え，その値を一定に保つ方が性能が向上すると言える．

5. 高遅延環境における iSCSI プロトコルの振る舞いと性能への影響

本節では，高遅延環境下でシーケンシャルリードアクセスを行った場合のパケットの振る舞いについて説明し，輻輳ウィンドウコントロール手法が高遅延環境下で有効である理由を考察する．

前節の実験結果より，高遅延環境においてはブロックサイズが多少小さくなっても輻輳ウィンドウを一定に保つ方が性能が向上するという結果を得た．ブロックサイズは一度に送信するデータ長であるため，一般的にはブロックサイズが大きい方がスループットは向上すると考えられる．しかし，あまりブロックサイズを大きくしすぎるとデータ送信側の送信バッファが溢れ，CWR エラーが生じる．この結果 TCP 実装により送信データ量が多すぎたと判断され，輻輳ウィンドウは大幅に減少する．その場合，高遅延環境においてはパケットを送信してから ACK が返信されるまでの時間が長くなり，何もせずに待っている無駄な時間が増加する．従って，一度に送信できるパケット数を増やすことで，待ち時間を短縮し無駄の少ない通信を行うことができると考えられる．

遅延が小さな場合と大きな場合それぞれにおいて，シーケンシャルリードアクセスを行った際のパケットの振る舞いを図 11，図 12 に示す．これらの図と図 13 に示したパケットの振る舞いは，図 2 における“ Data Send ”部分のパケットの振る舞いをさらに詳細に表したものである．また，Target から Initiator に向かう実線矢印はデータが含まれたパケット（Data），Initiator から Target に向かう点線矢印は確認応答パケット（Ack）を意味し，Data 矢印の本数が輻輳ウィンドウに等しくなる．遅延が小さな環境でシーケンシャルリードアクセスを行った場合，Target が送信したデータに対する ACK は比較的早い間隔で返信されるため，Target における通信の待ち時間は短い（図 11）．しかし，遅延が大きくなると応答時間も長くなるため，送信したデータに対する ACK の到着も遅くなり，Target における通信の待ち時間は大幅に増加する（図 12）．これらの環境においては輻輳ウィンドウが定期的に変動するため，一度に送信できるパケット数もその都度異なる．それに対し，輻輳ウィンドウコントロール手法を用いたシーケンシャルリードアクセスを行った場合のパケットの振る舞いが図 13 である．本手法を適用し，輻輳ウィンドウが一定値になった場合には Target から Initiator に送信される Data パケット数が一定値に保たれる．この場合，無駄な待ち時間の間にも CWR エラーが生じない範囲で最大限のパケットを送信し続けることができる．そのため，1 回のシーケンシャルリードアクセスにおけるブロックサイズはやや小さくなるが，その間隔は短い周期で繰り返されるため，効率的にデータを送信することができる．輻輳ウィンドウのコントロールを行わない図 12 の場合では，一度に送信できるパケット数が多い時には待ち時間は減少するが，1 回のシーケンシャルリードアクセスにおけるブロックサイズが大きいために CWR エラーが定期的に起こり，その度に輻輳ウィンドウは非常に小さな値となる．輻輳ウィンドウが減少している間の通信は，少数のパケットを送信した後，長い遅延時間を得て ACK を受け取るまでは待つことしかできず，結果としてシーケンシャルリードアクセス 1 回に要する時間が長くなってしまう．この非効率な振る舞いにより，高遅延時には性能が劇的に低下する結果となる．すなわち，遅延が大きくなるほど輻輳

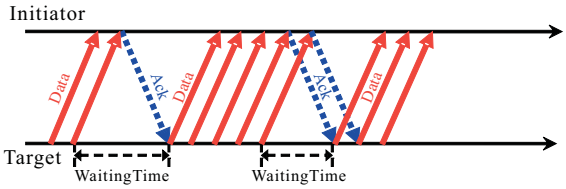

図 11 低遅延環境におけるシーケンシャルリードアクセス

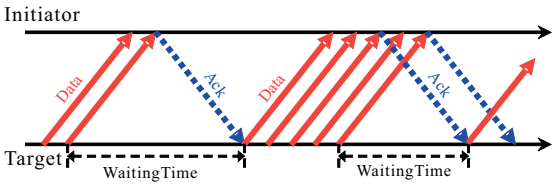

図 12 高遅延環境におけるシーケンシャルリードアクセス

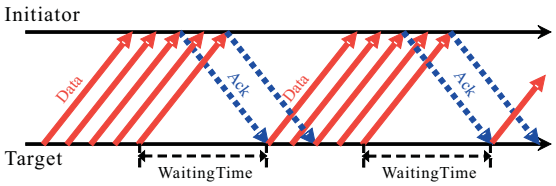

図 13 高遅延環境において提案手法を適用した場合のシーケンシャルリードアクセス

表 2 各遅延時間におけるスループット増加率

1 Way Delay Time	Throught Improvement
0ms	-9.1%
2ms	-7.9%
4ms	7.0%
8ms	19.7%
16ms	28.3%

ウィンドウの低下を抑える重要性が高まる．

表 2 に，前節の実験結果から求めた，提案手法を用いない場合に対する提案手法を用いた場合のスループット増加率を示す．遅延が小さい場合は，提案手法を適用するとブロックサイズがやや縮小することにより通信性能が少し低下するが，遅延が大きくなるにつれ提案手法を適用する方が性能は向上し，本実験における最大遅延時間である 16ms の場合，約 28%もの性能向上を実現することができた．

6. 関 連 研 究

iSCSI の関連研究としては文献 8)〜11)，TCP 輻輳ウィンドウに関連した研究としては文献 12)〜14) などが挙げられる．

文献 8) は iSCSI のソフトウェア実装とハードウェア実装を比較し，CPU 利用率という点を除いてはソフトウェア実装の方が性能が良いという結果を示している．文献 9) では，iSCSI と NFS の比較を行い，ファイル操作などの一般的な操作におけるパフォーマンスや，ベンチマークを用いた総合的なパフォーマンスを測定している．これらの研究は，iSCSI の性能を知る上では有用な研究であるが，システム内部の振る舞いについて把握し，性能を評価しているものではない．

文献 10) は，iSCSI の性能が，複数の階層構造を持つことやプロセスの重複により低下するものであると指摘し，“ quanta ”と呼ばれる固定のデータ単位でデータハンドリングを行うこと提案している．iSCSI 性能低下の原因についての着眼点は同じであるが，性能向上のアプローチが異なる．

文献 11) は，iSCSI Target の内部実装を考慮した iSCSI の性能評価を行っている．性能向上のため，カーネルや iSCSI ソフトウェアの設計に変更を加えるというアプローチであり，既存ソフトウェアには基本的に変更を加えず，ミドルウェアで対応している我々とは手法が異なるものである．また，高遅延環境における評価は行われていない．

文献 12) は，高速かつ高遅延なネットワークにおける既存 TCP の問題について触れ，パケット送信間隔を調節し，ネットワークへの負荷を抑制するものである．しかし，複数の iSCSI ストリームを用いることで帯域を有効に利用するというアプローチであるため，基本形態として単一コネクションにおける性能を評価した本研究とは根本的に異なる．

文献 13) では，高遅延環境においては CWR エラーを輻輳とみなし輻輳ウィンドウを減少させると通信性能を低下させるが，並列アプリケーションを実行した場合には CWR エラーを輻輳とみなす方が性能が向上することについて述べ，その解決法を紹介している．本実験は単一のコネクションを利用した実験を行っているが，iSCSI の使用用途によっては複数コネクションによる利用も想定される．文献で紹介されているアプローチは，CWR エラーを輻輳とみなす場合およびみなさない場合のどちらか一方の実装を行うものであり，これらを使い分けることはできない．そのため，単一または複数コネクションのどちらかを利用するかが明らかである場合には有効であると思われるが，これらが混在する環境においてはかえって性能が低下する可能性も考えられる．また，CWR エラーの回避手法が示されているが，この方法ではオーバヘッドが生じてしまうと述べられており，少ないオーバヘッドで CWR エラーを回避し，単一または複数コネクションのどちらにも対応できる本手法の方が適用性が高いものであると言える．

文献 14) では，長距離，大容量ネットワークにおける TCP の問題について触れ，新しいトランスポートプロトコルとして提案されている，HSTCP（HighSpeedTCP），Scalable TCP，FAST，SABUL（Simple Avaliable Bandwidth Utilization Library）の輻輳ウィンドウの振る舞いを紹介し，評価を行っている．これらの文献で紹介されている技術は，輻輳ウィンドウがシステム性能に関連していることについて議論されている点では共通しているが，既存の TCP を改変することで性能向上を目指すものである．従って，広く一般に普及している既存の TCP をそのまま利用した性能向上を目的とする本研究とは異なったものである．

7. ま と め

本稿では，iSCSI ストレージアクセス時に確認されるスループットのばらつきを抑制するために提案した輻輳ウィンドウコントロール手法を，iSCSI シーケンシャルリードアクセスに適用し，遅延時間が異なる場合における性能評価を行った．iSCSI はバースト性の

高い通信を行うプロトコルであるため，遅延が大きくなるほど輻輳ウィンドウ低下の影響が性能を劣化させてしまう結果となる．本手法を適用し輻輳ウインドウを限界値で一定に保った場合には，高遅延環境において最大約 28%のスループット向上が確認された．また，高遅延環境における iSCSI プロトコルのパケットの振る舞いについて述べ，本手法の有用性について考察を行った．今後は本手法をライトアクセスやランダムアクセスに適用し，さらなる評価を行いたい．

謝辞 本研究は，一部，文部科学省科学研究費特定領域研究課題番号 13224014 によるものである．

参 考 文 献

1) iSCSI Specification, http://www.ietf.org/rfc/rfc3720.txt?number=3270/
2) SCSI Specification, http://www.danbbs.dk/~dino/SCSI/
3) 豊田真智子，山口実靖，小口正人：“ iSCSI アクセス時の TCP 輻輳ウィンドウ制御を用いたシステム性能向上手法の一検討 ”，電子情報通信学会技術研究報告，CPSY2004-50，pp.1-6，December 2004.
4) 豊田真智子，山口実靖，小口正人：“ iSCSI ストレージアクセス時における TCP 輻輳ウィンドウとシステム性能の関連性評価 ”，FIT2004 第 3 回情報科学技術フォーラム，B-004，pp.107-109，September 2004.
5) L.Rizzo: “ dummynet, ” http://info.iet.unipi.it/~luigi/ip_dummynet/
6) InterOperability Lab Univ, of New Hampshire, http://www.iol.unh.edu/consortiums/iscsi/
7) 山口実靖，小口正人，喜連川優：“ 高遅延広帯域ネットワーク環境下における iSCSI プロトコルを用いたシーケンシャルストレージアクセスの性能評価ならびにその性能向上手法に関する考察 ”，電子情報通信学会第 14 回データ工学ワークショップ DEWS2003，4-B-02，March 2003.
8) P.Sarkar, S.Uttamchandani, and K.Voruganti: “ Storage over IP: When Does Hardware Support help?, ”*Proc. FAST 2003*，*USENIX Conference on File and Storage Technologies*，pp.231-244，January 2003.
9) P.Radkov, L.Yin, P.Goyal, P.Sarkar and P.Shenoy: “ Performance Comparison of NFS and iSCSI for IP-Networked Storage, ”*Proc. FAST 2002*，*USENIX Conference on File and Storage Technologies*，pp.101-114，March 2004.
10) P.Gurumohan，S.Narasimhamurthy，J.Hui：“ Quanta Data Storage: A New Storage Paradigm, ”*Proc. 12th NASA Goddard Conference on Mass Storage Systems and Technologies*，pp.101-107，April 2004.
11) 藤田智成，小河原成哲：“ iSCSI ターゲットソフトウェアの解析 ”，先進的計算基盤システムシンポジウム SACSIS2004，pp.335-342，May 2004.
12) 菅原豊，稲葉真理，平木敬：“ インテリジェント NIC を用いた広帯域ネットワーク向け TCP 通信方式 ”，情報処理学会研究報告 2004-OS-97，SWoPP2004，pp.57-64，August 2004.
13) 高野了成，石川裕，工藤知宏，松田元彦，児玉祐悦，手塚宏史：“ 並列アプリケーション実行における TCP/IP 通信挙動の解析 ”，IC2003，October 2003.
14) 熊副和美，堀良彰，鶴正人，尾家祐二：“ JGN を利用した高速トランスポートプロトコルの評価 ”，電子情報通信学会技術研究報告，NS2003-354，IN2003-309，pp.303-308，March 2004.