複数台 iSCSI Initiator を用いた TCP 輻輳ウィンドウ制御手法の性能評価

Performance Analysis of TCP Congestion Window Control Method using Multiple iSCSI Initiator

豊田 真智子♥　山口 実靖♦
小口 正人♠

Machiko TOYODA　Saneyasu YAMAGUCHI
Masato OGUCHI

コンピュータシステムにおけるデータ量の増大に伴い，効率的にストレージを管理したいという要望が高まっている．iSCSI を用いることによりデータセンタなどの遠隔地にデータをバックアップすることが容易となり，広域 IP-SAN の構築技術として期待されている．本稿では，iSCSI を用いて複数台サーバからストレージにアクセスする環境を想定し，TCP パラメータである輻輳ウィンドウを動的にコントロールすることにより，各サーバのスループットを均一化する手法を提案する．

As the volume of data computer systems process increases, it is important that storage is managed efficiently. Because iSCSI can make easy data backup of remote place, for example data center, iSCSI is expected to be used as technology of the wide area IP-SAN. In this paper, we suppose an environment in which storage is accessed from multiple servers using iSCSI and propose the method of flatting each server's throughput by controlling Congestion Window.

1. はじめに

近年急速に増加するストレージの利用効率向上や，管理コストの削減を目的としたストレージ統合にSAN(Storage Area Network)が広く用いられ，その実績は高い評価を得ている．既存インフラであるEthernetとTCP/IPを利用でき，安価なコストで構築可能なIP-SANが注目され，2003年2月にIETFにより承認されたiSCSIが，IP-SANの標準的なデータ転送プロトコルとして利用されている[1][2].iSCSIは，SCSIコマンドを，TCP/IPパケット内にカプセル化することにより，サーバ(Initiator)とストレージ(Target)間でブロックレベルのデータ転送を行う．SCSIケーブルを用いてサーバとストレージを直接接続するDASストレージシステムを使用した場合と同様に，ネットワーク越しのストレージへシームレスにアクセスすることができる．

我々はこれまで，iSCSIストレージアクセスの性能を向上させるため，TCPパラメータである輻輳ウィンドウを動的にコントロールする輻輳ウィンドウコントロール手法を提案し，InitiatorとTargetを1対1で接続した環境において評価をおこなってきた[3]．そこで本稿では，これまでの環境を複数台のInitiatorからTargetにストレージアクセスする環境に発展させ，各Initiatorの輻輳ウィンドウを動的にコントロールする複数台Initiator輻輳ウィンドウコントロール手法を提案する．iSCSIを用いたストレージアクセスとしては，遠隔バックアップなどで利用されるシーケンシャルリードアクセスを取り上げる．2つの提案手法を実装し，通常のiSCSIストレージアクセスと比較することにより提案手法の考察を行い，本手法の優位点を述べる．

2. 複数台 Initiator を用いた iSCSI ストレージアクセスにおける課題

TCP において輻輳制御のために用いられる輻輳ウィンドウは，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を意味する．通常，輻輳ウィンドウは ACK を１つ受信するごとに増加する．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，輻輳ウィンドウは減少する．iSCSI においては，輻輳ウィンドウが低下する原因として，送信側デバイスドライババッファが溢れることによる CWR が頻繁に検出される．これは，iSCSI が Read/Write コマンド発行後にデータが一斉送信されるというバースト性の高い通信が行われるためであると考えられる．TCP/IP のソケット通信のみの場合は，セルフクロッキングメカニズムによりバースト性は緩和される．しかし，iSCSI は Read/Write コマンドにより同期がとられるという特徴があるためこの機能が働かず，輻輳ウィンドウの成長も小さなものとなり，iSCSI 使用時の性能劣化は高遅延環境になるほど著しい[3]．このことから iSCSI を用いたストレージアクセスの性能向上のためには，CWR 検出頻度を減らし，輻輳ウィンドウの低下を抑制することにより大きな値で保つことが重要となる．

また，複数台のInitiatorからTargetにアクセスした場合，各Initiatorのスループットは上下に大きくばらつき，ある1台が高い性能を示す場合は残りのInitiatorの性能が低下するという振舞を示し，Initiatorごとに性能差が生じる様子が確認された[4]．Targetであるストレージデバイスを，Initiatorを保持するユーザに提供するような場合においては，ストレージを利用するユーザに平等に帯域を保証する必要があり，接続の度に通信速度が異なるという状況は好ましくない．このような環境において効率的なストレージアクセスを実現するためには，提供される回線のほぼ最大性能をうまく分配して割り当てるといった手法を確立することが望ましい．そこで，文献[3]において提案した輻輳ウィンドウコントロール手法を改良することにより，この利用に対応した手法の実装を試みた．

3. 複数台 Initiator 輻輳ウィンドウコントロール手法 I：スループット均一化手法

本章では，iSCSIストレージアクセスにおいて確認される

♥ 正会員　お茶の水女子大学大学院 人間文化研究科数理・情報科学専攻　現在NTT情報流通プラットフォーム研究所
toyoda.machiko@lab.ntt.co.jp
♦ 正会員　工学院大学 工学部情報通信工学科
sane@cc.kogakuin.ac.jp
♠ 正会員　お茶の水女子大学 理学部情報科学科
oguchi@computer.org

各Initiator間のスループットのばらつきを抑制し，かつ均一化を行う複数台Initiator輻輳ウィンドウコントロール手法Iの概要を述べ，提案手法の性能評価を行う．

3.1 提案手法 I の概要

複数台Initiator輻輳ウィンドウコントロール手法は，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ものである．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．これにより輻輳ウィンドウなどのTCPパラメータを確認することが可能となり，本手法ではこの仕組みをTargetに実装している．本手法における制御手順を以下に示す．

1. Targetで各Initiatorの輻輳ウィンドウをモニタし，変化を観察する．
2. 輻輳ウィンドウの振舞によって以下の異なる処理を行う．
   - 観察しているInitiatorのいずれかでCWRが検出され，輻輳ウィンドウが低下した場合
     i. エラーが検出されたInitiatorの輻輳ウィンドウの最大値(低下する直前の最大の輻輳ウィンドウ)をInitiatorごとに記録し，各Initiatorの輻輳ウィンドウ最大値の中で最も小さな値をすべてのInitiatorに通知する．
     ii. 通知した輻輳ウィンドウ値をTargetにおいても記録する．
   - すべてのInitiatorの輻輳ウィンドウが同じ値で一定値であると判断した場合
     i. その時の輻輳ウィンドウの限界値(CWRが検出されなかった場合の最大値)をすべてのInitiatorに通知し，通知した輻輳ウィンドウ値をTargetにおいても記録する．
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. Initiator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. この処理を，すべてのInitiatorの輻輳ウィンドウが同じ値で一定値と判断され，最大値と限界値の差が十分小さくなるまで繰り返す．

なお，輻輳ウィンドウが一定値となるのはLinux TCP実装の特徴であり，輻輳ウィンドウを一定値となるように収束させることにより，各Initiatorのスループットのばらつきが解消され，性能が安定する[3]．本手法適用後，すべてのInitiatorの輻輳ウィンドウはCWRが検出され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ている．

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（1500Bytes）からTCP/IPヘッダ（オプションを含む）を除いた1448Bytesである．

表1 使用計算機
Table.1 Machine Spec

CPU	Initiator: Intel PentiumIII 800MHz Target, Dummynet: Intel Xeon 2.4GHz
Main Memory	Initiator: 640MB Target, Dummynet: 512MB
OS	Initiator, Target: Linux2.4.18-3 Dummynet: FreeBSD 4.9 - RELEASE
NIC	Initiator, Dummynet: Intel PRO/1000MT Server Adapter Target: Intel PRO/1000XT Server Adapter

3.2 提案手法 I の性能測定実験

前節で述べた提案手法の性能評価を行うため，4台のInitiatorと1台のTargetを用いて多対1の環境を構築し，InitiatorからTargetへシーケンシャルリードアクセスを行う． Initiatorが指定するブロックサイズは，提案手法を用いない場合は1024KBに，提案手法を用いた場合はその初期値を1024KBに設定した．

3.2.1 実験環境

本実験は以下の環境で行った．InitiatorとTarget間は1000Base-Tスイッチングハブを介してGigabit Ethernetで接続した．広域IP-SANにおける遠隔データバックアップなどを想定した実験を行うため，InitiatorとTarget間に人工的な遅延装置であるFreeBSD Dummynet[5]を挿入した．本稿における実験結果としては，16ms (Round Trip Time: 32ms)における結果を示す．受信側のTCP広告ウィンドウサイズは，実験に影響を与えないように8MBと十分大きな値に設定した．本実験ではストレージアクセスのみの性能を評価するため，Initiatorのrawデバイスを使用することによりキャッシュの影響を排除した．また，iSCSIを利用したストレージアクセスにおける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，ディスクアクセスを伴わないようにした．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が提供するUNH実装を用い[4]，InitiatorにはUNH実装のInitiatorと同等の機能を持つ自作Initiatorを用いて実験を行った．

3.2.2 実験結果

前節の実験を行った結果として，提案手法を用いずにiSCSIシーケンシャルリードアクセスを行った場合の輻輳ウィンドウ，スループットの時間変化を図 1，2 に，提案手法 I を用いた場合の輻輳ウィンドウ，スループットの時間変化を図 3，4 に示す．これらの図における輻輳ウィンドウ低下の原因はすべて CWR 検出によるものである．

提案手法を用いない場合，各 Initiator のスループットが不安定であり，輻輳ウィンドウも不規則な増加減少を繰り返していることが確認される(図 1，2)．また，輻輳ウィンドウが大きく成長した場合にはスループットは高くなるが，あまり成長せず小さな値である場合にはスループットは低いことがわかる．そのため，各 Initiator へのデータ転送は不均

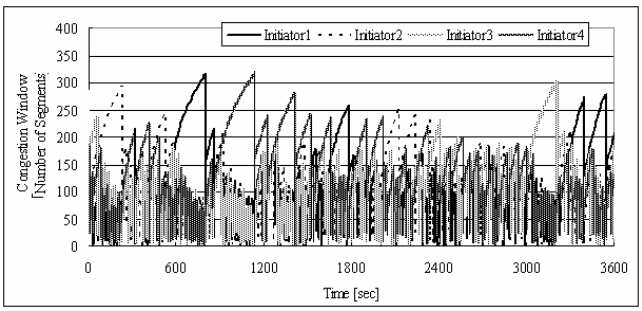

図 1　提案手法を用いない場合の輻輳ウィンドウ時間変化
Fig.1 Congestion Window not using the Proposed Method

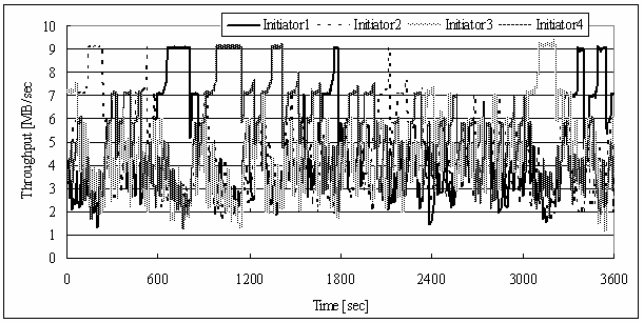

図 2　提案手法を用いない場合のスループット時間変化
Fig.2 Throughput not using the Proposed Method

一であり，その性能差は大きい．

一方提案手法を用いたストレージアクセスの場合，Target において CWR 検出の度に Initiator のミドルウェアが機能し，アクセスブロックサイズを調節することにより輻輳ウィンドウをコントロールする．最終的なブロックサイズが初期値と比較してやや低い値に設定されるため性能は少し低下するが，最終的に CWR は回避され，輻輳ウィンドウが一定値となる(図 3). その結果スループットは安定し，各 Initiator の性能差はほとんど見られない(図 4). また，CWR を回避した通信を行うことができるので，回線の最低帯域を保証することも可能となる．

なお，図 2 において，各 Initiator のスループットが約 9MB/sec 程度で打ち止めになっているが，これは Initiator 計算機の性能の限界のためである．

4. 複数台Initiator輻輳ウィンドウコントロール手法II：スループット均一化 + 性能向上手法

本章では，前章で提案した複数台Initiator 輻輳ウィンドウコントロール手法I に改良を加え，スループットのばらつき抑制かつ性能向上を目的とした複数台Initiator 輻輳ウィンドウコントロール手法II を提案し，その概要を述べる．

4.1 提案手法IIの概要

前章で提案した複数台Initiator 輻輳ウィンドウコントロール手法I を用いることにより，Target から各Initiator へのデータ転送を均一化し，信頼性の高いストレージアクセスを行うことができる．しかし，Target におけるCWR検出の度に，単純にアクセスブロックサイズを低下させるという実装であるため，安定した通信を行うことができる一方，エラー

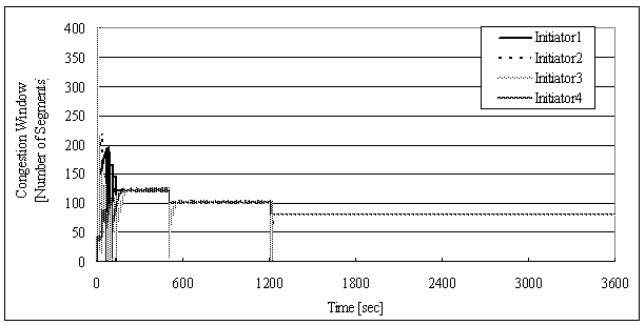

図 3　提案手法 I を用いた場合の輻輳ウィンドウ時間変化
Fig.3 Congestion Window using the Proposed Method(I)

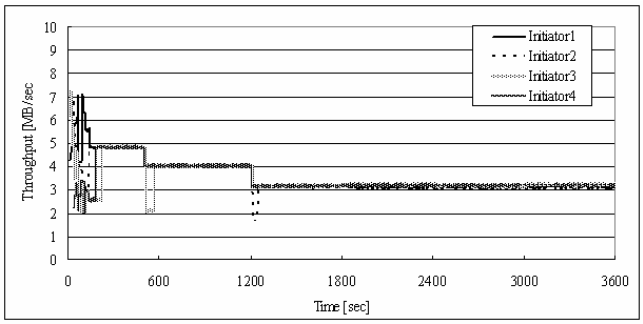

図 4　提案手法 I を用いた場合のスループット時間変化
Fig.4 Throughput using the Proposed Method(I)

頻度が高い場合にはアクセスブロックサイズが小さくなり，結果としてスループットが低下してしまう．ブロックサイズは一度に送信するデータ長であるため，この値を増加させることで送信するデータ量が増加し，スループットは向上する．しかし，ブロックサイズをあまり大きくしすぎると送信バッファが溢れてしまい，CWR を検出して輻輳ウィンドウが低下する．そのため，ブロックサイズと輻輳ウィンドウ双方のバランスを保ちつつ，スループットを向上することが重要となる．そこで，前章の実験結果を考慮し，CWRは検出されるが，輻輳ウィンドウが一定値となった場合には最適であると判断し，それ以降エラーが検出されてもブロックサイズの減少命令を発行しない，複数台Initiator 輻輳ウィンドウコントロール手法II を検討した．新たに検討した提案手法II は．提案手法Iの制御手順2が以下のように動作する．

2. 輻輳ウィンドウの振舞によって以下の異なる処理を行う．
   - 観察しているInitiatorのいずれかでCWRが検出され，輻輳ウィンドウが低下した場合
     i. それまでにすべてのInitiatorの輻輳ウィンドウが同じ値で一定値となったと判断していれば，最適であるとしてInitiatorへの通知は行わない
     ii. すべてのInitiatorが一定値であることを判断していない場合には，CWRが検出されたInitiatorの輻輳ウィンドウの最大値(低下する直前の最大の輻輳ウィンドウ)をInitiatorごとに記録し，各Initiatorの輻輳ウィンドウ最大値の中で最も小さな値をすべてのInitiatorに通知し，通知した輻輳ウィンドウ値をTargetにおいても記録する
   - 一度もCWRを検出せずに，すべてのInitiatorの

輻輳ウィンドウが同じ値で一定値であると判断した場合

i. その時の輻輳ウィンドウの限界値(CWRが検出されなかった場合の最大値)をすべてのInitiatorに通知し，通知した輻輳ウィンドウ値をTargetにおいても記録する．

本手法適用後，CWRは検出されるがすべてのInitiatorの輻輳ウィンドウはほぼ同じ値で一定に保たれ，その時のブロックサイズが最適値となる．なお，ミドルウェアが指定するブロックサイズは前章と同様の式を用いて計算した．

4.2 提案手法 II の性能測定実験

提案手法IIの評価を行うため，3.2節と同様の実験を行う．実験で使用した計算機は，表1のものを用い，実験環境は3.2.1節と同様の環境を構築した．実験結果として，図5，6に輻輳ウィンドウ，スループットの時間変化を示す．これらの図における輻輳ウィンドウの低下原因は，すべてCWR検出によるものである．

提案手法Iとは異なり，Targetにおいて一度すべてのInitiatorの輻輳ウィンドウが一定値であると判断されると通知を行わない．そのため，一定値となってからもCWRは検出され，検出の度に輻輳ウィンドウが低い値に設定されるが，すぐにほぼ同じ値にまで回復し再度一定値となる(図5)．CWR検出時にスループットも一時低下するが，その回復は早期であるため性能にはあまり影響を与えず，ほぼ安定した通信が行われている(図6)．さらに提案手法Iの場合よりも最終的なブロックサイズを大きな値で保つことにより，各Initiatorのスループットを均一化するだけでなく，性能向上が達成された．この際の各スループットの合計値は，1台のみのInitiatorアクセス時の最大スループットと近い値となっており，Initiatorが均一な本実験の環境においては，提案手法IIが理想的な手法であると言える．

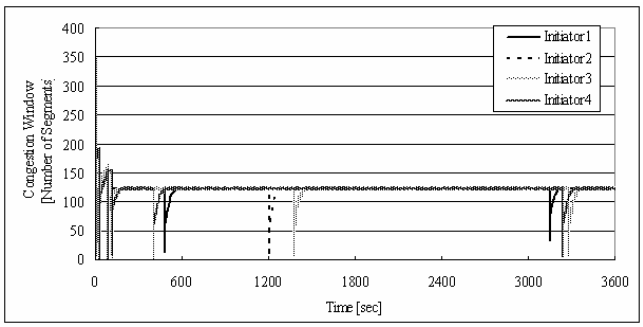

図 5 提案手法 II を用いた場合の輻輳ウィンドウ時間変化
Fig.5 Congestion Window using the Proposed Method(II)

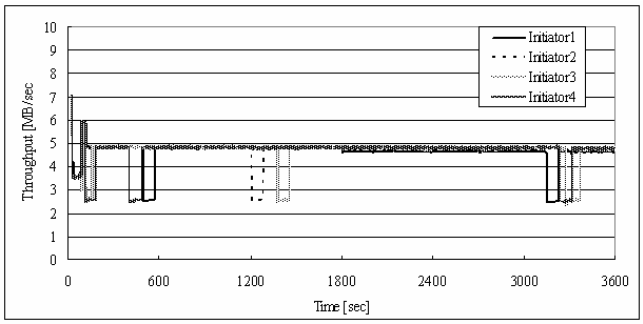

図 6 提案手法 II を用いた場合のスループット時間変化
Fig.6 Throughput using the Proposed Method(II)

5. まとめと今後の課題

本稿では，複数台の Initiator から Target にアクセスする環境を想定し，iSCSI ストレージアクセスにおける各 Initiator のスループットのばらつきを抑制する複数台 Initiator 輻輳ウィンドウコントロール手法を提案した．提案手法を用いることにより，各 Initiator へのデータ転送は均一化され，公平で効率的なストレージアクセスを実現することができた．また，提案手法 II を用いることにより，スループット均一化に加えて性能向上も実現させた．ストレージをサービスとして提供するような環境においては，ユーザに平等なストレージアクセスを提供する必要があり，最低使用帯域も保証可能な本手法の有効性は向上する．

今後は提案手法を用いたストレージアクセスの性能向上を実現させ，様々な環境の制御が行えるようにしたい．

[謝辞]

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

[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: “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”，先進的計算基盤システムシンポジウム（SACSIS2005）, pp.443-450, 2005年5月.
[4] 豊田真智子，山口実靖，小口正人：
“複数台のInitiatorを用いたiSCSIアクセスにおけるTCP輻輳ウィンドウとシステム性能の考察”，日本データベース学会Letters，Vol.4 No.2，pp.77-80, 2005年9月.
[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/

豊田 真智子 Machiko TOYODA
NTT 情報流通プラットフォーム研究所所属．2006 年お茶の水女子大学大学院人間文化研究科博士前期課程修了．iSCSI を用いたストレージエリアネットワークの研究に従事．日本データベース学会, 情報処理学会, 電子情報通信学会各正会員．

山口 実靖 Saneyasu YAMAGUCHI
工学院大学工学部情報通信工学科講師. 2002 年東京大学大学院工学系研究科電子情報工学専攻博士課程修了，工学博士．iSCSI を用いたストレージシステムの性能向上の研究に従事．日本データベース学会，情報処理学会，電子情報通信学会各正会員．

小口 正人 Masato OGUCHI
お茶の水女子大学理学部情報科学科助教授. 1995 年東京大学大学院工学系研究科電子工学専攻博士課程修了，工学博士．ネットワークコンピューティング・ミドルウェアに関する研究に従事．IEEE，ACM，日本データベース学会，情報処理学会，電子情報通信学会各正会員．