高遅延環境における小粒度 iSCSI アクセスの特性解析

Analysis of iSCSI Storage Access with Short Blocks in Long Delayed Network

山口 実靖♥ 小口 正人♦
喜連川 優♥

Saneyasu YAMAGUCHI Masato OGUCHI
Masaru KITSUREGAWA

規模拡張性の高さや導入コスト低さなどの利点を持つ SAN として IP-SAN や iSCSI が注目を集めている．本稿では，TCP の振る舞いを考慮した ショートブロックによる iSCSI ストレージアクセスの特性解析について述べる．まず，既存の iSCSI 実装を用いその性能を評価し，そのターンアラウンドタイム性能が必ずしも高く無いことや実装により性能が大きく異なる事を示す．次に開発した iSCSI 解析システムを用い，性能劣化が TCP の Nagle アルゴリズムや遅延 Ack に起因していることを示す．

IP-SAN is expected as scalable and cost-effective SAN, and iSCSI is also expected data transfer protocol of IP-SAN.

In this paper, authors describe detailed analysis of iSCSI storage access with short blocks.

First, we show turn around time of short block iSCSI storage access with several iSCSI implementations. We found differences among implementations are significantly large.

Second, we analyze these iSCSI implementations with our analysis system and show that performance degradations are caused by TCP Nagle algorithm and Delayed Ack.

1. はじめに

超大容量のデータを高速に処理するためのシステムとして，SAN(Storage Area Network)[1]が注目を集めており，その実績は高い評価を得ている．しかし現世代のSANは，FC (Fibre Channle) を用いた FC-SAN であり，FC の導入コストの高さ，FC 管理技術者の少なさ，FC の接続距離の限界，などの問題点も明らかとなってきている．これらの問題点を解決する SAN として，Ethernet と TCP/IP を用いた SAN である IP-SAN や，そのためのデータ転送プロトコルである iSCSI[1,2] に大きな期待が集まっている．我々は，iSCSI 用いたストレージアクセスの性能向上手法として文献[3]においてiSCSI システムを網羅的に観察することにより性能劣化原因の発見を可能とする iSCSI 解析システムを提案し，高遅延ネットワーク環境における iSCSI を用いた連続的なデータ転送のスループット性能を向上させるにはブロックサイズの拡大やそれに伴うローカルデバイス輻輳の回避が重要であることを述べた．

本稿では，ショートブロックサイズによる小粒度の iSCSI ストレージアクセスの性能について述べる．ショートブロックアクセスは DBMS やファイルアクセスなどに用いられ，ターンアラウンドタイムの短縮が重要であると考えられる．まず既存の iSCSI 実装を紹介し，小粒度の iSCSI アクセスの性能を紹介し，実装によりその性能が大きく異なることを示す．そして，それら各実装の振る舞いに対する詳細な解析を紹介し，これらの性能差が Nagle のアルゴリズム[4]や遅延確認応答[5]などの TCP の振る舞いに起因していること，これらを回避することによりその性を大きく向上できることを示す．

本稿は以下のように構成される.第2章で研究背景として，IP-SAN や iSCSI の重要性，関連する既存の研究成果について述べる．第3章において各実装における，iSCSI を用いた小粒度アクセスのターンアラウンドタイム性能について紹介し，実装により性能が大きく異なることを示す．第4章において前章の実験の振る舞いの解析を述べ，実装による性能の差の原因が TCP の振る舞いにあることを述べる．最後に，第5章において本稿をまとめる．

2. 研究背景

2.1 iSCSI

SAN はストレージ専用の高速ネットワークであり，ストレージを一カ所に集約し各サーバ計算機は SAN を用いてこれに接続する．サーバ毎に管理されていたストレージを一カ所に集約して管理することにより，管理コストは大幅に削減されると言われている．しかし現世代の SAN は FC を用いているため，FC の管理技術者が少ない，FC の接続距離には限界がある，FC の相互接続性は必ずしも高くない，FC の導入コストが高い，などの問題点も明らかとなってきておりこれらの問題を解決する IP-SAN と iSCSI に期待が高まっている．

IP-SAN とは，Ethernet と TCP/IP を用いて構築する SAN であり，管理可能技術者が多い，接続距離に限界がない，相互接続性が高い，導入コストが低い，などの利点を持っている．IP-SAN 用のデータ転送プロトコルとしては SCSI プロトコルを TCP プロトコルの中にカプセル化し IP ネットワーク上で転送するブロックレベルのプロトコル iSCSI が代表的なプロトコルである．多くの場合，物理層，トランスポート層には Ethernet が用いられ，代表的なプロトコルスタックはSCSI over iSCSI over TCP/IP over Ethernet となる．iSCSI は，2003年2月にIETF[2]に正式に承認され，現在各種 OS へのドライバ や HBA の提供が始まっている．

2.2 関連研究

文献[6]において，Ng らは独自の SCSI over IP 実装を用いて 8KB のブロックサイズにおけるシーケンシャルアクセスやランダムアクセスの性能を測定している．同測定から，ランダムリードの性能がネットワーク遅延時間の増加にともなり単調に減少することが確認されているが，TCP の振る舞いによりこの性能が大きく変化することについては言及されていない．

♥ 正会員 東京大学生産技術研究所
{sane,kitsure}@tkl.iis.u-tokyo.ac.jp
♦ 正会員 お茶の水女子大学理学部情報科学科
oguchi@computer.org

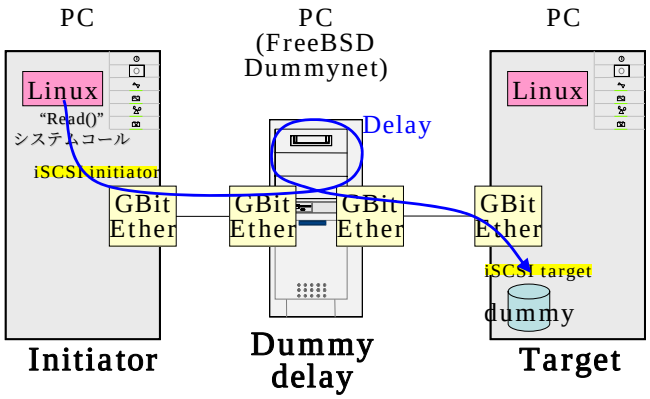

図 1 実験環境 1

Fig.1 Experiment Environment 1

3. iSCSI 実装の性能測定

本章において小粒度 iSCSI アクセスの性能を測定し，それを紹介する．

3.1 実験環境

性能評価実験は以下の環境で行った．図 1 のように， iSCSI Initiator(サーバ) と iSCSI Target(ストレージ)を Gigabit Ethernet で接続して TCP/IP 接続を確立する．Ethernet の接続は途中に人工的な遅延装置として FreeBSD Dummynet を挟んでクロスケーブルで接続をした．Initiator, Target, Dummynet はすべて PC 上に構築し，Initiator と Target には Linux を，遅延装置には FreeBSD をインストールした．Initiator, Target の PC の詳細を表 1 に，遅延装置の PC の詳細を表 2 に示す．また，iSCSI の実装としては以下のものを用いた (1)ニューハンプシャー大学 InterOperability Laboratory(以下，“IOL” と呼ぶ) [7] が配布する iSCSI 実装(iSCSI draft 18 準拠のもの), (2)同大学 IOL が配布する iSCSI 実装(iSCSI draft 20 準拠のもの), (3)Intel 社が配布する iSCSI 実装(draft 16 準拠)． また，これらの実装に対し我々が変更を施したもの(後述)も被実験実装として用いた．以後，ニューハンプシャー大学の iSCSI 実装で draft 18 準拠であるものを “UNH 18”と，draft 20 準拠のものを“UNH 20”と呼び，Intel の iSCSI(draft 16 準拠)を “Intel 16” を呼ぶ．

3.2 実験方法

前節の実験環境により，以下の実験を行い各実装の評価を行った．まず, Initiator 計算機 と Target 計算機において，iSCSI Initiator, iSCSI Target を起動させる．この際, iSCSI Target はメモリモードで起動させる．よって, iSCSI Target デバイスへのアクセスは物理的なディスクへのアクセスを伴わない．次に，Initiator 計算機から Target 計算機に対し iSCSI 接続を確立させる(Initiator 計算機の OS において遠隔ディスクのマウントを行う)．そして，作成したベンチマークソフトウェアにより，iSCSI 接続のディスクの raw デバイスに対して，システムコール **read()** を連続して発行しその性能の平均を測定する．

表 1 実験環境 2 : 使用計算機

Table 1 Experiment Environment 2 : PC Specifications

CPU	Pentium 4 2.8GHz
Main Memory	1GB
OS	Linux 2.4.18-3
Network Interface	Gigabit Ethernet Card Intel PRO/1000 XT Server Adapter

表 2 実験環境 3 : 使用計算機

Table 2 Experiment Environment 3 : PC Specifications

CPU	Pentium 4 1.5GHz
Main Memory	128MB
OS	FreeBSD 4.5-RELEASE
Network Interface	Gigabit Ethernet Card Intel PRO/1000 XT Server Adapter×2

3.3 性能測定結果

前節の実験により，各実装の性能を測定し，図 2 の UNH18(def), UNH20(def), Intel16(def)の結果を得た.同図は，片道遅延時間 4ms における各実装のターンアラウンドタイムを表している．“UNH18(def)” は UNH 18 実装を用いて測定したものであり，同実装に対し著者らが改変を行っていないものである．“(def)”は default を意味し後述する著者らが改変を行ったものと区別するために“(def)”と記す．同様に“UNH20(def)”は UNH 20 実装を用いて測定したものであり同実装に対して改変が行われていないもの，“Intel16(def)”は Intel 16 実装を用いて測定したものであり改変が行われていないものである．横軸はブロックサイズを表し，ベンチマークプログラムにおける システムコール **read()** の発行の際に引数として指定したサイズであり，実際にネットワークで転送される iSCSI PDU での Read コマンドのブロックサイズもこれに等しい(システムコール時に大きいブロックサイズを指定しても実際に発行される iSCSI PDU における Read コマンドのブロックサイズがこれよりも小さいことがあるが[3], 本稿で述べる小粒度のアクセスにおいてこれは発生しない)．縦軸はターンアラウンドタイムを表し，システムコール **read()** が発行されてからそれが終了するまでの時間を表している．

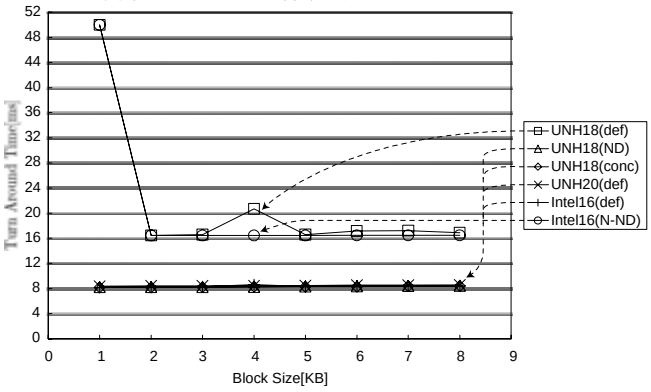

図 2 実験結果

Fig.2 Experimental Result

同結果より，ターンアラウンドタイムは UNH18(def) において約 16ms であり(ただしブロックサイズ 1KB が例外として，16ms から大きくはずれている)，UNH20(def), Intel(def) において約 8ms であることが確認された．すなわち，本実験結果の例においてターンアラウンドタイムは実装の違いにより，約 2 倍の性能差が現れること(1KB を除く)，ブロックサイズ 1KB に 6 倍の性能差が現れることが確認された．

4. 解析

本章では，開発した iSCSI 解析システム[3]を用いて前

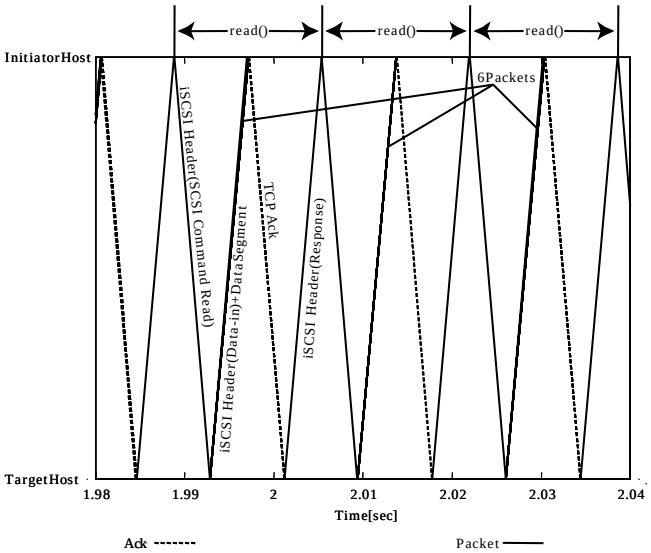

図 3 UNH18 実装, 片道遅延時間 4ms, Block Size 8KB, TCP パケット転送時間軸可視化図

Fig.3 Visualized TCP Packet Transfer, UNH 18, 4ms one way delay time, Block Size 8KB

章の実験を解析し，前述の性能差が現れる理由を示す.

4.1 UNH 18 の解析

片道遅延時間 4ms において UNH18 実装を用いブロックサイズ 8KB の **read()** を行ったときの TCP パケットの転送を可視化したものは図 3 の様になる. まず, 高遅延環境下におけるショートブロックアクセスのターンアラウンドタイムの多くが“データの転送時間”ではなく，ネットワークの“遅延時間”に費やされていることが視覚的に確認できる. よって，ネットワークの遅延時間の短縮や往復回数の削減がターンアラウンドタイムの短縮には効果が大きいと考えられる. 次に，同図より，iSCSI Target は SCSI Command Read の iSCSI PDU 受信の後にまず iSCSI Data-in PDU を送信し，Initiator からの TCP Ack の受信の後に iSCSI Response を送信していることが確認された. すなわち, 1 回の iSCSI Read は iSCSI PDU SCSI Command Read (I→T), iSCSI PDU Data-in (T→I), TCP Ack (I→T), iSCSI PDU Response (T→I), により構成され Initiator - Target 間の 2 往復を要している(ただし，“I→T”は Initiator から Target 方向，“T→I”は Target から Initiator 方向の意). これにより，“ストレージデバイスの動作時間に対してネットワーク遅延時間が十分に大きい”という仮定のものでは，ターンアラウンドタイムは，“4×片道遅延時間”と同程度になると言える. 同実装では, TCP の Nagle のアルゴリズムが有効となっており，かつ TCP に対する iSCSI Data-in PDU の送信要求と iSCSI Response の送信要求を別々に発行する. iSCSI Data-in PDU は TCP によりセグメントサイズ毎に分割され送信される. よって，分割後の最後のセグメントはセグメントサイズ未満の微少パケットとなる(PDU サイズをセグメントサイズで割った剰余がこのサイズとなる). この最後の微少パケット送信後に再度微少パケット(48B の iSCSI Response)の送信要求を受けた TCP 実装では Nagle のアルゴリズムが動作し，2 個目の微少パケットの送信は Ack 受信後まで延期されることとなる. これにより往復回数が 1 回増加する.

また, UNH 18 を用いて，ブロックサイズ 1KB で **read()** を行ったときにターンアラウンドタイム

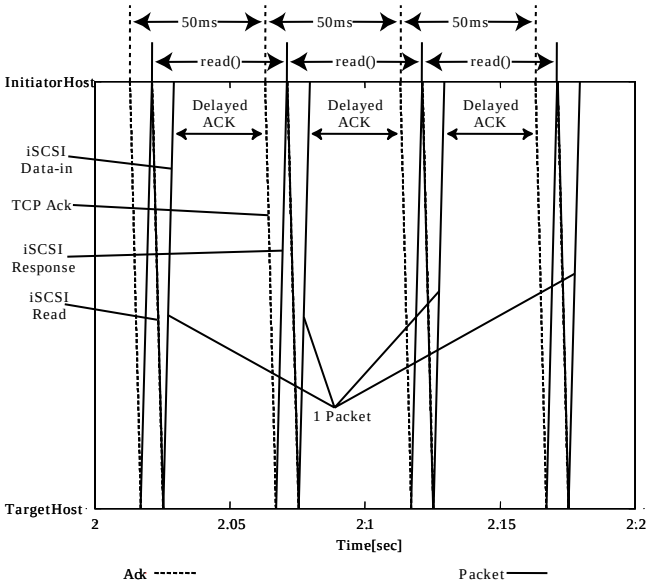

図 4 UNH18 実装, 片道遅延時間 4ms, Block Size 1KB, TCP パケット転送時間軸可視化図現

Fig.4 Visualized TCP Packet Transfer, UNH 18, 4ms one way delay time, Block Size 1KB

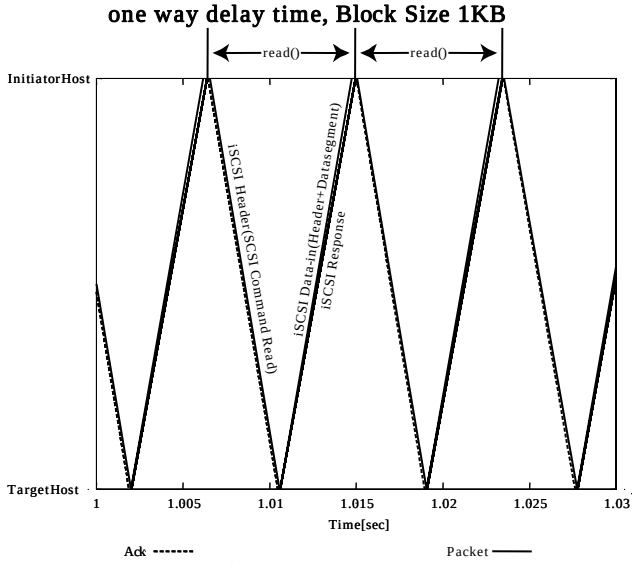

図 5 Intel 16 実装, 片道遅延時間 4ms, Block Size 8KB, TCP パケット転送時間軸可視化図

Fig.5 Visualized Packet Transfer, Intel 16, 4ms one way delay time, Block Size 8K

が非常に大きくなることが確認されている. UNH 18 実装を用いて，片道遅延時間 4ms，ブロックサイズ 1KB の **read()** を行ったときのパケット転送を可視化したものを図 4 に示す. 図 3 同様に, iSCSI Data-in PDU の送信は TCP Ack の受信を待ってから行われている. ただし，同図おいては iSCSI PDU (SCSI Command Read) に要求されたデータサイズは 1KB であり, Initiator 計算機の TCP 実装が受け取る TCP パケットは 1 個となる(Data-in PDU は 48B の iSCSI Header と 1KB の Data Segment で構成され，これは MSS より小さい). TCP の遅延確認応答アルゴリズムにより，孤立した単数の TCP パケットを受信した Initiator 計算機の TCP 実装は Ack の送信を大きく遅らせる. 同図より 50ms 毎に発生するカーネルタイマの発生まで Ack の送信を遅らせており，それにより Target による

iSCSI Response の送信が大きく遅れ，結果としてターンアラウンドタイムが著しく低下していることが確認された．

4.2 Intel 16 の解析

片道遅延時間 4ms において Intel 16 の実装を用いブロックサイズ 8KB の read の TCP パケットの転送を可視化したものを図 5 に示す．同図から UNH 18 実装と異なり Intel 16 実装では 1 回の システムコール **read()** の完了に必要とされるネットワークの往復は 1 回である(ターンアラウンドタイムは“2×片道遅延時間”と同程度)ことが確認でき，その結果 Intel 16 iSCSI 使用時のターンアラウンドタイムが UNH 18 使用時の約半分となることが確認できる．同実装では Nagle のアルゴリズムが無効されており，図より TCP Ack の受信を待たずに iSCSI Response を送信していることが UNH 18実装と比べて 往復回数が1回少なくなっている原因であることが分かる．

4.3 UNH 20 の解析

UNH iSCSI 20 の振る舞いの解析を行うと，図 5 同様に，システムコール **read()** 毎に必要とされるネットワークの往復回数は 1 回である．結果として ターンアラウンドタイムは“2×片道遅延時間”と同程度どなる．ただし，UNH 20 においては iSCSI Data-in PDU の後の iSCSI Response が省略されている．

4.4 考察と参考実験

以上より，高遅延環境下におけるショートブロック iSCSI アクセスの性能は TCP の振る舞いに強く依存し，これを考慮した iSCSI ドライバの実装が重要であると言える．参考のために (1) UNH 18 実装に対し Nagle のアルゴリズムの無効化を追加したもの, (2)UNH 18 実装に対しパケット結合層を追加したもの, (3)Intel 16 実装から Nagle のアルゴリズム無効化を削除したもの，の 3 実装の性能を評価した．(2)のパケット結合層とは iSCSI 層から送られる iSCSI Data-in PUD を一旦保持し，後に送られる iSCSI Response と結合してから TCP 層に渡すものである．この層を iSCSI 層と TCP 層の間に挿入し性能を測定した．これにより微少パケットの送信要求が 1 回削減され Nagle のアルゴリズムによる送信の遅延が回避されると考えられる．それぞれの性能は図 2 の“UNH 18 (ND)”, “UNH 18 (conc)”, “Intel 16 (N-ND)” の様になり，考察の正しさが確認された．

5. まとめと今後の課題

本稿では，ネットワーク遅延の大きい環境におけるショートブロックサイズ iSCSI アクセスの性能について述べた．シーケンシャルアクセスと異なり，ショートブロックサイズのアクセスにおいてはそのターンアラウンドタイム性能が重要となる．一般にネットワークが物理的にもつラウンドトリップタイムよりもターンアラウンドタイムを短くすることができず，ラウンドトリップタイムに近づけることが理想と言える．しかし，TCP の振る舞いを考慮せずに iSCSI ドライバの実装を行うとネットワークの往復回数を増やすことや，遅延確認応答の動作によるターンアラウンドタイム性能の著しい低下を招くことがあり，TCP 実装の動作に対する考察が重要であると言える．

本稿の例に置いては，Nagle のアルゴリズムを有効にした状態において Data-in PDU と Response PDU の送出要求を個別に TCP に対して行うと，TCP 実装が Response PDU の送出を保留してしまい，ネットワーク往復回数が 2 回となり，ターンアラウンドタイムが約 2 倍となった．また，Nagle のアルゴリズムが有効になっている状態において，微少(MSS 未満のサイズ)な **read()** を行うと，孤立した TCP パケット の受信を発生させ，それによる 遅延確認応答の動作を招くこととなりターンアラウンドタイムを著しく増加させてしまうことが確認された．

このように，SCSI プロトコルを TCP プロトコルの中にカプセル化して送信する iSCSI においては，その性能向上のためには TCP プロトコルの振る舞いを十分に考慮することが重要であると言える．

今後は，実ハードディスクデバイスを用いての性能の考察，TOE(TCP Offload Engine) を用いての性能の考察などを進めていく予定である．

[文献]

[1] 喜連川優: “ストレージネットワーキング”，オーム社出版局 (2002).

[2] Julian Satran at al.: “iSCSI”, http://www.ietf.org/internet-drafts/draft-ietf-ips-iscsi-20.txt (2003)

[3] 山口実靖 小口正人 喜連川優: “iSCSI 解析システムの構築と高遅延環境におけるシーケンシャルアクセスの性能向上に関する考察”，電子情報通信学会論文誌 D-1，pp. 216-231 (2004).

[4] John Nagle: “Congestion Control in IP/TCP Internetworks”, http://www.ietf.org/rfc/rfc0896.txt, (1984).

[5] R. Braden: “Requirements for Internet Hosts”, http://www.ietf.org/rfc/rfc01122.txt (1989).

[6] Wee Teck Ng et al.: “Performance Evaluation and Improving of Sequential Storage Access using iSCSI Protocol in Long-delayed High throughput Network”, Proc. of IEICE The 14th Data Engineering Workshop (2003).

[7] “University of New Hampshire InterOperability Lab”, http://www.iol.unh.edu/

山口 実靖 Saneyasu YAMAGUCHI

東京大学生産技術研究所 産学官連携研究員．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 を用いたネットワーク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，電子情報通信学会，情報処理学会，日本データベース学会各会員．

喜連川 優 Masaru KITSUREGAWA

1978 年東京大学工学部電子工学科卒．1983 年同大大学院工学系研究科情報工学博士課程修了，工学博士．同年同大生産技術研究所講師．現在，同教授．平成 15 年 4 月より，同所戦略情報融合国際研究センター長.データベース工学，並列処理，Web マイニングに関する研究に従事.本会理事，情報処理学会理事・フェロー，SNIA-Japan 顧問, ACM SIGMOD Japan Chapter Chair(H11-H14),電子情報通信学会データ工学研究専門委員会委員長(H9,10).VLDB Trustee, IEEE TKDE Assoc. Editor, IEEE ICDE, PAKDD, WAIM Steering Comm.Member.