ストレージシステムにおける Copy-on-Write Snapshot の高性能化方式

出口 彰† 下薗 紀夫† 山本 政行†

†日立製作所システム開発研究所 〒224-0817 神奈川県横浜市戸塚区吉田町 292 番地

E-mail: †{akira.deguchi.em, norio.shimozono.zf , masayuki.yamamoto.jw}@hitachi.com

あらまし ストレージシステムでは，プロセッサ高性能化や I/O 処理方式の効率化によりプロセッサ処理時間が短縮されているのに対し，制御情報のアクセスレイテンシが縮まないことが，高性能化を難しくしている．この問題に対して，ボリュームの I/O 処理を一つのコントローラに集約し，処理に必要となる制御情報を共有メモリからコントローラ内のローカルメモリにキャッシュすることで高性能化する方法が考えられる．しかし，ストレージシステムのスナップショット機能のように，多数のボリュームから共有される制御情報をキャッシュする場合，制御情報の一貫性を保つような処理が必要となり性能向上が困難である．本研究では，スナップショットデータの状況に応じ，制御情報のアクセス特性が変化することに着目し，状況に応じてキャッシュ可否の変更する方式を考案した．プロトタイプシステムによる検証の結果，大部分の制御情報アクセスをローカルメモリアクセスにできた．

キーワード ストレージシステム，スナップショット，高性能化，制御情報キャッシュ

Performance Improvement of Copy-on-Write Snapshot for Storage Systems

Akira DEGUCHI† Norio SHIMOZONO† and Masayuki YAMAMOTO†

†Systems Development Laboratory, Hitachi Ltd. 292 Yoshida-cho, Totsuka-ku,

Yokohama-shi, Kanagawa, 224-0817 Japan

E-mail: †{akira.deguchi.em, norio.shimozono.zf , masayuki.yamamoto.jw}@hitachi.com

Abstract In storage systems, it is becoming difficult to improve performance, because access latency to control information cannot be shortened even as processing becomes faster. For performance improvement, the storage system consolidates I/O processing for a volume to one controller and caches the control information that is needed for I/O processing from shared memory to local memory of the controller. However, in some storage functionality such as snapshot, control information is shared by many volumes. In such cases, processing to ensure cache consistency is necessary, making performance improvement difficult. In this paper, we focus on the fact that the access characteristic to the control information changes according to the state of the snapshot data. We propose a method to decide whether to cache the control information based on the state of the snapshot data. In our verification experiments, the majority of control information access was able to be executed by local memory.

Keyword Storage Systems, Copy-on-Write Snapshot, Performance, Control Information Caching

1. はじめに

ストレージシステムは，主にファイバチャネルと呼ばれる I/F で構成されるストレージエリアネットワーク(SAN)を介して，ホスト計算機に接続され，ホスト計算機で用いられるデータを保存する装置である．装置内部には，多数の HDD と大容量キャッシュメモリを搭載し，RAID 制御・キャッシュ制御やコピー制御などを行うことで，高性能・高機能を提供している．

小規模および中規模システムでは，2 個のコントローラを高速バスなどで接続し HDD を共有するデュアルコントローラ構成をとることが多い．一方 10 個以上のプロセッサや I/O ポートを搭載する大規模なシステムも存在し[4]，必要記憶容量，必要性能，必要ストレージ機能などシステム要件に応じて使い分けられる．

一般に，ストレージシステムは，装置の構成情報やキャッシュ上のデータを管理するためのキャッシュの管理情報などを持ち，I/O 処理はこれらの制御情報を参照・更新しながら実行される．そして，これらの制御情報は，コントローラに障害が発生しても，他のコントローラで I/O 処理を引き継げるように，複数のコントローラで共有されるメモリ(以後，共有メモリと呼ぶ)に格納されるのが一般的である．

近年，プロセッサ自体の高性能化，高性能化のための処理効率化によりプロセッサ処理時間は大幅に改善している一方で，制御情報アクセスに伴う共有メモリアクセス時間は短縮されず，ストレージ処理時間の大部分を共有メモリアクセス時間が占める状態となっている．このため，共有メモリアクセスがネックとなり，プロセッサ性能向上に応じた，性能向上が困難な状態になってきている．特に，ストレージ機能(バックアッ

プ向けボリューム複製機能，ディザスタリカバリ向けリモートコピー機能など)を用いる場合，I/O 処理の制御情報アクセスに加え，ストレージ機能特有の制御情報へのアクセスもあり影響が大きい．

このような問題の解決策として，プロセッサ近くにローカルメモリを配置し，当該ローカルメモリに制御情報を格納する方法が考えられる．具体的には，ボリューム毎にボリュームの処理を担当するコントローラを一つ決め，当該コントローラのローカルメモリに，I/O 処理に必要となる制御情報をキャッシュする．デュアルコントローラ構成を採るストレージシステムでは，コントローラ毎に処理担当のボリュームが予め決まっていることが多く，このような方法は効果的である．

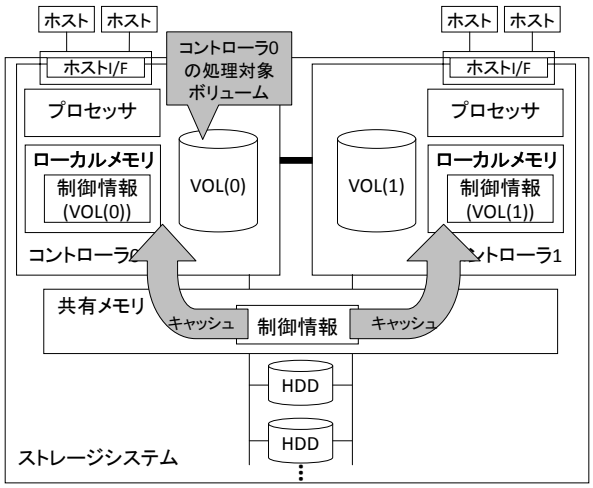

図 1　制御情報のキャッシュによる高性能化

しかし，上記の様にボリュームの粒度で処理担当を決める場合，複数のボリュームから共有される制御情報をキャッシュすることが難しい．例えば，ストレージシステムベースの Copy-on-Write スナップショット[6][7][8][9][10] (以後，単にスナップショットと呼ぶ)のように数百から数千のボリュームから参照および更新される制御情報をキャッシュする場合，コントローラ間で制御情報の一貫性を保証するための通信などが必要となり性能向上が難しい．

本研究の目的は，スナップショットの制御情報をローカルメモリにキャッシュし高性能化することである．本稿では，効率的なスナップショット制御情報のキャッシュ方式と，プロトタイプシステムによる検証結果を述べる。

2. ストレージスナップショット機能

スナップショットは，ある瞬間のユーザデータのイメージを作成するための機能である．具体的には，ユーザデータが格納されているボリューム(Primary Volume：以後，PVOL)に対しスナップショット作成指示を受け付けると，その瞬間の PVOL のイメージを別のボリューム(Secondary Volume：以後，SVOL)に作りだす．一瞬でユーザデータのスナップショットデータを作成することができるため，バックアップ運用，データ分析業務などに欠かせない機能となっている．

スナップショットは，スナップショット作成後，PVOL または SVOL が更新されない限り PVOL と SVOL のデータが同一であるという特徴を利用し，PVOL と SVOL のデータが同一の間は，PVOL から SVOL へデータコピーを行わないことにより記憶容量を節約する．記憶容量節約を実現するため，スナップショットの SVOL は仮想的なボリュームとなっている．さらに，PVOL または SVOL 更新時にスナップショットデータを格納するための記憶領域として，PoolVOL を持つ(図 2)．PVOL または SVOL が更新されるタイミングで，PVOL の旧データを PoolVOL にコピーすることでスナップショットイメージを保持する．この旧データの PoolVOL へのコピーのことを退避コピーと呼ぶ．PoolVOL の利用効率を高めるため PoolVOL は複数のスナップショットで共有可能である．図 2 の例では，PVOL1 の一つ目および二つ目の領域は PVOL と SVOL のデータが異なっており，PoolVOL にスナップショットデータが格納されている．一方，PVOL の最後の領域(データ「お」)は PVOL と SVOL で同一でありデータを共有している．

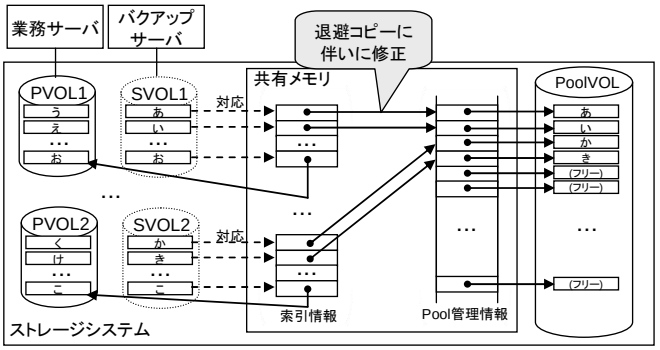

図 2　Copy-on-Write スナップショット

上述のように SVOL データの格納場所は状況により変化するため，データ格納位置を管理するための制御情報を持つ．これは，一般に，図 2 に示す様な索引情報と Pool 管理情報のような二種類の制御情報である．索引情報は PVOL の領域毎に一つのエントリがあり，スナップショットデータが格納されている PVOL の領域，または，PoolVOL の領域を管理する Pool 管理情報をポイントする．Pool 管理情報は PoolVOL の領域毎に一つのエントリがあり，PoolVOL の領域がフリーか，割り当て済みかなどの情報を管理する．

索引情報は PVOL 毎に存在するため，他の PVOL とは共有されないが，Pool 管理情報は PoolVOL を使用する全ての PVOL と SVOL で共有される．すなわち，どの SVOL のデータも任意の Pool 領域に格納される可能

性がある.また,図示していないが,SVOL がどの PVOL のスナップショットであるかを管理するための制御情報としてペア情報を持つ.

3. 課題と目標

3.1. 退避コピー時の制御情報更新

PVOL から PoolVOL へ旧データをコピーする退避コピーに伴い SVOL のデータ格納位置が変化するため,索引情報,Pool 管理情報を更新する必要がある.さらに,索引情報,Pool 管理情報は複数のコントローラから参照および更新される可能性がある.具体的には,索引情報は,PVOL ライトに伴う退避コピーで PVOL の担当コントローラから更新されるほか,SVOL ライトに伴う退避コピーで SVOL の担当コントローラから更新される可能性がある.Pool 管理情報は,同一の PoolVOL を使用する全 PVOL と SVOL で共有されており,各コントローラから更新される.上記のように複数のコントローラから参照および更新される制御情報をキャッシュする場合には,制御情報の一貫性を保証するための処理などが必要となる.

3.2. 集約方式と通信方式

制御情報をキャッシュし一貫性を保つ単純な方式として集約方式と通信方式が考えられる.

集約方式:集約方式は,図 3 に示す様に PoolVOL と当該 PoolVOL を使用する全 PVOL および SVOL の処理担当コントローラを同一にし,当該コントローラのローカルメモリに制御情報をキャッシュする方式である.この方式では,制御情報が複数コントローラで共有されず,一貫性保証処理が不要であり,処理効率が良い.

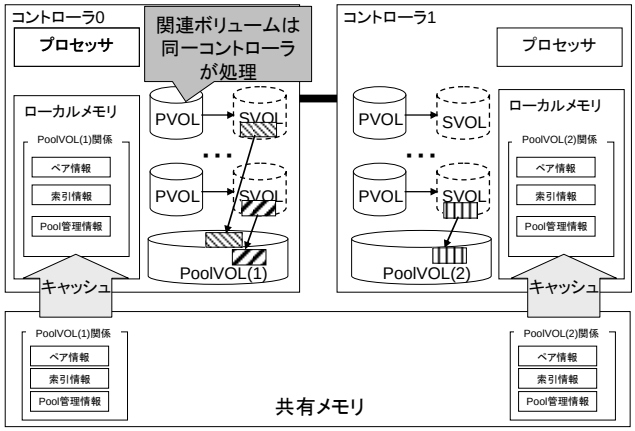

図 3　集約方式

通信方式:通信方式は,図 4 に示すように PVOL, SVOL, PoolVOL の担当コントローラを限定せず,PoolVOL と当該 PoolVOL を使用する PVOL, SVOL の処理を異なるコントローラが担当できる.スナップショットの制御情報は,各コントローラのローカルメモリにキャッシュし,一貫性の保証はメッセージ送受信によって実現する.具体的には,制御情報更新時,更新コントローラは,他コントローラに制御情報の更新をメッセージで通知する.メッセージを受信したコントローラは対象の制御情報をローカルメモリから破棄する.

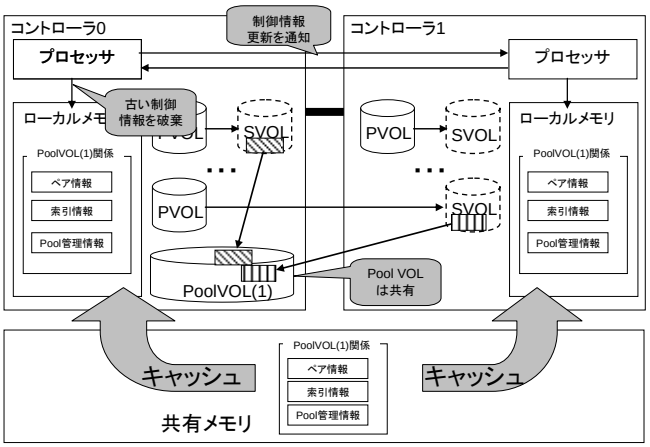

図 4　通信方式

集約方式と通信方式の間のトレードオフ:集約方式と通信方式はトレードオフの関係にある.集約方式は,スナップショットの制御情報をキャッシュでき,かつ,一貫性を保証するための処理が一切不要である.このため,制御情報キャッシュによる性能向上では,最大限の性能向上を実現できる.しかし,各コントローラに PoolVOL を用意しなければならず,PoolVOL の利用効率やユーザビリティが低下する.一方で,通信方式は一貫性保証のための通信により,集約方式より処理効率が低下する.しかし,各プロセッサから同一の PoolVOL を使用でき PoolVOL の利用効率は高い.

退避コピー比率と通信方式の性能の関係:通信方式のメッセージ送受信によりどの程度性能低下が発生するかを示す.メッセージ送受信オーバヘッド,制御情報破棄オーバヘッドを考慮し通信方式の性能を見積もった.退避コピー比率が向上すると,通信オーバヘッドが増える.図 5 は縦軸を PVOL ライト性能,横軸を退避コピー比率とし,通信オーバヘッドによる性能低下率を示している.退避コピー比率 0%の時点では,制御情報が更新されず,通信が発生しないため,両方式の性能は同じである.通信が発生する場合は,ストレージの I/O 処理時間に比べてメッセージ送受信オーバヘッドが大きく,退避コピー比率 10%で約 40%,退避コピー比率 20%以上で約 60%性能低下した.

3.3. 目標

集約方式同等の制御情報キャッシュによる性能向上効果と,PoolVOL の利用効率を両立するスナップショット方式を目指す.4 章では,通信方式をベースとした方式を提案する.

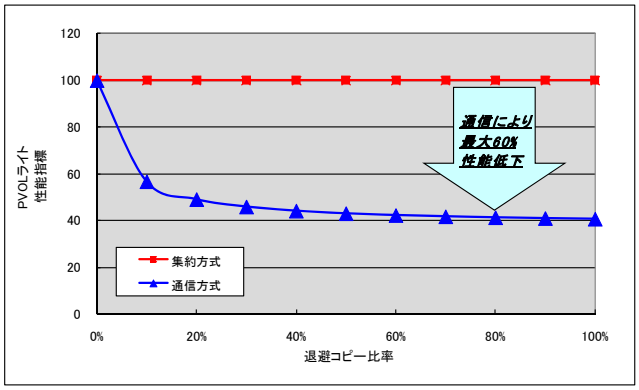

図 5　退避コピー比率と通信方式の性能の関係

4. 退避コピー時の一貫性保証処理の回避

4.1. 制御情報の分類

スナップショットには複数種類の制御情報があり，制御情報によってアクセス特性が異なるため，キャッシュした際の性能向上効果，一貫性保証処理オーバヘッドが異なる．どの制御情報をキャッシュする必要があり，どの制御情報をキャッシュすると一貫性保証処理オーバヘッドの影響が大きいかを明確にする．まず，図 6 に制御情報のアクセス頻度とアクセス局所性の組み合わせに対し，キャッシュ要否，キャッシュ時の一貫性保証処理オーバヘッドの性能影響をまとめた．図 6 の行は，I/O 当たりの参照頻度と I/O 当たりの更新頻度の組み合わせであり，列がアクセスの局所性である．アクセス局所性ありとは，一つのコントローラのみがアクセスすることを意味し，アクセス局所性なしは複数のコントローラから更新される可能性があることを意味している．

参照・更新頻度 ＼ アクセス局所性		アクセス局所性あり	アクセス局所性なし
IO当たりの参照少ない	IO当たり更新なし	○(一貫性保証処理不要)	○(一貫性保証処理OVH小)
	IO当たり更新あり	○(一貫性保証処理不要)	○(非キャッシュでもOVH小)
IO当たりの参照多い	IO当たり更新なし	○(一貫性保証処理不要)	○(一貫性保証処理OVH小) ・ペア情報
	IO当たり更新あり	○(一貫性保証処理不要) ・キャッシュ制御情報	×(一貫性保証処理OVH大) ・索引情報、Pool管理情報

○:問題なし　×:問題あり　OVH:オーバヘッド

図 6　問題となるアクセス特性

まず，アクセス局所性がある制御情報に関しては，参照および更新頻度に限らずキャッシュしても一貫性保証処理などが不要であり性能影響はない．アクセス局所性なしのうち，I/O あたりの更新がない制御情報は，一貫性保証処理オーバヘッドが小さくキャッシュしても性能への影響は小さい．I/O 当たりの参照頻度が少なく，かつ，I/O あたりの更新がある制御情報は，キャッシュしてしまうと一貫性保証処理オーバヘッドが大きいが，制御情報をキャッシュせず共有メモリアクセスを実行しても性能への影響が小さい．最後，アクセス局所性がなく，I/O 当たりの参照頻度，更新頻度が共に高い特性を持つ制御情報は，性能向上のためには，キャッシュしなければならないが，キャッシュしてしまうと一貫性保証処理オーバヘッドが大きく問題となる．

また，スナップショット機能の制御情報のアクセス特性は，図 6 に示すとおりである．ボリューム単位に管理されるキャッシュ管理情報，および，I/O 当たりの更新が発生しないペア情報は，単純にローカルメモリにキャッシュしても問題とならない．一方，索引情報，Pool 管理情報はアクセス局所性がなく，かつ，SVOL リード時などに多数回の参照行われるほか，退避コピーに伴い更新されるため，キャッシュした場合性能への影響が大きい．

次節に，索引情報，Pool 管理情報を効率的にキャッシュする方式を述べる．

4.2. 一貫性保証処理の回避

索引情報と Pool 管理情報をキャッシュし，一貫性保証処理を回避する方式を説明する．

(1) PVOL，SVOL 集約による索引情報のキャッシュ

索引情報は，PVOL と当該 PVOL に対するスナップショットに対して存在し，PVOL へのライト契機，または，SVOL のライト契機で更新される．このように，索引情報はアクセス局所性なしではあるが，更新するのは当該索引情報に対応する PVOL の担当コントローラ，または，SVOL の担当コントローラである．

索引情報には，上記のような特性があるため，スナップショット作成時に，PVOL または SVOL のどちらかの担当コントローラを変更し，PVOL と SVOL の担当コントローラを同一にすれば，アクセス局所性を持たせることができる(図 7)．

参照・更新頻度 ＼ アクセス局所性		アクセス局所性あり	アクセス局所性なし
IO当たりの参照少ない	IO当たり更新なし	○	○
	IO当たり更新あり	○	○
IO当たりの参照多い	IO当たり更新なし	○ ペア情報 ←	○ ペア情報
	IO当たり更新あり	○ 索引情報 ← キャッシュ管理情報	× 索引情報、Pool管理情報

PVOLとSVOLの担当プロセッサパッケージの同一化

○:問題なし　×:問題あり　OVH:オーバヘッド

図 7　索引情報のキャッシュ方法

PVOL と SVOL の担当コントローラを同一にするが，ストレージシステムが管理するボリューム数はコントローラ数より圧倒的に多いこと，異なる PVOL や PoolVOL は別のコントローラに処理を担当させることができることから考え，性能スケーラビリティの確保は可能である．

(2) アクセス特性変化による Pool 管理情報キャッシュ

Pool 管理情報は索引情報と異なり PoolVOL を共有する全ての PVOL および SVOL がアクセスする制御情報であり，全コントローラから参照および更新が行われる．よって，索引情報の様に PVOL と SVOL の担当コントローラを同一にするだけでは，アクセスを局所化することはできない．

本研究では，Pool 管理情報が管理対象とする Pool 領域がフリーから割当済みに遷移するのに伴い，当該 Pool 管理情報のアクセス特性が変化することに着目した．具体的には，フリー領域に対する Pool 管理情報は，領域を割り当てる時に，未使用領域をサーチするために参照され，割り当てに伴い割り当て済み情報の記憶などによる更新が発生する．そして，これらの参照および更新は全てのコントローラから行われる．一方，割り当て済み領域に対する Pool 管理情報は，SVOL リード時のデータ格納位置特定のために参照されるが，一旦，領域が割り当てられ，索引情報からポイントされる状態になった後は，更新が発生しないという特徴がある．また，割り当て先の PVOL と SVOL の I/O 時しかアクセスされないため，アクセス局所性も持つ(図 8)．

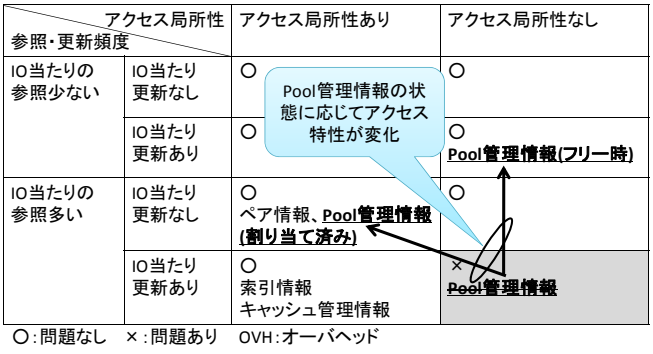

図 8　Pool 管理情報のキャッシュ方法

フリー領域を管理する Pool 管理情報と割当済み領域を管理する Pool 管理情報は上記のようなアクセス特性を持つ．そこで，Pool 領域割り当て時のアクセスでは共有メモリアクセスを行い，ローカルメモリへのキャッシュを行わず，一旦領域が割当たった後はローカルメモリにキャッシュ可能とするアクセス変更方式を提案する．これにより，一貫性保証処理を回避しつつ，大部分のアクセスをローカルメモリアクセスにすることができる．

5. 評価

5.1. アクセス変更方式性能と退避コピー率の関係

図 9 には，集約方式，通信方式，アクセス変更方式に対し，退避コピー比率と PVOL ライト性能の関係を示している．通信方式に比べ，通信オーバヘッドが削減により性能が大幅に改善した．退避率 100%では，集約方式から約 10%の性能低下となった．これは，フリーの Pool 領域に対する Pool 管理情報アクセスにおいて共有メモリアクセスを行うためである．

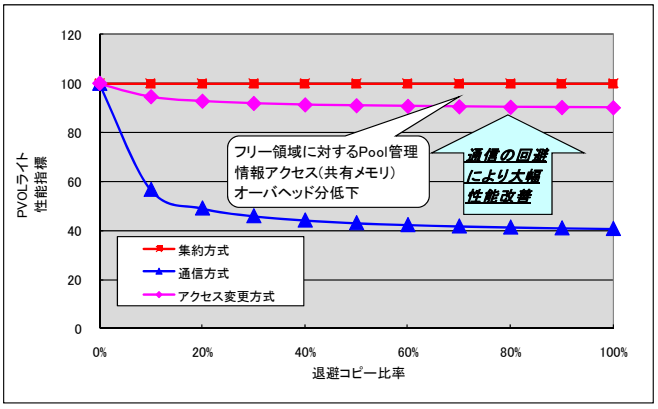

図 9　アクセス変更方式による性能改善

5.2. 通常運用性能の評価

プロトタイプシステムでの検証結果をまとめる．通常の運用状態に近い I/O パターンでの集約方式，通信方式，アクセス変更方式を比較した．通常の運用状態に近い I/O パターンとして，バックアップ処理が動作しない状態として運用時性能，バックアップ処理が動作している状態としてバックアップ時性能の二つを評価する．運用時性能は，ホスト I/O は全て PVOL に対して発行され，リードとライトの比率が 60:40 を想定した．PVOL のライトに伴う退避コピーの割合は，PVOL ライト全体の 10%を想定した．また，バックアップ時性能は，ホスト I/O の 60%が PVOL，40%が SVOL に対して発行され，PVOL のリードとライトの比率は 60:40 であり，SVOL はリード 100%とした．PVOL のライトに伴う退避コピーの割合は，通常運用時同様に PVOL ライト全体の 10%を想定した．結果を表 1 に示す．

退避コピー比率 100%では集約方式から約 10%性能が低下したが，通常の運用に近い運用時性能で約 5%，バックアップ時性能で約 3%の性能低下に収まり，ほぼ同等性能となった．

表 1　通常運用状態の性能比較結果

I/O パターン	集約方式	通信方式	アクセス変更方式
運用時性能指標	1.0	0.56	0.95
バックアップ時性能指標	1.0	0.68	0.97

5.3. 方式比較

各方式の比較結果を表 2 にまとめる．性能面では，集約方式は全制御情報をキャッシュ可能であるため，理想的な性能値である．アクセス変更方式は 5.1 節および 5.2 節で述べたように，集約方式とほぼ同等の性能である．これに対し，通信方式は，最悪の場合集約方式の 40%まで性能低下する可能性がある．

次に，PoolVOL の利用効率に関しては，集約方式以外の方式は任意のコントローラが担当する PVOL と SVOL から同一の PoolVOL を使用することができる．これに対して，集約方式は PoolVOL を共有できず PoolVOL の利用効率が悪い．

表 2　方式比較結果

観点	集約方式	通信方式	アクセス変更方式
性能	○ (全制御情報をキャッシュ可能であり理想の性能値)	× (最悪で集約方式から60%低下)	○ (集約方式ほぼ同等性能)
PoolVOL 利用効率	× (共有不可)	○ (共有可能)	○ (共有可能)

6. 関連研究

多くのストレージシステムでスナップショット機能が提供されているほか[5][6][7][8][9][10]，スナップショット方式の比較などが研究されている[3]．しかし，制御情報のキャッシュによる高性能化を狙った製品，研究はない．

7. まとめ

本稿では，ストレージシステムにおける Copy-on-Write スナップショットの高性能化目的に，スナップショット機能の制御情報を複数のコントローラのローカルメモリに効率よくキャッシュする方式を提案した．

一般に，スナップショットは，業務ボリュームとスナップショットの差分のみを，複数の業務ボリュームで共有される Pool と呼ばれる記憶領域に保存することで記憶容量を節約する．このため，Pool 領域を管理する制御情報は，全てのコントローラから参照および更新される可能性があり，ローカルメモリにキャッシュすると一貫生を保証するための処理が発生し期待通りの性能向上ができないという問題がある．

性能向上に対して問題となる制御情報のアクセス特性は，高頻度リードであり，I/O 単位で更新され，アクセス局所性がない制御情報である．本研究では，性能向上に対して問題となる制御情報が，Pool の割り当て状況に応じて，局所性なしから局所性ありに変化することに着目した．そして，Pool の割り当て状況に応じてキャッシュ可否を変更するアクセス変更方式を提案した．これにより，大部分の制御情報アクセスをローカルメモリアクセスにすることができた．

参　考　文　献

[1] Alain Azagury, Michael E. Factor and Julian Satran, “Point-in-Time Copy : Yesterday, Today and Tomorrow”, Proc. of the Tenth Goddard Conference on Mass Storage Systems and Technologies in Cooperation with the Nineteenth IEEE Symposium on Mass Storage Systems,pp.259-270, 2002.

[2] Marc Staimer, “Backup in a Snap : A Guide to Snapshot Technologies”, Storage Magazine Vol.8 No.7, pp.11-21 2009.

[3] Weijun Xiao, Qing Yang, Jin Ren, Changsheng Xie, Huaiyang Li, “Design and Analysis of Block-Level Snapshots for Data Protection and Recovery”, IEEE Transactions on Computers, vol. 58, no. 12, pp. 1615-1625, December 2009.

[4] 高橋直也，黒須康雄，“キャッシュメモリと共有メモリをもつディスクアレーの高速化手法”，信学論，vol.J86-D-I, no.6, pp.375-388, June 2003.

[5] Hitachi Data Systems Corp., “Hitachi ShadowImage Heterogeneous In-System Replication Software”, 2005.
http://www.hds.com/pdf/DISK-552-00.pdf

[6] Hitachi Data Systems Corp., “Hitachi Copy-on-Write Snapshot Software”, 2007.
http://www.hds.com/assets/pdf/ds-hitachi-copy-on-write-snapshot-software.pdf

[7] EMC Corp., “EMC Symmetrix TimeFinder Family of Software”, 2007.
http://japan.emc.com/collateral/software/data-sheet/1700-timefinder.pdf

[8] IBM Corp., “IBM System Storage DS8000 Series : IBM FlashCopy SE”, 2008.
http://www.redbooks.ibm.com/redpapers/pdfs/redp4368.pdf

[9] 3PAR Inc., “3PAR Virtual Copy”, 2010.
http://www.3par.com/SiteObjects/DA04D9276E5FE206825FD25A7064F291/vc-br-10.1.pdf

[10] IBM Corp., “IBM XIV Storage System Snapshots Reinvented”, 2008.
http://www.xivstorage.com/materials/white_papers/ibm_xiv_snapshots_paper.pdf