情報処理学会論文誌 Vol.61 No.2 275–286 (Feb. 2020)

オンライン処理とバッチ処理の処理負荷を分散制御する入出力制御方式の実装と評価

田辺 雅則[1,2,a)] 横山 和俊[3] 長尾 尚[1,4] 谷口 秀夫[1]

受付日 2019年5月8日, 採録日 2019年11月7日

概要：銀行のオンラインシステムに代表される業務システムでは，オンライン処理とバッチ処理が別計算機で実行される．しかし，計算機の処理能力向上とともに，計算機資源を十分に利用しない時間帯が生じ，分割損が生じている．両処理の計算機資源の利用量は時間帯で異なるため，両処理を同じ計算機で実行するシステム構成とすることで，分割損を軽減し，計算機資源の利用効率の向上が期待できる．しかし，このシステム構成においては，2 つの処理の負荷を分散させ，バッチ処理の負荷がオンライン処理の応答時間に悪影響を与えないようにすることが強く求められる．しかし，負荷分散は，プロセス優先度を用いたプロセッサ割当て制御だけでは不十分であり，ディスクに対する I/O 要求の負荷分散が必要である．本論文では，オンライン処理とバッチ処理のファイル I/O の特徴として入出力データ長に着目し，ディスクに対する I/O 要求の短期の負荷分散を行うディスク I/O 制御方式を提案し，FreeBSD に実装し評価した結果を述べる．

キーワード：ディスク，I/O，スケジューリング，オンライン処理，バッチ処理

Implementation and Evaluation of I/O Control Mechanism Controlling Workload Distribution Based on I/O Characteristics of Online and Batch Processing

Masanori Tanabe[1,2,a)] Kazutoshi Yokoyama[3] Takashi Nagao[1,4] Hideo Taniguchi[1]

Received: May 8, 2019, Accepted: November 7, 2019

Abstract: Some IT systems, including banking systems, perform online and batch processing on a daily basis. To separate the impact of both processings, typical banking systems employ individual computers dedicated to each processing. However, with the improvement of computer performance, computer resources of each dedicated system are not fully utilized. Since the use of computer resources for online and batch processing is different in time, it is expected to improve the utilization of computer resources by executing both processings on the same computer. In this system, it is strongly demanded that the workload of both processings are distributed so that the workload of batch processing does not affect the response time of online processing. It is not enough only controlling to assign each workload to processor units of a computer using priority of online and batch processing, and it is also needed to distribute I/O requests to a disk. In this paper, we propose an effective method for maximizing disk I/O performance under composite workloads. Our basic idea depends on the fact that workloads of online and batch processing (1) have different peak times in a day, and (2) have different I/O patterns of disk access and lengths of data. We describe the I/O control mechanism in FreeBSD and the evaluation of it in this paper.

Keywords: disk, I/O, scheduling, online processing, batch processing

1. はじめに

銀行のオンラインシステムに代表される業務システムで

[1] 岡山大学大学院自然科学研究科
Graduate School of Natural Science and Technology, Okayama University, Okayama 700–8530, Japan
[2] 株式会社 NTT データ
NTT Data Corporation, Koto, Tokyo 135–6033, Japan
[3] 高知工科大学情報学群
School of Information, Kochi University of Technology, Kami, Kochi 782–8502, Japan
[4] 株式会社日立製作所
Hitachi, Ltd., Chiyoda, Tokyo 100–8280, Japan
[a)] tanabems@s.okayama-u.ac.jp

は，オンライン処理とバッチ処理に分類される処理が毎日実行される．オンライン処理は，処理時間が比較的短いものが多く，サービスの応答時間が重視される．たとえば，入出金処理がある．また，利用者からの要求に基づく処理であるため，トランザクション量（単位時間あたりの処理量）は時刻変化し，あらかじめ計画して実行できない．このため，計算機資源の利用として，トランザクション量の変化に合わせた最適配分が重視される．一方，バッチ処理は，処理時間が短いものから長いものまで様々であり，スループットが重視される．たとえば，給与振込処理がある．また，バッチ処理は，定型的な処理であるため，事前に計画した処理として定めた時刻に実行を開始できる．このため，計算機資源の利用として，定めた時間で最適な配分と効率的な利用が重視される．

これらの性質の違う処理を実行するため，業務システムは，両処理を別の計算機で実行するシステム構成とすることが多い．一方，近年，計算機の処理能力が著しく向上したため，計算機資源を十分に利用しない時間帯が生じ，分割損が生じている．そこで，オンライン処理とバッチ処理の計算機資源の利用量は時間帯で異なるため，両処理を同じ計算機で実行するシステム構成とすることで，分割損を軽減し，計算機資源の利用効率の向上が期待できる．しかし，両処理を同じ計算機で実行する場合，バッチ処理がオンライン処理の応答時間に与える影響を軽減する必要がある．オンライン処理は昼間帯に実行され，バッチ処理は主に夜間帯に実行することで，長期の時間的な負荷分散は可能である．しかし，バッチ処理の増加とともに，バッチ処理を昼間帯で実行することも必要になっており，同時走行時の短期の時間的な負荷分散をできることが強く求められる．両処理の負荷を長期だけではなく短期にも分散させ，バッチ処理の負荷がオンライン処理の応答時間に悪影響を与えないようにすることが必要である．

計算機資源の利用量を処理の特徴にあわせて負荷分散できる方法として，プロセス優先度によるプロセッサ割当て制御法がある．しかし，プロセスの処理やプロセスが発行したシステムコール処理は，この制御法によって負荷分散できるが，割込み処理はこれらの処理に比べ優先して実行されるため負荷分散できない．つまり，たとえば，低優先度のプロセスが依頼した入出力要求による割込み処理は，高優先度のプロセスの処理より優先して実行される．したがって，入出力処理が比較的多いオンライン処理やバッチ処理においては，不十分である．具体的には，低優先度のバッチ処理がI/O要求を実行した場合，高優先度のオンライン処理はバッチ処理のI/O割込み処理の影響を受け，応答時間が遅くなってしまう．したがって，短期の時間的な負荷分散を実現するには，入出力処理を制御する方法に工夫が必要である．

本論文では，オンライン処理とバッチ処理が混在する環境において，バッチ処理によるオンライン処理の応答時間への影響を軽減するディスクI/O制御方式を提案する．具体的には，オンライン処理とバッチ処理の特徴を示し，バッチ処理が実行するディスクへのI/O要求がオンライン処理に与える影響と課題を述べ，オンライン処理のI/O要求を優先して実行するディスクI/O制御方式を提案する．また，評価として，提案方式を実装し，銀行オンラインシステムのモデル評価により有効性を示す．

2. オンライン処理とバッチ処理の混在環境

2.1 オンライン処理とバッチ処理の特徴

オンライン処理は，利用者からの要求に基づいて実行され，応答時間を重視することから，処理時間が短いものが多い．このため，プロセッサ利用時間が短く，ディスクへのI/O要求も少ない．一方，バッチ処理は，オンライン処理に比べ，プロセッサ利用量も多く，多量なデータのI/O要求を実行する．銀行オンラインシステムに代表されるシステムについて，オンライン処理とバッチ処理の特徴を述べる．

(1) オンライン処理とバッチ処理の処理の流れ

オンライン処理における1つのトランザクション処理の主な処理の流れは，利用者からの要求内容をディスクに書き込み，処理を実行し，実行結果をディスクに書き込み，利用者にその結果を返す．バッチ処理の主な処理の流れは，計算処理を実行し，結果をディスクに書き込むことを繰り返し行う．

(2) トランザクション数とバッチ処理数

オンライン処理とバッチ処理の計算機資源の利用量は時間帯で異なる．ある銀行システムにおけるバッチ処理について，通常営業日5日間の時間帯ごとのバッチ処理数の1日平均の実測データを**図1**に示す．オンライン処理は，主に昼間帯（6時から18時）に実行される．これに対し，昼間帯（6時から18時）のバッチ処理数は，夜間/深夜帯（18時から6時）に比較して少ないが，一定数存在している．

オンライン処理の計算機資源の利用量として，1日のピーク時間帯である午前中のオンライン処理のトランザク

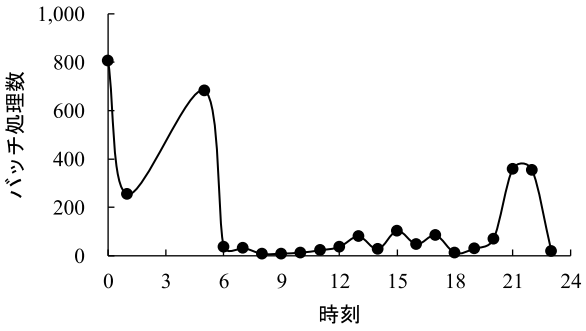


図1 1日のシステム運用におけるバッチ処理数
Fig. 1 Number of batch processing in a day.

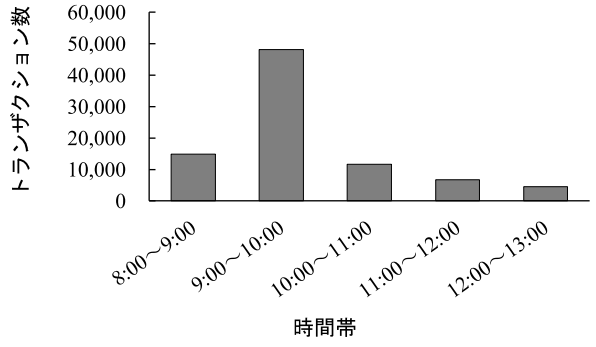


図 2 1 日のピーク時間帯のトランザクション数
Fig. 2 Number of transactions during peak times in a day.

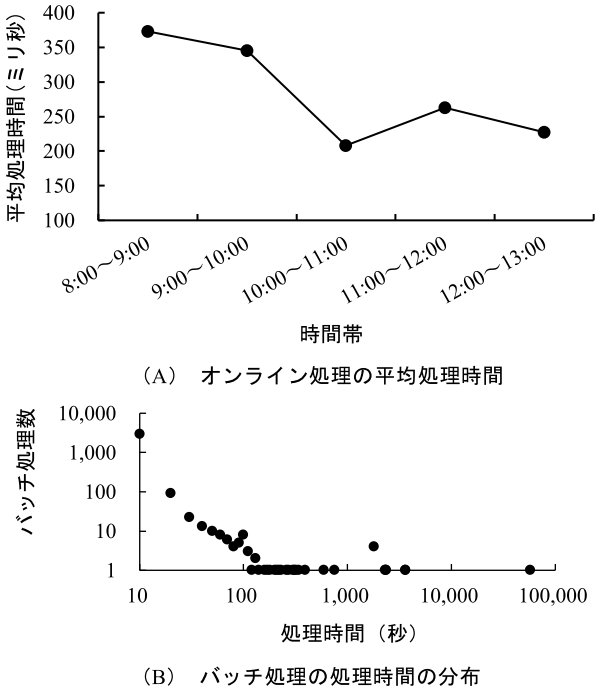


図 3 オンライン処理とバッチ処理の処理時間
Fig. 3 Execution time of online and batch processing.

ション数について，図 1 と同様な環境における実測データを**図 2** に示す．9 時から 10 時のピーク時間におけるオンライン処理の負荷は約 13TPS（トランザクション数は約 48,000）である．

(3) 処理時間

図 3 に図 1 と同様な環境におけるオンライン処理の平均処理時間とバッチ処理の処理時間の分布を示す．図 3 (A) より，オンライン処理の平均処理時間は，約 200 ミリ秒から 370 ミリ秒である．なお，最小処理時間は 160 ミリ秒程度である．なお，図 3 (A) のオンライン処理の平均処理時間には，通信にかかる処理時間も含まれている．また，図 3 (B) より，バッチ処理の処理時間は，短いものから長いものまで，様々である．1 日に実行されるバッチ処理の処理時間を単純平均した場合，バッチ処理の平均処理時間は約 30 秒である．

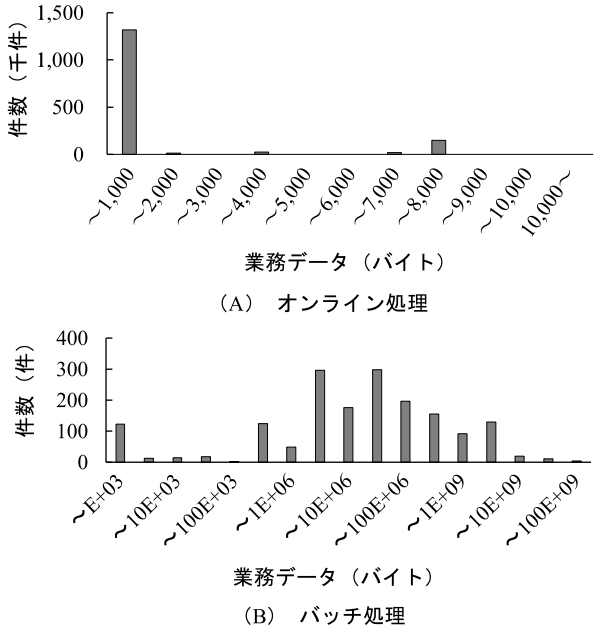


図 4 書き込みデータ長の分布
Fig. 4 Distribution of written data length.

(4) 書き込みデータ長

オンライン処理とバッチ処理は，ディスクに書き込むデータ長に大きな違いがある．図 1 と同様な環境におけるオンライン処理とバッチ処理の業務データのデータ長の分布を**図 4** に示す．図 4 (A) より，オンライン処理の書き込みデータ長は，1,000 バイト以下が大半を占める．一方，図 4 (B) より，バッチ処理の書き込みデータ長は，数 MB から 100 MB である．

2.2 課題

オンライン処理とバッチ処理の計算機資源の利用量の大半は，時間帯で異なるため，長期の負荷分散は可能である．しかし，オンライン処理が多い昼間帯にもバッチ処理が実行される．このため，以下の課題に対処し，短期の時間的な負荷分散を行う．

ディスクドライバは，ディスク装置の I/O 要求の完了による割込み制御によって，I/O 要求の実行が完了した後の処理を開始する．このため，バッチ処理の I/O 要求が実行されると，オンライン処理の I/O 要求はバッチ処理の I/O 要求の完了まで待たされる．この様子を**図 5** に示す．図 5 では，バッチ処理が I/O 要求の実行をディスクドライバに要求し，その後，オンライン処理は I/O 要求の実行をディスクドライバに要求する様子を示している．ディスクドライバは，バッチ処理による I/O 要求を受け付けた後，前処理を実行し，実 I/O 処理をディスクに要求する．そして，実 I/O 処理の完了（割込み）を待つ．その後，実 I/O 処理の完了（割込み）を契機に，I/O 要求完了後処理（割込み処理）を実行する．この I/O 要求完了後処理（割込み処理）は他の処理より優先して実行される．バッチ処理は，

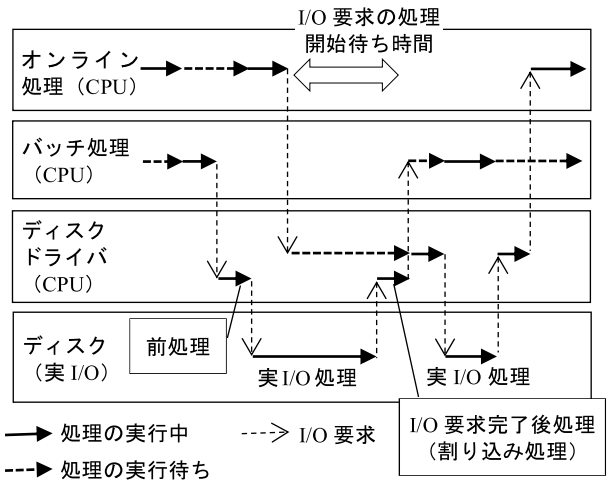


図 5 I/O 要求の処理開始待ち時間
Fig. 5 Wait time of I/O request processing start.

ディスクドライバによる I/O 要求の処理完了後，後続の処理を開始する．一方，オンライン処理による I/O 要求は，バッチ処理の I/O 要求の実行後，バッチ処理の I/O 要求完了後処理（割込み処理）が終わるまで実行されない．つまり，この間，オンライン処理による I/O 要求が，ディスクドライバによって待たされるため，オンライン処理の I/O 要求の処理開始待ち時間が発生する．したがって，オンライン処理の I/O 要求を優先して実行させるという課題がある（課題 1）．

また，バッチ処理の I/O 要求のデータ長が大きい場合，つまりバッチ処理が長い処理時間の I/O 要求を実行した場合，オンライン処理の I/O 要求の処理開始待ち時間は長くなる．この待ち時間によるオンライン処理への影響をいかに軽減するかが課題である（課題 2）．

なお，計算機資源の利用量を処理の特徴にあわせて負荷分散できる方法として，プロセス優先度によるプロセッサ割当て制御法がある．しかし，プロセスの処理やプロセスが発行したシステムコール処理は，この制御法によって負荷分散できるが，割込み処理はこれらの処理に比べ優先して実行されるため負荷分散できない．また，バッチ処理の実 I/O 時間を短縮することはできない．したがって，オンライン処理の応答時間を保証するには，プロセス優先度による優先制御では解決できないため，新たな入出力制御方式が必要である．

3. データ長を考慮したディスク I/O 制御方式

3.1 基本機能

図 4 に示したように，書き込みデータ長は，オンライン処理がバッチ処理に比べ非常に小さい．そこで，書き込みデータ長の長大化を防ぎ，かつ，書き込みデータ長が小さい I/O 要求を優先的に実行するディスク I/O 制御方式を提案する．データ長を考慮したディスク I/O 制御方式は，2 つの特徴を持つ．1 つは，I/O 要求のデータ長が大きい場合は，データ長を規定値以下になるように分割して実 I/O を行う．これにより課題 2 に対処できる．もう 1 つは，データ長が短いものから実 I/O を行う．これにより課題 1 に対処できる．この方式の基本構造を**図 6** に示し，以下に説明する．

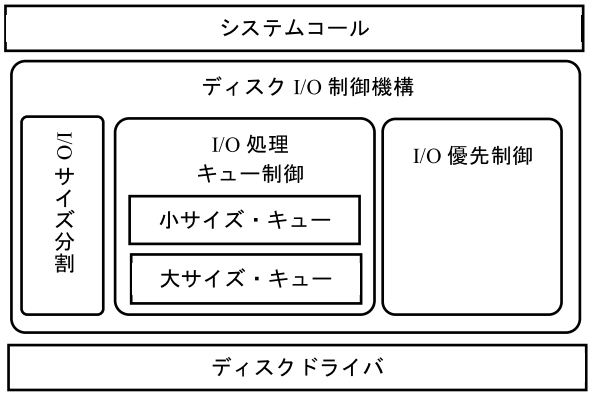


図 6 ディスク I/O 制御方式の基本構造
Fig. 6 Overview of disk I/O control mechanism.

(1) I/O サイズ分割機能

書き込みデータ長が非常に大きいものについて，分割し（図 6 では規定値を「大サイズ」としている），1 つ 1 つの I/O 要求の処理時間を短くする．これにより，書き込みデータ長が非常に大きいバッチ処理の I/O 要求を分割して小さくし，バッチ処理の実 I/O 時間を短くしてオンライン処理への影響を抑制する．

(2) I/O 処理キュー制御機能

I/O サイズの違いごとに実行待ちキューを用意して管理する．各キューは I/O 要求到着順である．小サイズ・キューは，I/O サイズが小さい I/O 要求のキューであり，オンライン処理の I/O 要求に相当する．一方，大サイズ・キューは，I/O サイズ分割機能によって分割された I/O 要求のキューであり，バッチ処理の I/O 要求に相当する．

(3) I/O 優先制御機能

小サイズ・キューの I/O 要求を大サイズ・キューの I/O 要求より優先させて，ディスクドライバに I/O 要求の実行を依頼する．つまり，I/O サイズが小さい I/O 要求を優先して実行する．なお，ディスクドライバ処理中に新たな I/O 要求が発生しても，優先制御を確実に行うために，I/O 優先制御がディスクドライバに依頼する処理は 1 度に 1 つである．これにより，データ長が短いオンライン処理の実 I/O を優先して実行し，バッチ処理の I/O 要求完了後処理の影響を抑える．

3.2 処理の流れ

I/O 制御機構の処理の流れを**図 7** に示し，以下に説明する．

(1) システムコール発行により，OS に I/O 要求の実行が要求される．

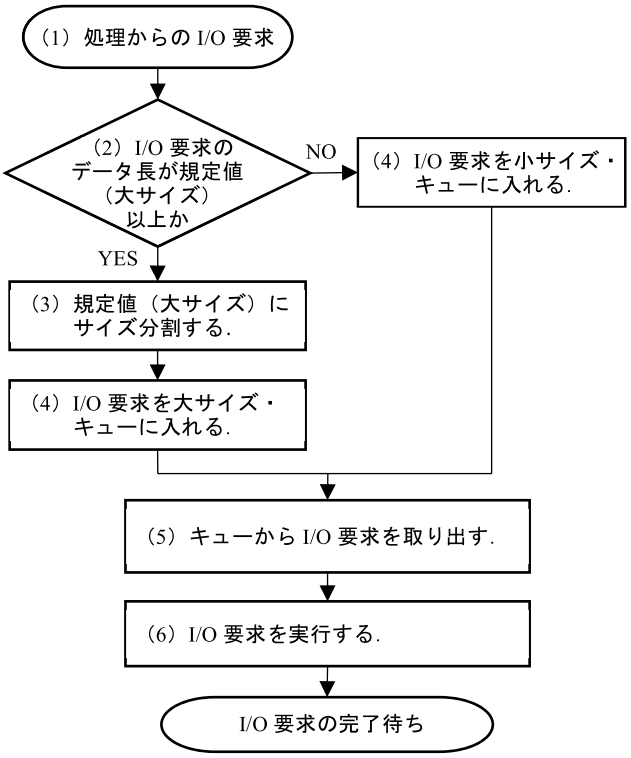


図 7 ディスク I/O 制御機構の処理の流れ
Fig. 7 Flow of disk I/O control mechanism.

(2) I/O 要求のデータ長が規定値（大サイズ）以上かどうかを判定する．規定値（大サイズ）以上の場合，I/O 要求を規定値（大サイズ）に分割する．
(3) I/O 要求を規定値（大サイズ）に分割する．
(4) I/O 要求のサイズに基づきキューに入れる．
(5) キューから I/O 要求を取り出す．優先度の高い小サイズ・キューから優先的に I/O 要求を取り出す．
(6) I/O 要求を実行し，処理結果を待つ．I/O 要求は，その割込み処理により，呼び出し元に戻る．

4. 実装と評価

4.1 評価観点

提案したディスク I/O 制御方式を FreeBSD に実装し，その有効性を以下の評価項目で明らかにする．

（評価項目 1）課題 1 および課題 2 の対処に対するディスク I/O 制御方式の評価として，オンライン処理の負荷（TPS）とオンライン処理の処理時間の関係を明らかにし，ディスク I/O 制御方式を使用したオンライン処理の処理時間が，プロセス優先度によるプロセッサ割当て制御を使用したオンライン処理の処理時間よりも短くなること．

（評価項目 2）課題 2 の対処に対する I/O サイズ分割機能の評価として，バッチ処理の I/O 要求を分割して小さくし，バッチ処理の 1 回の実 I/O 時間を短くすることでオンライン処理への影響を抑制することができること．

（評価項目 3）プロセス優先度によるプロセッサ割当て制御では解決できないことに対する評価として，プロセス優先度によるプロセッサ割当て制御ではオンライン処理の I/O 要求の処理時間が短くならないこと，すなわちディスク I/O 制御方式の方がオンライン処理の I/O 要求の処理時間が短くなること．

表 1 測定に使用した計算機環境の諸元
Table 1 Specification of a computer.

CPU	Intel i5 3.2GHz 4 コア ハイパースレッドなし	
ディスク	I/O パス	SATA3.0
	DISK	7400 回転 32MB キャッシュ
	ファイルシステム	UFS 32KB ブロック
	ファイルシステムのキャッシュ	使用しない
OS	FreeBSD 11.0-RELEASE-p1	

上記の評価項目 1 と評価項目 3 で評価対象とする制御方式は，以下の 3 つである．また，評価項目 2 については，I/O 要求優先における I/O サイズ分割機能を評価対象とする．

(1) 無優先
(2) CPU 優先（オンライン処理優先）

プロセス優先度によるプロセッサ割当て制御によって，オンライン処理を優先的に実行する．なお，オンライン処理の I/O 要求は優先的に実行されない．

(3) I/O 要求優先（オンライン処理優先）

ディスク I/O 制御方式を使用した優先制御によって，オンライン処理の I/O 要求を優先的に実行する．

4.2 評価環境

ディスク I/O 制御方式の評価に使用する計算機の諸元を**表 1** に示す．銀行システムのオンライン処理では，トランザクションデータのように失うことが許容されないデータは，書き出しのたびに，永続的なディスク等にデータを保存することが必須である．このため，ファイルシステムのキャッシュを使用しない．そこで，オンライン処理およびバッチ処理がデータを書き出すために使用するデータ用ディスクは，ファイルシステムのキャッシュの影響をなくすため，UFS のキャッシュを使用しない設定とする．

バッチ処理の I/O 要求について，I/O 要求を処理する効率は，書き込みデータ長が大きいほど良い．ところが，書き込みデータ長が大きいほど，オンライン処理の I/O 要求の処理開始待ち時間は長くなり，オンライン処理への影響が大きい．そこで，大サイズは，オンライン処理の I/O 要求の処理時間と同等で，かつ，できるだけ大きいサイズに設定する．具体的には，ディスク I/O 制御方式の I/O サイズ分割機能によって分割する大サイズは，オンライン処

理の 1,000 バイトのデータ書き込み時間と同程度の書き込み時間となるデータ長とする．表 1 の計算機を使用した場合，1,000 バイトのデータ書き込み時間は 0.926 ミリ秒，10,000 バイトの書き込み時間は 1.26 ミリ秒である．このため，分割後の I/O サイズについて，大サイズを 10,000 バイトとする．

4.3 評価用プログラム

4.3.1 処理内容

ディスク I/O 制御方式を評価するため，ある銀行システムのオンライン処理とバッチ処理を擬似した評価用プログラムを使用する．具体的には，オンライン処理は，外部からデータをオンラインで受信し，集計・加工処理を実行し，その結果を外部に送信する．このようなオンライン処理では，送受信するデータはディスク等に保存される．そこで，評価用プログラムのオンライン処理は，受信したデータをディスクに保存し，その後，データベース等を使用して集計・加工処理を実行後，処理結果データをディスクに保存し，処理結果データを送信する．また，オンライン処理は外部からの多数の要求を受け付けて処理を実行する．バッチ処理は，銀行におけるデータベース等に格納されているデータ等の集計・加工処理を実行する．バッチ処理は，大量データを順次処理するため，データベースよりも高速なアクセスが期待できるファイルを使用することが多い．そこで，評価用プログラムのバッチ処理は，データベースからデータを取り出し，集計・加工処理を実行し，処理結果データをファイルに書き出す．また，オンライン処理が実行される時間帯では，少数のバッチ処理が実行される．評価用プログラムの処理の流れを**図 8** に示す．

図 8 (A) は，オンライン処理の 1 トランザクション処理を擬似した処理（オンライン処理）であり，処理実行（初期処理），要求内容データ書き出し，処理実行（主処理），DB 処理実行結果待ち，結果データ書き出し，処理実行（後処理）からなる．表 1 の計算機を使用したオンライン処理の処理時間は，オンライン処理の同時起動数 20 の場合，オンライン処理の平均処理時間は 193 ミリ秒，平均 CPU 使用率は 40%であり，図 3 (A) のオンライン処理の処理時間相当である．

図 8 (B) は，バッチ処理を擬似した処理（バッチ処理）であり，処理実行（初期処理），処理実行（主処理），DB 処理実行結果待ち，結果データ書き出し，処理実行（後処理）からなり，処理実行（主処理），DB 処理実行結果待ち，結果データ書き出しの処理を繰り返す．全体の処理時間が 30 秒となるように，繰返し回数を 150 回とした．表 1 の計算機を使用したバッチ処理の処理時間は，バッチ処理の同時起動数 4 の場合，バッチ処理の平均処理時間は 28 秒，平均 CPU 使用率は 30%であり，図 3 (B) のバッチ処理の平均処理時間相当である．

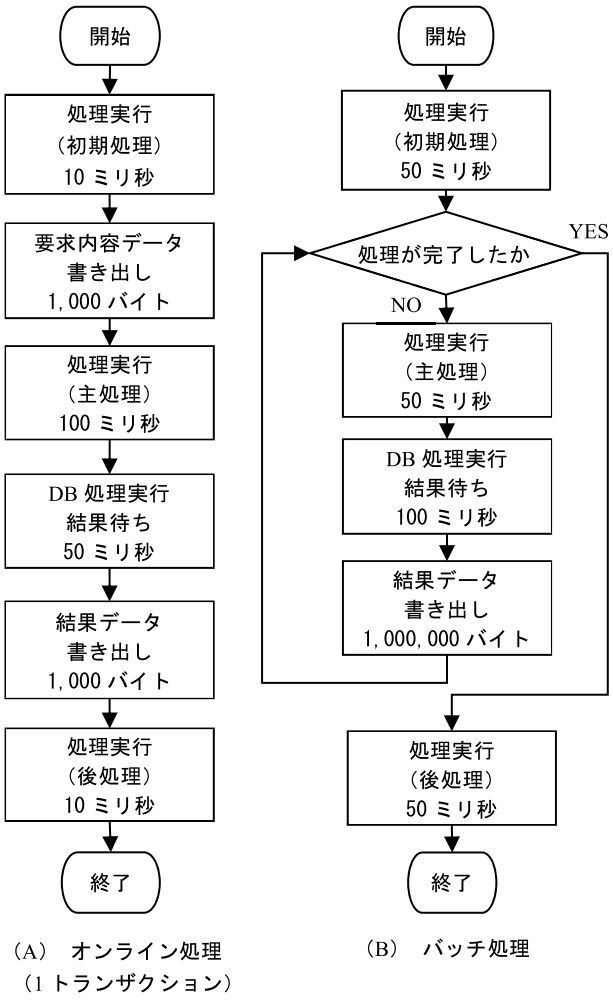


図 8 評価用プログラムの処理の流れ
Fig. 8 Flow of evaluation programs.

4.3.2 起動条件

オンライン処理とバッチ処理の起動は，オンライン処理の負荷，および，バッチ処理数が多い場合と少ない場合の組合せとする．また，両処理の同時起動により，プロセッサ資源がボトルネックとなることを避けるため，最大の同時起動数は，CPU 使用率が 90%程度になるように決める．評価用プログラムの処理時間を考慮したオンライン処理とバッチ処理の起動条件を以下に示す．

(1) オンライン処理
  (A) オンライン処理の到着は，0〜3 秒の間隔でランダムに到着するものとする．
  (B) 同時起動数は 10，20，30，40 とする．

(2) バッチ処理
  (A) バッチ処理は，同時に起動したバッチ処理の完了後，次のバッチ処理を同時に起動する．
  (B) バッチ処理の同時起動数は，少ない場合として 2，多い場合として 4 とする．

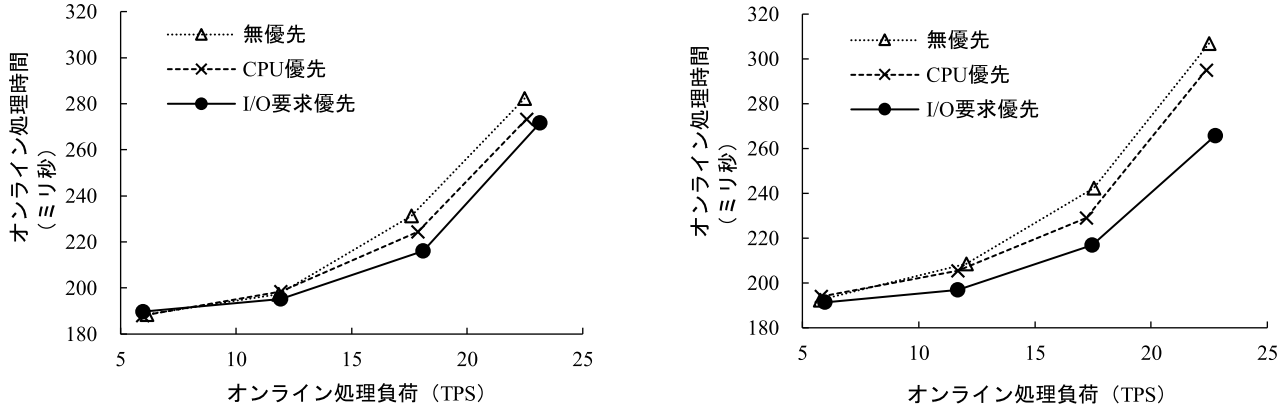


図 9 オンライン処理時間
Fig. 9 Execution time of online processing.

4.4 結果と考察

4.4.1 処理時間

評価項目 1 について結果を示す．オンライン処理とバッチ処理を同時に起動したときのオンライン処理の処理時間を**図 9** に示す．図 9 より以下のことが分かる．

(1) バッチ処理の同時起動数に関係なく，I/O 要求優先はオンライン処理負荷が大きいほどオンライン処理時間が他方式に比べ短く，効果が大きい．たとえば，バッチ処理の同時起動数が 2 の場合，オンライン処理の負荷 11.8 TPS（無優先），11.9 TPS（CPU 優先），および，11.9 TPS（I/O 要求優先）を比較すると，I/O 要求優先は無優先より 1.1%（2.18 ミリ秒），CPU 優先より 1.7%（3.38 ミリ秒）短い．一方，オンライン処理の負荷 17.6 TPS（無優先），17.9 TPS（CPU 優先），および，18.1 TPS（I/O 要求優先）を比較すると，I/O 要求優先は無優先より 6.7%（15.47 ミリ秒），CPU 優先より 3.8%（8.52 ミリ秒）短く，I/O 要求優先の効果は大きくなる．

(2) 同じようなオンライン処理負荷であっても，I/O 要求優先はバッチ処理負荷が大きい（バッチ処理の同時起動数が多い）ほど，効果が大きい．たとえば，先に述べたように，バッチ処理の同時起動数が 2 の場合，オンライン処理の負荷 17.6 TPS（無優先），17.9 TPS（CPU 優先），および，18.1 TPS（I/O 要求優先）を比較すると，I/O 要求優先は無優先より 6.7%（15.47 ミリ秒），CPU 優先より 3.8%（8.52 ミリ秒）短い．一方，バッチ処理の同時起動数が 4 の場合，オンライン処理の負荷 17.5 TPS（無優先），17.2 TPS（CPU 優先），および，17.5 TPS（I/O 要求優先）を比較すると，I/O 要求優先は無優先より 10.5%（25.6 ミリ秒），CPU 優先より 5.3%（12.2 ミリ秒）短く，I/O 要求優先の効果は大きくなる．

オンライン処理のみを起動したとき（バッチ処理の同時起動数なし）のオンライン処理の処理時間を**図 10** に示す．図 10 より以下のことが分かる．

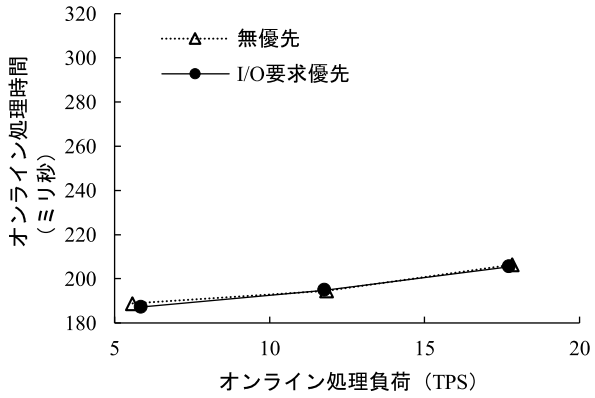


図 10 バッチ処理の同時起動数無のオンライン処理時間
Fig. 10 Execution time of online processing only.

(3) I/O 要求優先は，無優先に比較して，オンライン処理への影響は軽微である．無優先と I/O 要求優先の処理時間の差は，0.24%（0.5 ミリ秒）から 0.84%（1.6 ミリ秒）であり，ほぼ同じである．

(4) オンライン処理とバッチ処理が同時に実行されるとき，I/O 要求優先のオンライン処理の処理時間の増加は，無優先と比較して小さい．たとえば，図 10 と図 9 (B) のバッチ処理の同時起動数 4 を比較する．無優先の場合，同時起動数なしのオンライン処理負荷 17.8 TPS（無優先）に対する同時起動数 4 のオンライン処理負荷 17.5 TPS（無優先）の処理時間の増加は 14.8%（35.9 ミリ秒）である．これに対し，I/O 要求優先の場合，同時起動数なしのオンライン処理負荷 17.7 TPS（I/O 要求優先）に対する同時起動数 4 のオンライン処理負荷 17.5 TPS（I/O 要求優先）の処理時間の増加は 5.2%（11.3 ミリ秒）であり小さい．なお，オンライン処理負荷が小さい場合は，この差が少なくなる．

次に，オンライン処理とバッチ処理を同時に起動したときのバッチ処理の処理時間を**図 11** に示す．図 11 より以下のことが分かる．

(5) I/O 要求優先は，無優先および CPU 優先と比べ長

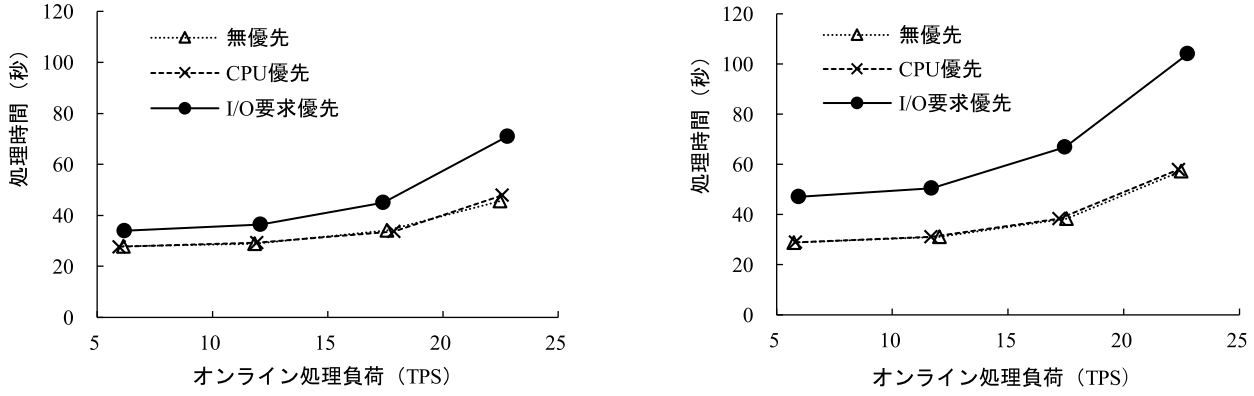


図 11 バッチ処理時間
Fig. 11 Execution time of batch processing.

表 2 I/O サイズ分割の有無とオンライン処理の処理時間
Table 2 Execution time of online processing with or without I/O request division.

バッチ処理の同時起動数	2				4			
オンライン処理の同時起動数	30		40		30		40	
オンライン処理の負荷（TPS）	18.1	17.4	23.2	22.8	17.5	17.7	22.8	22.4
I/O サイズ分割	有	無	有	無	有	無	有	無
オンライン処理の処理時間（ミリ秒）	216.0	218.6	271.6	276.3	216.8	235.9	265.8	288.4

い．これは，I/O 要求優先によって，オンライン処理の I/O 要求を優先的に実行した結果である．なお，バッチ処理の処理時間は，図 11 (A) のバッチ処理の同時起動数 2 のオンライン処理負荷 6.2 TPS（I/O 要求優先）の場合，6.1 TPS（無優先）より 6.1 秒，6.0 TPS（CPU 優先）より 6.3 秒長い程度（約 22.6%）である．また，最も影響が大きい場合つまり，図 11 (B) のバッチ処理の同時起動数 4 のオンライン処理負荷 22.8 TPS（I/O 要求優先）の場合，22.5 TPS（無優先）より 46.7 秒，22.4 TPS（CPU 優先）より 46.0 秒長い（約 79.4%）．いずれの場合もバッチ処理を計画的に実行することにより，大きな問題とはならない．

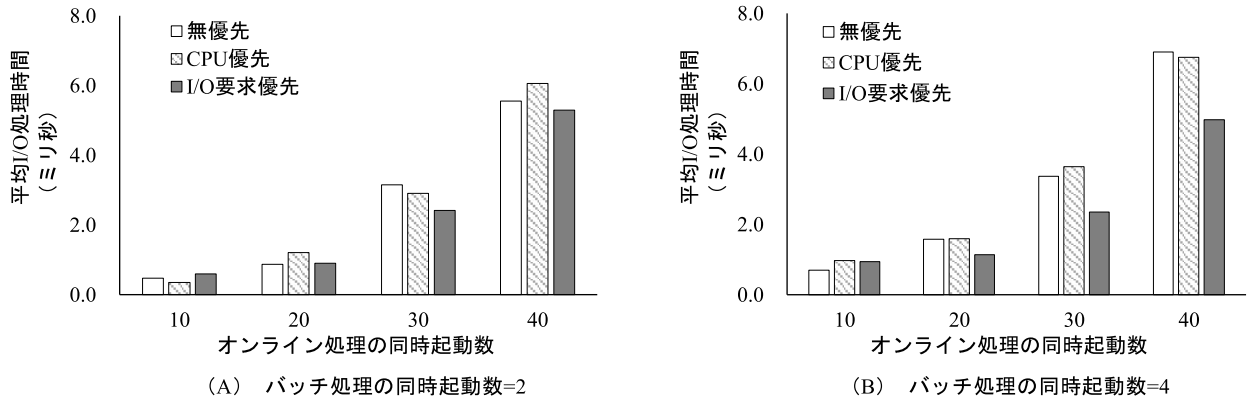
4.4.2 I/O サイズ分割の効果

3.1 節（1）I/O サイズ分割機能で述べたように，I/O サイズ分割により，バッチ処理の 1 回の実 I/O 時間を短くし，バッチ処理の I/O 要求がオンライン処理に与える影響を抑制する．ここでは，評価項目 2 について，ディスク I/O 制御方式の I/O サイズ分割機能の有効性を評価する．具体的には，ディスク I/O 制御機構における I/O サイズ分割機能がある場合とない場合について，オンライン処理の処理時間を比較する．**表 2** にディスク I/O 制御方式における I/O サイズ分割機能の有無とオンライン処理の処理時間を示す．オンライン処理の処理時間は，I/O サイズ分割を行った場合（I/O サイズ分割が「有」）は I/O サイズ分割を行わなかった場合（I/O サイズ分割が「無」）より短い．この効果は，オンライン処理の同時起動数の影響は小さく，バッチ処理の同時起動数が多い場合に大きい．たとえば，オンライン処理の処理時間は，バッチ処理の同時起動数 4 の場合は約 8%も短縮されるが，バッチ処理の同時起動数 2 の場合は 1 から 2%の短縮である．

4.4.3 I/O 要求の処理時間

評価項目 3 として，オンライン処理の I/O 要求の処理時間を評価する．オンライン処理の I/O 要求の処理時間の平均値（以降，平均 I/O 処理時間とよぶ）を**図 12** に示す．図 12 より，以下のことが分かる．

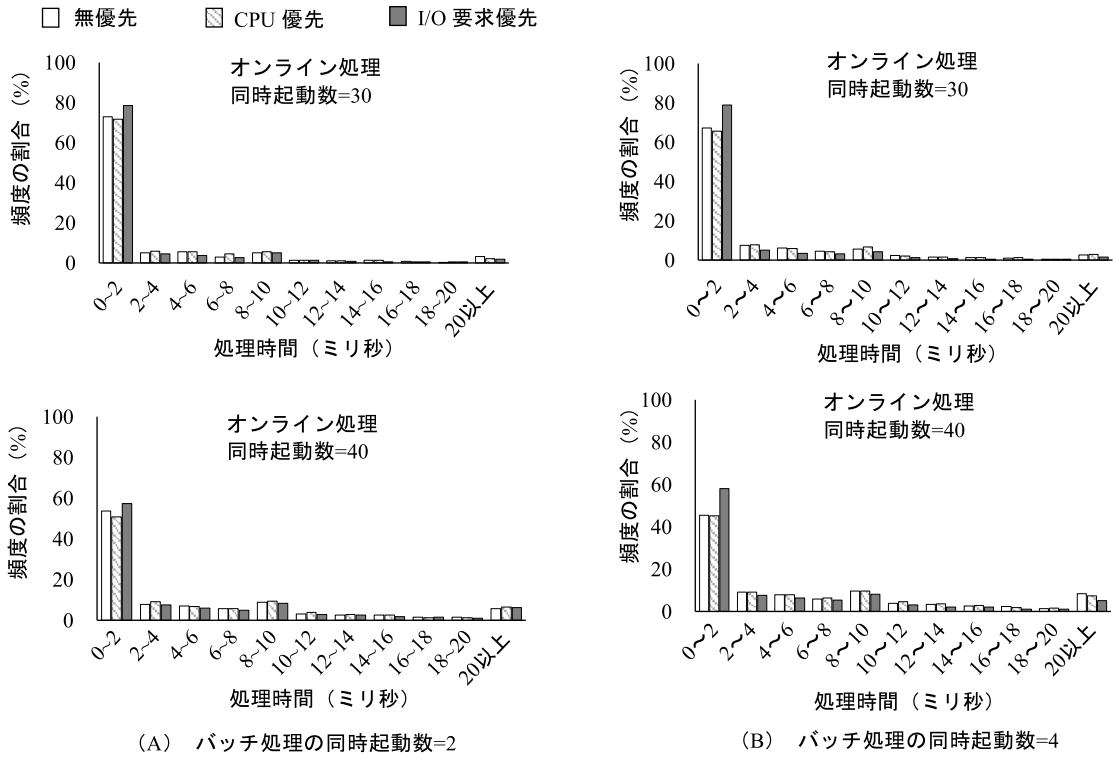
（1）オンライン処理の同時起動数が多い，つまりオンライン処理の負荷（TPS）が大きい場合，I/O 要求優先の平均 I/O 処理時間は，無優先および CPU 優先に比べて短い．この効果は，バッチ処理が多いときほど大きい．たとえば，バッチ処理が少なく（同時起動数 2），オンライン処理の同時起動数 40 のとき，I/O 要求優先の平均 I/O 処理時間は，無優先に比べ 4.7%（0.26 ミリ秒）短く，CPU 優先に比べ 12.7%（0.77 ミリ秒）短い．バッチ処理が多く（同時起動数 4），オンライン処理の同時起動数 40 のとき，I/O 要求優先の平均 I/O 処理時間は，無優先に比べ 27.8%（1.9 ミ



（A） バッチ処理の同時起動数=2　　（B） バッチ処理の同時起動数=4

図 **12** オンライン処理の I/O 要求の処理時間

Fig. 12 Execution time of I/O request in online processing.



（A） バッチ処理の同時起動数=2　　（B） バッチ処理の同時起動数=4

図 **13** オンライン処理の I/O 要求の処理時間分布

Fig. 13 The distribution of execution time of I/O requests in online processing.

リ秒）短く，CPU 優先に比べ 26.3%（1.78 ミリ秒）短い．

（2）CPU 優先は，バッチ処理が少なく（同時起動数 2），オンライン処理の同時起動数 40 および 20 のとき，無優先よりも平均 I/O 処理時間が悪化している．また，バッチ処理が多いとき（同時起動数 4），オンライン処理の同時起動数，つまりオンライン処理の負荷（TPS）にかかわらず，平均 I/O 処理時間は無優先と同等もしくは悪化している．

したがって，I/O 要求優先の制御は，バッチ処理の多少（同時起動数）にかかわらず，オンライン処理の負荷（TPS）が大きいとき，オンライン処理の平均 I/O 処理時間を短くできる．一方，CPU 優先の制御では，平均 I/O 処理時間が悪化する（長くなる）ことがある．

次に，オンライン処理の I/O 要求の処理時間の分布を**図 13** に示す．いずれの方式も，バッチ処理の多少やオンライン処理の同時起動数に関係なく，I/O 要求の処理時間の大半は 2 ミリ秒未満である．I/O 要求の処理時間が 2 ミリ秒から 10 ミリ秒の間である場合も見られるものの，I/O 要求の処理時間が 10 ミリ秒以上である場合は非常に少ない．そこで，I/O 要求の処理時間を，2 ミリ秒未満（区分 1），2 ミリ秒から 10 ミリ秒未満（区分 2），10 ミリ秒以上（区分 3）に区分し，その割合を**図 14** に示す．図 14 より，以下のことが分かる．

（1）オンライン処理の同時起動数にかかわらず，I/O 要求優先の場合は他の場合に比べ区分 1 の割合が最も高い．これは，バッチ処理の同時起動数が多い場合と少ない場合のいずれも同じである．また，区分 1 と区分 2 の和も同様

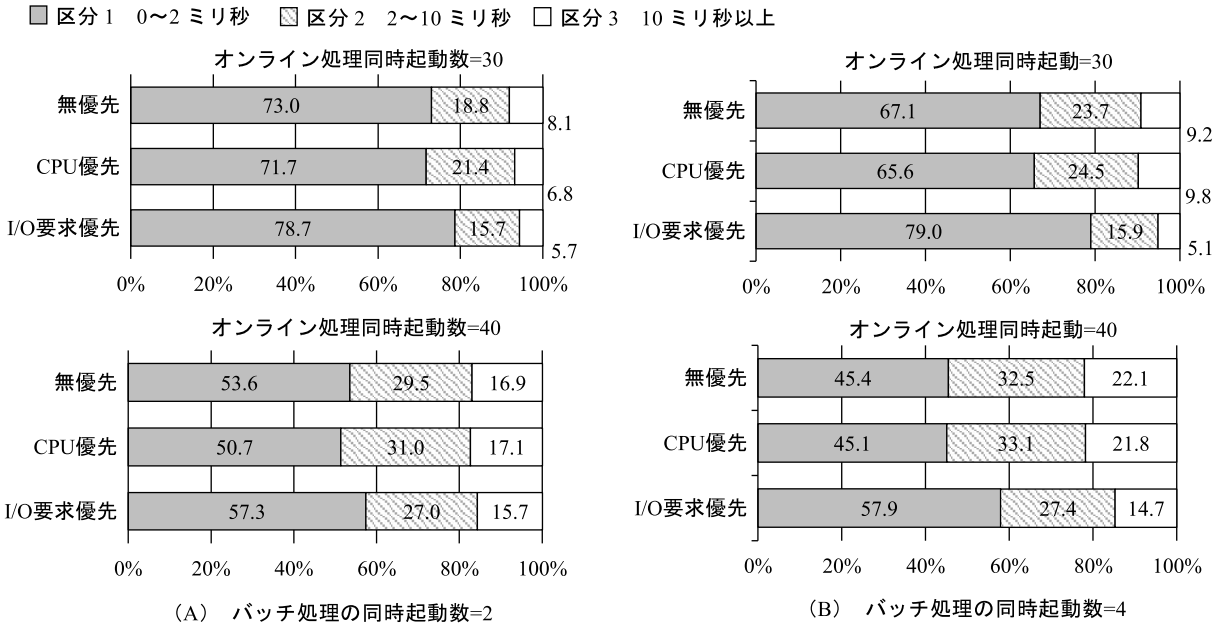


図 14 オンライン処理の I/O 要求の処理時間区分ごとの割合
Fig. 14 Time ratio of execution time of I/O requests in online processing.

である．したがって，I/O 要求優先の制御は，他の方式に比べ，多くの I/O 要求について I/O 要求の処理時間を短くできる．

（2）CPU 優先は，無優先より区分 1 の割合が小さいため，I/O 要求の処理時間が短い場合は少ない．

5. 関連研究

文献 [1], [2], [3], [4] は，I/O 要求に対するシーク時間や回転待ち時間により，I/O 要求の処理時間が大きくなることに着目し，書き込みデータのディスク上の書き込み位置等の情報と I/O 要求の処理時間を予測し，I/O 要求を効率的に実行するスケジューリング手法を提案している．文献 [5] は，入出力スループットの向上を目的として，入出力要求を並べ替えて HDD のシーク時間を最小とする I/O 要求のスケジューリング手法を提案している．また，近年普及している SSD は入出力要求の処理時間が短いため，OS と入出力デバイス間の入出力要求の授受に要する時間が入出力時間に影響を与えやすい．文献 [6] は入出力要求をまとめて発行することで授受の回数を削減している．文献 [7], [8] は，SSD ではメモリセルへの読み書きにおいて排他が必要となるため，読み書き対象の領域が分散するように入出力要求を並べ替えて排他の発生を防いでいる．これらの方法では，オンライン処理の I/O 要求をバッチ処理の I/O 要求より優先させて実行することはできない．

文献 [9] は，アプリケーションの I/O 要求の実行にあたって，I/O 要求の優先度，I/O 要求の開始時間と終了時間をもとにして，優先的に実行する I/O 要求を決定し，それらの I/O 要求を複数まとめ，スループットを向上させるために I/O 要求の実行順番を入れ替える方式を提案している．この方式では，実行順番が入れ替わることがあるため，銀行システムのオンライン処理では使用できない．また，この方式では，オンライン処理の I/O 要求をまとめて実行することになり，オンライン処理の I/O 要求開始待ち時間が増加することがある．その結果，オンライン処理の応答時間の最大値が大きくなり好ましくない．

リアルタイム処理と QoS 保証を実現する場合として，I/O 要求の優先度制御が提案されている．文献 [10] は，リアルタイムシステムにおいて，サービス時間を短縮することとデッドラインを守ることを実現するため，I/O 要求の優先度とシーク時間を基にした評価関数を用いて，実行する I/O 要求を決定することで，リアルタイム処理のデッドラインを守りながらサービス時間の短縮を図る手法を提案している．また，文献 [11] は，リアルタイム処理とベストエフォートな処理が混在する環境において，I/O 要求を実行するスケジューラが，リアルタイム処理のデッドライン等の情報に基づき，各 I/O 要求の処理時間を予測し，その予測に基づいて，ベストエフォート処理およびリアルタイム処理の I/O 要求の優先度を決定する手法を提案している．文献 [12] は，デッドライン制御に関して Deferrable Schedule と組み合わせることで平均応答時間を改善している．文献 [13], [14] は，ストレージスタック（ファイルシステムレベル，ブロックレベル，デバイスレベル等）における各階層の I/O 要求の優先度制御に関して，各階層にある I/O 要求の情報（処理時間や優先度，デッドライン等）をそれぞれの階層の優先度制御で利用する制御方式を提案している．しかし，いずれの提案もデッドラインを規定しスケジュールできることが前提となっており，負荷変動が大きく，かつデッドラインを規定し難いオンライン処理には

適さない．

文献 [15] は，I/O 要求の処理時間を調整する制御法を提案している．この提案では，高い優先度を持つ I/O 要求が優先的に実行されるように I/O 要求の処理時間を調整することで，優先度の高い処理の I/O 要求の完了待ちによる待ち時間を短くできることを示している．しかし，この方法はスループットが低いという問題がある．

6. おわりに

オンライン処理とバッチ処理が同じ計算機で走行する環境において，バッチ処理によるオンライン処理の応答時間への影響を軽減するために，バッチ処理の I/O 要求に対して，短期の時間的な負荷分散を行うディスク I/O 制御方式を提案した．本制御方式は，オンライン処理の I/O 要求の書き込みデータ長が小さいことに着目し，書き込みデータ長を考慮した入出力制御を行うことで，I/O 要求の短期の時間的な負荷分散を行う．提案方式を FreeBSD に実装し，銀行オンラインシステムのモデルとしてオンライン処理とバッチ処理を使用し評価した．

提案方式により，オンライン処理の処理時間は，オンライン処理の負荷が小さいとき，プロセッサ割当ての優先制御に比較して，1.7%短くなる．また，オンライン処理の負荷が大きいとき，プロセッサ割当ての優先制御に比較して，バッチ処理の負荷が小さいとき 3.8%短く，バッチ処理の負荷が大きいとき 5.3%短くなる．また，提案方式はバッチ処理の負荷に関係なく，オンライン処理に対してバッチ処理の I/O 要求の影響を軽減でき，オンライン処理およびバッチ処理の負荷が大きいほど，バッチ処理の I/O 要求に対する短期の時間的な負荷分散の効果が大きくなる．

また，提案方式により，オンライン処理の I/O 要求の処理時間は，オンライン処理の負荷が大きく，バッチ処理が多い場合，プロセッサ割当ての優先制御に比較して 26.3%短くできる．なお，プロセス優先度によるプロセッサ割当て制御法では，バッチ処理の負荷が大きいとき，オンライン処理の I/O 要求の処理時間が同等もしくは悪化することがある．

残された課題として，I/O サイズ分割機能で，分割する大サイズの大きさが処理時間に与える影響の明確化がある．

参考文献

[1] Huang, L. and Chiueh, T.-C.: Implementation of a Rotation-Latency-Sensitive Disk Scheduler, Technical Report ECSL-TR81, SUNY, Stony Brook University at New York (Mar. 2000).

[2] Iyer, S. and Drushel, P.: Anticipatory scheduling: A disk scheduling framework to overcome deceptive idleness in synchronous I/O, *Proc. 18th ACM Symposium on Operating Systems Principles* (Oct. 2001).

[3] Valente, P. and Checconi, F.: High Throughput Disk Scheduling with Fair Bandwidth Distribution, *IEEE Trans. Comput.*, Vol.59, No.9, pp.1172–1186 (2010).

[4] 田邨優人，中島耕太，山本昌生，前田宗則：CPU 処理と I/O 処理の高速性を両立する I/O 制御機構，情報処理学会研究報告，システムソフトウェアとオペレーティングシステム，Vol.2017-OS-141, No.22 (2017).

[5] Basavaraju, G., Kondugari, P.K. and Shankaraiah: Accelarting the performance of Disk with Re-Oredering of An Input/Output Requests at Virtual Machine Monitor Level, *Proc. IEEE International Conference on Computer Communication and Systems* (*ICCCS14*), pp.245–247 (Feb. 2014).

[6] Son, Y., Yeom, H.Y. and Han, H.: Optimizing I/O Operations in File Systems for Fast Storage Devices, *IEEE Trans. Comput.*, Vol.66, No.6, pp.1071–1084 (2017).

[7] Jo, M.H. and Ro, W.W.: Dynamic Load Balancing of Dispatch Scheduling for Solid State Disks, *IEEE Trans. Comput.*, Vol.66, No.6, pp.1034–1047 (2017).

[8] He, S., Wang, Y., Sun, X., Huang, C. and Xu, C.: Heterogeneity-Aware Collective I/O for Parallel I/O Systems with Hybrid HDD/SSD Servers, *IEEE Trans. Comput.*, Vol.66, No.6, pp.1091–1098 (2017).

[9] Bruno, J., Brustoloni, J., Gabber, E., Mcshea, M., Özden, B. and Silberschatz, A.: Disk Scheduling with Quality of Service Guarantees, *Proc. IEEE International Conference on Multimedia Computing and Systems*, pp.7–11 (June 1999).

[10] Chen, S., Stankovic, J.A., Kurose, J.F. and Towsley, D.: Performance Evaluation of Two New Disk Scheduling Algorithms for Real-Time Systems, *Real-Time Systems*, Vol.3, No.3, pp.307–336 (Sep. 1991).

[11] Le Moal, D., Molaro, D. and Campello, J.: A Real-Time File System for Constrained Quality of Service Applications, *IPSJ Trans. Advanced Computing Systems*, Vol.3, No.1, pp.61–76 (Mar. 2010).

[12] Han, S., Chen, D., Xiong, M., Lam, K.-Y., Mok, A.K. and Ramamritham, K.: Schedulability Analysis of Deferrable Scheduling Algorithms for Maintaining Real-Time Data Freshness, *IEEE Trans. Comput.*, Vol.63, No.4, pp.979–994 (Apr. 2014).

[13] Kim, S., Kim, H., Lee, J. and Jeong, J.: Enlightening the I/O Path: A Holistic Approach for Application Performance, *the 15th USENIX Conference on File and Storage Technologies* (*FAST '17*), pp.345–358 (Mar. 2017).

[14] Yang, S., Harter, T., Agrawal, N., Kowsalya, S.S., Krishnamurthy, A., Al-Kiswany, S., Kaushik, R.T., Arpaci-Dusseau, A.C. and Arpaci-Dusseau, R.H.: Split-level I/O Scheduling, *Proc. SOSP '15 Proc. 25th Symposium on Operating Systems Principles* pp.474–489 (Oct. 4-7, 2015).

[15] 長尾　尚，谷口秀夫：入出力要求数の制御によりサービス時間を調整する制御法の実現と評価，電子情報通信学会論文誌 D，Vol.J94-D, No.7, pp.1047–1057 (2011-07-01).

田辺 雅則

1990 年京都工芸繊維大学電子工学科卒業．1992 年同大学大学院工芸科学研究科博士前期課程修了．同年 NTT データ通信株式会社（現，株式会社 NTT データ）入社．現在，同社にて銀行向けのシステム開発に従事．

横山 和俊 （正会員）

1988 年広島大学工学部第二類（電気系）卒業．1990 年同大学大学院工学研究科博士前期課程修了．2006 年岡山大学大学院自然科学研究科博士後期課程修了．1990 年 NTT データ通信株式会社（現，株式会社 NTT データ）入社後，オペレーティングシステム，分散処理の研究開発に従事．2012 年高知工科大学情報学群教授．博士（工学）．電子情報通信学会会員．

長尾 尚 （正会員）

2009 年岡山大学工学部情報系卒業．2011 年同大学大学院自然科学研究科博士前期課程修了．2011 年株式会社日立製作所横浜研究所に入社．現在，同社にてストレージシステム向けオペレーティングシステムの研究開発に従事．

谷口 秀夫 （正会員）

1978 年九州大学工学部電子工学科卒業．1980 年同大学大学院修士課程修了．同年日本電信電話公社電気通信研究所入所．1987 年同所主任研究員．1988 年 NTT データ通信株式会社開発本部移籍．1992 年同本部主幹技師．1993 年九州大学工学部助教授．2003 年岡山大学工学部教授．2010 年岡山大学工学部長．2014 年岡山大学理事・副学長．博士（工学）．オペレーティングシステム，実時間処理，分散処理に興味を持つ．著書『並列分散処理』（コロナ社）等．電子情報通信学会，ACM 各会員．本会フェロー．