令和 4 年度

修士学位論文

ハードディスクの動作環境の違いを考慮した入出力性能調整法

I/O Performance Regulation Method Considering Differences in Hard Disk Operating Environments

1255111　　谷　玲治

指導教員　　横山和俊

2023 年 2 月 28 日

高知工科大学大学院 工学研究科 基盤工学専攻

情報学コース

要 旨

ハードディスクの動作環境の違いを考慮した入出力性能調整法

谷　玲治

OS 上で稼働するプログラムの多くは, プロセッサ処理と入出力処理を繰り返しながら, 利用者にサービスを提供する. 我々はこれまでに, 利用者の利便性向上のため, プロセッサ性能や入出力性能の調整法を提案している. この入出力性能調整法では, 入出力デバイスが受け付けられる数である許容値を算出し, 入出力回数が許容値を超えないように制限する. また, 入出力処理実行後に目標の入出力時間になるまで遅延処理をすることで入出力処理時間を一定に調整している. しかし, ハードディスクの動作環境が異なり入出力処理の挙動に差がある場合, 従来手法では, 許容値が固定的であるため, 調整精度が悪くなる場合がある. そこで入出力性能調整結果を監視し, 調整精度が悪い場合, 許容値を動的に変更する許容値の補正法を提案する. また, 評価により, 提案手法は, 複数のハードウェア動作環境において, 入出力時間を精度よく調整できること, 補正によって最適な許容値を算出でき, その値を維持できることを示す.

キーワード　オペレーティングシステム, 入出力処理, 入出力スケジューリング, 性能調整

Abstract

I/O Performance Regulation Method Considering Differences in Hard Disk Operating Environments

Reiji Tani

Most of the programs running on the OS provide services to users by repeating processor processing and I/O processing. We have proposed a method to regulate processor performance and input/output performance for user convenience. In this I/O performance adjustment method, the allowable value, which is the number of I/O devices that can be accepted, is calculated and the number of I/O time is limited so that it does not exceed the allowable value. In addition, the I/O processing time is adjusted to a constant value by delaying the I/O processing until the target I/O time is reached after the I/O processing is executed. However, when there is a difference in the behavior of input/output processing due to different operating environments of hard disks, the conventional method may result in poor adjustment accuracy because the allowable value is fixed. Therefore, we propose a method to monitor the results of I/O performance regulation and to dynamically change the allowable value when the accuracy of regulation is poor. The evaluation also shows that the proposed method can precisely regulate hte I/O performance in multiple hardware operating environments, and can calculate and maintain the optimum allowable value by the correction.

key words Operating System, I/O processing, I/O Scheduling, Performance Tuning

目次

目次

図目次

表目次

第 1 章

はじめに

現在, 一台の計算機上で同時に複数のソフトウェアを動作させる利用形態が一般的となった. 例えば動画を視聴しながらエディタで文書を作成し, ウイルス対策ソフトウェアでファイルを検査させるといった利用形態である. この時, 利用者は他のソフトウェアの影響により直接操作するソフトウェアの入力受付や画面表示が滞るとストレスを感じる場合がある. 利用者がソフトウェアの利用を開始するとき, 計算機内ではオペレーションシステム（以降, OS と呼ぶ）が該当ソフトウェアに対応するプロセスを生成し, ソフトウェアの動作を管理する. 具体的には, OS がどのプロセスにどの時間だけプロセッサを割り当てるか, プロセスの入出力要求をどの順番で入出力デバイスに発行するか, を制御する. この制御によって, OS が他プロセスを優先すると, 利用者が直接操作するソフトウェアの動作が遅れることがある. このような動作の遅れを防ぐには, 利用差が直接操作するソフトウェアに対応するプロセス（ここでは重要なプロセスと呼ぶ）が使用する計算機資源を他プロセス（ここでは重要ではないプロセスと呼ぶ）にかかわらず一定に保ち, 重要なプロセスの動作時間を安定させることが重要である. すなわち, 重要なプロセスに割り当てるプロセッサの時間や入出力要求の処理に要する時間（以降, 入出力時間と呼ぶ）を一定に保つことが重要であり, これらを実現できるプロセッサスケジューラと入出力スケジューラが必要となる. スケジューラとは, プロセスに計算機資源を割り当てる制御機構のことである. プロセッサスケジューラとして優先度順スケジューラやラウンドロビンスケジューラ, 多段フィードバック待ち行列スケジューラなどが存在する. また入出力スケジューラとして Linux では noop, deadline, acticipatory, cfq と 4 つのアルゴリズムが実装されている.

これまで, 著者らは, 割り当てるプロセッサ時間を一定に保つプロセッサ性能調整法 [1] と,

入出力時間を一定に保つ入出力性能の調整法 [2] を提案した. 入出力性能の調整法では, 利用者は, 重要なプロセスに要求入出力性能と呼び値を設定して調整対象にする. 本調整法では, 調整対象プロセスの入出力要求を, 調整対象はないプロセスの入出力要求よりも先に処理することで入出力時間を保証するとともに調整対象プロセスの起床を遅延させる. これらにより, 要求入出力性能に合わせて, プロセスの入出力時間を調整する. 文献 [3] で提案された入出力順序保証方式では, 要求入出力性能が同じ他の調整対象プロセスが先に入出力要求を発行する場合においても, 理想の入出力時間で入出力処理を完了することを保証するとともに, スループット向上を実現した.

しかし, ハードディスクの動作環境が異なり, 入出力処理の挙動に差が場合, 調整対象プロセスの要求入出力性能とそのプロセス数によって一意に許容値が算出される従来手法では, これに対応できない.

そこで, 入出力性能調整動作を監視し, 利用している許容値で入出力性能調整がうまくできていないと判断した場合に許容値を補正する許容値補正法を提案する. 提案手法では, 理想の入出力時間に対して, 各入出力処理に要した実際の入出力時間が超過した時間を蓄積し, 閾値を超えた時, 許容値を利用していた区間の調整精度を計測し, 調整精度が一定水準を満たさない場合, 補正を行う. 評価により, 提案手法によって許容値を補正することで, 評価に利用した 3 種のハードディスクの動作環境において精度良く調整できることを示す. またハードディスクに最適な許容値に補正することを示す.

第 2 章

関連研究

この章では, 入出力性能調整法の関連研究について説明する.

2.1 入出力性能調整法

これまでにソフトウェア (以降, プロセス) の入出力処理速度を一定に保つ手法として入出力性能調整法を提案している. 実行速度を調整できる機能があることで, 利用者はプロセスを自身が使いやすい速度にコントロールできるようになるため, ソフトウェアやサービスの利便性の向上につながる. 本制御法は調整したいプロセスに対して, 独占的な入出力資源を提供しているため, 他プロセスの入出力要求の影響を受けない性能調整が可能である. ここでは文献 [2], [3], [4] を紹介する.

2.1.1 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評価

文献 [2] では, 入出力性能を調整する制御法として, OS が入出力デバイスに発行可能な入出力要求数を調整する制御法を提案している. 入出力要求数に制限をかけることで, 既にデバイスドライバや入出力デバイス内に入出力要求が存在しており, 調整したいプロセスの入出力処理がそれらの処理待ちになったとしても利用者が望む入出力処理性能 (入出力時間) を担保している. 本手法により, 他プロセスの入出力要求の影響を受けない精度の良い性能調整を実現している.

2.1.2　入出力スループットの低下を抑制する入出力性能の調整法

文献 [2] の入出力性能調整法では, 調整対象のプロセスが複数実行されている場合, 全てのプロセスで精度のよい性能調整は実現できるが, OS が入出力デバイスに発行可能な入出力要求数を必要以上に制限してしまい, 入出力デバイスのスループットを低下させてしまう問題を抱えていた. この問題を解決する手法として, 文献 [3] では入出力順序保証方式を提案している. 本手法は他の調整対象のプロセスの影響による入出力処理の待ち時間の増加は (1) 自プロセスよりも高い入出力性能を与えられたプロセスが入出力要求を発行していること, (2) 同じ入出力性能を与えられたプロセスが自プロセスより先に処理されること, が要因で発生すると考えた. この二点を踏まえた上で最長の待ち時間を見込む, つまり (1)(2) が先に処理されたとしても, 利用者が望む入出力処理性能 (入出力時間) を担保できる最大値で制限をかけることで精度の良い性能調整とスループットの向上を実現している.

2.1.3　Tender オペレーティングシステムにおける資源「入出力」の実現と評価

文献 [4] では, 分散指向永続オペレーティングシステム Tender 上に入出力性能調整法を実現している. Tender では各入出力デバイス (入出力資源) をルートとした入出力木で管理し, 親ノードが持つ入出力資源内で子ノードに入出力資源を割り当てる. 入出力資源を持つ各ノードとプロセスを関連づけることでノードが持つ入出力資源を利用できる. 一方, 関連づけられていない資源については一切利用できない. 入出力木で資源を管理することで, プロセスグループ単位の性能調整の保証とその管理が容易となる. 本件では, 性能調整の精度低下を防ぐため, 発行可能な入出力要求数を更新する契機として次の 6 つを定義している.

(1) 入出力に対するプロセスの関連付け

(2) 入出力に対するプロセスの関連付け解除

(3) 入出力要求発行可否判定

(4) 遅延処理終了後

(5) 入出力の性能変更

(6) 許容係数の変更

(1)(2)(5)(6) については, 発行可能な入出力要求数の算出式の項が変更されるため, 再計算が必要な契機である. (3) については調整対象プロセスの入出力要求が発行され, 入出力処理が完了するまでは, その調整対象プロセスによる入出力要求が発生しないため, 算出式の対象から外す契機である. (4) は算出式の対象に戻す契機である.

2.2　関連研究の問題点

文献 [2] により, 他プロセスの入出力要求の影響を受けない精度の良い性能調整を実現し, 文献 [3] により, スループット向上を実現した. しかし, 本手法を複数種類のハードディスクを用いて検証したところ, 調整精度の結果がハードディスクごとに異なり, [3] の結果と比較して精度が悪くなる場合があることが確認された. ハードディスクの動作環境の違いによって調整精度が悪くなる具体的な要因は 5 章 1 項に記述する. これまでの入出力性能調整法では要求入出力性能と調整対象プロセスの数によって固定的に許容値が算出されていたため, ハードディスクの動作環境が異なり, 入出力処理の挙動に差がある場合に対応できない. そこで, 入出力性能調整動作を監視し, 性能調整が精度よく実行できる許容値に動的に変化させることができればどのハードディスク環境でも精度よく性能調整ができると考えた. なお, [4] においても許容値を動的に変化させて場面はあるが, 調整対象プロセスが入出力処理実行中は調整すべき入出力要求が発生しないため, 許容値を大きくし, ハードウェア割り込み処理が発生したタイミングで元の許容値に戻すことでスループットを向上させることを目的としており, ハードディスクの動作環境の違いに対応したものではない. そこで本研究では, 特にハードディスクの動作環境の違いを考慮し, ハードディスクごとに最適な値に発行可能な入出力要求数を補正する許容値補正法を提案する.

第 3 章

入出力処理の流れ

3.1 入出力処理

プロセスが入出力デバイス内部のデータを読み書きする場合のシステムコールの処理の流れを図 3.1 に示す．まずプロセスは入出力処理を行う際, read/write システムコールを実行し, CPU の動作モードをユーザモードからカーネルモード変更し, プロセスの実行状態（コンテキスト）を保存して OS 内部に遷移する．OS はまず初めにメモリ上のデータにアクセスする．読み込みの場合はデータが既にメモリ上に存在する時, そのデータをプロセスの領域に書き写し, 処理を終了する．書き込みの場合, 多くは非同期書き込みを実行するため, メモリ上にデータを書き込んで処理を終了する．メモリ上に読み込むデータが存在しない場合や書き込んだデータを入出力デバイスに反映する場合に限り, 入出力デバイスへのアクセスが発生する．OS は入出力デバイスごとにキューを持ち．アクセスする入出力デバイス用の待ちキューにプロセスの入出力要求を格納し, 順番が来るまでプロセスを待ち状態にする．次に, OS が入出力デバイスの状態を監視し, 入出力要求を受け取り可能であれば, 待ちキューから入出力要求を取り出し, デバイスドライバに渡す．デバイスドライバは, 入出力要求を入出力デバイスが解釈可能なコマンドに変化して発行する．入出力デバイスは, 受け取った入出力要求を解釈し, 記憶媒体を読み書きする（実 I/O 処理と呼ぶ）．この後, 入出力デバイスは, OS に入出力要求の処理完了を通知する．OS が通知を受け取ると, 対応するプロセスを起床する．そして, プロセスは結果を確認し, 自身の次の処理に進む．以上がシステムコールの流れである．

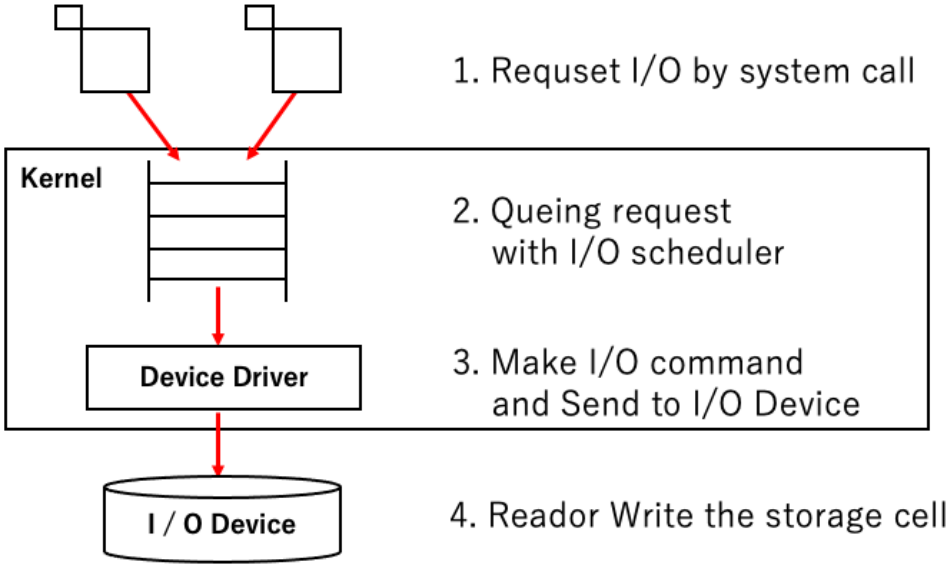

図 3.1　入出力処理の流れ

3.2　入出力要求の並列受付

入出力デバイスが入出力要求を受け取り可能となっても, OS が状態を検出し, 待ちキューから取り出され, デバイスドライバで入出力要求を変換するまでに時間を要する. このため, 入出力デバイスが直ちに入出力要求を受け取れるとは限らない. この結果, OS には入出力要求があるにも関わらず, 入出力デバイスは入出力要求の受け取り待ちとなり, 入出力スループットが低下してしまう. このような背景により, Intel 社は同時に 32 個の入出力要求を受け取り可能な入出力でナイスのインタフェース Advanced Host Controller Interface（AHCI）[5] を実現しており, 個人向け計算機に普及している. これにより, 複数の入出力要求を同時に変換し, 入出力デバイスに蓄積することで入出力要求を絶え間なく処理できる. したがって, 入出力デバイスの処理性能を引き出すには, OS ができる限り多くの入出力要求を発行することが重要となる.

3.3　入出力処理の留意点

ここで, あるプロセスの入出力システムコール（以降, 入出力時間）に要する時間が不安定になる要因として次の二点挙げられる. 第一に, 待ちキューに登録されたプロセスは, キュー内部の先行するプロセスの入出力要求を入出力デバイスに発行するまで待ち状態となる. つまり, 先行するプロセスによって待ち時間が生じる. 第二に, 入出力デバイスは, 自身の稼働率を向上させるために, 複数の入出力要求を受け付けて, 入出力デバイス内部の待ちキューに登録する. この動作により, 入出力デバイス 内部においても, 先行する入出力要求による記憶媒体の読み書きが完了するまでの待ち時間が発生し得る. このように, 複数プロセスが入出力要求を発行する環境では, あるプロセスの入出力時間は, 他プロセスの入出力要求の処理を待つことにより, 長くなり動作速度が不安定になり得る.

第 4 章

入出力性能調整法

4.1 基本方式

入出力時間を一定に保つ入出力性能調整法の基本方式を述べる．入出力性能の調整法の目的は利用者が指定する要求入出力性能の入出力デバイスが存在し, この入出力デバイスを調整対象プロセスが占有するかのようにみせることである．ここで, 要求入出力性能とは, 入出力デバイスの処理能力そのものを 100%とした百分率の値である．この目的のために, 他プロセスの動作に関わらず調整対象プロセスの入出力時間を一定に保つ．文献 [3] の入出力順序保証方式を以下に説明する．

本調整法は, 入出力デバイス内部での入出力要求の処理に要する時間（以降, 実 I/O 時間と呼ぶ）を要求　入出力性能で割った値を理想の入出力時間とする．調整対象プロセスの要求入出力性能 P のとき, 理想の入出力時間の算出式を以下に示す．

$$\text{理想の入出力時間} = \frac{100}{P} \times \text{実 } I/O \text{ 時間}$$

入出力順序保証方式の処理の流れを図 4.1 に示し, 以下に説明する．

(1) プロセスがシステムコールを発行する．

(2) 調整対象プロセスがいつ入出力要求を発行しても, 理想の入出力時間内に入出力デバイスが処理できるよう, 入出力デバイスに発行する入出力要求数を許容値以下に制限する．

(3) プロセスを待ちキューに登録して待ち状態にする．

(4) デバイスドライバを介して入出力要求を発行する．

(5) 待ち処理に同期を発送する．

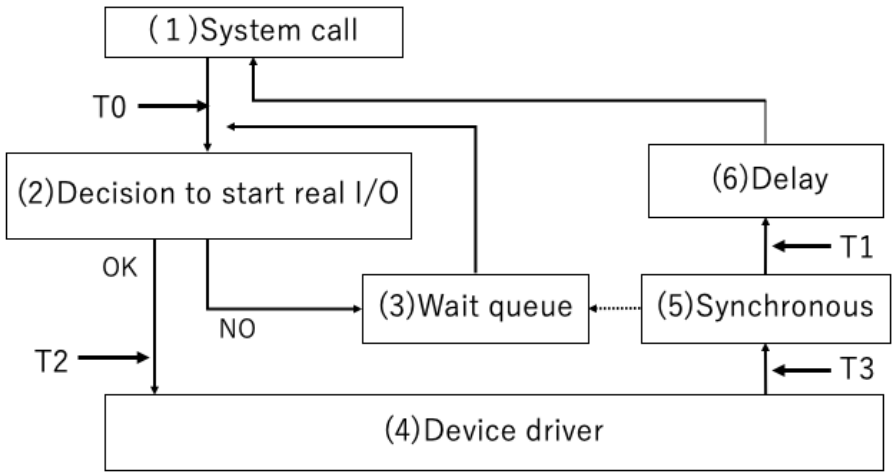

図 4.1　入出力性能調整の基本的な流れ

(6) 理想の入出力時間になるまで, 調整対象プロセスの起床を遅延する.

4.2　許容値の算出

調整対象プロセスの実 I/O 処理が理想の入出力時間内に終了するよう, 入出力デバイス内の入出力要求数を許容値と呼ぶ値以下に制限する.

まず, 要求入出力性能 P の調整対象プロセスが 1 個だけ存在するときの許容値 A は以下である.

$$A = max(1, \frac{100}{p} - 1)$$

次に, 調整対象プロセスが複数存在する場合について述べる. 同じ要求入出力性能を持つ調整プロセスが存在する場合, どちらが先に入出力要求を発行できるかはプロセスが起床する順に依存している. そのため, 上記の式では, 片一方の調整対象プロセスの実行が遅れてしまうため, うまく入出力調整ができない. そこで, 片一方の調整対象プロセスが先に入出力要求を発行することを前提に許容値の計算を行うことでどちらが先に入出力要求を発行しても双方の入出力調整がうまくいくようにしている. 調整対象プロセス i の要求入出力性能 P_i

とするとき, 許容値 A の算出式を以下に示す.

$$A_i = \frac{100}{p_i} - \sum_i f(i,j) - 1$$

$$f(x,y) = \begin{cases} 1 & P_y < P_x \\ 0 & \text{上記以外} \end{cases}$$

$$A = max(\min_i(A_i, 1))$$

4.3　遅延方式

入出力時間が理想の入出力時間になるように, 調整対象プロセスの入出力システムコールの完了時刻を遅延させる. 遅延時間 T_s の算出式を以下に示す.

$$T_s = \text{理想の調整時間} - (T_1 - T_0)$$

ここで, 実 I/O 時間は入出力デバイスごとに異なるため, 事前に決定できない. そこで, 次の方法で計測する. 入出力デバイス内部の入出力要求が 1 つの場合, 実 I/O 時間は, 入出力要求の発行から処理完了通知の受信までの時間（図 4.2 の T_2 と T_3 の差分）である. 一方, 入出力デバイス内部の入出力要求が複数の場合, 実 I/O 時間は, 入出力要求の処理完了通知の間隔（T_3 の間隔）である. したがって, 以下の式により実 I/O 時間を決定する.

$$\text{実 } I/O \text{ 時間} = \min(T_3 - T_2, T_3\text{の間隔})$$

ここで, 入出力デバイスの内部動作によって, 個々の実 I/O 時間には, バラつきが生じ得る. このバラつきの影響を抑えるため, 式 (5.7) で算出する実 I/O 時間を過去 10 回分蓄積し, 平均値を実 I/O 時間に用いる.

遅延には, OS が提供するスリープ機能を用いる. スリープ機能では, タイマ割込みで経過時間を計測するため, 指定した遅延時間（T_s）と実際にプロセスを停止した時間に差が生じ, 理想の入出力時間から乖離し得る. そこで, 遅延時間（T_s）と実際の停止時間の差分を次回の遅延処理へ繰り越す. これにより, 個々の入出力時間のバラつきを平準化する. スリープによる遅延を図 4.2 に示す.

4.3　遅延方式

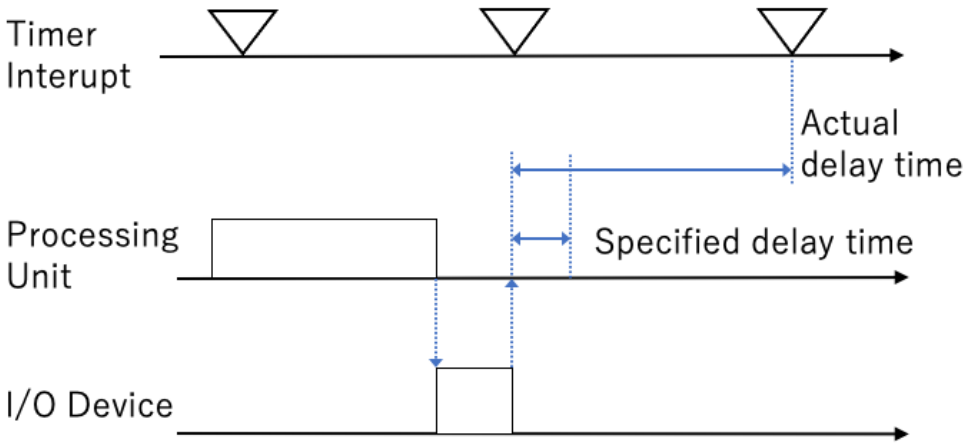

図 4.2　プロセスの起床の遅延

第 5 章

提案手法

5.1 HDD の違いによる調整精度低下の要因

入出力順序保証方式を用いて, 許容値を算出することで, 調整対象プロセスが複数存在する場合においても, 全ての調整対象プロセスにおいて高い調整精度を保持し, かつ入出力デバイスのスループットが向上した [2]. しかし, プラットフォームに搭載されているハードディスクの動作環境が異なることで, 入出力性能調整の精度が低下する場合がある.

一例として, 入出力デバイスファームウェアの機能で入出力デバイス内部キューに蓄積されている入出力要求が並び替えられる. そのため, 調整対象プロセスの入出力要求が後回しにされることで, 理想の入出力時間までに入出力処理が完了しない場合がある. その様子を図 5.1 に示す. また, 同一ハードウェア環境においても, OS パーティション方式が GPT と MBR とでは, 調整精度に 10%〜15%程度の差が見られた.

ここで, SATA（Serial ATA）インタフェースにおいては, NCQ（Native Command Queuing）が規格されている. NCQ では入出力デバイス内部に入出力要求を蓄積するキューが実装されている. このキュー内の入出力要求は, ドライブの回転方向に昇順になることやドライブヘッダの移動量を減らすことを考慮し, 並び替えを行うことでスループットを向上させる [6]. これにより入出力デバイスのパフォーマンスが向上する. 一方で各入出力処理に要する時間が不安定になる. 加えて, SCSI（Small Computer System Interface）やエンタープライズ向けサーバのディスクとしてよく用いられる SAS（Serial Attached SCSI）では, TCQ（Tagged Command Queuing）が規格され, NCQ 同様に入出力要求の並び替えが発生する.

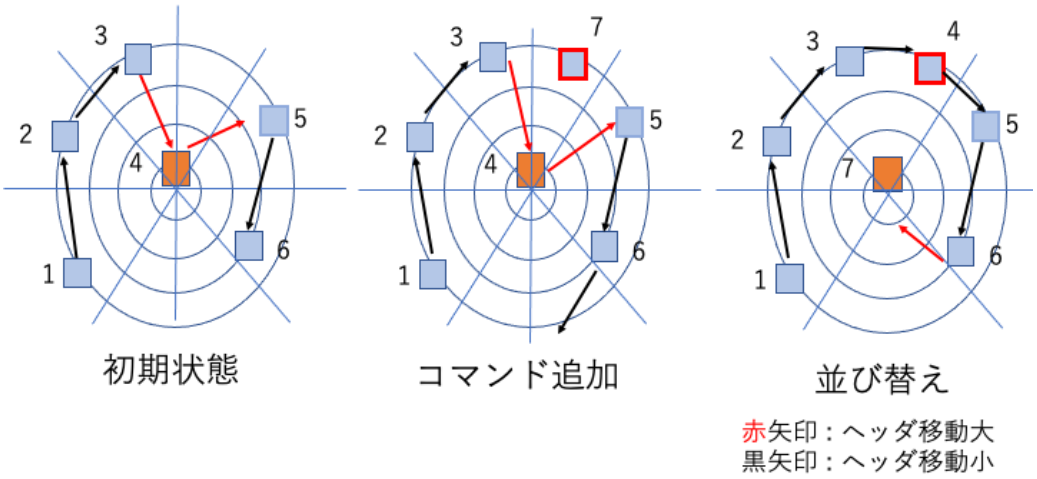

図 5.1　入出力デバイス内部キューのソート

このようにハードディスクの動作環境が異なり, 入出力処理の挙動に差がある中で, 調整対象プロセスの要求入出力性能とそのプロセス数によって固定的に許容値が算出される従来手法では, これに対応できない.

5.2　実装要件

入出力性能の調整法に対する要件を以下にまとめる.

1. 外部記憶装置ごとに調整可能

   本研究はハードディスクの動作環境の違いに注目しているが, 過去研究では全ての外部記憶装置を対象にしている. そして, プロセスが必要とする入出力性能は外部記憶装置ごと, 本研究では, 複数台搭載されているハードディスクごとに調整できる必要がある.

2. デバイスドライバの変更無し

   個々のデバイスドライバに調整機構を組み込むと制御のための実装や保守の工数が増大してしまう. そのため, 個々のデバイスドライバは変更せず, 全てのデバイスドライバに共通な処理機構に組み込む必要がある. 具体的には FreeBSD 上の抽象レイヤーである

CAM 層に実装する. そのため, 文献 [7] のような各デバイスドライバで入出力要求コマンドの NCQ 優先度を設定変更する方法は採用しない.

5.3 提案手法

ハードディスクの動作環境ごとに適切な許容値を算出するにあたり, 入出力処理動作に影響を与えている要素を網羅し, 固定的に許容値を定めることは, 要素が膨大になること, ハードディスクや基盤ソフトウェアが新たに登場するたびに許容値算出に必要なパラメータを追加 , 変更する必要があるため, 現実的ではない. そこで, 入出力性能調整動作を監視し, 現在利用している許容値では入出力性能調整の精度が悪くなる判断した場合に許容値を補正する許容値補正法を提案する.

許容値補正法では, 新たに調整対象プロセスが起動, または終了するたびに従来手法を用いて許容値 A を算出する. これは全てのハードディスクで共通の理想的な許容値である. 入出力性能調整を利用している間, 各調整対象プロセスの理想の入出力時間に対して, 各入出力処理に要した実際の入出力時間が超過した時間 T_{over_i} を T_{ama} に蓄積する. 蓄積された T_{ama} が決められた閾値に達した時, 蓄積対象の区間における平均調整精度 Reg を計測し, その調整精度 Reg が一定水準を満たさない場合, 許容値 A を１つ下げる. 調整精度 Reg に問題がない場合, そのままの許容値 A を利用し, 調整を続ける. そして, 補正の有無に限らず, T_{ama} はリセットする. この許容値補正処理を繰り返すことで, そのハードディスク環境で入出力性能調整が精度良く動作する許容値に補正する. ここで, 蓄積時間の閾値を T_b, 調整精度の水準を R_b とする. T_{ama} が T_b を超過するまでに調整対象プロセスの入出力処理が n 回発生した時, 許容値 A の補正の式を示す.

$$A = A - f(T_{ama}, Reg)$$

$$f(T_{ama}, Reg) = \begin{cases} 1 & T_{ama}{>}T_b, Reg{>}R_b \\ 0 & \text{上記以外} \end{cases}$$

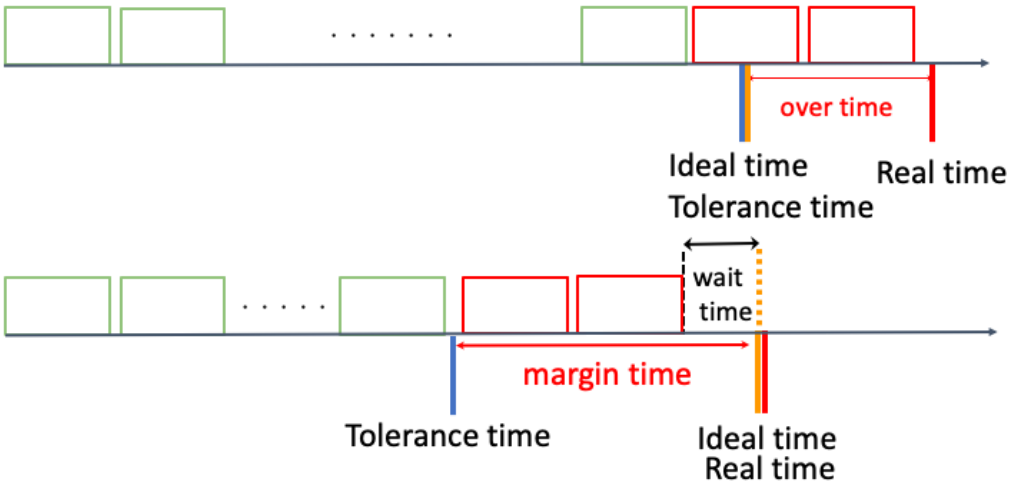

図 5.2　補正によるマージン

$$T_{ama} = \sum_{i}^{n} T_{over\_i}$$

$$Reg = \frac{\sum_{i}^{n} \frac{i\text{ 番目の実際の入出力時間}}{\text{理想の入出力時間}}}{n}$$

5.4　期待される効果

許容値を 1 つ下げるごとに理想の入出力時間と利用する許容値から推定される理論上の入出力時間（以降, 許容値時間と呼ぶ）の間に実 I/O 時間 1 つ分のマージンが発生する．そのため, 許容値時間に対して実際の入出力時間が超過しても, マージンが存在することで理想の入出力時間までに相殺できる．また, ハードディスクの動作環境ごとに超過度合いが異なる場合にも, 許容値の下げ幅, つまりマージンの幅を変えることで対応する．マージンを確保して入出力性能調整を改善する様子を図 5.2, 許容値時間を算出する式を次に示す．

$$\text{許容値時間} = (\text{許容値} + 1) \times \text{実 } I/O \text{ 時間}$$

そして, 補正された最適な許容値を維持することで高い調整精度と入出力性能調整利用時の入出力スループットを維持する．以上より, ハードディスクの動作環境に適合した許容値

に補正することで調整対象プロセスの入出力処理をユーザが求める要求入出力性能で実現できる.

一方で理想的な許容値に比べて, 利用する許容値が小さくなるほど, 入出力デバイスに同時に発行可能な入出力要求が減らされるため, 従来手法より入出力スループットが低下することになる.

第 6 章

評価

6.1 入出力調整法の実装

本調整法を FreeBSD Ver11.2 に実装した．本調整法は, 利用者が指定した入出力性能を有する入出力デバイスがあたかも存在するように見せる機能である．このため, 入出力デバイスを扱うデバイスドライバに本提案法のためのプログラムを追加した．具体的には FreeBSD の CAM 層に図 2 の基本方式と許容値補正法のプログラムを追加した．また, 利用者が要求入出力性能を設定するために, 以下のシステムコールを追加した.

$$int\, set\_iorate(int\, pid,\, char * dev_path,\, int\, req_iorate)$$

pid は, プロセス ID であり, 例えば ps コマンドを用いることで取得できる． 次に, dev_path は, デバイスファイルパスであり, /dev 以下のデバイスファイルの絶対パスを指定する．最後に, req_iorate は, 要求入出力性能であり, 1 ～ 100 の値を設定する．0 を指定することで要求入出力性能を解除できる.

6.2 最適な許容値の調査

まず, 許容値補正法で補正される許容値はプラットフォームで用いるハードディスクの動作環境に対して最適な許容値であることが望ましい．つまり調整対象プロセスの入出力処理が一定に動作し, その上で入出力スループットが最大になる必要がある．なお, スループットの大きさは利用する許容値の大きさに比例する．事前にハードディスクの動作環境ごとの最適な許容値を示すために, 許容値補正法の実験環境ごとに次の調査実験を行った.

表 6.1　実験環境

Environment	PC1	PC2	PC3
Processor	3.4GHz 2core/4thread		3.2GHz 4core/4thread
Memory	DDR3 8GB		
Partition	GPT	MBR	
Interface	SATA3.0 6Gb/s		
I/O device	WDC WD1200BE VS-22UST0 120GB	WDC WD10TPVX 16JC3T3 1TB	Hitachi HDP72505 GLA360 500GB

表 6.2　測定パタン

Request I/O performance	10, 15, 20, 25, 30(%)
Allowance	2 ∼9

まず, 評価指標について説明する．入出力時間を精度良く調整できたか否かは, 各入出力要求における実際の入出力時間と理想の入出力時間がどの程度近いか, 調整精度で評価する．ここでは, 関連研究 [2] に従い, 良い調整精度は 1.1 以下とする

$$\text{調整精度} = \frac{\text{実際の入出力時間}}{\text{理想の入出力時間}}$$

次に, 調査環境について説明する．表 6.1 に示す 3 種の計算機について調査した．評価プログラムとして, 2GB のテキストファイルに対してランダム読み込みを 5000 回繰り返す処理を用いた．このプログラムを 10 つのプロセスで走行させた．ファイルアクセスの競合を回避するために, プロセスごとにテキストファイルを用意した．10 個のプロセスの内 1 プロセスを調整対象プロセスとした．調整対象プロセスの要求入出力性能と利用する許容値を表 6.2 の通りに変化させた．全ての組み合わせに対して 50 回実験を行った．

調査実験の結果を図 6.1 に示す．各調査環境の最適な許容値と従来手法によって算出される許容値とを比較したとき, PC1 では各要求入出力性能における最適な許容値は同一の値を

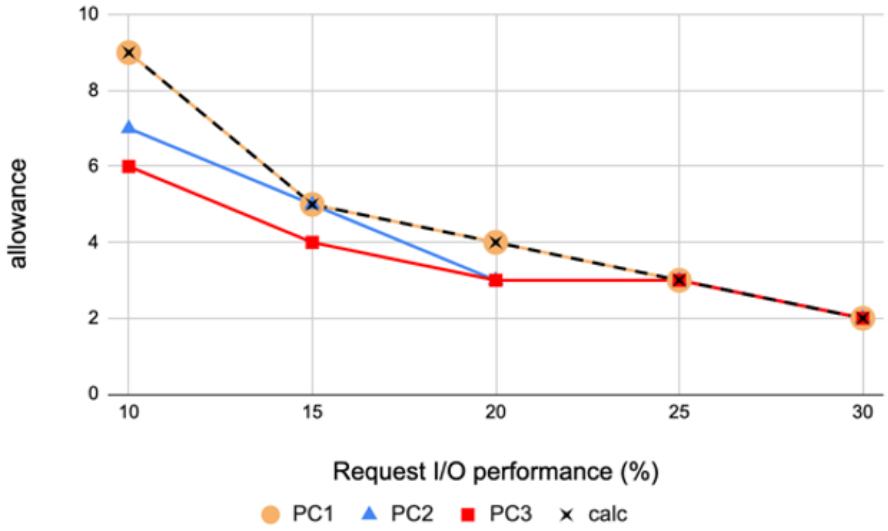

図 6.1　最適な許容値

示している．PC2 では, 要求入出力性能 10%で 2, 20%で 1 ずれが生じる．PC 3 では要求入出力性能 10%で 3, 15%で 1, 20%で 1 ずれが生じる. この結果からも, 従来手法がハードディスクの動作環境にマッチした許容値算出ができていないことが示される. この結果を許容値補正の正解データとして, 許容値補正法が最適な許容値に補正できているか評価する.

6.3　提案手法の実験条件

本調整法が入出力時間をうまく調整できるか否かは 6.2 同様に調整精度で評価する．ここでは, 従来研究に従い, 良い調整精度を 1.1 以下としている．各ハードディスクの動作環境において, 最適な許容値に補正できているか否かは正解データと許容値補正の推移を比較して評価する．入出力スループットは, 単位時間あたりに OS が入出力デバイスから受け取った入出力要求の完了通知（I/O per seconds, IOPS) で評価する.

次に評価環境について説明する．6.2 同様に表 6.1 に示す 3 種の計算機に, 本調整法を実装した FreeBSD Ver11.2 を動作させた．評価プログラムとして, 6.2 同様に 2GB のテキストファイルに対してランダム読み込みを 5000 回繰り返す処理を用いた．このプログラムを 10 つのプロセスで走行させた．ファイルアクセスの競合を回避するために, プロセスごとに

表 6.3　測定パタン

Request I/O performance	10, 15, 20, 25, 30(%)
Accumulation time border	5 second
Regulation ratio border	1.1

テキストファイルを用意した．10 個のプロセスの内 1 プロセスを調整対象プロセスとした．調整対象プロセスの要求入出力性能を 10%, 15%, 20%, 25%, 30%と変化させる．また, 蓄積時間の閾値を 5 秒, 調整精度の水準を 1.1 以下とした．これらを表 6.3 にまとめる．本実験を従来手法と本調整法とで実験を行う．最後に, 測定データの加工方法を述べる．ハードディスクドライブは入出力要求を受け取っていない時間が続くと, 省電力化のためにディスクの回転速度を低下または回転を停止する．このため, プロセスの起動直後は実 I/O 時間が安定しない場合がある．またプロセッサのスケジューリングによって, プロセスの終了時刻にばらつきがある．一部のプロセスが終了すると, 入出力要求の発行数が少なくなってしまう．そこで, これら影響を省くために, プロセスの起動後 30 秒とプロセスの終了 30 秒を省いた期間の平均値を結果に用いた．

6.4　調整精度

各実験環境において, 要求入出力性能を 10%, 15%, 20%, 25%, 30%と変化させた調整精度の結果を図 6.2 から図 6.4 に示す．

(1) 全ての実験環境において, 要求入出力性能の値が何%であっても, 提案手法の調整精度は 1.1 以下となり, 良い調整精度を示している．

(2) PC 1 については, 全ての要求入出力性能において従来手法で算出される許容値と正解データで示される許容値が同一であるため, 従来手法と提案手法間に調整精度の差がない．

(3) PC2 については, 要求入出力性能 10%, 20%において従来手法で算出される許容値と正

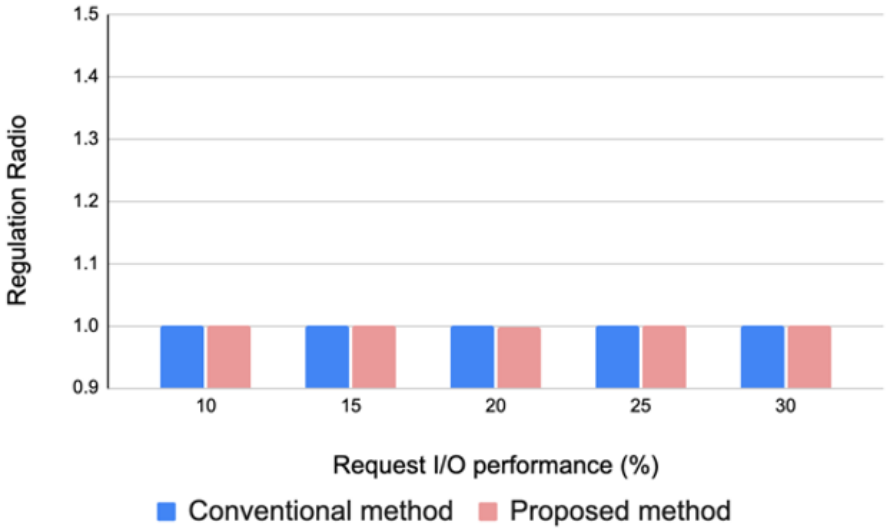

図 6.2　PC1 調整精度

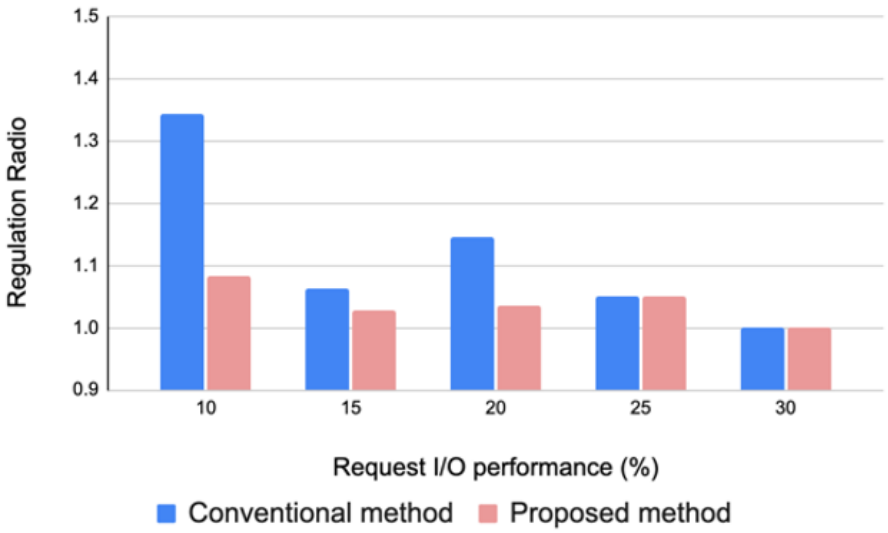

図 6.3　PC2 調整精度

解データで示される許容値に差がある．そのため, 従来手法の調整精度が悪い．提案手法については許容値を補正できた結果, 要求入出力性能 10%,20%においても良い調整精度を示している．

(4) PC3 については要求入出力性能 10%, 15%, 20%において従来手法で算出される許容値と正解データで示される許容値に差がある．そのため従来手法の調整精度が悪い．提案

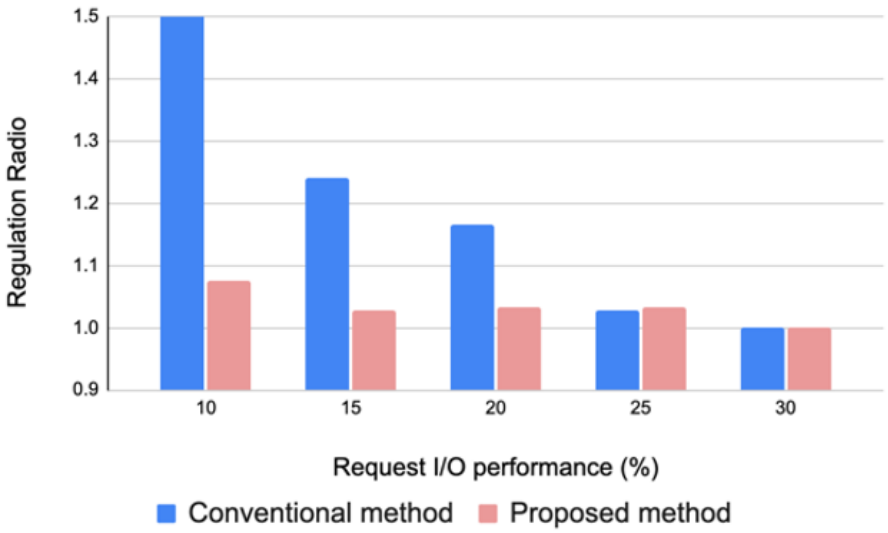

図 6.4　PC3 調整精度

手法については許容値を補正できた結果, 要求入出力性能 10%,15%,20%においても良い調整精度を示している.

以上より, 提案手法を用いることで 3 種全てのハードディスクにおいて, 調整精度が改善することがわかる.

6.5　許容値の推移

各実験環境において, 要求入出力性能を 10%, 15%, 20%, 25%, 30%と変化させた許容値補正の推移を図 6.5 から図 6.7 に示す.

(1) PC1 については, 全ての要求入出力性能において従来手法で算出される許容値と正解データで示される許容値が同一である. 許容値の推移としても提案手法による許容値の補正は発生せず, 一定の許容値を維持できている.

(2) PC2 については, 要求入出力性能 10%, 20%, 25%, 30%において従来手法で算出された許容値から補正されて, 調整プロセスの入出力処理が約 800 回以内に正解データの許容値に補正され, それを維持する結果となった. しかし, 要求入出力性能 15%については

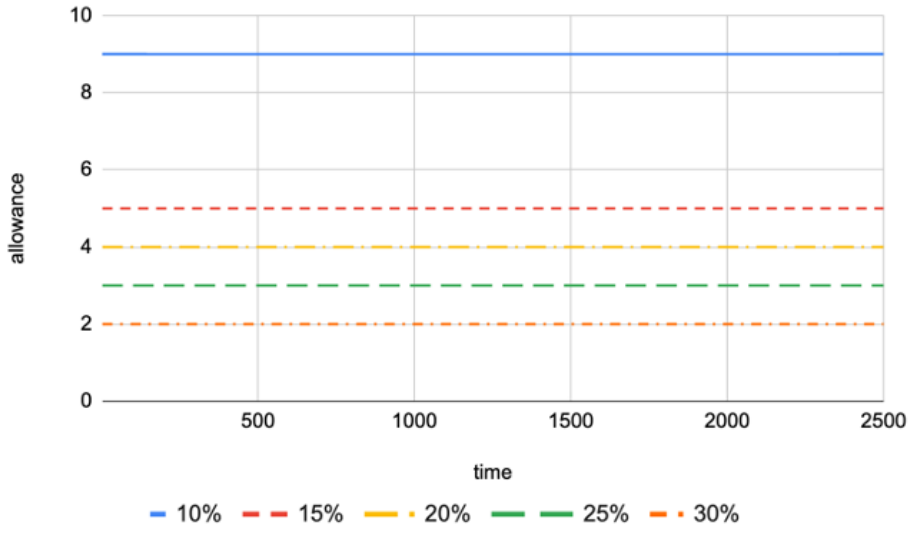

図 6.5　PC1 許容値の推移

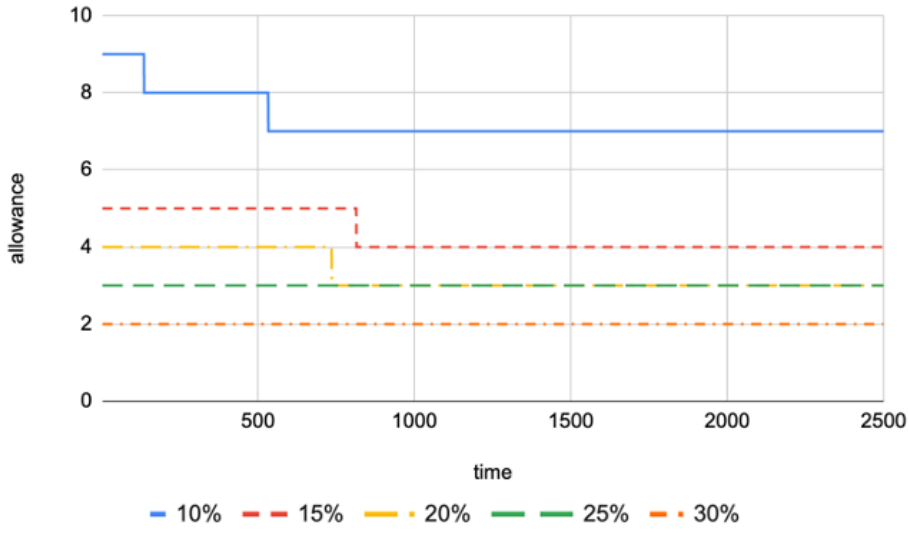

図 6.6　PC2 許容値の推移

正解データの許容値は 5 である一方で補正される許容値は 4 まで下がっている.

(3) PC3 については, 全ての要求入出力性能において従来手法で算出された許容値から補正されて, 調整プロセスの入出力処理が約 750 回以内に正解データの許容値に補正され, それを維持する結果となった.

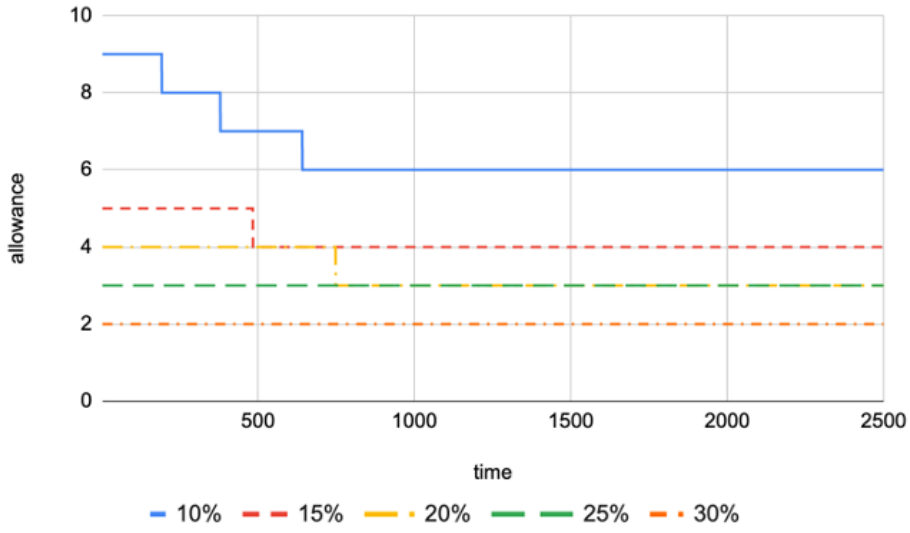

図 6.7　PC3 許容値の推移

以上より, 提案手法を用いることでハードディスクの動作環境に最適な許容値を算出し, 入出力性能調整が利用できる可能性が高い. そのため, 入出力性能調整利用時における入出力スループットを維持できている. 一方で PC2 おける要求入出力性能 15%の時のように最適な許容値より低い許容値に補正される可能性もある. そのため, 蓄積時間の閾値や調整精度の水準の最適化などを今後検討する必要がある.

6.6　スループット

PC2 における, 要求入出力性能を 10%, 15%, 20%, 25%, 30%と変化させたスループットを図 6.8 に示す. 15%, 25%, 30%については従来手法と提案手法で許容値が同じため, スループットに差はない. 許容値に差が出る 10%, 20%については, 提案手法を用いた場合, 9IPOS, 11IPOS 程度低い結果となった. 両条件とも従来手法で算出される許容値より 2 低い値に補正されていることから, 補正により許容値が 1 下がるごとにスループットが 5IPOS 程度低くなると予想される. また 20%については最適な許容値より 1 低い値に補正しているため, 目標より 5IPOS 低い値といえる.

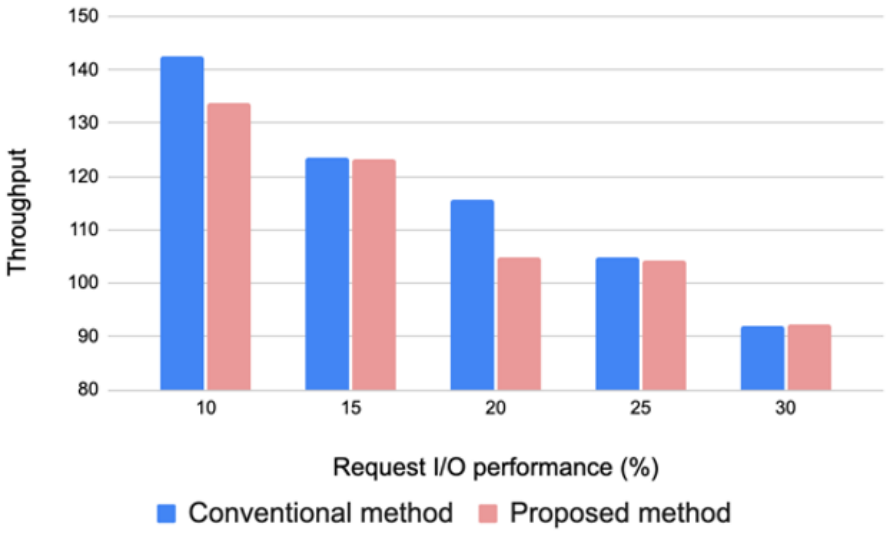

図 6.8　PC2 スループット

以上より, 許容値を補正し, 調整精度の精度を向上させる提案手法を用いてもスループットの低下は小さいものである. そのため, 全ての条件で調整精度が高い提案手法を用いるほうがユーザにとってはメリットが大きいといえる.

第 7 章

考察

7.1 入出力スループット向上に向けた許容値解放の検討

ここでは, 入出力スループット向上について考察をする. 入出力性能調整法では, 許容値を用いて入出力要求数を制限することで調整対象プロセスの入出力性能を一定に調整している. しかし, 許容値が制限されることでデバイス本来の処理性能を下回ることになるため, システム全体の入出力スループットが低下する.

ここで, 許容値の制限が必要となるのは調整対象プロセスの入出力処理実行時であり, 非調整プロセスの入出力処理実行時には許容値の制限を必要としていない. そのため, 調整対象プロセスで入出力処理が発生しない区間においては許容値を解放して, デバイス本来の処理性能で動作することで非調整プロセスに割り当て可能なリソースが最大となり, 入出力スループット向上すると考える. 本節では, 次のアクセスパターン 4 種を対象にプロセッサ処理中は許容値を解放することによる調整精度への影響と入出力スループットが向上するか調査, 考察する.

(1) 低頻度単一アクセス（LS）
(2) 低頻度連続アクセス（LC）
(3) 高頻度単一アクセス（HS）
(4) 高頻度連続アクセス（HC）

表 7.1　パラメータ

要求性能	15%
許容値	制限：5, 解放：32
解放時間	5 秒
I/O アクセス	単一アクセス：10 回, 連続アクセス：100 回
CPU 処理	低頻度アクセス：10 秒, 高頻度アクセス：1 秒

7.2　調査実験

入出力スループットを向上させるためには, 許容値による制限と解放を切り替える必要がある．しかし, OS はプロセスの入出力処理がどのタイミング発生するか分からないため, 事前に入出力要求の発生を予測し, 切り替えることはできない．そこで本考察では, 調整対象プロセスの入出力処理が一定時間発生しなかった場合に許容値を解放し, 入出力処理が再開したタイミングで許容値の制限に戻すことでスループット向上させる手法を前提に調査を行う．

評価環境として, 前述した PC2 を用いた．評価プログラムとして, アクセスパターンごとに実験パラメータに示した CPU 処理と I/O 処理を交互に繰り返すものを用いた．また, 許容値の解放が発生するアクセスパターン LS と LC については許容値の解放がない場合も調査し, 比較する．本実験の I/O 処理については 6.2 同様に 2GB のテキストファイルに対してランダム読み込みを合計 5000 回行う処理を用いた．このプログラムを 10 つのプロセスで走行させた．ファイルアクセスの競合を回避するために, プロセスごとにテキストファイルを用意した．10 個のプロセスの内 1 プロセスを調整対象プロセスとした．調整対象プロセスの要求入出力性能を 15%とした．許容値については制限時は, PC2 の最適な許容値である 5 を用い, 解放時は, デバイス本来の値である 32 を用いる．調整対象が入出力処理を行っていない閾値を 5 秒とする．これらを表 7.1 にまとめる．

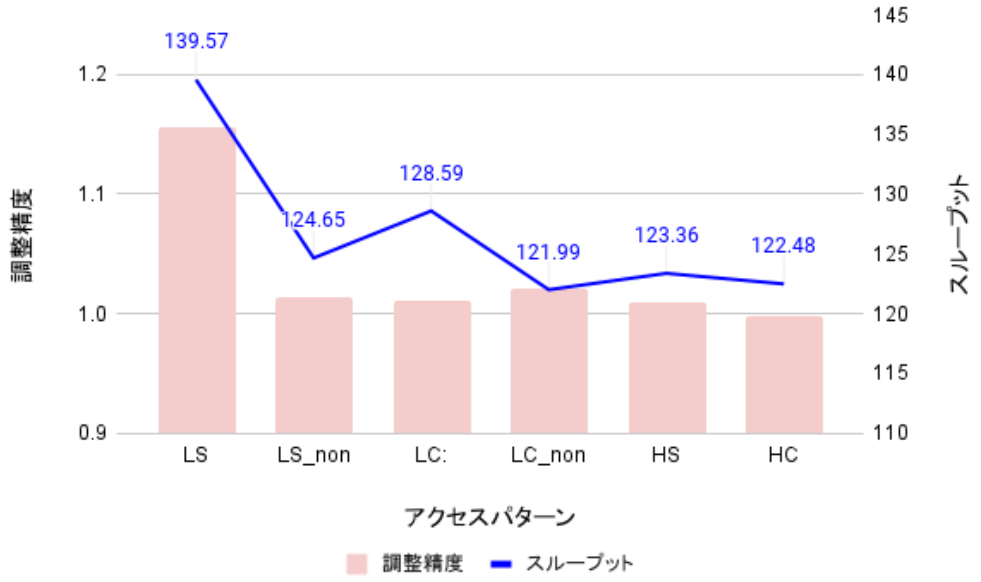

図 7.1　調整精度とスループットの変化

7.3　結果と考察

調査結果を図 7.1 に示す．高頻度アクセスである HS と HC については, 許容値の解放がないため従来と同様の結果である．低頻度アクセスである LS と LC については, LS は許容値を解放することでスループットは 15IPOS ほど上昇した一方で, 調整精度が 1.15 程度まで劣化している．LC は許容値を解放することでスループットが 7IPOS ほど上昇し, 調整精度はほぼ変わらない結果となった．以上よりアクセスパターン LC においては, 高い調整精度とスループット向上の両立が成り立つため, 調整対象プロセスが入出力処理を行わない場合に許容値を解放する手法を提案することは有用といえる．

許容値を切り替えることで調整精度が大きく劣化することはできるだけ避けなければならない．劣化する具体的な要因として, 調整対象プロセスの入出力要求発生時にデバイスへの入出力要求発行数が許容値以下の状態である必要があるが, 入出力要求発生時に許容値による制限に切り替えても入出力性能調整にするまでに間に合わない．

連続アクセスである LC と HC については, まとまったアクセスの先頭で許容値の制限をかけなおすことで先頭のアクセス以外をうまく調整することで全体の調整精度を担保するこ

とは可能である．そして, LC については, まとまったアクセスが終了し, 一定時間入出力要求が発生しないため, 許容値を解放することで入出力スループットの向上も見込める．

一方で, 単一アクセスについては許容値を固定することが望ましいと考える．LS は低頻度アクセスであるため, 調整がうまくいかない場合でもユーザに与えるストレスは小さい．そのため, 許容値は常に解放しておくこと, つまり調整機能を利用せず, 入出力スループットの向上させるほうがユーザのメリットが大きい．HS は高頻度アクセスであるため, 入出力処理の割合が大きくなり調整できない場合ユーザに与えるストレスは大きい．そのため, 許容値を制限することで調整精度を向上させるほうが望ましい．

第 8 章

おわりに

本研究では, 入出力性能調整動作を監視し, 利用している許容値で入出力性能調整の精度が悪い場合に許容値を補正する許容値補正法を提案した. これまでの, 入出力順序保証方式では, 要求入出力性能が同じ他の調整対象プロセスが先に入出力要求を発行する場合において, 理想の入出力時間で入出力処理を完了することを保証するとともに, スループット向上を実現している. しかし, ハードディスクの動作環境が異なり, 入出力処理の挙動に差がある中, 調整対象プロセスの要求入出力性能とそのプロセス数によって一意に許容値が算出される従来手法では, これに対応できない.

提案手法では, 理想の入出力時間に対して, 各入出力処理に要した実際の入出力時間が超過した時間を蓄積し, 決められた閾値を超えた時, 許容値を利用していた区間の調整精度を計測し, 調整精度が水準に満たない場合, 補正を行う. 評価により, 提案手法によって許容値を補正することで, 評価に利用した 3 種のハードディスクにおいて精度良く調整できることを示した. またハードディスクの動作環境に最適な許容値に補正し, 維持できる可能生が高いため, 入出力調整利用時におけるスループットの維持にも適している.

残された課題として, 最適な許容値よりも低い値に補正される可能性もあるため, 蓄積時間の閾値や調整精度の水準の最適化などを進める必要がある. また考察で検討した調整対象プロセスの入出力要求が発生していないときに許容値を解放し, システム全体の入出力スループットの向上させる調査をより進めるとともに, 具体的な手法を考える必要がある.

謝辞

本研究を進めるにあたり, 懇切丁寧なご指導をしていただきました指導教員の横山和俊教授に深く感謝申し上げます. 誠にありがとうございました. またお忙しい中, 本研究の調査をお引き受けいただきました, 敷田幹文教授, 植田和憲講師に深く感謝申し上げます. さらに活発な議論をして頂きました岡山大学 谷口秀夫教授, 日立製作所 長尾尚様に深く感謝申し上げます. 特に, 長尾尚様には調査, 実装に際して, 多数のご助力を頂きましたことを改めて感謝申し上げます. そして, 共に研究を行った北雄大さんを始めとする研究室の皆様に心より感謝申し上げます.

参考文献

[1] 谷口秀夫, ”入出力時間の制御によりサービス時間を調整する制御法”, 信学論 (D), Vol.J83-D-I, No.5, pp.469-477, May 2000.

[2] 長尾尚, 山内利宏, 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評価”, 情報処理学会, Vol.2010-OS-114 No.10, 2010.

[3] 長尾尚, 田辺雅則, 横山和俊, 谷口秀夫 ”各プロセスの入出力性能の調整による入出力スループットの低下を抑制する制御法の実現と評価”, 信学論 (D), Vol.J103-D, No.3,pp.159-170, May 2020.

[4] 一井晴那, 長尾尚, 山内利宏, 谷口秀夫, ”Tender オペレーティングシステムにおける資源「入出力」の実現と評価”, 情報処理学会, Vol.2011-OS-118 No.19, 2011.

[5] AHCI Specification, https://www.intel.com/content/www/us/en/io/serial-ata/ahci.html.

[6] David A.Deming, ”The Essential Guide to Serial ATA and SATA Express”, CRC Press, 2019

[7] Adam Manzanares, Filip Blagojevic, Cyril Guyot, ”A Tail of Latency, IOPs, & IO Priority”, 2017