Vol.2012-ARC-200 No.6
Vol.2012-OS-121 No.6
2012/5/7

Tender オペレーティングシステムにおける 入出力性能調整機能の評価

一井 晴那[1]　山内 利宏[1]　谷口 秀夫[1]

概要：プログラムは，PU 処理と I/O 処理を繰り返しながら処理を進める．このため，プロセスが利用できるプロセッサ性能の量や入出力性能の量を制御することにより，プログラムの実行速度を調整できる．我々は，***Tender*** に資源「入出力」による入出力性能調整機能を実現した．資源「入出力」を導入することにより，プロセスの入出力性能を保証および制限することができる．ここでは，***Tender*** に実現した入出力性能調整機能の評価について述べる．具体的には，占有状態における調整精度，他プロセスの影響，および許容値の更新有無がスループットに及ぼす影響について述べる．

Evaluation of Regulating I/O performance on *Tender* Operating System

Haruna Ichii[1]　Toshihiro Yamauchi[1]　Hideo Taniguchi[1]

Abstract: Executing a program repeats PU processing and I/O processing, hence we can regulate program execution time by regulating CPU performance or I/O performance which process can use. We proposed the mechanism for regulating I/O performance by ***I/O*** resources on the ***Tender*** operating system and implemented it. ***I/O*** resources enable a process to guarantee and regulate I/O performance of process. In this paper, we evaluate the mechanism for regulating I/O performance on ***Tender*** Operating System. The evaluation results show the precision without other process, the influence on precision by the other process, and the influence on throughput by tolerance.

1. はじめに

計算機の高性能化により，1 台の計算機上で多くのサービスを実行できるようになった．しかし，個々のサービスの実行速度は，ハードウェア性能や他のサービスの影響を受ける．例えば，ウィルス対策プログラムのように，高負荷な処理が実行されると，Web ブラウザやテキストエディタのプログラムの起動や応答性は著しく低下する [1]．そこで，プログラムの実行速度を調整できれば，資源競合時であっても，バックグラウンドで実行するプログラムの実行速度を制限することにより，フォアグラウンドで提供しているプログラムの実行速度の低下を抑制できる．

プログラムは，PU 処理と I/O 処理を繰り返しながら処理を進める．したがって，プロセスが利用できるプロセッサ性能の量や入出力性能の量を制御することにより，プログラムの実行速度を調整できる．我々は，プロセッサ性能の調整法 [2] を提案した．また，分散指向永続オペレーティングシステム ***Tender***(The ENduring operating system for Distributed EnviRonment)[3] において，資源「演算」を提案し，資源「演算」によるプロセッサ性能調整機能 [4] を実現した．さらに，プロセッサ性能の調整のみでは入出力処理の影響を受け，プログラムの実行速度の調整精度が低下することを踏まえ，入出力性能を調整する制御法として，入出力要求数を調整する制御法 [1] を提案し

[1] 岡山大学大学院自然科学研究科
Graduate School of Natural Science and Technology, Okayama University

た．また，***Tender*** において，資源「入出力」を提案し，資源「入出力」による入出力性能調整機能 [5] を実現した．資源「入出力」を導入することにより，プロセスの入出力性能を保証および制限するだけでなく，プロセスの入出力デバイスの使用可否を制御できる．また，入出力処理に優先度の概念を導入することにより，入出力処理の優先度制御を行うことができる．さらに，資源「入出力」は様々な入出力デバイスの入出力処理インタフェースを包括したインタフェースをプロセスに提供する．

本稿では，***Tender*** に実現した入出力性能調整機能の評価について述べる．具体的には，占有状態における調整精度，他プロセスが及ぼす影響について評価する．また，入出力ドライバへ同時要求する入出力要求数のしきい値 (以降，許容値と名付ける) の更新有無がスループットに及ぼす影響を評価する．

I/O スケジューリングについては，シーク時間が短くなるように入出力要求の順番を入れ替えることで磁気ディスクの入出力性能を向上させる研究 [6][7] がある．これらの研究は，ハードウェア性能を最大限に引き出す．また，プロセスの優先度に基づき入出力要求の順番を入れ替えることにより，優先度の高いプロセスの入出力処理を優先的に実行する研究 [8]，予約した資源量の多いプロセスの入出力処理を優先的に実行する資源予約型 I/O スケジューリング法の研究 [9][10] がある．これらの研究は，より重要度の高い入出力要求を優先的に実行することにより，多数のプロセスが入出力処理を行う場合において，重要なプロセスの入出力性能の低下を抑制する．一方，***Tender*** に実現した入出力性能調整機能は，ユーザが要求する入出力性能に合わせて入出力時間を調整する．

2. *Tender* オペレーティングシステム

Tender[3] では，OS の操作する対象を資源として，分離し独立化している．資源には，資源名と資源識別子を付与し，資源操作のインタフェースを統一している．さらに，各資源を操作するプログラム部品（資源管理処理部と呼ぶ）を資源ごとに分離し，共有プログラムを排除している．また，各震源の管理情報も資源ごとに分離し，各資源の管理表の間の参照関係を禁止している．

以上の資源の分離と独立化により，以下の利点がある．

(1) 資源の事前生成や保留による資源の作成や削除を伴う処理の高速化
(2) OS の動作および内部情報の理解や把握が容易になり，OS の理解を支援
(3) プログラムを部品化できるため，機能の追加や変更が容易

3. 「入出力」管理

3.1 資源「入出力」

資源「入出力」は，***Tender*** の資源であり，入出力処理に要する時間の程度（以降，入出力の程度と名付ける）と入出力デバイスの種類を持つ．資源「入出力」には，性能調整入出力と優先度入出力がある．性能調整入出力は入出力の程度として，要求入出力性能を持つ．要求入出力性能とは，他プロセスが走行せず，ハードウェアを占有して走行したときの入出力性能を 100 %として 1 %単位で指定する．指定可能な範囲は 1〜100 %である．性能調整入出力が持つ入出力の程度は，1 回の入出力処理に要する時間を調整するものであるため，資源「演算」と異なり，総和に制限はない．

優先度入出力が持つ入出力の程度は，入出力要求を実行する優先順位となる優先度を示す．優先度は固定値であり，動的に変更されることはない．

プロセスが入出力処理を行うには，資源「入出力」とプロセスを関連付ける必要がある．資源「入出力」とプロセスを関連付けることにより，プロセスは，自身に関連付けられた資源「入出力」を利用して入出力要求を発行する．この時，プロセスは，自身に関連付けられた資源「入出力」の入出力の程度に従って，入出力要求がスケジュールされ，入出力性能が調整される．したがって，資源「入出力」を関連付けられていないプロセスは，入出力要求をスケジュールすることができないため，入出力処理を実行できない．つまり，プロセスに対して使用を禁止したい入出力デバイスが存在する場合，その入出力デバイスを種類として持つ資源「入出力」にプロセスを関連付けないことにより，入出力処理を禁止できる．なお，資源「入出力」は 1 回の入出力処理に要する時間を調整するものであるため，資源「入出力」とプロセスの関連付けの関係は，入出力デバイスごとに 1：1 である．この様子を**図 1** に示す．例えば，プロセス A は入出力 A に関連付けられており，同じ入出力デバイスの種類を持つ入出力 B を関連付けることはできない．しかし，プロセス B のように，入出力デバイスの種類が異なる入出力 B と入出力 C を 1 つのプロセスに関連付けることはできる．

3.2 提供機能とインタフェース

資源「入出力」を管理する入出力管理が提供する機能を以下に示す．

（機能 1）資源「入出力」の生成
（機能 2）資源「入出力」の削除
（機能 3）資源「入出力」に対するプロセスの関連付け
（機能 4）資源「入出力」に対するプロセスの関連付けの解除

表 1 入出力管理のインタフェース

通番	形式	機能
(機能 1)	create_io(iodeg, dev_kind);	iodeg で指定した入出力を生成し，入出力識別子 ioid を返す．dev_kind 毎に管理する．iodeg は，1〜100 %の場合は要求入出力性能を意味し，0〜-255 の場合は優先度を意味する．優先度は値が大きいほど高い．
(機能 2)	delete_io(ioid);	入出力識別子 ioid を持つ資源「入出力」を削除する．
(機能 3)	attach_io(ioid, rid);	入出力識別子 ioid を持つ資源「入出力」に，プロセス識別子 rid を持つプロセスを関連付ける．
(機能 4)	detach_io(ioid, rid);	入出力識別子 ioid を持つ資源「入出力」とプロセス識別子 rid を持つプロセスの関連付けを解除する．
(機能 5)	ctrl_io(ioid, value, io_op);	(1)io_op が CHANGE_IODEG の場合，入出力識別子 ioid を持つ資源「入出力」の入出力の程度を指定された値 value に変更する． (2)io_op が CHANGE_K の場合，入出力識別子 ioid を持つ資源「入出力」の入出力デバイスの種類から許容係数を変更する入出力デバイスを決定し，当該入出力デバイスに対応する許容係数を指定された値 value に変更する．許容係数 (詳しくは，3.3.2 項で述べる) は許容値の算出時に使用する係数である．
(機能 6)	ctrl_io(ioid, pid, dst_node, disk_option, *buf, dst_addr, size, disk_rw, io_op);	入出力識別子 ioid を持つ資源「入出力」から使用する入出力デバイスを特定し，入出力デバイスへ入出力要求を発行する．引数として，入出力識別子 ioid，プロセス識別子 pid，ドライブ番号 dst_node，ディスク I/O 方式 disk_option，メモリアドレス*buf，読み込みまたは書き込み開始セクタ dst_addr，読み込みまたは書き込むセクタ数 size，読み込みまたは書き込みの指定 disk_rw，および操作種別に DISK_REAL_IO を指定する．

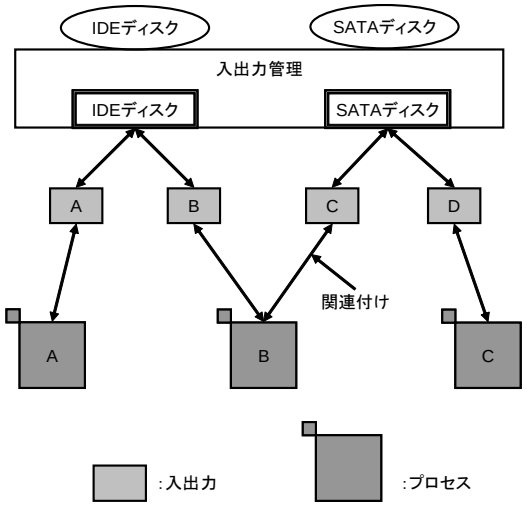


図 1 資源「入出力」とプロセスの関係

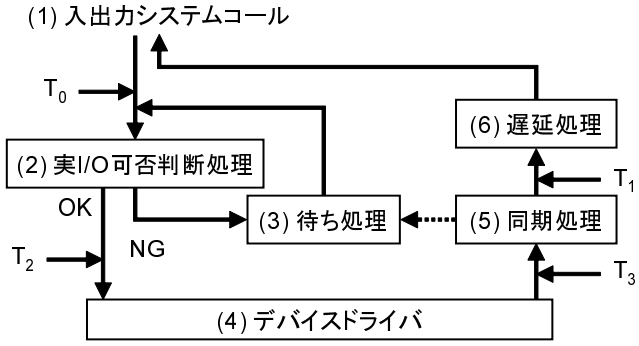


図 2 入出力性能調整機能の基本方式

（機能 5）入出力の程度または許容係数の変更の変更
（機能 6）入出力要求の受付

これらの機能のインタフェースを**表 1** に示し，具体的な資源「入出力」の利用手順を述べる．利用者は，最初に，(機能 1) により，利用したい入出力の程度と入出力デバイスの種類を指定して資源「入出力」を生成する．次に，(機能 3) により，生成した資源「入出力」に入出力処理を行うプロセスを関連付ける．(機能 6) により，プロセスは，自身に関連付けられた資源「入出力」へ入出力要求を発行し，入出力処理を行う．このとき，プロセスの入出力時間は，当該プロセスに関連付けられた資源「入出力」が持つ入出力の程度にあわせて調整される．入出力デバイスの使用を停止する場合，利用者は，(機能 4) により，資源「入出力」とプロセスの関連付けを解除する．(機能 2) により，使用しない資源「入出力」を削除する．

3.3 入出力性能調整機構

3.3.1 基本方式

入出力性能調整機能の基本方式を**図 2** に示し，各処理内容を説明する．

実 I/O 可否判断処理は，実 I/O 可否判断規則に基づき，デバイスドライバへの実 I/O 要求数 (実 I/O 処理を行うプロセス数) を制御し，他プロセスの影響を抑制して調整精度の向上を図る．詳細は 3.3.2 項で説明する．

待ち処理は，実 I/O 処理を許可されなかったプロセスを管理し，入出力優先度に基づき，待ちキューにつないで待ち状態にする．なお，性能調整入出力，優先度入出力の順

に高入出力優先度とし，性能調整入出力間では要求入出力性能が高いもの，優先度入出力間では優先度が高いものを高入出力優先度とする．また，同入出力優先度は LRU で管理する．同期処理からの同期により，入出力優先度の最も高いプロセスを起床させる．

同期処理は，待ち処理に同期を送信し，入出力優先度の最も高いプロセスを起床させる．

遅延処理は，要求入出力性能に基づき，遅延時間分だけプロセスを待ち状態にする．遅延時間は，要求入出力性能から算出される理想の調整後の入出力時間 (以降，理想の入出力時間と名付ける) から入出力処理に要した時間 $(T_1 - T_0)$ を減算した値である．遅延時間の算出式を以下に示す．遅延時間が遅延可否閾値以下の場合，遅延処理を行うことができない．これによる調整精度の低下を防ぐため，遅延時間が遅延可否閾値以下の場合，次の遅延処理に繰り越す．

$$\text{遅延時間} = \frac{100}{\text{要求入出力性能}} \times \text{実 I/O 時間} - (T_1 - T_0) \quad (1)$$

ここで，実 I/O 時間とは，実 I/O 処理に要する時間であり，その算出式を以下に示す．

$$\text{実 I/O 時間} = MIN((T_3 - T_2), T_3\text{の間隔}) \quad (2)$$

3.3.2 被調整プロセス数を考慮した制御

デバイスドライバは到着順に実 I/O 要求を処理するため，要求入出力性能の高い被調整プロセスほど，理想の入出力時間内に実 I/O 処理が終了するように，実 I/O 処理開始時期を制御する必要がある．例えば，デバイスドライバに既に複数の実 I/O 要求がある場合，要求入出力性能の高いプロセスの実 I/O 要求をデバイスドライバに発行しても，その要求は優先して処理されないため，実 I/O 処理が理想の入出力時間内に完了しない．したがって，デバイスドライバ内の実 I/O 要求数を制限しなければ，調整精度が低下する．このため，実 I/O 可否判断処理では，他プロセスの影響により，理想の入出力時間内に実 I/O 処理が終了しないことで調整精度が低下しないように，実 I/O 可否判断規則に基づき，デバイスドライバへの実 I/O 要求数を制限する．具体的には，被調整プロセスの実 I/O 処理が要求入出力性能から算出される理想の入出力時間内に終了するように，許容値を決定する．許容値の算出式を式 (3) に示す．

$$\text{許容値} = MAX(1, \frac{100}{\sum_{i=1}^{k} P_i} - 1) \quad (3)$$

ここで，P_i は，実 I/O 要求を行っていない被調整プロセスの上位 i 番目の要求入出力性能である．このため，実 I/O 要求を行っていないプロセスは許容値の算出に使用する対象となる．また，許容係数 k は，許容値の算出において考慮するプロセス数であり，利用者が求められる調整精度の高さにあわせて設定される．共存プロセスが入出力処理を実行できなくなることを防ぐため，許容値は少なくとも 1 である．したがって，実 I/O 可否判断処理では，デバイスドライバへの実 I/O 要求数と許容値を比較し，許容値の方が大きい場合，実 I/O 処理を許可し，そうでない場合，待ち処理を行う．

許容値は入出力性能調整の調整精度に影響を与えるため，許容値を正確に定めることは重要である．3.2 節で示す入出力管理の各機能を使用することにより，式 (3) 中の k もしくは P_i が変化する．したがって，入出力性能を正確に調整するために，式 (3) 中の k もしくは P_i が変化するごとに許容値を更新する必要がある．許容値の更新契機を以下に示す．

(契機 1) 資源「入出力」に対するプロセスの関連付け
(契機 2) 資源「入出力」に対するプロセスの関連付け解除
(契機 3) 実 I/O 可否判断処理前
(契機 4) 遅延処理終了後
(契機 5) 資源「入出力」の要求入出力性能の変更
(契機 6) 許容係数の変更

(契機 1) において，資源「入出力」にプロセスを関連付ける場合，許容値の算出に使用する対象となるプロセスが増加する．また，(契機 2) において，資源「入出力」とプロセスの関連付けを解除した場合，許容値の算出に使用する対象となるプロセスが減少する．つまり，式 (3) の P_i が変化するため．(契機 1) と (契機 2) において許容値を更新する必要がある．(契機 3) において，実 I/O 可否判断処理を行うプロセスは遅延処理が終了するまで，入出力要求を発行することがないため，許容値の算出に使用する対象とならない．また，(契機 4) において遅延処理後のプロセスは，入出力要求を発行できるため，許容値の算出に使用する対象となる．つまり，式 (3) の P_i が変化するため，(契機 3) と (契機 4) において許容値を更新する．(契機 5) において，入出力の程度を変更した資源「入出力」にプロセスを関連付けられている場合，関連付けられているプロセスの入出力の程度も変更される．つまり，変更する資源「入出力」が要求入出力性能を持つ場合，式 (3) の P_i が変化するため，(契機 5) において許容値を更新する必要がある．(契機 6) について，許容係数 k を変更する場合，許容値の算出において考慮するプロセス数が変更される．つまり，式 (3) の k が変化するため．(契機 6) において許容値を更新する必要がある．

以上より，(契機 1)〜(契機 6) において，許容値の更新を行う．このため，各契機に対応した処理を終了後，許容値の更新処理を行う．

許容値の算出時，実 I/O プロセス管理リスト内の被調整プロセスの数が許容係数 k 以下の場合，実 I/O 処理中でない全ての被調整プロセスが許容値の算出対象となる．また，被調整プロセスが存在しない場合，許容値を設定する必要はないため，許容値は最大プロセス数となる．

3.4 期待される効果

資源「入出力」の導入により, プロセスの入出力性能を保証および制限することができる．また, 資源「入出力」の特徴として, 以下の 2 点を示す.

(1) 入出力処理に優先度の概念を導入

資源「入出力」の入出力の程度として, 優先度を導入したことにより，3.3.1 節で示した制御法に加え，優先度制御により入出力性能調整を行うことができる．このため，使用者の利用目的に合わせて，プロセスの入出力性能を調整することができる．例えば，入出力時間を調整する場合，性能調整入出力により入出力時間を調整することができる．一方，入出力時間の調整を要求しないが，入出力処理に優先順位をつけたい場合，優先度入出力により，入出力処理に優先度を付与することができる.

(2) 被調整プロセスを考慮した制御

許容値は式 (3) より算出する．3.3.2 項で示すように，入出力管理の提供する機能を使用する毎に，またはプロセスが入出力処理を行う毎に許容値の算出対象となるプロセスは変化する．このため，各契機において，許容値を再度算出して更新することにより，入出力デバイス内の入出力要求数を適切に制限することができる．したがって，被調整プロセスの実 I/O 開始時期が遅れ，理想の入出力時間以内に実 I/O 処理が終了しないという状態を防ぐ．これにより，許容値を更新しない場合と比べて，更新する場合は被調整プロセスの調整精度が向上する.

4. 評価

4.1 評価目的

Tender に実現した入出力性能調整機能の調整精度を評価する．3.3.2 項で述べたように，許容値は入出力性能調整の調整精度に影響を与えるため，3.3.2 項で述べた更新契機で更新し，適切な値を定めることが重要である．許容値を更新しないことが入出力性能へ及ぼす影響について，スループットと調整精度の観点から評価し，許容値を更新する重要性を示す.

4.2 評価条件

Celeron(2.0GHz) プロセッサと IDE HDD である ST340016A(容量 40GB，7200RPM，2MB キャッシュ) を搭載した計算機に資源「入出力」による入出力要求数を調整する制御法を実装した ***Tender*** で測定を行い，入出力性能の調整精度に着目して評価を行った．評価で使用する評価プログラムは，I/O 処理として磁気ディスク装置から 512 バイトを読み込む入力処理を繰り返す．なお，入力位置はランダムである．また，入出力システムコールの発行前後に取得したハードウェアクロックの差を測定し，入出力時間とする．CPU 処理時間と I/O 処理時間の比率を処理比率と名付ける．調整精度の評価尺度として，調整比を用いる．調整比は,

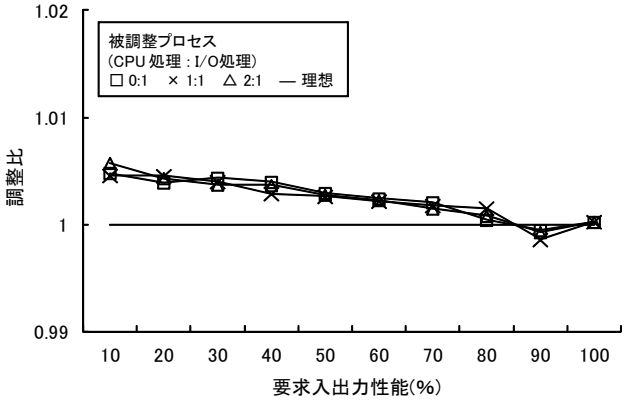


図 3 占有状態における要求入出力性能と調整比

$$\text{調整比} = \frac{\text{入出力時間の実測値}}{\text{理想の入出力時間}} \tag{4}$$

である．ここで，入出力時間の実測値とは，入出力システムコールを発行してから戻り値が返却されるまでの時間である．また，理想の入出力時間とは，実 I/O 時間を要求入出力性能で割った値である．以降の評価では，評価プログラムによる CPU 処理と I/O 処理を 1002 回繰り返し，入出力時間を測定し，最初と最後を除いた 1000 回の入出力時間を評価に利用する．なお，すべての評価において，許容係数は 1 である.

4.3 占有状態の調整精度

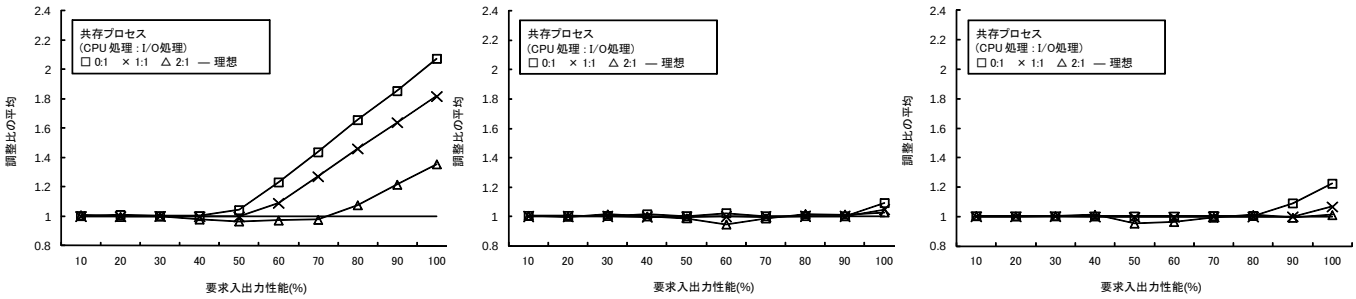
占有状態における要求入出力性能と調整比 (平均値) の関係を**図 3** に示す．図 3 から，以下のことがわかる.

(1) 被調整プロセスの要求入出力性能と処理比率に関わらず，理想の調整比と測定値との差は約 0.005 であり，調整比は 1 付近であるため調整精度は高い.

(2) 要求入出力性能が 90 %の時，調整比が 1 以下となる.

(2) の原因は，遅延時間が 1msec 以下となることである．本実装では，遅延可否閾値が 1msec であるため，3.3.1 項より，遅延時間が 1msec 以下である場合，次の遅延処理に繰り越す．要求入出力性能が 90 %の場合，遅延時間が 1msec となる可能性が高い．このため，遅延時間を繰り越す回数が多いほど，調整比は 1 以下となる．したがって，要求入出力性能が 90 %のときは，調整比が 1 以下になりやすい.

次に調整比の最大値と頻度分布について考察する．被調整プロセスの処理比率が 1：1 の場合において，調整比とその頻度分布を**図 4** と**図 5** に示す．図 4 より，最大値のみが平均値や中央値から大きく外れていることがわかる．例えば，要求入出力性能が 10 %の時，1.3 である．また，図 5 から調整比の大半は 1 付近であることがわかる．したがって，占有状態において調整精度は高いといえる.



(A) 被調整プロセス（CPU:I/O=0:1） (B) 被調整プロセス（CPU:I/O=1:1） (C) 被調整プロセス（CPU:I/O=2:1）

図 6 共存プロセスを 1 つだけ同時走行させた場合の調整比

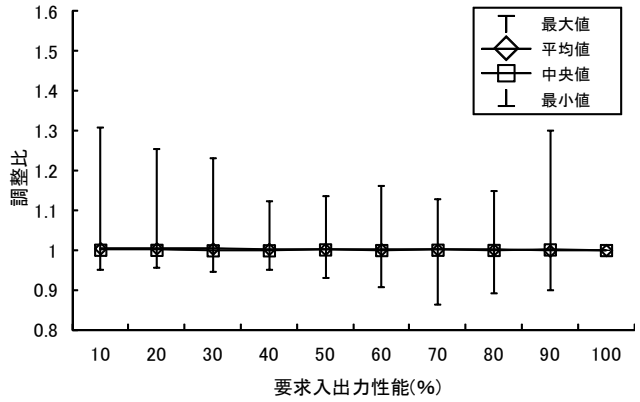


図 4 調整比（CPU：I/O = 1：1）

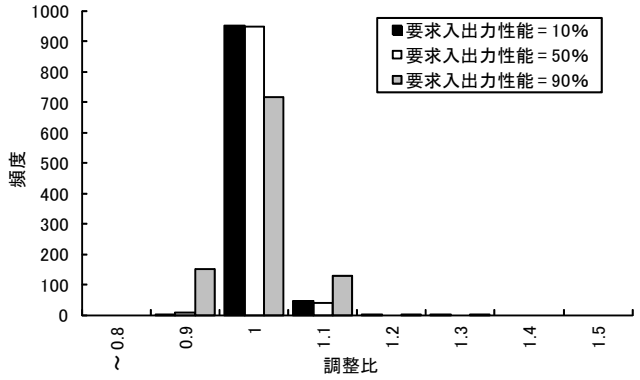


図 5 調整比の頻度分布（CPU：I/O = 1：1）

4.4 他プロセスの影響

共存プロセスを 1 つだけ同時走行させた場合における被調整プロセスの要求入出力性能と調整比（平均値）の関係を**図 6** に示す．図 6 から以下のことがわかる．

(1) 被調整プロセスの要求入出力性能が低い場合，共存プロセスの影響は小さく，調整精度は高い．
(2) 被調整プロセスの要求入出力性能が高い場合，共存プロセスの入出力処理の比率が高くなるにつれ，共存プロセスの影響は大きくなり，調整精度は低下する．
(3) 被調整プロセスの入出力処理の比率が高くなると，共存プロセスの影響は大きくなり，調整精度は低下する．

表 2 許容値の更新パターン

	(契機 3)	(契機 4)
条件 1	○	○
条件 2	○	×
条件 3	×	○
条件 4	×	×

○：有効 ×：無効

結果は文献 [1] と同じ傾向を示している．調整精度が低下する原因は，被調整プロセスと共存プロセスが同時期に入出力要求を発行する確率が高くなり，デバイスドライバ内で被調整プロセスの実 I/O 開始待ちが発生するからである．

4.5 許容値の更新とスループット

許容値の更新契機は 3.3.2 項で述べた．評価プログラムを実行した際，(契機 3) と (契機 4) による許容値の更新が多い．したがって，本評価では，(契機 3) と (契機 4) における許容値の更新を有効もしくは無効にすることで，スループットの変化を評価する．本評価における許容値の更新パターンを**表 2** に示す．要求入出力性能が 30 %，40 %，50 %の 3 つの被調整プロセス (被調整プロセス 1，被調整プロセス 2，および被調整プロセス 3) と 1 つの共存プロセスを同時に走行させた．共存プロセスと要求入出力性能が 30 %の被調整プロセスの処理比率は 0：1 であり，要求入出力性能が 40 %と 50 %の被調整プロセスの処理比率は 2：1 である．4 つ全てのプロセスが走行している間のスループットを評価する．このため，スループット算出対象範囲は，4 つのプロセスのうち 4 番目に起動する被調整プロセス 1 が 1 回目の入出力処理を終了してから，最初に 1002 回目の入出力処理を終了するプロセスが 1002 回目の入出力処理を終了するまでの間とする．表 2 の各条件におけるスループットを**図 7** に示す．図 7 から以下のことがわかる．

(1) (契機 3) において許容値の更新を無効にするより，有効にする方がスループットが高い．
(2) (契機 4) における許容値の更新の有無はスループットへの影響が小さい．

(1) について，例えば，条件 1 と条件 3 では，条件 3 に

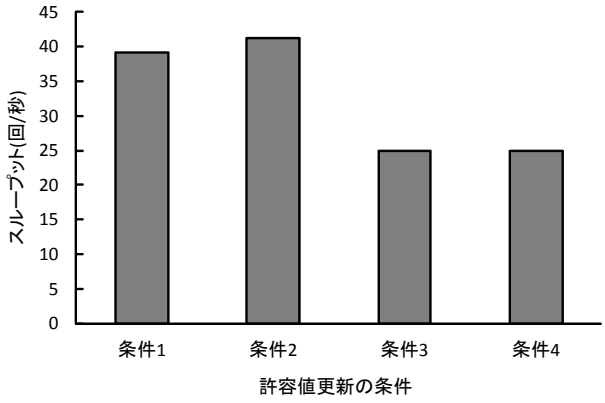


図 7 許容値の更新の有無におけるスループットの変化

比べて，条件 1 はスループットが 15 回/秒多い．したがって，(契機 3) における許容値の更新有無がスループットに及ぼす影響は大きいといえる．条件 3 では，(契機 3) において許容値を更新しないため，許容値の値は入出力処理中のプロセスを考慮した値のままである．このため，更新する場合と比べて，更新しない場合は許容値が小さい．これにより，デバイスドライバへの発行可能な実 I/O 要求数が小さくなり，スループットが低下する．条件 1 と条件 3 における各被調整プロセスの各時間間隔におけるスループットをそれぞれ**図 8** と**図 9** に示す．測定開始から 60 秒経過するまでの間は 4 つの全てのプロセスが走行しているため，測定開始から 60 秒経過するまでの間において，10 秒間隔でスループットを算出する．図 9 からわかるように，(契機 3) で許容値を更新しない場合，要求入出力性能が 40 %と 50 %のプロセスが走行している間，許容値に対する要求入出力性能が 40 %と 50 %のプロセスの影響が大きいため, 要求入出力性能が 30 %のプロセスはほぼ入出力処理を行っていないため，スループットは 0 に近い．一方，図 8 では，(契機 3) で許容値を更新しているため，要求入出力性能が 30 %のプロセスが入出力処理を行うことができる．このため，スループットは約 9 回/秒である．

(2) について，条件 1 と条件 2 では，条件 1 に比べて，条件 2 はスループットが 2 回/秒多い．したがって，(契機 4) における許容値の更新有無がスループットに及ぼす影響は小さいといえる．(契機 4) では，入出力処理が終了する被調整プロセスが許容値の算出対象として再び加わり，許容値を更新する．このため，(契機 4) で許容値を更新する場合，更新前に比べて，更新後は許容値が小さくなる可能性が高い．しかし，入出力処理要求が発行されれば，(契機 3) で再び許容値を更新するため，(契機 4) において更新した許容値による影響は小さい．また、同期処理で起動したプロセスが実 I/O 処理を行う場合は，(契機 4) で更新した許容値により実 I/O 処理の可否を判断する．ここで，実 I/O 処理を禁止したとしても，スループット全体へ与える影響

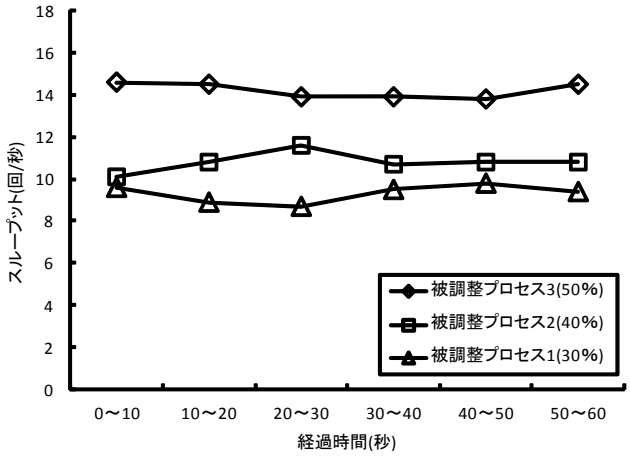


図 8 条件 1 における被調整プロセスの経過時間ごとのスループット

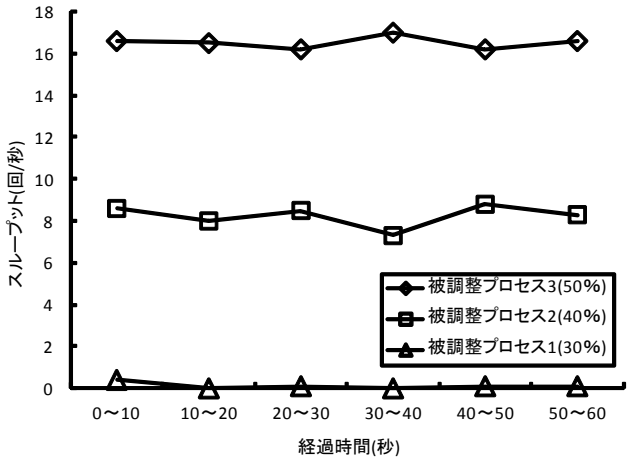


図 9 条件 3 における被調整プロセスの経過時間ごとのスループット

は小さい．なぜなら，許容値は少なくとも 1 であるため，実 I/O 処理が禁止されても，デバイスドライバには少なくとも実 I/O 要求が 1 以上存在するからである．このため，(契機 4) における許容値の更新有無はスループットへの影響が小さい．

各条件における要求入出力性能が 40 %と 50 %の被調整プロセスの調整比について，理想値と測定値の差を**図 10** に示す．理想値と測定値の差は絶対値で表す．図 10 より，(契機 4) で許容値を更新しない場合，調整精度が低下している．例えば，要求入出力性能が 50 %の被調整プロセスにおいて，条件 1 に比べて，条件 2 は調整比の理想値と測定値の差が約 0.06 大きい．これは，(契機 4) で許容値を更新しない場合，許容値の値は入出力処理を終了したプロセスを考慮していない値のままである．このため，(契機 4) で許容値を更新する場合に比べて，更新をしない場合は許容値が大きい．これにより，デバイスドライバへの発行可能な実 I/O 要求数が大きくなり，入出力優先度の高い被調整プロセスの実 I/O 処理時間が理想の入出力時間内に終了

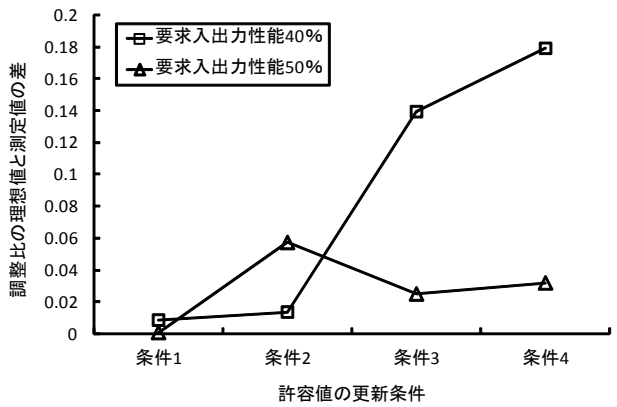


図 10 許容値の更新有無における調整比の理想値と測定値の差

せず，調整精度が低下する．

以上より，スループット，調整精度の両方が高い条件 1 が一番適切な更新条件であることがわかる．

5. おわりに

Tender の資源「入出力」による入出力性能調整機能の評価について示した．被調整プロセスが単独で走行する場合，高い精度で入出力性能を調整できる．また，共存プロセスが存在する場合でも，要求入出力性能が 50 %未満であれば，高い精度で調整できる．なお，要求入出力性能が 50 %以上であっても，被調整プロセスの入出力要求と共存プロセスの入出力要求が同時期に発生する確率が低い場合，高い精度で調整できる．また，実 I/O 可否判断処理前と遅延処理後の両方において許容値を更新をする場合，スループットと調整精度が高い．

残された課題として，複数プロセスをグループとし，グループ単位で入出力性能を調整する方法の設計と実現がある．

参考文献

[1] 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).
[2] 谷口秀夫：サービス処理時間を調整するプロセスのスケジュール法，電子情報通信学会論文誌 (D-I)， Vol. 81, No. 4, pp. 386–392 (1998).
[3] 谷口秀夫，青木義則，後藤真孝，村上大介，田端利宏：資源の独立化機構による ***Tender*** オペレーティングシステム，情報処理学会論文誌，Vol. 41, No. 12, pp. 3363–3374 (2000).
[4] 田端利宏，乃村能成，谷口秀夫：***Tender*** オペレーティングシステムにおける資源「演算」を利用したプロセスグループの実行性能調整法，電子情報通信学会論文誌 (D-I), Vol. 87, No. 11, pp. 961–974 (2004).
[5] 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，情報処理学会研究報告，Vol. 2011-OS-118, No. 19, pp. 1–8 (2011).
[6] Margo I. Seltzer, Peter M. Chen and John K. Ousterhout: Disk scheduling Revisited, *Proceedings of the Winter 1990 USENIX Technical Conference*, pp. 313–324 (1990).
[7] H.P.Chang, R.I.Chang, W.K.Shih, R.C.Chang: GSR: A global seek-optimizing real-time disk-scheduling algorithm, *Journal of Systems and Software*, Vol. 80, No. 2, pp. 198–215 (2007).
[8] R. Abbott and H. Garcia-Molina: Scheduling real-time transactions: a performance evaluation, *ACM Trans. Database Syst.*, Vol. 17, No. 3, pp. 513–560 (1992).
[9] Anna Povzner, Tim Kaldewey, Scott Brandt, Richard Golding, Theodore M. Wong and Carlos Maltzahn: Efficient guaranteed disk request scheduling with fahrrad, *SIGOPS Oper. Syst. Rev.*, Vol. 42, No. 4, pp. 13–25 (2008).
[10] Ting Yang, Tongping Liu, Emery D. Berger, Scott F. Kaplan and J. Eliot B. Moss: Redline: First Class Support for Interactivity in Commodity Operating Systems, *OSDI*, USENIX Association, pp. 73–86 (2008).