Vol.2020-DPS-184 No.10
Vol.2020-EIP-89 No.10
2020/9/11

入出力性能調整法の調整精度向上法

長尾尚[†1]　田辺雅則[†2]　横山和俊[†3]　谷口秀夫[†2]

概要：サービスを提供するプログラムは，プロセスとして OS に制御され，プロセッサ処理と入出力処理を繰り返す．このため，利用者へ適切な応答速度を提供するためには，プロセッサ処理や入出力処理の実行速度を調整できる必要がある。また，この調整機能は処理分散を行う上でも有効である．これまで，著者らはプロセッサ性能や入出力性能の調整法を提案した．ここで，入出力性能の調整法では，入出力要求の処理完了後に，OS が提供するスリープ機能を用いて入出力システムコールの終了を遅延することで，入出力時間を一定に保つ．近年，入出力要求を短時間に処理できる SSD が普及している．SSD の実 I/O 時間が短いため，入出力時間を一定に保つための遅延時間がスリープ機能の動作可能な最短時間よりも短くなり，上手く入出力時間を調整できない．そこで，スリープ機能の動作可能な最短時間よりも短い時間の遅延が必要な場合，ビジーループを用いて入出力システムコールの終了を遅延させる方式を提案する．評価により，提案方式は，実 I/O 時間が短い入出力デバイスにおいても入出力時間を精度良く調整できることを示す．

キーワード：入出力スケジューラ，性能調整，オペレーティングシステム，仮想計算機

Improving Accuracy for I/O Performance Regulating Method

TAKASHI NAGAO[†1]　MASANORI TANABE[†2]
KAZUTOSHI YOKOYAMA[†3]　HIDEO TANIGUCHI[†2]

Abstract: A program that provides service is controlled by operating system (OS) as process. And, the process repeats processor processing and I/O processing. Thus, it is necessary to regulate execution speed of processor processing and I/O processing in order to provide users with appropriate execution speed. And, this regulating execution speed is also valid for distributed processing. We proposed the regulating method for the processor performance and I/O performance. When a process finishes I/O process, the I/O performance regulating method delays the wakeup of the process in order to regulate I/O time by sleep function. Solid state drive (SSD) can execute I/O at short time. So, calculated delay time to regulate I/O time is shorter than shortest time supported by the sleep function. As a result, regulating accuracy decreases. In this paper, we propose selected busy loop methods that execute busy loop when the calculated delay time is shorter than the shortest time supported by the sleep function. The evaluation shows that the proposed method can increase regulating accuracy with the device that can execute I/O request at short time.

Keywords: I/O Scheduling, Regulating Performance, OS, Virtual Machine

1. はじめに

計算機の性能向上に伴い，一台の計算機上で複数のソフトウェアを同時に動作させることが一般的になっている．例えば，ウィルス対策ソフトウェアにファイルを検査させつつ，動画を視聴しながら文章を作成する，といった利用が挙げられる．このとき，他のソフトウェアによって，利用者が直接操作するソフトウェアの入力受付や表示が滞ると，利用者が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ティングシステム（以降，OS と呼ぶ）が当該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 は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．この制御によって他プロセスが優先されると，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，利用者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）が使用できる計算機資源を他プロセス（ここでは，重要でないプロセスと呼ぶ）に関わらず一定に保ち，当該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，当該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の調整法[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，調整対象プロセスの入出力要求を調整対象で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，利用者が指定する性能（以降，要求入出力性能と呼ぶ）に合わせて，プロセスの入出力時間を調整する．

†1 株式会社 日立製作所
Hitachi, Ltd.
†2 岡山大学大学院自然科学研究科
Graduate School of Natural Science and Technology, Okayama University
†3 高知工科大学情報学群
School of Information, Kochi University of Technology

Vol.2020-DPS-184 No.10
Vol.2020-EIP-89 No.10
2020/9/11

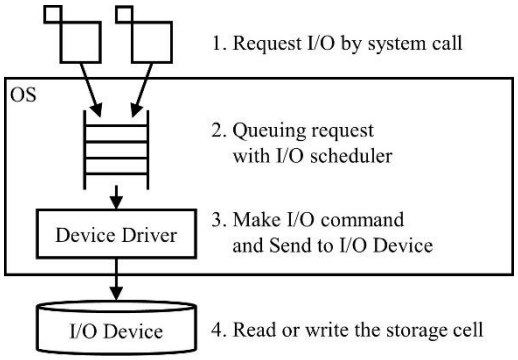


図 1　入出力処理の流れ

Figure 1　Overview of I/O.

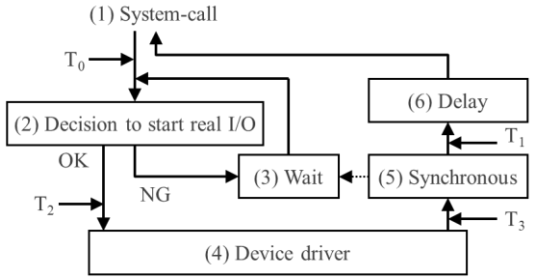


図 2　基本方式

Figure 2　Basic Method.

ここで，文献[2]の方式では，OS が提供するスリープ機能を利用し，調整対象プロセスの起床を遅延させる．スリープ機能では，タイマ割込によって経過時間を監視する．このため，本調整法がタイマ割込の周期よりも短い時間を指定してスリープ機能を呼び出すと，スリープ機能は，指定された時間が経過しても，タイマ割込が発生するまで認識できず，時間通りに調整対象プロセスを起床できない．

近年，入出力デバイスとして，半導体メモリを用いた SSD(Solid State Drive)が普及している．SSD は，磁気ディスクを用いた HDD と異なり，磁気ヘッドの移動や磁気ディスクの回転といった機械動作がない．このため，SSD の実 I/O 時間は，HDD に比べて短い．本調整法は，要求入出力性能と実 I/O 時間を用いて，調整対象プロセスの起床の遅延時間を算出する．したがって，SSD を利用した場合，算出する遅延時間がタイマ割込の周期よりも短くなり，本調整法が入出力時間を上手く調整できないことがある．

そこで，算出する遅延時間がスリープ機能の動作可能な最短時間よりも短い場合，スリープ機能でなく，ビジーループによって調整対象プロセスの起床を遅延させる方式を提案する．ビジーループは，プロセッサを占有して調整対象プロセスの遅延時間を監視する．このため，ビジーループ実施中は，他プロセスが実行可能状態であっても，プロセッサを割り当てることができず，他プロセスの処理速度を低下させる悪影響がある．そこで，提案方式では，ビジーループによって調整対象プロセスの起床を遅延する際，一定期間ごとに実行可能状態の他プロセスの有無を判定し，実行可能状態のプロセスが存在する場合にはビジーループを終了する．これにより，他プロセスへの悪影響を抑える．また，評価により，提案方式により，SSD を利用した場合にも，入出力時間を上手く調整し，かつ他プロセスの処理時間の増加を抑えることができることを示す．

2. 入出力処理の流れ

プロセスは入出力デバイス内のデータを読み書きする場合，OS が提供する入出力用システムコールを呼び出す．このシステムコールの処理流れを図 1 に示す．OS は当該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，プロセスを待ち状態にする．この待ちキューは，OS の入出力スケジューラによって整列される．例えば，Linux ではシーク時間の最小化や公平性の担保といった目的に応じて，noop, deadline, acticipatory, cfq の 4 種類の入出力スケジューラを実現している．

OS は入出力デバイスの状態を監視しており，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形式に変換して発行する．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 は通知を受け取ると，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．

複数のプロセスが入出力要求を発行すると，これらのプロセスは，OS 内の待ちキューに登録される．後端に登録されたプロセスは，デバイスドライバがキュー内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つ．さらに，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絶え間なく記憶媒体を読み書きする．この動作により，後から受け付けた入出力要求は，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待つ．このように，複数のプロセスが入出力要求を発行すると，あるプロセスの入出力システムコールに要する時間（以降，入出力時間と呼ぶ）は，他プロセスの入出力要求の処理を待つことにより，長くなり得る．

3. 入出力性能調整法[2]

3.1 基本方式

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．本調整法

Vol.2020-DPS-184 No.10
Vol.2020-EIP-89 No.10
2020/9/11

は，調整対象プロセスの入出力時間を入出力デバイスが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（以降，理想の入出力時間と呼ぶ）に常に保つことで，上記の機能を実現する．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．

$$\text{理想の入出力時間} = \frac{100}{P} \times \text{実 I/O 時間} \quad (1)$$

文献[2]にて提案した入出力順序保証方式を図 2 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．許容値の算出方法は，後述する．入出力デバイス内の入出力要求数が許容値以上の場合，入出力要求の発行を許可せず，（3）の処理を行う．許容値未満の場合，（4）の処理を行う．

（3）プロセスを待ちキューに登録して待ち状態にする．このキューは，要求入出力性能の降順で整列する．要求入出力性能が同じプロセスは，入出力用システムコールの発行順に整列する．同期処理から待ち解除の指示を受けると，キューの先頭からプロセスを起床する．

（4）デバイスドライバを介して入出力要求を発行する．

（5）待ち処理に同期を送信する．

（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．

3.2 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．

最初に，調整対象プロセスが 1 個だけ存在する場合について，許容値の決定式を述べる．理想の入出力時間には，当該プロセスの入出力要求の処理に要する実 I/O 時間が含まれる．このため，理想の入出力時間から 1 回分の実 I/O 時間を差し引いた時間だけ，他プロセスの実 I/O 処理を待つことができる．したがって，この時間内に入出力デバイスが処理できる入出力要求数を許容値とする．例えば，要求入出力性能が 20%の場合，理想の入出力時間は実 I/O 時間の 5 倍である．自身の実 I/O 時間を除いた時間内に，入出力要求を 4 個処理できるため，許容値は 4 となる．なお，他プロセスの入出力要求を発行できなくなることを防ぐため，許容値は 1 以上にする．以上より，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は以下である．

$$max\left(1, \frac{100}{P} - 1\right) \quad (2)$$

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．プロセスが入出力用システムコールを発行したとき，すでに入出力デバイス内の入出力要求数が許容値以上であれば，待ちキューに登録される．複数の調整対象プロセスがある場合，要求入出力性能が低いプロセスは，この待ちキューにおける待ち時間が要求入出力性能の高いプロセスによって長くなり得る．これにより，長くなり得る最長の時間を見込んで，許容値を小さくする．具体的には，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．

$$A_i = \frac{100}{P_i} - \sum_j f(i,j) - 1 \quad (3)$$

$$f(x,y) = \begin{cases} 1 & P_y \geq P_x \text{のとき} \\ 0 & \text{上記以外のとき} \end{cases} \quad (4)$$

$$A = max\left(\min_i(A_i), 1\right) \quad (5)$$

3.3 遅延処理

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用システムコールの完了時刻を遅らせる．遅延時間 T_S の算出式を以下に示す．

$$T_S = \text{理想の入出力時間} - (T_1 - T_0) \quad (6)$$

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 2 の T_2 と T_3 の差分）である．一方，入出力デバイス内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．

$$\text{実 I/O 時間} = min(T_3 - T_2, T_3\text{の間隔}) \quad (7)$$

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，ばらつきが生じ得る．このばらつきの影響を抑えるため，式(7)で算出する実 I/O 時間を過去 10 回分だけ蓄積し，この平均を実 I/O 時間として用いる．

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，タイマ割込で経過時間を監視するため，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ばらつきを平準化する．

4. 従来方式の問題

遅延処理で利用するスリープ機能では，タイマ割込によって経過時間を監視する．このため，本調整法がタイマ割込の周期よりも短い遅延時間を指定してスリープ機能を呼び出しても，スリープ機能は，時間通りに遅延処理を終了

Vol.2020-DPS-184 No.10
Vol.2020-EIP-89 No.10
2020/9/11

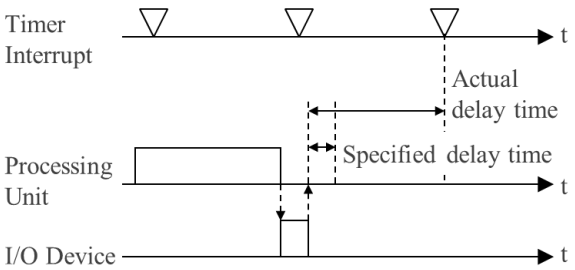


図 3　プロセスの起床の遅延
Figure 3　Delay of wake-up process.

できない．ここで，本調整法は，式(1)と式(6)を用いて遅延時間を算出する．このため，実 I/O 時間が短い場合や要求入出力性能が高い場合，算出する遅延時間が短くなり，スリープ機能が時間通りに遅延処理を終了できず，入出力時間を上手く調整できなくなる可能性がある．

ここで，入出力デバイスとして，半導体メモリを用いた SSD が普及している．SSD は，磁気ディスクを用いた HDD と異なり，磁気ヘッドの移動や磁気ディスクの回転といった機械動作がない．このため，SSD の実 I/O 時間は，HDD に比べて短い．これにより，本調整法は，入出力時間を上手く調整できない場合がある．図 3 を用いて，実 I/O 時間が短いことにより，入出力時間を上手く調整できないことを説明する．OS が調整対象プロセスの入出力要求の処理完了を認識すると，実 I/O 時間を用いて遅延時間を算出する．ここで，実 I/O 時間が短い場合，算出する遅延時間がタイマ割込の周期よりも短くなり得る．この短い遅延時間を指定してスリープ機能を呼び出した場合，スリープを開始してから指定した遅延時間を経過しても，タイマ割込が発生するまで，OS はプロセスを起床できない．このため，指定した遅延時間よりも実際の遅延時間が長くなり，実際の入出力時間が理想の入出力時間よりも長くなってしまう．

本調整法では，指定した遅延時間と実際の遅延時間の差を次回の遅延処理に繰り越す．このため，実際の入出力時間が理想の入出力時間よりも長くなると，次の入出力処理では遅延処理を実施しないことで，実際の入出力時間を理想の入出力時間よりも短くする．

これにより，SSD を利用した環境では，実際の入出力時間の平均値と理想の入出力時間の差が小さいものの，個々の入出力時間と理想の入出力時間の差が大きくなってしまう．

5. 遅延処理の改善

算出した遅延時間がタイマ周期よりも短い場合には，調整対象プロセスにプロセッサを割り当て続け，指定した遅延時間が経過したことを監視するビジーループを用いる．これにより，算出した遅延時間通りに調整対象プロセスを起床できるようにする．この方式を選択的ビジーループ方式と呼ぶ．選択的ビジーループ方式の処理流れを図 4 に示し，説明する．

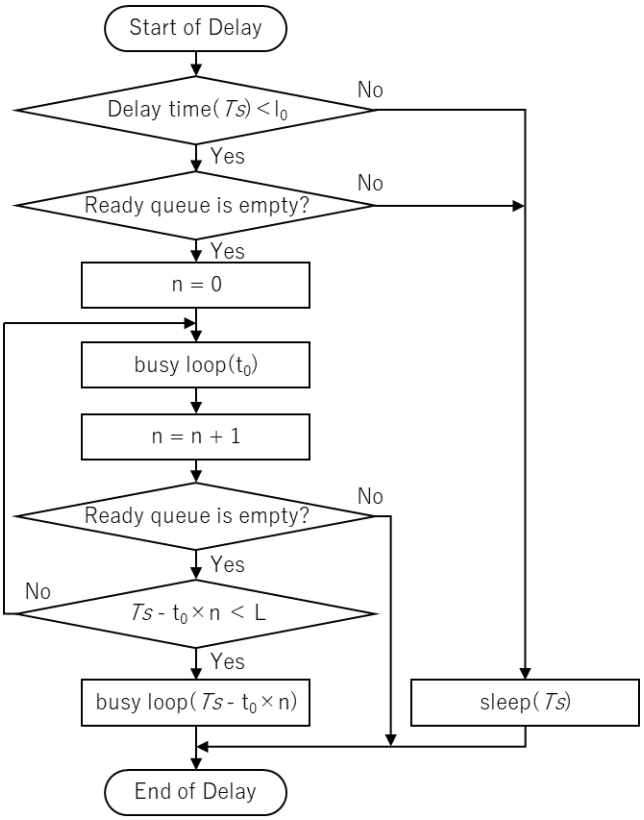


図 4　選択的ビジーループ方式
Figure 4　Selective busy loop method.

OS ごとにスリープ機能が遅延できる最短時間が異なる．ビジーループを用いるか否かを判定するために，ビジーループ選択閾値(l_0)を導入する．指定した遅延時間がビジーループ選択閾値よりも小さい場合，ビジーループを選択する．そうでない場合，スリープ機能を用いて調整対象プロセスの起床を遅延させる．

ビジーループ実施中は，調整対象プロセスにプロセッサを割り当てるため，他プロセスが実行可能状態であっても，当該プロセスのプロセッサ処理を実行できない．このため，他プロセスの処理速度を低下させてしまう悪影響がある．そこで，ビジーループを実施する前に，実行可能状態の他プロセスの有無を判定する．実行可能状態のプロセスが存在する場合，上記の悪影響を抑えるため，ビジーループでなく，スリープ機能を用いて調整対象プロセスの起床を遅延させる．実行可能状態の他プロセスが存在しない場合，ビジーループを実施する．さらに，ビジーループ実施中に他プロセスが実行可能状態に遷移した場合の悪影響を抑えるため，ビジーループ実施単位(t_0)を導入する．ビジーループ実施単位の時間だけビジーループするたびに，実行可能状態の他プロセスが存在するか否かを判定する．もし，他プロセスが存在した場合，以後のビジーループを実施せずに，調整対象プロセスの遅延処理を終了する．他プロセス

Vol.2020-DPS-184 No.10
Vol.2020-EIP-89 No.10
2020/9/11

表 1　評価環境

Table 1　Environment of Evaluation.

プロセッサ	2.5GHz 4cores
メモリ	8GB
入出力デバイス	SATA3.0 接続 HDD と SSD

が存在しない場合，ビジーループを続行する．

さらに，実行可能状態のプロセスが存在せず，ビジーループを一定回数継続できた場合，調整対象プロセスの入出力時間の調整を優先し，他プロセスの状態に関わらず，ビジーループを実施する．このために，ビジーループ継続閾値(L)を導入する．実行可能状態のプロセスの有無を判定した後，指定した遅延時間までの残り時間がビジーループ継続閾値未満の場合，以降の他プロセスの状態に関わらず，残り時間だけビジーループを実施する．

以上により，提案方式は，算出する遅延時間が短い場合にも指定通りの時間に調整対象プロセスを起床できる．さらに，他プロセスが存在していても，実行可能状態でなければ，ビジーループを継続することで，調整対象プロセスの入出力性能を上手く調整できる．

6. 評価

6.1 条件

まず，評価指標を述べる．本調整法が入出力時間を上手く調整できるか否かを調整精度と呼ぶ値で評価する．調整精度の算出式を以下に示す．調整精度は，1 に近いほど調整の精度が良く，1 より大きいほど入出力時間が過大，1 より小さいほど入出力時間が過少であることを示す．

$$\text{調整精度} = \frac{\text{実際の入出力時間}}{\text{理想の入出力時間}} \tag{8}$$

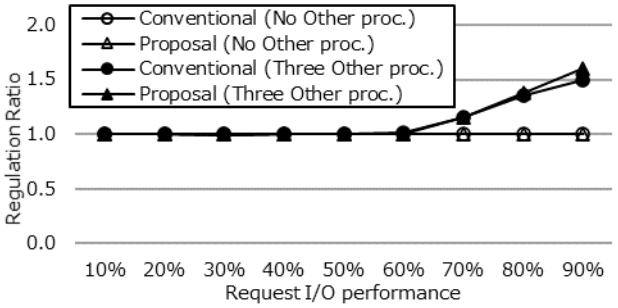
提案方式は，指定した遅延時間が短い場合にも，指定した時間通りに調整対象プロセスの起床を遅延できる．この効果を遅延精度と呼ぶ値で評価する．遅延精度の算出式を以下に示す．遅延精度は，1 に近いほど指定した遅延時間と実際の遅延時間の差が小さく，指定した時間通りに遅延できることを示す．

$$\text{遅延精度} = \frac{\text{実際の遅延時間}}{\text{指定した遅延時間}} \tag{9}$$

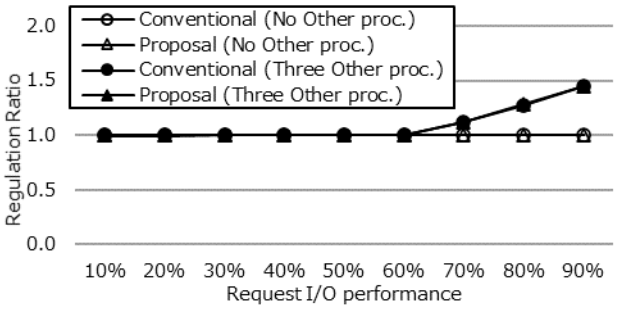
ビジーループを用いることで，他プロセスの処理速度が低下し得る．この影響を評価するために，後述する評価プログラムの開始から終了までの時間を用いる．この時間を他プロセス実行時間と呼ぶ．

次に，評価環境を述べる．表 1 の計算機に，本調整法を実装した FreeBSD Ver11.0 を動作させた．入出力デバイスとして，HDD と SSD を接続し，それぞれについて，次の測定を実施した．

評価プログラムとして，raw デバイスに対してランダム読み込みと単純な四則演算を実行するプロセッサ処理を 10000 回繰り返す処理を用いた．ここで，プロセッサ処理と入出力処理に要する時間が 1:1 になるよう，直前の読み込みに要した入出力時間だけ，プロセッサ処理を実行する．この評価プログラムを用いて，調整対象プロセスだけが走行する場合，および調整対象プロセス 1 つと他プロセス 3 つが走行する場合，を測定した．



(A) HDD



(B) SSD

図 5　平均の調整精度の分布

Figure 5　Average of Regulation Ratio.

最後に，提案方式のパラメタの設定を述べる．ビジーループ選択閾値(l_0)は 5 ミリ秒，ビジーループ実施単位(t_0)は 0.1 ミリ秒，ビジーループ継続閾値(L)は 1 ミリ秒に設定した．

6.2 調整精度

平均の調整精度を図 5 に示す．図 5 より，次のことが分かる．

（1）HDD と SSD のどちらにおいても，従来方式と提案方式のいずれも平均の調整精度が良い．

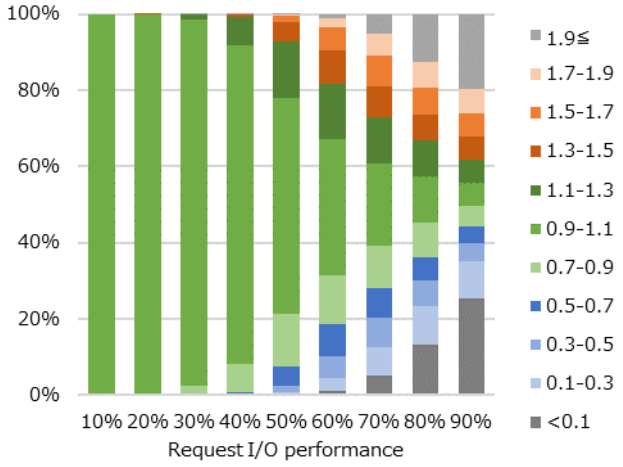
（2）他プロセスが存在する場合であっても，要求入出力性能が 60%以下の場合，従来方式と提案方式のいずれも平均の調整精度が良い．

このように，従来方式と提案方式のいずれも平均の調整精度が良い．

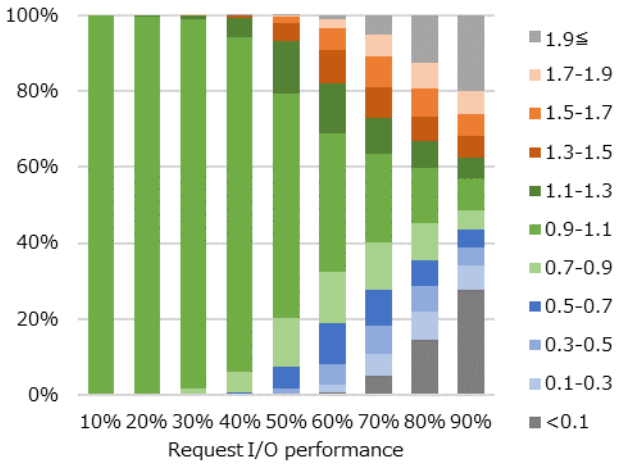
次に，調整精度の分布を図 6 と図 7 に示す．図 6 は，HDD 環境における調整精度の分布であり，図 7 は，SSD 環境における調整精度の分布である．図 6 より，次のことが分かる．

（3）要求入出力性能が低い場合，従来方式と提案方式の

Vol.2020-DPS-184 No.10
Vol.2020-EIP-89 No.10
2020/9/11



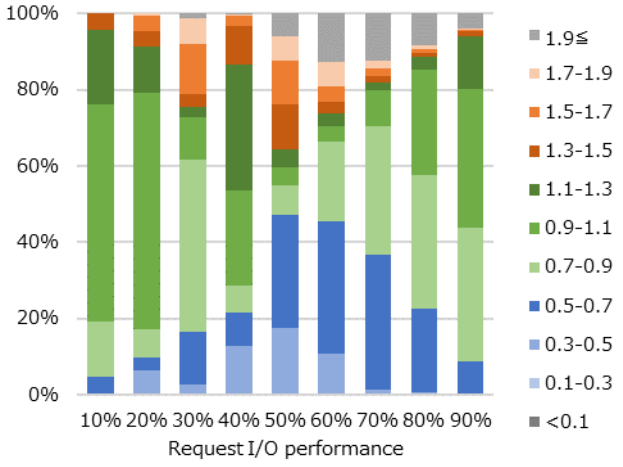
(A) Conventional Method



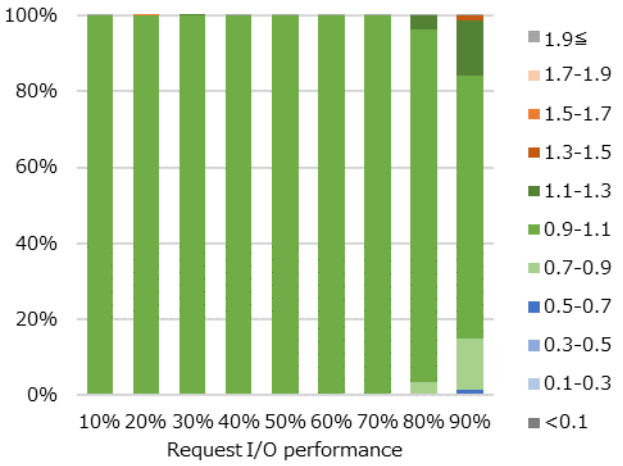
(B) Proposal Method

図 6　HDD 環境における調整精度の分布

Figure 6　Distribution of Regulation Ratio with HDD



(A) Conventional Method



(B) Proposal Method

図 7　SSD 環境における調整精度の分布

Figure 7　Distribution of Regulation Ratio with SSD

いずれも個々の調整精度が良い．例えば，要求入出力性能が 30%以下の場合，両方式とも 0.9～1.1 の調整精度が全体の 95%以上を占める．

（4）要求入出力性能が高くなると，従来方式と提案方式のいずれも，個々の調整精度がばらつく．例えば，要求入出力性能が 90%のとき，両方式とも，0.7 以下の調整精度や 1.1 以上の調整精度がそれぞれ全体の 40%以上を占める．

このように，HDD 環境において，提案方式は，従来方式と同じ性質を示す．これは，HDD 環境では，実 I/O 時間が長く，算出する遅延時間がビジーループ選択閾値を超えることで，提案方式が従来方式と同じくスリープ機能を用いて遅延処理を実施するためである．

次に，図 7 より，次のことが分かる．

（5）従来方式では，要求入出力性能が 30%以上になると，0.9～1.1 の調整精度の割合が低くなり，個々の入出力時間を上手く調整できない．例えば，要求入出力性能が 50%のとき，0.9～1.1 の調整精度は，全体の 4%に過ぎない．1.5 以上の調整精度が全体の 23%を占めており，これらの超過した時間を全体の 54%を占める 0.9 以下の調整精度の入出力処理で補っている．

（6）提案方式では，要求入出力性能に関わらず，個々の調整精度が良い．例えば，要求入出力性能が 80%以下の場合，0.9～1.1 の調整精度は，全体の 92%を占める．要求入出力性能が 90%の場合であっても，0.9～1.1 の調整精度は，全体の 69%を占める．

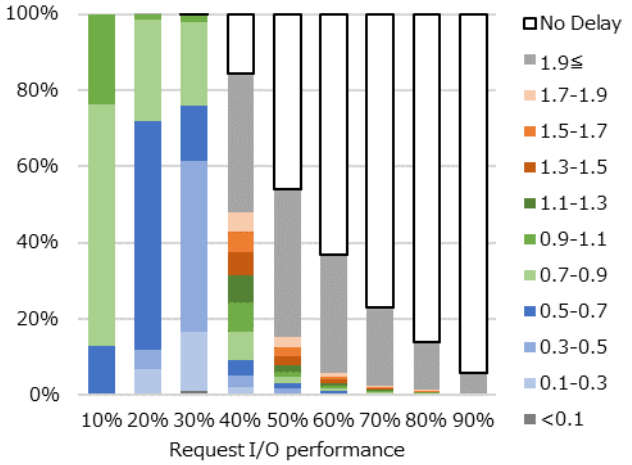
以上より，SSD 環境では，実 I/O 時間が短いために，従来方式では個々の調整精度が悪化する．一方で，提案方式の調整精度は良い．

6.3　遅延精度

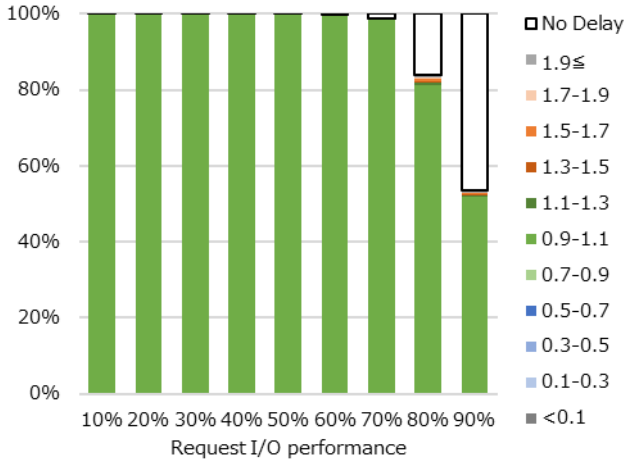
図 7 の条件における遅延精度の分布を図 8 に示す．図 8 より，次のことが分かる．

（1）従来方式では，要求入出力性能が高い場合，遅延処

Vol.2020-DPS-184 No.10
Vol.2020-EIP-89 No.10
2020/9/11



(A) Conventional Method



(B) Proposal Method

図 8　SSD 環境における遅延精度の分布

Figure 8　Distribution of Delay Ratio with SSD

理を実施しない場合と大きな値の遅延精度が高い割合を占める．例えば，要求入出力性能が 50%の場合，遅延処理を実施しない場合は，全体の 46%を占める．さらに，1.9 以上の遅延精度は，全体の 38%を占める．これは，スリープ機能を用いて遅延処理を実施することで，実際の遅延時間が指定された遅延時間よりも大きな値になり，理想の入出力時間よりも実際の入出力時間が長くなる．これにより，従来方式は，次回の入出力時間を理想の入出力時間よりも短くして入出力時間を平準化するために，この差を繰り越す．次回の入出力処理では，算出した遅延時間から繰り越し時間を減算することで，遅延時間が負の値となり，遅延処理を実施しなくなる．このように，従来方式では，指定した遅延時間よりも長く遅延してしまう入出力処理と遅延処理を実施しない入出力処理を繰り返す．

（2）提案方式では，要求入出力性能に関わらず，遅延精度が良い．例えば，要求入出力要求が 70%以下の場合，0.9

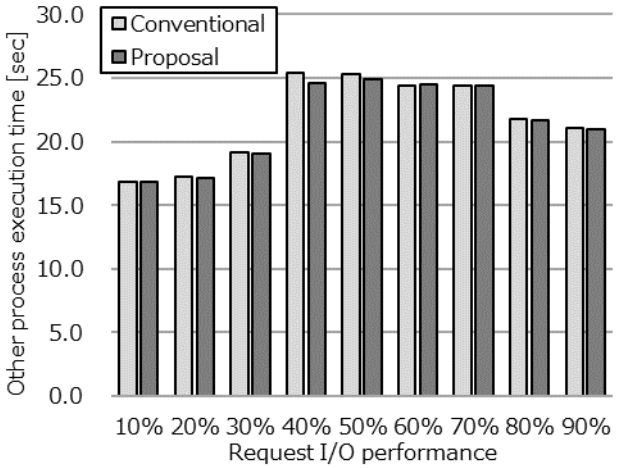


図 9　SSD 環境における平均の他プロセス実行時間

Figure 9　Average of Other Process Execution Time with SSD

～1.1 の遅延精度は，全体の 98%を占める．要求入出力性能が 90%においても，全体の 51%を占める．提案方式では，ビジーループによって遅延時間を監視するため，遅延精度が高い．要求入出力性能が高い場合，提案方式においても，遅延処理を実施しない場合が多くなる．これは，次の理由による．要求入出力性能が高い場合，平均の実 I/O 時間から算出する理想の入出力時間と実 I/O 時間の差が小さくなる．実 I/O 時間のばらつきによって，入出力要求の処理終了までの時間が長くなると，遅延処理の実施前にすでに理想の入出力時間を超過してしまい，遅延処理を実施しない．

以上より，従来方式は，指定した遅延時間通りに上手くプロセスを起床できない一方で，提案方式は，指定した遅延時間通りにプロセスを起床できることが分かる．

6.4 他プロセス実行時間

調整対象プロセスの他に 3 つのプロセスを動作させた場合における 3 つのプロセスの実行時間を図 9 に示す．図 9 より，従来方式と提案方式の他プロセス実行時間にほとんど差がないことが分かる．例えば，要求入出力性能が 60%のとき，従来方式での他プロセス実行時間は 24.4 秒である一方で，提案方式では 24.5 秒である．これは，提案方式は，0.1 ミリ秒ごとに実行可能状態の他プロセスが存在するかを判定し，ビジーループを終了することで，他プロセスにプロセッサを割り当てられるためである．

このように，提案方式は，他プロセスの状態に応じてビジーループの実施有無を制御することで，ビジーループによる他プロセスへの悪影響を抑えられることが分かる．

7. 関連研究

複数のプロセスが動作する環境において，重要なプロセスの入出力時間を保証する研究として，デッドライン制御がある．例えば，Earliest Deadline First(EDF)を基礎にして，SCAN-EDF [3]とリアルタイムシステム向けに拡張した

RBED [4]，デッドラインを超過した入出力要求を公平に処理する Fair-EDF [5]，および EDF と Deferrable Schedule を組み合わせた DS-EDF[6]がある．また，入出力デバイスの帯域を重要なプロセスに予約する YFQ [7]と SASLO[8]が提案されている．他にも，各プロセスに設定した重みに基づき，各プロセスの入出力要求を処理する Budget Fair Queueing(BFQ)[9]が提案されている．これらの研究では，重要なプロセスの入出力時間が一定時間以下になるように保証できる．

一方で，重要でないプロセスの入出力処理を制限することで，重要なプロセスの入出力時間の増加を防ぐ研究として文献[10-12]がある．これらのスケジューラは，入出力要求を大量に発行するプロセスが他プロセスの入出力時間を長大化させることを防ぐ．

これらの研究では，重要なプロセスの最長の入出力時間を一定以下に保証することができる．しかしながら，他プロセスに関わらず，重要なプロセスを一定の速度に保つ研究はなされてない．

8. おわりに

入出力性能の調整法では，OS のスリープ機能を用いて調整対象プロセスの起床を遅延させ，入出力時間を調整する．スリープ機能は，タイマ割込の周期で経過時間を監視するため，本調整法が指定する遅延時間がタイマ割込の周期よりも短い場合，スリープ機能が時間通りに調整対象プロセスを起床できず，個々の入出力時間を上手く調整できない場合がある．

そこで，算出した遅延時間が短い場合には，ビジーループを用いて遅延処理を実行する選択的ビジーループ方式を提案した．提案方式では，ビジーループによって，他プロセスにプロセッサが割り当たらないことで，処理速度が低下する悪影響を抑えるために，一定時間ごとに実行可能状態のプロセスが有無を判定し，当該プロセスが存在する場合にはビジーループを終了する．

評価により，提案方式は，実 I/O 時間が短い SSD 環境においても，個々の入出力処理において，指定した遅延時間通りに調整対象プロセスを起床でき，この結果，個々の入出力時間を精度良く調整できることを示した．さらに，他プロセスの処理時間への悪影響を抑えられることを示した．

謝辞 本研究の一部は、JSPS KAKENHI 18K11244 による。

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，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.03, pp.159-170, March 2020.

[3] A. L. N. Reddy, J. Wyllie and K. B. R. Wijayaratne, “ Disk scheduling in a multimedia I/O system, ” ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM), Vol.1, Issue 1, pp.37-59, Feb. 2005.

[4] A. Povzner, T. Kaldewey, S. Brandt, R. Golding, T. M. Wong, and C. Maltzahn, “Efficient guaranteed disk request scheduling with Fahrrad,” Proceedings of Third ACM European Conference on Computer Systems (EuroSys ’08), pp.13-25, Apr. 2008.

[5] Y. Peng and P. Varman, “Fair-EDF: a latency fairness framework for shared storage systems,” Proceedings of the 11th USENIX Conference on Hot Topics in Storage and File Systems (HotStorage'19), pp1-8, July 2019.

[6] S. Han, D. Chen, M. Xiong, K. Lam, A. K. Mok and K. Ramamritham, “Schedulability Analysis of Deferrable Scheduling Algorithms for Maintaining Real-Time Data Freshness,” IEEE Transactions on Computers, Vol.63, No.4, pp.979-994, Apr. 2014.

[7] J. Bruno, J. Brustoloni, E. Gabber, B. Ozden and A. Silberschatz, “Disk scheduling with quality of service guarantees,” Proceedings of IEEE International Conference on Multimedia Computing and Systems (ICMCS'99), Vol.2, pp.400-405, June 1999.

[8] N. Li, H. Jiang, D. Feng and Z. Shi, “Storage Sharing Optimization Under Constraints of SLO Compliance and Performance Variability,” IEEE Transactions on Services Computing, Vol.12, No.1, pp.58-72, Jan. 2019.

[9] P. Valente and F. Checconi, “High Throughput Disk Scheduling with Fair Bandwidth Distribution,”IEEE Transactions on Computers, vol.59, no.9, pp.1172-1186, Sep. 2010.

[10] Y. Wu, B. Jia and Z. Qi, “IO QoS: A New Disk I/O Scheduler Module with QoS Guarantee for Cloud Platform,” Proceedings of 2012 4th International Symposium on Information Science and Engineering (ISISE'12), pp.441-444, Dec. 2012.

[11] Z. Yang, H. Fang, Y.Wu, C. Li, B. Zhao and H. H. Huang, “Understanding the effects of hypervisor I/O scheduling for virtual machine performance interference,” Proceedings of 4th IEEE International Conference on Cloud Computing Technology and Science Proceedings (CloudCom'12), pp.34-41, Dec. 2012.

[12] X. Ding, A. Xiong and C. Yang, “Optimation of Xen scheduler for multitasking,” Proceedings of 4th International Conference on Software Engineering and Service Science (ICSESS'13), pp.754-757, May 2013.