Vol.2015-ARC-215 No.15
Vol.2015-OS-133 No.15
2015/5/27

AnT オペレーティングシステムにおける OS サーバ間通信での優先度継承による優先度逆転の抑制法

鴨生 悠冬[1]　山内 利宏[1]　谷口 秀夫[1]

概要：プロセスの優先度に基づくスケジュールでは，優先度逆転の現象を抑制することが求められる．特に，マイクロカーネル構造 OS は，OS 機能をプロセスとして実現する．このため，優先度逆転は OS 処理にも影響を与える．そこで，本稿では，OS サーバ間通信時に AP プロセスの優先度を伝達し，その値を OS サーバ間で継承することにより，優先度逆転を抑制する方法を提案する．また，優先度逆転の抑制効果とオーバヘッドの観点から提案手法を評価する．

1. はじめに

計算機で様々なサービスを提供するために，オペレーティングシステム（以降，OS) はサービスを実現するプロセスの実行を制御している．このため，OS のプロセス実行制御機能，特にスケジュール機能は，サービスの要望に則した実行制御を行う必要がある．

多くのスケジュール機能は，プロセスの優先度に基づくスケジュールを行う．この場合，サービスの優先度に合わせるため，サービスを実現するプロセスの優先度に従ったスケジュール制御が必要であり，プロセス実行時の優先度逆転の現象を極力抑制することが求められる．ここで優先度逆転とは，低優先度のプロセスがプロセッサを解放するまで高優先度のプロセスが処理を実行できず，2 つのサービスの優先度が逆転する状態である．したがって，優先度逆転が発生すると，必要とされる優先度に合わせたサービス提供に支障をきたしてしまう．特に，マイクロカーネル構造 OS[1][2][3] の場合，OS 機能を OS サーバとしてプロセスで実現するため，OS 処理にも優先度逆転を生んでしまう．

我々は，マイクロカーネル構造を有する ***AnT*** オペレーティングシステム（An operating system with adaptability and toughness）（以降，***AnT***）[4] を開発している．***AnT*** では，サーバプログラム間通信機構により OS サーバを含むプロセス間で処理依頼や処理結果の情報を授受する．

そこで，本稿では，AP プロセスからの処理依頼に基づく，OS サーバ間通信時に AP プロセスの優先度を伝達し，その値を OS サーバ間で継承することにより，優先度逆転を抑制する方法を提案する．また，既存手法と提案手法を比較評価する．

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

2. *AnT* オペレーティングシステム

AnT は，プロセス間の通信を高速化するため，コア間通信データ域（ICA: Inter-core Communication Area）を利用したデータ複写レス授受機能を有する．ICA の特徴として以下の 3 つがある．

（1）ページを単位とし，n ページ分の領域の確保と解放
（2）確保した領域 (n ページ) の実メモリ連続の保証
（3）2 仮想空間での領域の貼り替え

ICA は，ページを最小単位として管理される領域であり，ICA へのアクセスは，プロセスごとの仮想空間のページテーブルを通して行われる．ここで，ページテーブルへの書き込みを貼り付けと呼び，ページテーブルからの削除を剥がしと呼ぶ．また，貼り替えとは，剥がしと貼り付けを行うことを意味する．プロセス間の複写レスでのデータ授受の様子を**図 1** に示す．ICA を利用したプロセス間でのデータ授受は，授受するデータを格納した ICA をデータ授受元プロセスの仮想空間から剥がし，データ授受先プロセスの仮想空間へ貼り付けることで行われる．

サーバプログラム間通信 [4] の基本機構を**図 2** に示す．ICA を利用することにより，プロセス間でデータ複写レスでの通信を実現している．具体的には，OS サーバへ渡す引数や通信制御の情報（以降，依頼情報）を制御用の ICA（以降，制御用 ICA）に格納し，扱うデータをデータ用の ICA（以降，データ用 ICA）に格納する．カーネルは，プロセスごとに通信のための依頼キューと結果キューを持

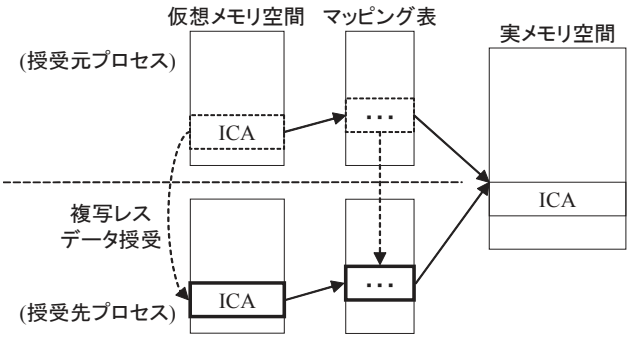

図 1 複写レスデータ授受の様子

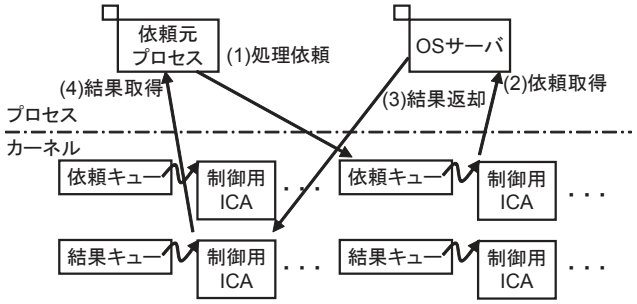

図 2 サーバプログラム間通信の基本機構

ち，いずれのキューも FIFO に従う．また，同期型と非同期型の通信インタフェースを同様な形式で提供し，両インタフェースを選択して利用できる．基本的な通信の流れを以下に述べる．

（1）依頼元プロセスが処理依頼を行うと，カーネルは OS サーバの依頼キューに依頼情報を格納した制御用 ICA を登録し，依頼元プロセスから制御用 ICA を剥がす．

（2）カーネルは，依頼キューから依頼情報を格納した制御用 ICA を取得し，OS サーバに制御用 ICA を貼り付ける．その後，OS サーバは，取得した依頼の処理を実行する．

（3）OS サーバが結果返却を行うと，カーネルは依頼元プロセスの結果キューに結果情報を格納した制御用 ICA を登録し，OS サーバから制御用 ICA を剥がす．

（4）カーネルは，結果キューから結果情報を格納した制御用 ICA を取得し，依頼元プロセスに制御用 ICA を貼り付ける．その後，依頼元プロセスは，結果受取を行う．

スケジュール機能は，プロセスの優先度に基づいて行っている．また，タイムスライスとプリエンプション機能を持つ．なお，OS サーバの優先度は，AP プロセスの優先度より高く設定される．

3. 優先度逆転を抑制する制御法

3.1 優先度逆転の問題

AnT では，OS サーバの優先度を AP プロセスの優先度より高く設定する．これは，OS サーバが OS 機能を有するためである．また，プロセスの優先度は固定である．さらに，OS サーバは処理依頼や処理結果の情報取得を FIFO 順に行う．このため，以下の 2 つの優先度逆転の問題が発生する．

（問題 1）他の低優先度の AP プロセスの依頼により，高優先度の AP プロセスの依頼の実行が遅延し，高優先度の AP プロセスの応答時間が長くなる．

この問題の例として，OS サーバの依頼キューに低優先度と高優先度の AP プロセスの依頼が順に登録されている場合を挙げる．OS サーバは，先に依頼キューに登録された低優先度の AP プロセスの依頼を実行し，その後，高優先度の AP プロセスの依頼を実行する．すなわち，高優先度の AP プロセスの依頼の実行は，OS サーバによる低優先度の AP プロセスの依頼の実行時間だけ遅延する．

（問題 2）低優先度の AP プロセスの依頼を OS サーバが処理する間，高優先度の AP プロセスの実行開始が遅延する．

この問題の例として，OS サーバの依頼キューに低優先度の AP プロセスの依頼が登録されており，高優先度の AP プロセスと OS サーバが READY 状態である場合を挙げる．OS サーバの優先度は AP プロセスより優先度が高いため，初めに OS サーバが実行を開始する．OS サーバは低優先度の AP プロセスの依頼を実行し，依頼取得待ちにより WAIT 状態となる．その後，高優先度の AP プロセスは実行を開始する．すなわち，高優先度の AP プロセスの実行開始は，OS サーバによる低優先度の AP プロセスの依頼の実行時間だけ遅延する．

3.2 対処

前節で述べた（問題 1）と（問題 2）は，依頼取得を FIFO 順に行うこと，および OS サーバの優先度が固定であり AP プロセスの優先度より高いことに起因する．これらの要因に対して，以下の 2 つの対処を行う．

（対処 1）依頼元 AP プロセスの優先度に基づいた依頼取得

この対処により，高優先度の AP プロセスの依頼は，OS サーバの依頼キューより優先して取得される．すなわち，高優先度の AP プロセスの依頼の実行は，依頼キューに存在する他の低優先度の AP プロセスの依頼によって遅延されない．

（対処 2）依頼元 AP プロセスの優先度を OS サーバに継承

この対処により，依頼を実行中の OS サーバの優先度は，依頼元 AP プロセスと同じ優先度になる．すなわち，OS サーバが依頼を実行中であっても，実行中の依頼の依頼元 AP プロセスより高い優先度の AP プロセスは，実行権を奪い実行を開始できる．以降，依頼元 AP プロセスの優先度を OS サーバに継承することを優先度継承と呼ぶ．

表 1 各制御法の概要

通番	制御法	依頼取得順	OS サーバの優先度変更	
			依頼登録時	依頼取得時
(1)	固定優先度 FIFO（FPFIFO）	FIFO 順	なし	なし
(2)	固定優先度 ICA 優先度順（FPICA）	ICA 優先度順	なし	なし
(3)	可変優先度取得時変更（VPGET）	ICA 優先度順	なし	ICA 優先度に変更
(4)	可変優先度登録時変更（VPSET）	ICA 優先度順	ICA 優先度より OS サーバの優先度が低い場合，ICA 優先度に変更	ICA 優先度より OS サーバの優先度が高い場合，ICA 優先度に変更

3.3 実現方式

前節にて示した（対処 1）と（対処 2）の実現方式を述べる．

（対処 1）の実現には，依頼キューから取得する依頼を選択する際に，依頼キュー中の依頼の依頼元 AP プロセスの優先度を把握する必要がある．そこで，OS サーバ間通信時に授受する制御用 ICA に依頼元 AP プロセスの優先度を保持させる．以降，制御用 ICA に保持させた依頼元 AP プロセスの優先度を ICA 優先度と呼ぶ．

（対処 2）は，ICA 優先度を用いて OS サーバの優先度を変更することで実現する．優先度継承の契機として，以下の 2 つがある．

（契機 1）OS サーバの依頼取得時

（契機 2）OS サーバへの依頼登録時

なお，（対処 2）を行うと，同じ優先度の AP プロセスと OS サーバが存在することになる．この場合，OS サーバを優先する．これは，OS サーバが OS 機能を有するためである．

3.4 制御法

既存手法と優先度逆転を抑制する制御法を**表 1** に示し，以下に説明する．表 1 における依頼取得順は，依頼キューから依頼をどのように取得するかを示す．表 1 における OS サーバの優先度変更は，依頼取得時と依頼登録時に OS サーバの優先度をどのように変更するかを示す．

（1）固定優先度 FIFO 法（FPFIFO）は，既存手法である．プロセスの優先度を固定とし，OS サーバの優先度を AP プロセスの優先度よりも高く設定する．依頼取得を FIFO 順に行う．

（2）固定優先度 ICA 順法（FPICA）は，（対処 1）を行った優先度逆転を抑制する制御法である．プロセスの優先度に関しては FPFIFO と同様であるが，依頼取得を ICA 優先度順に行う．

（3）可変優先度取得時変更法（VPGET）は，（対処 1）と（対処 2）を行った優先度逆転を抑制する制御法である．依頼取得を ICA 優先度順に行うとともに，依頼取得時に OS サーバの優先度を ICA 優先度に変更する．

（4）可変優先度登録時変更法（VPSET）は，（対処 1）と（対処 2）を行った優先度逆転を抑制する制御法である．依頼取得を ICA 優先度順に行うとともに，依頼登録時と依頼取得時に OS サーバの優先度を ICA 優先度に変更する．ただし，依頼登録時の OS サーバの優先度変更は，ICA 優先度が OS サーバの優先度よりも高い場合に行う．これは，低優先度の AP プロセスの処理依頼によって，OS サーバの優先度が不用意に下がることを防ぐためである．また，依頼取得時の OS サーバの優先度変更は，ICA 優先度が OS サーバの優先度よりも低い場合に行う．これは，OS サーバの優先度を低下させる契機として OS サーバの優先度変更を行うためである．

3.5 動作例

低優先度，中優先度，および高優先度の AP プロセスを以降，AP_l，AP_m，および AP_h と呼ぶ．AP_h の OS サーバへの同期処理依頼と OS サーバの依頼取得について，各制御法の処理流れを**図 3** に示し，以下に説明する．なお，OS サーバの依頼キューには，AP_l の同期依頼が登録されている．また，AP_l は WAIT 状態，AP_h は READY 状態，OS サーバは他の AP_l の返却不要の非同期依頼の処理による WAIT 状態である．

（1）FPFIFO では，以下のように動作する．

（i）AP_h は，OS サーバに同期処理依頼を行い，OS サーバの依頼キューの末尾に依頼を登録する．

（ii）OS サーバは，FIFO 順に依頼を取得するため，依頼キューの先頭の AP_l の依頼を取得する．

（2）FPICA では，以下のように動作する．

（i）AP_h は，OS サーバに同期処理依頼を行い，OS サーバの依頼キューの末尾に依頼を登録する．

（ii）OS サーバは，ICA 優先度順に依頼を取得するため，依頼キューの中で最も ICA 優先度の高い AP_h の依頼を取得する．

（3）VPGET では，以下のように動作する．

（i）AP_h は，OS サーバに同期処理依頼を行い，OS サーバの依頼キューの末尾に依頼を登録する．

（ii）OS サーバは，ICA 優先度順に依頼を取得するため，

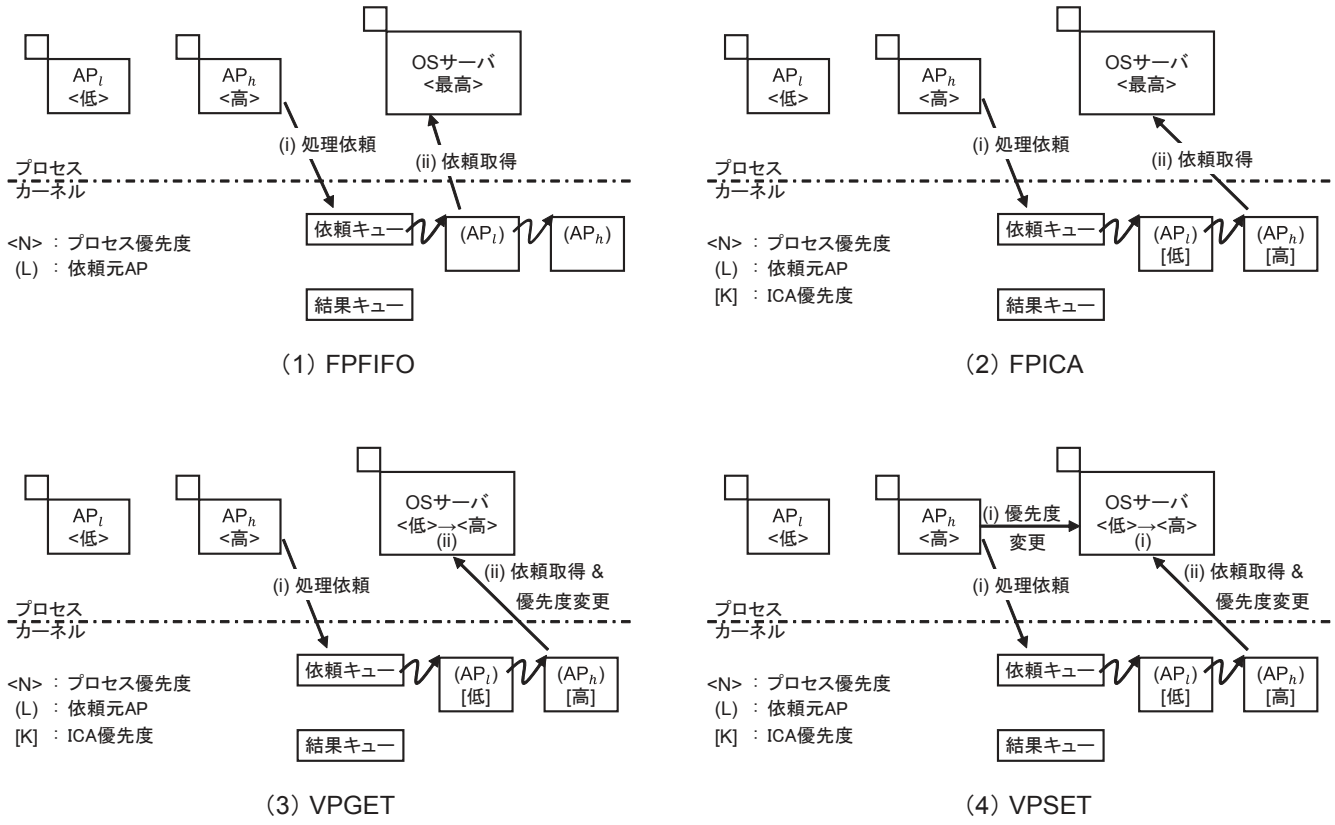

図 3 各制御法の動作例

依頼キューの中で最も ICA 優先度の高い AP_h の依頼を取得する．この時，OS サーバの優先度を ICA 優先度に変更する．

（4）VPSET では，以下のように動作する．

（i）AP_h は，OS サーバに同期処理依頼を行い，OS サーバの依頼キューの末尾に依頼を登録する．この時，ICA 優先度より OS サーバの優先度が低いため，OS サーバの優先度を ICA 優先度に変更する．

（ii）OS サーバは，ICA 優先度順に依頼を取得するため，依頼キューの中で最も ICA 優先度の高い AP_h の依頼を取得する．この時，ICA 優先度より OS サーバの優先度が高くないため，OS サーバの優先度を変更しない．

3.6 優先度逆転の抑制効果とオーバヘッド

各制御法の優先度逆転の抑制効果および各制御法のオーバヘッドは，それぞれ以下の関係となる．

＜優先度逆転の抑制効果＞

FPFIFO < FPICA < VPGET < VPSET

＜オーバヘッド＞

FPFIFO < FPICA < VPGET < VPSET

各制御法の優先度逆転の抑制効果が上記の関係となる理由を以下に示す．

（1）FPFIFO < FPICA

FPICA は，依頼取得を ICA 優先度順に行うため，AP_h の依頼を優先して実行できる．このため，OS サーバが処理中に WAIT 状態になる場合，FPFIFO に比べ，高優先度の AP プロセスが早期に結果受取を行える．

（2）FPICA < VPGET

FPICA では，OS サーバが処理中に WAIT 状態にならない場合，OS サーバが全ての依頼を実行するまで，高優先度の AP プロセスが結果取得を行えない．一方，VPGET は，依頼取得時に依頼元プロセスの優先度を継承する．このため，OS サーバが処理中に WAIT 状態にならない場合でも，高優先度の AP プロセスの結果取得は，OS サーバによる中優先度または低優先度の AP プロセスの依頼取得後に行われるため，FPICA に比べ，優先度逆転を抑制できるといえる．

（3）VPGET < VPSET

VPGET は, 依頼取得時に優先度継承を行う．一方, VPSET は，依頼登録時に優先度継承を行う．このため，VPSET は，依頼登録後から依頼取得までの間に生じる優先度逆転を抑制できる．

優先度逆転の抑制のための処理が増えるほど，オーバヘッドは増加するため，オーバヘッドは，上記の関係となる．

Vol.2015-ARC-215 No.15
Vol.2015-OS-133 No.15
2015/5/27

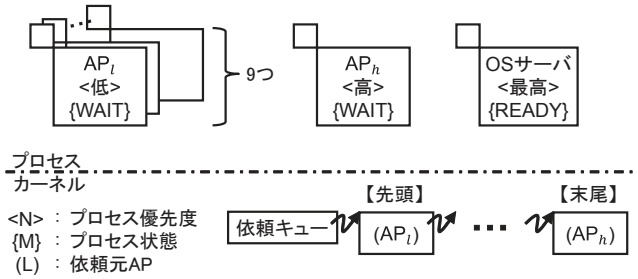

図 4 各制御法の測定前の様子

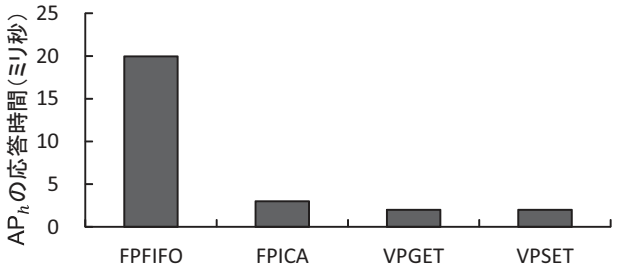

図 5 AP$_h$ の応答時間

4. 評価

4.1 測定環境

Pentium4（2.8Ghz）を搭載した計算機を利用し，各制御法を ***AnT*** に実現した．各制御法の優先度逆転の抑制効果および各制御法のオーバヘッドについて，以降に述べる．なお，以降の測定には RDTSC 命令を使用し，測定結果は 100 回実行した場合の平均実行時間である．また，処理依頼の際に，OS サーバへ渡す引数，および戻り値はなしとする．

4.2 優先度逆転の抑制効果

4.2.1 各制御法の比較

高優先度プロセス AP$_h$ が結果受取を行うまでの時間を比較した．測定前の様子を**図 4** に示し，以下に説明する．低優先度の 9 つの AP$_l$，高優先度の 1 つの AP$_h$，および最高優先度の OS サーバが存在し，OS サーバの依頼キューには 9 つの AP$_l$ と 1 つの AP$_h$ の同期依頼が順に登録されている．AP$_l$ と AP$_h$ は結果受取待ちのため WAIT 状態であり，OS サーバは READY 状態である．OS サーバは，依頼処理として PU 処理 1 ミリ秒，WAIT 処理 1 ミリ秒を行う．既存手法と比較するため，OS サーバの優先度は最高優先度とする．測定区間は，OS サーバの依頼取得の開始から，AP$_h$ が結果受取を行うまでである．以降，この測定区間の時間を AP$_h$ の応答時間と呼ぶ．抑制効果は，応答時間が短いほど高い．

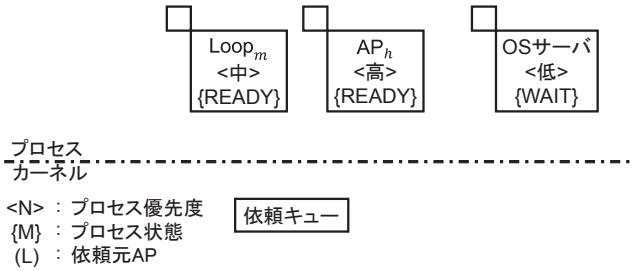

図 6 OS サーバより高優先度の AP が存在する場合の測定前の様子

なお，PU 処理は，特定領域のインクリメントを繰り返すプロセッサ処理であり，WAIT 処理は，指定時間だけ実行権を放棄するシステムコールを用いた待ち処理である．

測定結果を**図 5** に示す．図 5 より，以下の 2 つのことが分かる．

（1）FPICA，VPGET, および VPSET の AP$_h$ の応答時間は，FPFIFO に比べ，非常に短い．

これは, 依頼取得順が異なるためである．FPICA, VPGET, および VPSET の OS サーバは，（対処 1）により ICA 優先度順に依頼を取得するため，初めに AP$_h$ の依頼を実行する．一方，FPFIFO の OS サーバは，FIFO 順に依頼を取得するため，最後に AP$_h$ の依頼を実行する．この差が AP$_h$ の応答時間の差として表れている．上記から，FPICA, VPGET, および VPSET の AP$_h$ の依頼は，他の AP$_l$ の依頼によって遅延していないことが分かる．したがって，FPICA，VPGET, および VPSET は，3.1 節の（問題 1）を解決している．

（2）VPGET と VPSET の AP$_h$ の応答時間は，FPICA に比べ，短い．

これは，VPGET と VPSET の OS サーバの優先度が，依頼取得時に ICA 優先度に変更されるためである．VPGET と VPSET の OS サーバは，AP$_l$ の依頼取得時に低優先度となり，AP$_h$ に切り替わる．一方，FPICA の OS サーバは常に最高優先度であるため，AP$_l$ の依頼の WAIT 処理開始時に，AP$_h$ に切り替わる．この差が AP$_h$ の応答時間の差として表れている．上記から，VPGET と VPSET の AP$_h$ の実行開始は，OS サーバの AP$_l$ の依頼の実行によって遅延していないことが分かる．したがって，VPGET と VPSET は，3.1 節の（問題 2）を解決している．

4.2.2 可変優先度制御法の比較

ここでは，可変優先度制御法において，依頼登録時に優先度継承を行う場合とそうでない場合の比較を行う．測定前の様子を**図 6** に示し，以下に説明する．

2 ミリ秒の PU 処理を行う中優先度の AP プロセスを Loop$_m$ と呼ぶ．Loop$_m$，高優先度の AP$_h$，および最高優先度の OS サーバが存在し，Loop$_m$ と AP$_h$ は READY 状態であり，OS サーバは依頼取得待ち（WAIT 状態）であ

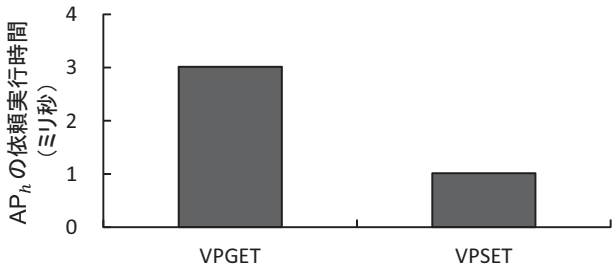

図 7 AP_h の依頼実行時間

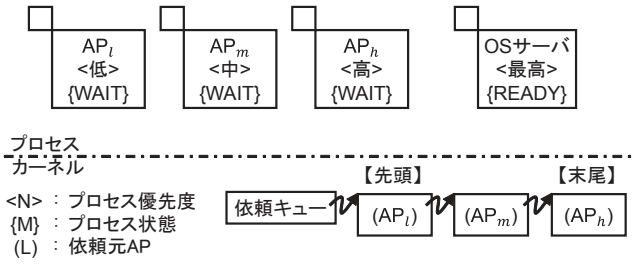

図 8 オーバヘッドの測定前の様子

る．AP_h は，OS サーバに同期処理依頼を行う．OS サーバは，依頼処理として PU 処理 1 ミリ秒を行う．測定区間は，AP_h が OS サーバに処理依頼を行い，結果受取を行うまでである．以降，この測定区間の時間を AP_h の依頼実行時間と呼ぶ．抑制効果は，依頼実行時間が短いほど高い．

測定結果を**図 7** に示す．図 7 から，VPGET の AP_h の依頼実行時間は，VPSET に比べ，2 ミリ秒長いことが分かる．これは，VPGET の OS サーバは $Loop_m$ によって依頼の実行を妨害されるが，VPSET の OS サーバは妨害されないためである．VPGET の OS サーバの依頼取得までの処理流れを以下に示す．

（1）AP_h は OS サーバに同期処理依頼を行う．

（2）OS サーバより優先度の高い $Loop_m$ に切り替わり，$Loop_m$ は 2 ミリ秒の PU 処理を行う．

（3）$Loop_m$ から OS サーバに切り替わり，AP_h の依頼を取得する．この時，優先度継承により，OS サーバは高優先度となる．

VPSET の OS サーバの依頼取得までの処理流れを以下に示す．

（1）AP_h は OS サーバに依頼を登録する．この時，優先度継承により，OS サーバの優先度は高優先度となる．

（2）$Loop_m$ より優先度の高い OS サーバに切り替わり，OS サーバは AP_h の依頼を取得する．

4.3 オーバヘッド

4.3.1 測定内容

複数の AP プロセスが結果受取を行うまでの時間を比較した．測定前の様子を**図 8** に示し，以下に説明する．

低優先度の AP_l，中優先度の AP_m，および高優先度の AP_h が存在し，OS サーバの依頼キューには AP_l，AP_m，および AP_h の同期依頼が順に登録されている．AP_l，AP_m，および AP_h は結果受取待ちのため WAIT 状態であり，OS サーバは READY 状態である．OS サーバは，依頼を受取ると，依頼内容を実行せずに，結果を返却する．各 AP プロセスは，結果受取後，直ちに終了する．既存手法と比較するため，OS サーバの優先度は最高優先度とする．測定区間は，OS サーバの依頼取得の開始から，全ての AP プロセスが結果受取を行うまでである．以降，OS サーバの依頼取得の開始から，全ての AP プロセスの応答時間を全 AP プロセスの応答時間と呼ぶ．オーバヘッドは，FPFIFO に対する提案手法の全 AP プロセスの応答時間の長大化とする．

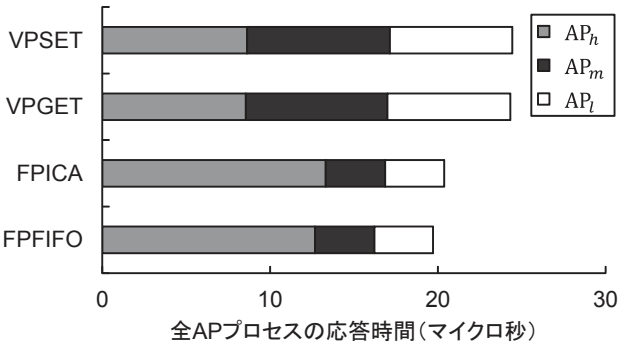

図 9 全 AP プロセスの応答時間

4.3.2 結果と考察

測定結果を**図 9** に示す．凡例は全 AP プロセスの応答時間の内訳を表し，AP_h，AP_m，および AP_l はそれぞれ，OS サーバの依頼取得の開始から AP_h が結果受取を行うまでの時間，AP_h が結果受取を行ってから AP_m が結果受取を行うまでの時間，および AP_m が結果受取を行ってから AP_l が結果受取を行うまでの時間を表す．図 9 より，以下の 4 つのことが分かる．

（1）VPGET と VPSET の OS サーバの依頼取得の開始から AP_h が結果受取を行うまでの時間は，FPFIFO と FPICA に比べ，短い．

これは，AP_h の結果受取を行うまでに OS サーバが行う処理が異なるためである．具体的には，FPFIFO と FPICA の AP_h の結果受取は，OS サーバによる全ての AP の依頼の実行と結果返却後に行われる．一方，VPGET と VPSET の AP_h の結果受取は，OS サーバによる AP_h の依頼の実行と結果返却，AP_m の依頼の取得後に行われる．すなわち，VPGET と VPSET では，AP_h の結果受取までに，AP_l の依頼の取得と結果返却，AP_m への結果返却を実行しない．

（2）FPFIFO と FPICA の全 AP プロセスの応答時間は，

VPGET と VPSET に比べ，短い．
これは，測定区間中の FPFIFO と FPICA のコンテキストスイッチの回数が VPGET と VPSET に比べ，少ないためである．前述したように FPFIFO と FPICA の AP_h の結果受取までに，OS サーバは全ての AP の依頼の実行と結果返却を行う．このため，AP_h の結果受取後，AP_m と AP_l が順に結果受取を行う．すなわち，測定区間中の FPFIFO と FPICA のコンテキストスイッチの回数は，3 回である．一方，VPGET と VPSET の AP_h の結果受取までに，OS サーバは AP_h の依頼の実行と結果返却，AP_m の依頼の取得を行う．このため，AP_m の結果受取は，OS サーバによる AP_m の依頼の実行と結果返却，AP_l の依頼の取得後に行われる．同様にして，AP_l も結果受取を行う．すなわち，測定区間中の VPGET と VPSET のコンテキストスイッチの回数は，5 回であり，FPFIFO と FPICA に比べ，2 回多い．
（3）FPICA の全 AP プロセスの応答時間は，FPFIFO に比べ，わずかに長い．
これは，FPFIFO と FPICA の依頼取得時のキュー操作の違いによるものである．FPFIFO では，依頼キューの先頭から依頼を取得する．一方，FPICA では，依頼キューの中から最も ICA 優先度の高い依頼を探索し取得する．すなわち，FPICA は，FPFIFO に比べ，最も ICA 優先度の依頼を探索する時間だけ，全 AP プロセスの応答時間が長い．
（4）VPGET の全 AP プロセスの応答時間は，VPSET に比べ，わずかに短い．
これは，VPGET と VPSET の OS サーバの優先度変更の回数の違いによるものである．VPGET では，依頼取得時に OS サーバの優先度変更を行う．一方，VPSET では，依頼取得時に加え，依頼登録時にも OS サーバの優先度変更を行う．すなわち，VPGET は，VPSET より OS サーバの優先度変更の回数が 1 回少ない．

4.3.3 分析

前項より，優先度逆転を抑制する制御法のオーバヘッドには，以下の 3 つが存在することがわかる．
（1）キュー中の最も ICA 優先度の高い依頼を探索するオーバヘッド
（2）OS サーバの優先度変更によるオーバヘッド
（3）コンテキストスイッチの回数の増加によるオーバヘッド
（1）のオーバヘッドは, FPICA, VPGET, および VPSET に存在する．このオーバヘッドは，キューに登録されている依頼数に比例する．このため，キューに登録されている依頼数を変化させて，依頼の探索時間を測定した．測定結果を**図 10** に示す．測定結果から，OS サーバの依頼キューに登録されている依頼数が 1 増加するごとに探索時間が約 13 クロック増加していることが分かる．ここで，依頼取得の実行時間は約 4800 クロックであり，キューに登録されている依頼数を 30 とした際のオーバヘッドは約 480 クロックである．したがって，キューに登録されている依頼数が 30 以下であれば，オーバヘッドの約 10%以下となる．

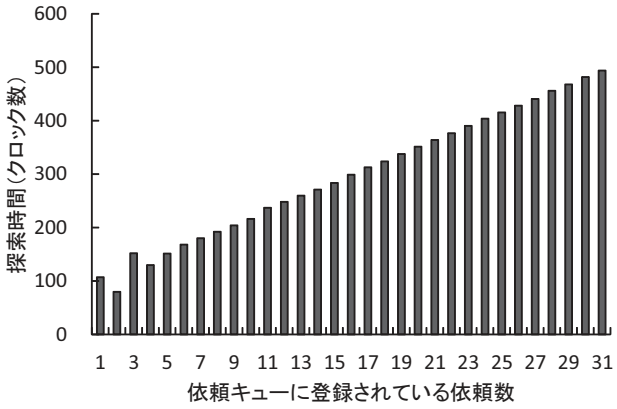

図 10 依頼の探索時間

（2）のオーバヘッドは，VPGET，および VPSET に存在する．VPGET では OS サーバの依頼取得時，VPGET では依頼登録時，依頼取得時，および結果返却時に OS サーバの優先度変更を行う場合がある．優先度変更の実行時間は約 0.17 マイクロ秒である．ここで，依頼登録，依頼取得，および結果返却の実行時間は，それぞれ約 1.0 マイクロ秒，約 1.7 マイクロ秒，約 1.6 マイクロ秒である．したがって，オーバヘッドは，約 10〜17%である．
（3）のオーバヘッドは，VPGET，および VPSET に存在する．VPGET では OS サーバの依頼取得時，VPGET では依頼登録時，依頼取得時，および結果返却時にコンテキストスイッチが発生する場合がある．コンテキストスイッチが発生すると，OS サーバからプロセスに，プロセスから OS サーバにコンテキストスイッチを行うため，コンテキストスイッチの回数が最大 2 回増加する．コンテキストスイッチの実行時間は，約 1.6 マイクロ秒であるため，オーバヘッドは最大 3.2 マイクロ秒である．このため，オーバヘッドは大きいといえる．
最もオーバヘッドが大きい制御法は VPSET である．OS サーバの依頼キューに登録されている依頼数を 30 とした際，VPSET のオーバヘッドの最大値は，約 480 クロック (約 0.14 マイクロ秒)+約 0.17 マイクロ秒*3+約 3.2 マイクロ秒*3=約 10.3 マイクロ秒である．したがって，OS サーバの固有処理の時間が 1 ミリ秒程度であれば，VPSET のオーバヘッドは 1%以下になる．

5. 関連研究

本稿の提案手法は，OS サーバ間通信における優先度逆転を抑制する．具体的には，OS サーバ間通信時に AP プロセスの優先度を伝達することで，依頼元 AP プロセスの

優先度を考慮した依頼取得を可能にする．また，依頼登録時と依頼取得時に伝達した優先度を OS サーバ間で継承している．

文献 [5] では，優先度逆転の抑制により，実時間性を向上させている．具体的には，実時間処理のためのプロセス間通信のポートを作成し，依頼元プロセスの優先度順の依頼取得と依頼取得時と登録時の優先度継承を実現している．しかし，本稿の FPICA に値する手法の評価を行っていない．また，本稿の VPGET や VPSET に値するオーバヘッドの要因に関して考察していない．

文献 [6] では，共有資源におけるデッドロックの対処を参考に，代替資源を用意することにより優先度逆転に対処できることを示している．具体的には，優先度の異なる OS サーバを複数用意するか，OS サーバの処理実行中に他の依頼を実行可能にすることにより優先度逆転に対処できる．

文献 [7] では，プロセス間通信における，リソース不足，なりすまし，および信頼できない AP プロセスと OS サーバ間の通信の脅威を示し，プロセス間通信の基盤設計によりこの問題に対処できることを示している．一方で，プロセス間通信の基盤設計だけでは，優先度逆転の抑制やデッドロックは回避できないとしている．

文献 [8] では，実時間システムで用いるバリアプロトコルにおける優先度逆転を抑制する手法を提案している．この手法では，バリアに参加するスレッドの優先度を参加しているスレッドの中で最も高い優先度に一時的に向上させることにより優先度逆転を抑制している．

6. おわりに

AnT における OS サーバ間通信時の優先度逆転の 2 つの問題を述べた．1 つ目は，他の低優先度の AP プロセスの依頼により，高優先度の AP プロセスの依頼の実行が遅延し，高優先度の AP プロセスの応答時間が長くなる問題である．2 つ目は，低優先度の AP プロセスの依頼を OS サーバが処理する間，高優先度の AP プロセスの実行開始が遅延する問題である．また，これらの問題に対して 2 つの対処を述べた．1 つ目は，依頼元 AP プロセスの優先度に基づいた依頼取得である．2 つ目は，OS サーバ間通信時に依頼元 AP プロセスの優先度を OS サーバに継承させる方法である．

次に，これらの対処を実現する制御法として，固定優先度 ICA 優先度順法（FPICA），可変優先度取得時変更法（VPGET），および可変優先度登録時変更法（VPSET）を提案した．

また，既存手法と提案手法を優先度逆転の抑制効果とオーバヘッドの観点から比較し，評価した．提案手法は，優先度逆転を抑制でき，その効果は，FPICA，VPGET，VPSET の順（FPICA < VPGET < VPSET）に高くなる．すなわち，VPSET が優先度逆転を最も抑制できる制御法である．しかし，提案手法のオーバヘッドは，FPICA，VPGET，VPSET の順（FPICA < VPGET < VPSET）に大きくなり，VPSET のオーバヘッドが最も大きい．VPSET のオーバヘッドには，キュー中の最も高い依頼を探索するもの，OS サーバの優先度変更によるもの，およびコンテキストスイッチの回数の増加によるものがある．また，VPSET のオーバヘッドを示し，OS サーバの固有処理の時間が 1 ミリ秒程度であれば，VPSET のオーバヘッドは 1%以下であることを述べた．

今後の課題として，実サービスにおける提案手法の評価がある．

参考文献

[1] Liedtke, J.: Toward real microkernels，Communications of the ACM, Vol.39, No.9, pp.70–77 (1996).
[2] Tanenbaum, A.S., Herder, J.N., and Bos, H.: Can we make operating systems reliable and secure?, IEEE Computer Magazine, Vol.39, No.5, pp.44–51 (2006).
[3] Black, D.L., Golub, D.B., Julin, D.P., Rashid, R.F., Draves, R.P., Dean, R.W., Forin, A., Barrera, J., Tokuda, H., Malan, G., and Bohman, D.: Microkernel operating system architecture and mach, Journal of Information Processing, Vol.14, No.4, pp.442–453 (1992).
[4] 岡本 幸大，谷口 秀夫: ***AnT*** オペレーティングシステムにおける高速なサーバプログラム間通信機構の実現と評価, 電子情報通信学会論文誌 (D), Vol.J93-D, No.10, pp.1977–1987 (2010).
[5] Kitayama, T., Nakajima, T., and Tokuda, H.: RT-IPC:An IPC Extension for Real-Time Mach, USENIX Microkernels and Other Kernel Architectures Symposium, pp.91–104 (1993).
[6] Levine, G.: Priority Inversion with Fungible Resources, ACM SIGAda Ada Letters, Vol.31, No.2, pp.9–14 (2011).
[7] Herder, J.N., Bos, H.,Gras, B., Homburg, P., and Tanenbaum, A.S.: Countering IPC Threats in Multiserver Operating Systems, In Proc.14th PRDC, pp112–121 (2008).
[8] Röck, H., Auerbach, J., Kirsch, C.M., Bacon, D.F.: Avoiding unbounded priority inversion in barrier protocols using gang priority management, Proceedings of the 7th International Workshop on Java Technologies for Real-Time and Embedded Systems, pp.70–79 (2009).