1H-3 データ部の初期化によるプロセスの再起動機構

田端 利宏　　谷口 秀夫　　牛島 和夫
九州大学 大学院システム情報科学研究科

1 はじめに

プロセスの生成と消滅処理は、オペレーティングシステム（以降、OSと略す）の処理の中でも負荷の大きな処理である。例えば、makeプログラムでは、コンパイラが繰り返し実行されるため、プロセスの生成と消滅が繰り返し起こる。このため、プロセスの生成と消滅を高速化することは重要である。***Tender***[1]では、プロセスがプロセッサを利用できる程度を持つ資源「演算」を導入した。本稿では、資源「演算」を利用し、プロセスのデータ部を初期化して、高速にプロセスを再起動する機構を提案し、実測評価した結果について報告する。

2 *Tender*オペレーティングシステム

2.1 資源の独立化機構

TenderではOSの操作する対象を資源として、分離し独立化した。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を資源毎に分離し、共有プログラムを排除した。また、各資源の管理情報も資源毎に分離し、各資源の管理表の間のポインタを禁止した。このように、資源の分離と独立化を行なうことで、資源の事前生成や保留により、資源の作成や削除を伴う処理を高速化している[2]。更に、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である。

2.2 資源「演算」

Tenderでは、プロセッサの割当単位を資源として分離・独立化し、「演算」と名付けた[3]。資源「演算」とは、プロセッサを利用でき、プログラムを実行できる程度 (演算の程度と名付ける) を持つ資源である。演算をプロセスと関連付けることにより、プロセスの実行が可能になる。つまり、利用者は、プロセスと演算を生成し、プロセスに演算を関連付けることで、演算が持つ演算の程度に応じてプロセスを走行させることができる。このため、演算の程度を変更することで、プロセッサを利用できる程度を変更できる。これにより、プロセスを生成し、演算の関連付けを行なわないことによるプロセスの作り置きや、演算の関連付けと解除によるプロセスの実行の停止と再開が容易に実現できる。

Mechanism of Restarting a Process by Initializing Data Segments
Toshihiro TABATA, Hideo TANIGUCHI and Kazuo USHIJIMA
Graduate School of Information Science and Electrical Engineering, Kyushu University
Email:tabata,tani,ushijima@csce.kyushu-u.ac.jp

3 プロセスの再起動機構

プロセスの再起動とは、任意の時点でプロセスの実行を停止し、プロセスのデータ部を初期化し、プロセスを最初から実行できる環境を作ることである。

3.1 プロセスの再起動の特徴と利点

プロセスの再起動を行うことにより、以下の特徴と利点がある。

(1) 同じプログラムを実行する場合に、プロセスを作り直さずに、再起動することで、高速に処理を開始することが可能となる。これは、同じプログラムを繰り返し実行する場合に有効である。

(2) プロセスを作り直すと、一度利用していた資源を解放しなければならないが、プロセスの再起動では、再起動前に利用していた資源を、再起動後に利用できる。これにより、一度生成した資源を繰り返し利用することが可能となる。

3.2 実現方式

プロセスの再起動は、プロセスの実行を停止が必要である。任意の位置でのプロセスの実行の停止とスケジューリング対象からの除外は、資源「演算」を導入により、容易に実現できる。次に、データ部を初期化しプロセスを最初から実行できる環境を構築するための処理を行う。この処理は、システムコールで実装した。プロセスの再起動のインタフェースを表1に示し、処理内容を説明する。プロセスの各部分（テキスト部、データ部、BSS部）のアドレス位置と大きさを取得する。仮想記憶空間を再起動プロセスのものに切替え、データ部とBSS部の内容を初期化し、プロセスに渡す引数をユーザスタックに複写する。最後に、仮想記憶空間を元のものに切替え、プロセス管理表のエントリを初期化する。

4 実測評価

プロセスの再起動、およびプロセスの生成と消滅の処理時間の測定を行った。プロセスAが、プロセスBを再起動、または生成、消滅する時間を測定した。プロセスBが実行を停止した状態で、プロセス再起動のシステムコールをプロセスAが発行し、終了するまで時間を測定した。プロセスの生成と消滅も同様に、プロセスAが実行を停止したプロセスBを消滅し生成する処理の時間を測定した。***Tender***では、プロセスの生成と消滅を行う際に、プロセスを構成する資源を再利用し、プロセスの生成と消滅を高速化できる[2]。プロセスの生成と消滅処理では、ディスクI/Oが発生しない環境で測

表 1 プロセスの再起動インタフェース

形式	引数の内容
procreset(pid, argv)	pid で指定したプロセスのデータ部を初期化し、argv で指定された文字列を引数として渡し、再起動する。

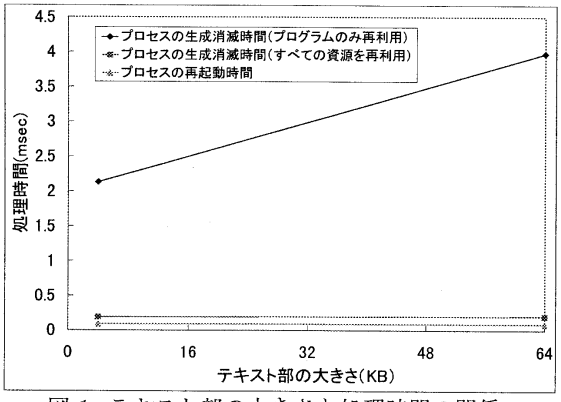

図 1 テキスト部の大きさと処理時間の関係

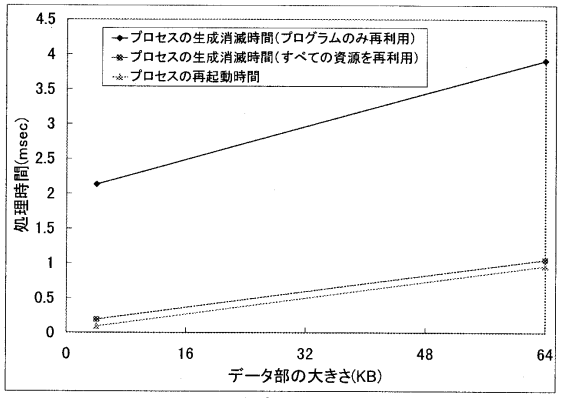

図 2 データ部の大きさと処理時間の関係

定を行うために、プログラムのみを再利用した場合と、プロセスを構成するすべての資源を再利用できた場合の処理時間を測定した。

測定に使用した計算機は、PentiumII 450MHz の計算機で、ハードウェアクロックのカウンタを利用して処理時間を測定した。測定内容は、テキスト部の大きさを可変にし、データ部と BSS 部を 4KB に固定した場合と、データ部の大きさを可変にし、テキスト部と BSS 部を 4KB に固定した場合の処理時間を測定した。

テキスト部の大きさを可変にした場合の測定結果を図 1に示す。プロセスの再起動処理では、テキスト部の大きさにかかわらず約 90usec で処理を実行できる。すべての資源を再利用してプロセスを生成・消滅する場合には、テキスト部の内容まで再利用できるので、テキスト部の大きさに関係なくほぼ一定の時間で処理を実行できる。その処理時間は、約 200usec であり、プロセスの再起動の方が高速であることがわかる。これは、プロセスの再起動では、プロセスの実体を消さずにプロセスを再起動するためである。

データ部の大きさを可変にした場合の測定結果を図 1 に示す。プロセスの再起動処理では、データ部の内容を読み込むためのメモリコピーが必要であり、データ部の大きさに比例した処理時間がかかる。同様にプロセスを生成・消滅する場合も、データ部の内容を読み込むためのメモリコピーが必要であるため、データ部に比例した処理時間がかかる。プロセスの再起動は、プロセスの実体を消さないため、すべての資源を再利用したプロセスの生成と消滅に比べ、約 100usec 高速である。

プログラムのみを再利用してプロセスを生成・消滅する場合には、データ部の内容を読み込む際にメモリコピーが必要であるため、データ部の大きさに比例した処理時間がかかる。今回の測定では、BSS 部を可変にした場合の測定を行っていないが、プロセスの再起動処理やプロセスの生成・消滅処理での処理内容が、データ部に対するものと同一であるため、データ部の結果とほぼ同じ結果が出ると推測できる。

測定結果から、プロセスの再起動時間にかかる処理時間を定式化した結果を示す。

$$(\text{プロセスの再起動時間}\ (msec)) = 14.4 \times (\text{データ部}\ (KB) + BSS\text{部}\ (KB)) + 34.3$$

5 まとめ

本稿では、資源を分離・独立化した ***Tender*** の特徴について述べ、プロセッサの割当単位を資源として分離・独立化した「演算」について述べた。演算を用い、プロセスの実行を任意の場所で停止し、データ部の初期化によるプロセスの再起動の特徴と実現方式について述べた。実測評価した結果、プロセスの再起動の処理時間はテキスト部の大きさに対して一定であり、データ部と BSS 部の大きさに比例することを明らかにした。

今後の課題としては、プロセスの再起動時にプロセスの確保していた資源の解放、およびその情報の受け渡しの検討がある。

参考文献

[1] 谷口秀夫：“分散指向永続オペレーティングシステム ***Tender*** ”, 情報処理学会コンピュータシステムシンポジウム, シンポジウム論文集 Vol.95, No.7, pp.47-54(1995).

[2] 田端 利宏, 谷口 秀夫：“プロセス構成要素を再利用したプロセスの生成と消滅の高速化,” 情処研報, Vol.98, No.33, pp.79-86 (1998).

[3] 田端利宏, 谷口秀夫：“***Tender*** オペレーティングシステムの資源「演算」によるプログラム実行速度調整機能の実現と評価”, 情報処理学会論文誌, Vol.40, No.6, pp.2523-2533 (1999).