走行中のプロセス間で共有されたプログラムの部分入れ替え法

7H－2

後藤 真孝　谷口 秀夫　牛島 和夫
九州大学工学部

1　はじめに

現代の社会では様々なところで計算機が使われており、計算機に対してより高い信頼性を必要としている。しかし、計算機サービスにはソフト面やハード面で色々な障害が起こり得る。また、長期のサービス運用によって、サービスの内容を変更する必要性も生じる。これらの対処法として、走行中のプログラムの部分的変更が有効である。それが可能ならば、走行中のプロセスを終了させる必要はなくなり、修正後、再起動を行う必要もなくなる。

プロセスとして走行しているプログラムを変更する方法については、プロセスを冗長構成にしてプロセス単位で入れ替える方法[1]がある。この方法では、プロセスを単位として入れ替えるため、わずかな変更であってもプロセス全体を入れ替える必要がある。また、プロセスを継続したままでプログラムの一部を入れ替える方法も報告されているが、入れ替えを行うタイミングに厳しい制約があるもの[2]や、入れ替え対象プログラムが、１つのプロセス単独で実行されている場合の実現法[3]といったものであった。

本稿では、複数のプロセスがプログラムを共有している場合についての、部分入れ替え処理の課題を示し、対処法と実現方式を述べる。

2　入れ替え処理の課題と対処

2.1　入れ替え可能条件

図1のプログラムを例に考える。今、プロセスはプログラム部分(以下モジュール)Yを実行していると仮定した時、モジュールX,Y,Zはそれぞれ呼出中、走行中、未使用の状態と名付ける。これにより、全てのプログラム部分の状態は、３つの状態のどれかである。各状態でプログラムの部分入れ替えが可能なためのプログラム部分への条件は、文献[3]に述べた。

この条件を基に、複数のプロセスでプログラムを共有している場合の、プログラム部分に対する入れ替え可能条件を表1に示す。表1において、項番４から７が共有されたプログラム部分の入れ替えに特有な条件である。条件(a),(b),(c),(d)の内容は、以下の通りである。

(a) 入れ替えるモジュールのアドレス解決が必要
(b) 入れ替えるモジュールの外部変数の引き継ぎが必要

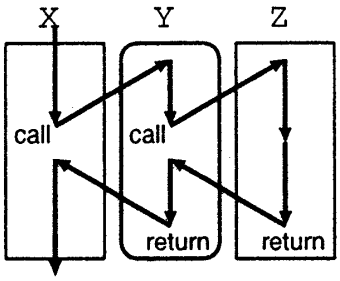

図１ 簡単なプログラムの例

表１ テキスト共有時の入れ替え条件

	モジュールの状態	条件
1	すべて未使用	(a),(b)
2	すべて走行中	入れ替え不可能
3	すべて呼出中	(a),(b),(c),(d)
4	未使用、走行中	入れ替え不可能
5	未使用、呼出中	(a),(b),(c),(d)
6	走行中、呼出中	入れ替え不可能
7	未使用、走行中、呼出中	入れ替え不可能

表２ 管理表の種類と主な役割

管理表の種類	役割
プログラム管理表	管理のための各テキスト毎の区別
プロセス管理表	プロセスの走行状態のデータ保持
モジュール管理表	入れ替えに関するデータを保持

(c) 別のプログラム部分を呼び出したアドレス(戻りアドレス)を変更しない
(d) 別のプログラム部分を呼んだ関数の内部変数を変更しない

2.2　入れ替え可能状態の把握

入れ替え可能か否かを把握するには、以下の点に留意する必要がある。ここで、入れ替えるプログラム部分の最小単位は関数と仮定し、モジュールと名付ける。

(1) 複数のプログラムに対して、同時にモジュール入れ替えの要求を行えるようにするため、プログラム対応にプロセス毎の走行状態を保持する必要がある。
(2) 同一プログラム中の異なるモジュールに対して、同時に入れ替えの要求を行えるようにするため、モジュール対応に入れ替えに関するデータを保持する必要がある。

上記の留意点からプログラム、プロセス、モジュール毎の管理表を用意する。各管理表の主な役割を表2に示す。

2.3　入れ替え可能の保証

入れ替えるモジュールが複数のプログラムから共有されている場合、そのプログラム部分が有限時間内で入れ

Mechanism for exchanging program shared by running processes.
Masataka Goto, Hideo Taniguchi, Kazuo Ushijima
Faculty of Engineering, Kyushu University

表 3 システムコールの種類と処理内容

システムコール	処理内容
状態監視開始	プログラム管理表を作成する
状態監視終了	プログラム管理表以下、モジュール管理表、プロセス管理表を全て解放する
入れ替え要求	モジュール管理表を作成し、管理表に入れ替え部分に関する情報を書き込む
モジュール突入	プロセス管理表が無ければ作成し、管理表にモジュール突入情報を書き込む
モジュール脱出	プロセス管理表が無ければ作成し、管理表にモジュール脱出情報を書き込む
タイムアウト時間設定	プロセス管理表にタイムアウト時間を書き込む

替え可能な状態になるという保証は無い。当該モジュールを実行しようとしているプロセスを、停止させることによって、入れ替え可能状態への移行を促進させる。入れ替え対象モジュールを利用しているプロセスへの影響を考えて、最大停止時間を設ける。これをタイムアウト時間と名付ける。タイムアウト時間は、プロセス毎に自由に設定変更可能にする。これにより、入れ替えのための実行停止がサービスへ大きな影響を与えることを防げる。例えば、最初はどのプロセスも無停止に設定し、停止しても影響が小さそうなプロセスから徐々に、タイムアウト時間を伸ばしていく方法がある。

3 実現

本機能は、OS のプロセス機能を実現しているプログラム部分を、改修しない形で実現する。このように、実現部分を局所化することにより、既存の OS への組み込みが容易に行える。

入れ替え処理に必要なシステムコールとして、入れ替えのための走行状態の監視を開始するもの、終了するもの、および、入れ替えを要求するものがある。また、モジュール間の移動を告げるシステムコールも必要である。更に、入れ替え可能を保証するための処理により、タイムアウト時間を設定するシステムコールも必要である。

各システムコールの主な処理内容を表 3に示す。

4 測定と評価

測定に用いたテストプログラムは、図 1の構造をした無限ループで、入れ替えは対象モジュールは Y とした。また、各モジュール毎の処理時間は、X が 4 秒、Y が 1 秒、Z が 5 秒とした。

実現したシステムコールの処理時間を表 4に示し、プロセス数と入れ替え処理時間の関係を図 2に示す。

表 4からモジュール突入や脱出のシステムコールは、監視 ON の状態の場合に比べ、監視 OFF の状態の場合は約 1/3 の処理時間になっている。したがって、監視の開始と終了のシステムコールの効果は大きい。

図 2 から、入れ替え処理時間は、プロセス数にあまり影響を受けないことが分かる。入れ替え処理時間は、モジュールの処理時間の最大のものとほぼ一致している。これは、監視開始を宣言後、状態把握が必要な全プロセスの状態を得るには、これらの全プロセスがモジュールへ突入またはモジュールから脱出し、当該システムコールを発行しなければならないからである。したがって、プログラムを多くのモジュールに分割し、モジュールの処理時間の最大を小さくすると、入れ替え時間は小さくなる。

表 4 システムコール処理時間 (相対値)

システムコールの種類	監視 ON	監視 OFF
監視開始	2.30	
監視終了		1.48
監視の開始と終了		12.12
モジュール突入	4.17	1.59
モジュール脱出	4.43	1.55
入れ替え要求		5.20
タイムアウト設定	2.39	2.39

値は getpid システムコールの処理時間を 1 としたときの相対値

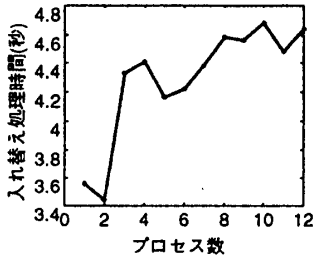

図 2 入れ替え処理時間

5 まとめと今後の課題

プログラムをモジュール単位で分割し、プロセス毎のモジュールの走行状態を OS が把握することにより、走行中のプロセス間で共有されたプログラム部分の入れ替えを実現した。入れ替えのために必要なシステムコールは、プログラムの実行時間にあまり影響を与えない。また、入れ替え時間は、モジュールの処理時間の最大のものとほぼ一致する。

今後は、入れ替え時間とプロセス数とタイムアウト時間の関係を明らかにする。また、共有するプログラム部分と非共有なプログラム部分の両方を扱う方法を確立する。さらに、走行状態の監視を高速化する。

参考文献

[1] Hiroshi Muramatsu, Masahiro Date, Hiroshi Yoshida, Masaharu Kitaoka and Norio Kurobane :"Operating System SXO for Continuous Operation", IFIP92. Vol.1, pp.615-621(1992)

[2] 中橋晃文, 小林博, 西山修治, 保里裕之, 雑賀敏昌 :"フォルトトレラントコンピュータのソフトウェア保守技術の考察", 情処第 43 回全国大会予稿, pp.6-35(1991)

[3] 伊藤健一, 箱守聰, 横山和俊, 谷口秀夫 :"走行中プログラムの部分入れ替え法", コンピュータシステムシンポジウム論文集 Vol.94 No.10, pp.71-78(1994)