

A P の動的入替えに関する一考察

4K-9

伊藤 健一

谷口 秀夫

N T T データ通信（株）

九州大学 工学部

1. はじめに

近年、利用者のシステムに対する要望として、24時間連続のサービスを要求する声が高まっている。そのためには、以下の課題を満たす必要がある。

- (1) 故障レス（または故障がサービスに影響しない）
- (2) 障害復旧・改修のための保守時間の縮小

このうち、(1)は信頼性向上の研究がハード、ソフトの両面から進んでいる¹⁾²⁾。(2)については、保守作業開始までの時間短縮としてリモートメンテナンス等の検討がなされている。しかし、保守作業中のサービス停止は避けられない。我々は、計算機に障害が発生してもサービス処理に影響を与えずに障害を回復させるための制御方式について、研究を進めている。ここでは、サービスを実現している応用プログラム（A P と略す）を動的に入替える条件について述べる。

2. プログラムを変更する方法

プログラムを変更する方法として、次のものが考えられる。

(1) プロセス終了型

サービスを構成するプロセスを一旦終了し、修正されたプログラムから新しいプロセスを生成しサービスを再開する方法である。

(2) プロセス継続型

プロセスを存在させたまま、プロセスの利用するテキスト領域やデータ領域の一部を入替える方法である。

それぞれの長所、短所を表1に示す。プロセス継続型の方がサービスを実現するプロセスを走行させ続けることが可能になるため、24時間連続サービスを実現するためには有利である。従来、プロセスを継続しながらプログラムを修正する方法として、パッチによる修正が使われてきた。しかし、従来の方

表1 プログラムを変更する方法

	プロセス終了型	プロセス継続型
長所	1. OSのプロセス制御機能を利用するため処理が簡単	1. プロセスが資源を再確保する必要がない 2. プロセスの状態が保持できる
短所	1. プロセスの保持している資源や状態を引き継ぐのが困難 2. ループを形成する処理は終了のタイミングが難しい	1. プロセスが走行し続けるので、プロセス状態を把握しづらい

法には、以下の問題点があった³⁾。

- (1) 修正データの作成をマシン語レベルで行なうため、修正箇所の特定制や修正データの作成に時間がかかる。
- (2) 実メモリ空間を直接修正するため、元のメモリ領域より大きいパッチは当てられない。
- (3) プログラム構造がブロック化されておらず、プロセスの走行状態が特定できないため、パッチ後に部分的な初期化処理が必要である。

3. ダイナミックリンク機能の利用

我々は上記の問題点を解決するために、メモリ上のアドレス解決を動的に行なうダイナミックリンク機能を利用してプログラムを入替える方式を検討した。ダイナミックリンク機能を利用することにより、以下の利点がある。

- (1) ロードセグメント（L S と略す）毎にコンパイルできるため、高級言語での修正が可能。
- (2) ロードにより実メモリにL Sが配置されるため、L Sの大きさには制限がない。
- (3) ダイナミックリンク機能で使用するテーブルに手を加えることにより、プロセスがプログラムのどの部分を走行しているかをL Sを単位として把握することが可能になる。

A Study of Dynamic Program Restructuring.
Ken'ichi ITO* and Hideo TANIGUCHI**
* : NTT DATA Communications Systems Corporation
** : Kyushu University

4. LSを入替えるための条件

プログラムを修正する場合には、以下の条件を満たす必要がある。

- (1) 呼出し・返却値のインタフェースが一致していること。
- (2) 修正によるアルゴリズムの矛盾（ループカウンタのリセット等）が生じないこと。

これらの条件は、動的入替えについても同じである。

さらにプロセス継続型では、プログラム修正箇所とプロセスの位置関係が重要になる。プロセスの位置を無視してLSを入替えると、実行しているテキスト内容や戻り番地になっているアドレスが変更され、プロセスの実行に影響が出る。

LSを入替える場合には、プロセスの位置を把握して処理に影響が出ないようにする必要がある。入替え対象となるLSから見た場合、プロセスの走行状態は以下に示す3つに分類される。

- 未使用：入替え対象のLSを実行する前か、既に実行を終えリターンした状態。
- 走行中：入替え対象のLS自体を実行している状態。
- 呼出し中：入替え対象のLSから更に他のLSを呼出し、実行している状態。呼出し先の処理が終了後、対象のLSに実行が戻る。

図1にLSとプロセスの位置関係を示し、以下に、各状態の時の入替えに必要な条件について述べる。

(1) 未使用の時

これは、プロセスが入替え対象のLSを使用していない（図1のX部分）時であり、新たな使用時には入替えたLSの先頭から走行する。そのため、以下の条件が必要になる。

- (a) 入替えたLSのアドレス解決が必要。
- (b) 入替えたLSの外部変数の値の保証（引き継ぎ）が必要。

(2) 走行中の時

これは、入替え対象のLSを利用しているプロセスが存在する（図1のY部分）ため、入替えを行なうと実行のためのテキストが変更され

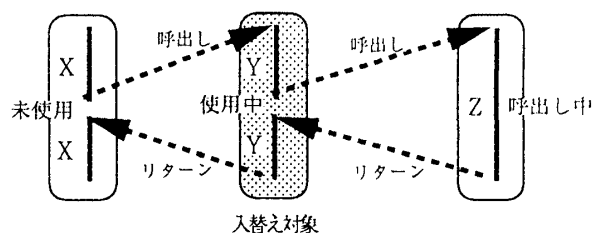


図1 プロセスとLSの位置関係

プロセスに影響が出る。そのため入替えは不可能である。

(3) 呼出し中の時

これは、入替え対象のLSを実行していたプロセスが、更に別のLSを呼び出して実行している状況（図1のZ部分）である。そのため、別のLSで処理を終えたプロセスが、戻り番地として入替え対象のLSのアドレスを利用する。また、呼出し側の関数における内部変数などが呼び出し時にスタックにより受け渡されている可能性があるため、未使用の時の条件(a)(b)に追加して、以下の条件も必要になる。

- (c) 別のLS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LSを呼んだ関数の内部変数を変更しない。

5. まとめ

本稿では、AP入替えをダイナミックリンク機能を利用して行なう場合の利点と、入替えを行なう際に必要となるロードセグメントの条件について述べた。現在、プロセスの状態を把握する方法について検討中であり、プロセスの状態を入替え可能状態に動かす方法と、入替えの前後で必要になるプロセスの走行を制御する方法について検討する予定である。

参考文献

- [1] Parag K. Lala (当麻監訳)：フォールト・トレランス入門、オーム社 (1988)
- [2] 萩原他：特集「フォールトトレラント分散システム向けアルゴリズム」、情報処理、Vol. 34, No. 11, pp. 1335-1374 (1993)
- [3] 遠城、谷口、伊藤：コンパイラを用いたバッチデータ作成方法、情報処理学会ソフトウェア工学研究会 89-SE-67 (1989)