システムソフトウェアと
オペレーティング・システム 79－7
（1998. 8. 6）

# 入替え条件を緩和した
# 実行中プログラム部分入替え法の評価

中島 雷太　谷口 秀夫

九州大学大学院システム情報科学研究科

〒 812-8581 福岡県福岡市東区箱崎 6-10-1
TEL 092(642)4051
Email : nakajima, tani@csce.kyushu-u.ac.jp

あらまし
我々は、実行中プログラムの一部分を入れ替える新たな制御法を提案している。具体的には、プログラム実行状態を分類し、入替え条件、プログラム状態の管理法、有限時間内での入替えを保証する制御法、サービスへの影響を軽減する方法を示し、実装および評価を行なっている。本論文では、入替え条件の緩和について述べる。さらに、入出力処理を行なうプログラムとプロセッサ処理を行なうプログラムの二通りについて、入替え時間の観点から評価を行ない、入替え条件の緩和による問題点を述べる。

キーワード　プログラム入替え、ソフトウェア、プログラム、プロセス、信頼性

# Evaluation of Mechanism for Exchanging Program Part of Running under the Relaxed Conditions of Exchange

Raita Nakajima and Hideo Taniguchi

Graduate School of information science and electrical engineering, Kyushu University.

〒 812-8581 6-10-1 Hakozaki, Higashi-ku, Fukuoka, Japan
TEL 092(642)4051
Email : nakajima, tani@csce.kyushu-u.ac.jp

abstract
We have proposed that the mechanism of exchanging program part of running. To be concrete, we have described problems that how to classify the state of executed program parts, to make conditions of exchangeable program parts clear, how to manage the state of executed program parts, how to guarantee an exchange in finite time, and how to reduce the influence on service. And we have reported the evaluation of the mechanism. In this paper, we examine about the relaxed conditions of exchange and describe the evaluation of exchanging time about two varieties of programs. One program is I/O processing program and the other is processor processing program. Last we clarify the problem due to the relaxed condition of exchange.

keywords　exchanging program, software, program, process, reliability

## 1 まえがき

現在では、計算機の普及に伴い、日常生活のあらゆる場面で計算機が利用されている。また、計算機の性能向上によりサービスも多様化しつつある。これに伴い、計算機によるサービスが不調をきたすと、さまざまな形で我々に影響を与え、最悪の場合、社会活動を麻痺させてしまう危険性もはらんでいる。したがって、これまで以上に計算機に対して、より高い信頼性が求められる。しかし、どんなに信頼性が向上しても、ソフトウェアやハードウェアにおいて生じる、様々な障害を避けることは難しい。また、計算機による長期サービスを提供する場合、ユーザの要望などにより、サービスの内容を変更する必要性も生じると考えられる。しかしながら、社会に与える影響を考えると、これら諸々の変更や修正は、すみやかに完了されることが望ましい。

計算機を停止することなく、動作中のプログラムの一部分を変更できれば、サービスの継続的な提供が可能となる。このような背景から、実行中プログラムの一部分を入れ替える方法が提案されている。一つは、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として、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。また、プロセスを走行させたままプログラムの一部分を入れ替える方法 [2] も提案されている。

我々は、これらの方法について問題点を指摘 [3],[4] し、新たな部分入替え法として、次の大きな二つの特徴を持ったプログラム部分入替え法を提案した [3]。一つは、サービスプログラムが、入替え契機を意識する必要がない点である。もう一つは、入替え対象プログラム部分が、入替え対象でない別のプログラム部分を呼び出している場合にも、プログラムの入替えを可能としている点である。さらに、複数のプロセス間で共有されたプログラム部分の入替えも考慮している [4]。また、実装し評価することにより、その有効性も示した。しかし、文献 [4] では、入れ替えるプログラム部分に要求する入替え条件が厳しいため、実用に大きな制限となる問題点がある。また、入替え時間についての評価において、入出力処理を行なうプログラムに関する評価しか行なっていない。

そこで、本論文では、入替え条件の緩和について述べる。さらに、入出力処理を行なうプログラムとプロセッサ処理を行なうプログラムの2通りについて、入替え時間の観点から評価を行ない、入替え条件の緩和による問題点を明らかにする。

以降では、プロセス間で共有する単位を単にプログラムと呼び、そのプログラムの一部分を、プログラム部分と呼ぶ。プログラムとプログラム部分は、ともに最小単位を関数とし、その関数をモジュールと名づける。実行単位によってプログラムが共有されている場合を単に共有されていると呼ぶ。

## 2 入替え法

### 2.1 基本方式

実行中プログラムの一部分を入れ替える方式について、文献 [3]、[4] の内容を基に、以下の項目について簡単に説明する。

(1) プログラム実行状態の分類
(2) 入替え可能条件
(3) プログラム状態の管理法
(4) 有限時間内での入替えを保証する制御法
(5) 入替え対象プログラムのサービスへの影響を軽減する方法

入替え対象プログラム部分の実行状態を、図1に示すように、以下の3つに分類する。

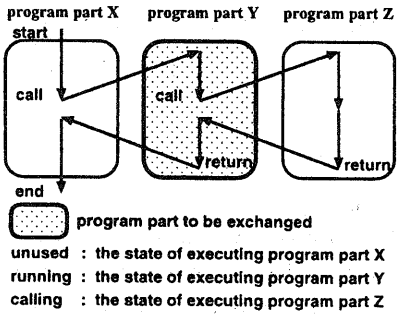


図1 プログラム部分の実行状態の関係

(1) 未使用:実行する前、あるいは、実行を終えた状態
(2) 走行中:実行中の状態
(3) 呼出中:別のプログラム部分を呼び出している状態

プログラム部分の実行状態によって、以下の入替え条件を満たす必要がある。

(条件1) 呼び出しと返却のインタフェースの一致

(条件2) 処理の矛盾の回避

(条件3) 戻り値のアドレスを変更しない

(条件4) スタティックリンクの場合、メモリ上の外部変数の参照アドレスが同じであること

(条件5) 外部変数の値が入替えの前後で同じであることの保証が必要か否か

(条件6) アドレス渡しによる外部変数や内部変数の参照や更新がどのように行なわれているか

(条件5)、(条件6)については、変更する内容に応じて、プログラム個別の対処が必要である。入替え対象プログラム部分が複数のプロセスによって共有されている場合も含めた、プログラム部分の実行状態と入替え条件の関係を、表1に示す。

表1 プログラム部分状態と入替え条件の関係

| 通番 | 実行状態 | 入替え条件 |
|---|---|---|
| (1) | 未使用 | (条件1), (条件2) |
| (2) | 走行中 | 入替え不可能 |
| (3) | 呼出中 | (条件1), (条件2)<br>(条件3), (条件4)<br>(条件5), (条件6) |
| (4) | 未使用かつ走行中 | 入替え不可能 |
| (5) | 未使用かつ呼出中 | (条件1), (条件2)<br>(条件3), (条件4)<br>(条件5), (条件6) |
| (6) | 走行中かつ呼出中 | 入替え不可能 |
| (7) | 未使用かつ走行中かつ呼出中 | 入替え不可能 |

次に、一つのプログラムについて複数箇所のプログラム部分に変更の要求があり、更に多数のプログラムに対して、同時にこの要求がある状態を想定し、このような場合にも入替え出来るようなプログラム状態の管理法を述べる。具体的には、プログラム状態の管理に対して、以下の対処を行なう。

(1) 複数のプログラムに対して、同時に入替えを可能にするため、プログラム毎に走行状態を保持する

(2) モジュールがプロセスで共有されていても、正しい走行状態を把握するため、走行状態の把握はプロセス毎に行なう

(3) 同一プログラム中の異なるモジュールに対して、同時に入替えの要求を行なえるようにするため、モジュール対応に入替えに関するデータを保持する

以上の要求を満たすために、プログラム、プロセス、モジュール単位で状態を管理する。

有限時間内で入替えを実行し完了できるために、「入替え対象プログラム部分が有限時間内で走行中の状態でなくなること」を保証する機構を確立する。つまり、入替え要求があると、入替え対象モジュールに走行を移そうとしているプロセスを停止させ、その他のプロセスを走行させ続ける。その後、入替え対象プログラム部分が入替え可能状態であることを検知した直後に入替えを実施する。これにより、有限時間内での入替えを保証する。ただし、入替え対象モジュール内で停止しているプロセスが存在する場合、その停止中のプロセスが有限時間内にプログラム実行を再開する保証がない時はこの限りではない。

サービスへの影響の軽減法として、文献[4]では、プロセスの停止に制限時間（タイムアウト時間）を設けている。プロセス毎にタイムアウト時間の設定変更を可能にし、停止時間を制御する。タイムアウト時間による制御の様子を、図2に示す。タイムアウト時間を用いることにより、停止のために悪影響を受けるプロセスは止めず、逆に、重要度の低いプロセスを長めに止めるということが出来る。

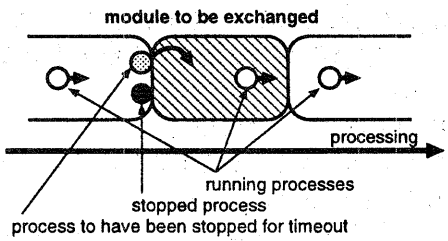


図2 タイムアウト時間による制御

### 2.2 課題と対処

文献[4]に示したように、プログラム部分の実行状態が「未使用」または「呼出中」に入替えを可能とすると、表1に示した（条件1）から（条件6）までの全てを満たしていなければならず、入れ替えようとするプログラム部分の記述に対する制約が大きい。そこで、プログラム部分入替え法の利便性を高めるために、入替え条件を緩和する。具体的には、入替え対象プログラム部分の実行状態が「未使用」の時のみに入替えを行なうようにする。これにより、入替え条件は（条件1）と（条件2）の二つのみに緩和でき、実用性を高めることができる。

しかしながら、この際には、入替え時間の増加が懸念される。

## 3 評価と考察

### 3.1 実現機能

入替えのために六つのシステムコールを実現した[4]。

(1) 状態監視開始と状態監視終了

指定するプログラムを実行しているプロセスに対し、状態把握処理の開始または終了を宣言するシステムコールである。

(2) 入替え要求

指定するプログラムの、指定する部分の入替えを要求するシステムコールである。入替えが完了するまで待つ。

(3) モジュール突入とモジュール脱出

プロセスが、自分の走行状態をオペレーティングシステムに告げるシステムコールである。入替え要求があれば、入替え対象モジュールに突入する時、あるいは他のモジュールを脱出して入替え対象モジュールに返ってくる時、入替え終了まで待たされる。入替え終了まで待たず、タイムアウト時間により待ちが解除されることもある。

(4) タイムアウト時間設定

タイムアウト時間を設定するシステムコールである。指定されたプロセス識別子を持つプロセス管理表のタイムアウト時間を、指定時間に設定する。

### 3.2 測定条件

測定に用いたプログラムの流れを図3に示す。

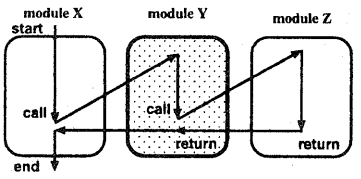


図3 プログラムの処理の流れ

モジュールYが、入替え対象プログラム部分である。

表2 プログラムの各モジュール処理時間（秒）(場合1)

| type | module X | module Y | module Z |
|---|---|---|---|
| A | 1 | 1 | 8 |
| B | 1 | 2 | 7 |
| C | 1 | 3 | 6 |
| D | 1 | 4 | 5 |
| E | 1 | 5 | 4 |
| F | 1 | 6 | 3 |
| G | 1 | 7 | 2 |
| H | 1 | 8 | 1 |

表3 プログラムの各モジュール処理時間（秒）(場合2)

| type | module X | module Y | module Z |
|---|---|---|---|
| A | 1 | 1 | 8 |
| B | 2 | 1 | 7 |
| C | 3 | 1 | 6 |
| D | 4 | 1 | 5 |
| E | 5 | 1 | 4 |
| F | 6 | 1 | 3 |
| G | 7 | 1 | 2 |
| H | 8 | 1 | 1 |

モジュール処理時間の関係として二つ用意した。様子を表2と表3に示す。各モジュールの処理時間の合計は10秒とし、AからHの八つの場合を想定した。表2の場合、モジュールXの処理時間が一定（1秒）である。このため、「未使用」の時のみに入替えを可能とした場合には、AからHのどの場合にも入替え時間はほぼ一定になると推察することができる。一方、

表3は、モジュールYの処理時間を一定（1秒）とした場合である。「呼出中」に入替えが可能か否かが入替え時間に与える影響を明らかにするために、この場合を設定した。

プロセスは、疑似乱数を用いて不定間隔で起動し、共有プロセス数が1から10までの場合について、モジュールYに対して入替えを行ない、その入替え時間を測定した。測定は20回ずつ行ない、それらの平均値を測定結果とした。

また、測定に用いたプログラムとして、入出力処理を行なうプログラムと、プロセッサ処理を行なうプログラムを用意した。測定の組合せを表4に示す。

表4　入替え可能条件と測定条件の組合せ

| （測定条件）／（入替え可能条件） | 入出力処理プログラム 場合1（表2） | 入出力処理プログラム 場合2（表3） | プロセッサ処理プログラム 場合1（表2） | プロセッサ処理プログラム 場合2（表3） |
|---|---|---|---|---|
| 未使用 or 呼出中 | (図4) | (図6) | (図8) | (図10) |
| 未使用 | (図5) | (図7) | (図9) | (図11) |

なお、文献[4]では、表4において、図4に相当する箇所について、入替え対象プログラムの状態把握を、入替え要求直前に行なう場合と入替え対象プロセスの起動時に行なう場合について測定を行ない、評価を行なっている。一方、本論文では、入替え時間の観点から、入替え可能とするプログラム実行状態を「未使用」または「呼出中」とした時と、「未使用」のみとした時について、比較・評価を行なうことで、入替え内容の記述に関する入替え条件の緩和の有効性と問題点を明らかにする事を主点としている。したがって、入替え対象プログラムの状態把握をいつ行なうかと入替え条件の緩和は密接な関係にないため、状態把握は入替え対象プロセスの起動時に行なうようにした。

### 3.3　入出力処理プログラムの入替え時間

測定結果を図4から図7に示す。

文献[4]では、図4を基に、入替え時間について次の特徴があることを述べている。

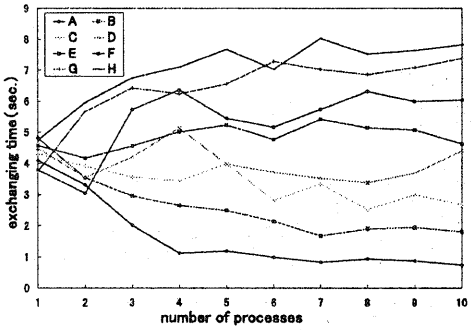


図4　プロセス数と入替え時間の関係
（入出力処理プログラムの場合，
プログラム部分が未使用または呼出中の時に入替え可能，
各モジュールの処理時間は表2の場合）

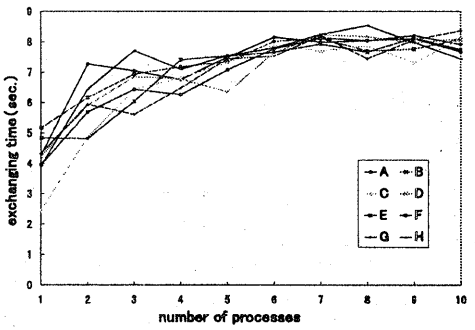


図5　プロセス数と入替え時間の関係
（入出力処理プログラムの場合，
プログラム部分が未使用の時のみに入替え可能，
各モジュールの処理時間は表2の場合）

(1) プログラムを共有するプロセス数が少ない時、入替え時間は最大モジュール処理時間の影響を受ける

(2) プログラムを共有するプロセス数が多くなると、入替え時間は入替え対象モジュールの処理時間に近づく

プログラムを共有するプロセス数が少ない時には、最大モジュール処理時間のモジュールを走行している確率が最も高くなるため、入替え時間は、モジュール突入、脱出システムコールを発行するまでの時間になる。一方、プロセス数が増加すると、入替え対象モジュールを走行している確率が高くなるため、入替え対象モジュールの処理時間に影響される。

上記の理由により、図6の入替え時間は、各typeとも同じような値になっている。

次に、入替え可能条件が「未使用」のみであ

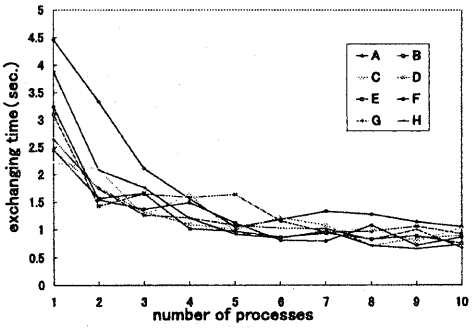


図 6 プロセス数と入替え時間の関係
(入出力処理プログラムの場合,
プログラム部分が未使用または呼出中の時に入替え可能,
各モジュールの処理時間は表3の場合)

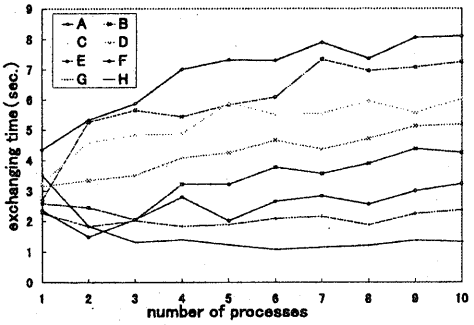


図 7 プロセス数と入替え時間の関係
(入出力処理プログラムの場合,
プログラム部分が未使用の時のみに入替え可能,
各モジュールの処理時間は表3の場合)

る図5と図7について考察する。図5においては、プログラムを共有するプロセス数が少ない時は、図4と同様の現象になっている。しかし、プロセス数が増加すると、大きく様子が異なり、各typeにおける差はない。これは、入替え可能条件が「未使用」のみであるため、入替え時間は、モジュールYとモジュールZの処理時間の和に近づくが、表4では、この処理時間の和が一定(9秒)のためである。

上記のことは、図7についても言える。つまり、図7では各type毎に、モジュールYとモジュールZの処理時間の和が異なるため、プロセス数が多くなると、入替え時間は、それぞれ異なってくる。

最後に、プログラムを共有するプロセス数が少ない場合と、プロセス数が多い場合の入替え時間について考察する。

プロセス数が一つの時、処理時間が最大の

表 5 プロセス数と入替え時間の関係(秒)

| type | 20 processes | 50 processes |
|---|---|---|
| 図5のA | 8.59 | 8.82 |
| 図7のA | 8.59 | 8.82 |
| 図7のD | 5.56 | 5.83 |
| 図7のH | 1.53 | 1.83 |

モジュールを走行している確率が高い。この時の入替え時間は、このモジュールからの脱出を示すシステムコールを発行するまでの時間にほぼ等しくなる。したがって、プロセス数が一つの時の入替え時間の期待値は、最大モジュール処理時間の半分となる。例えば、type Aでは、モジュールZが最大モジュール処理時間8秒であり、図5から図7において、いずれもプロセス数が一つの時の入替え時間は、約4秒になっている。

プログラムを共有するプロセス数が多い時、入替え時間は入替えができない実行状態のモジュールの処理時間の総和になる。図4と図5はこれを裏付けている。しかし、図5では約8.5秒程度に収束し、図7では、モジュールZの処理時間に収束している。これは、プロセス数が最大で10と十分でなかったためと推察する。そこで、図5のtype A、および図7のtype Aとtype Dとtype Hについて、プロセス数を20、50と大きくして入替え時間を測定した。結果を表5に示す。プロセス数を大きくすることにより、入替え時間は大きくなり、上記の考察を裏付けている。

### 3.4 プロセッサ処理プログラムの入替え時間

測定結果を図8から図11に示す。

まず、入替え可能条件が「未使用」または「呼出中」である図8と図10について考察する。図8や図10の入替え時間は次の特徴を持つ。

(1) プログラムを共有するプロセス数が多くなると、入替え時間はほぼ単調増加する

プログラムを共有するプロセス数が少ない時は、図4や図6と同様になっている。しかし、プロセス数が増加すると、入替え時間の様子が異なり、各typeにおいてほぼ単調増加となっている。これは、プロセス数が多くなると、プロセッサ処理開始のためのREADY状態での待ちが発生し、入替え時間に大きな影響を与え

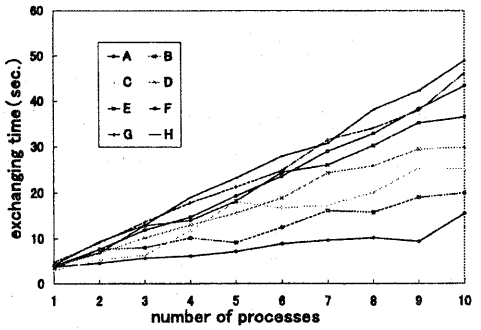


図 8 プロセス数と入替え時間の関係
(プロセッサ処理プログラムの場合,
プログラム部分が未使用または呼出中の時に入替え可能,
各モジュールの処理時間は表2の場合)

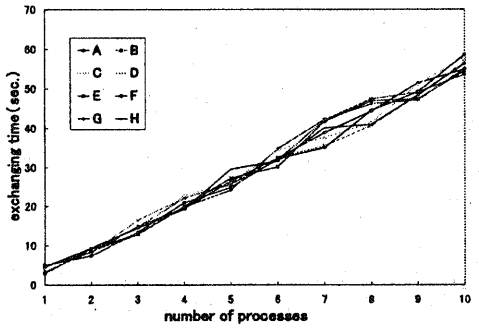


図 9 プロセス数と入替え時間の関係
(プロセッサ処理プログラムの場合,
プログラム部分が未使用の時のみに入替え可能,
各モジュールの処理時間は表2の場合)

るためである。type Aでもほぼ単調増加となるため、入替え対象モジュールの処理時間の影響以上に、READY状態での待ちの影響が大きいと言える。図8において、各typeで入替え時間に差が生じるのは、図4と同様に入替え対象モジュールの処理時間が違い、READY状態における待ちの影響度が異なるためである。

図10においては、図6と同様に各typeともモジュールYとモジュールZの処理時間の和が一定(9秒)のため、各typeにおける差はあまりない。ただし、入替え時間について、次の特徴が見られる。

(2) 入替え時間のばらつきが目立つ

入替え可能条件が「未使用」または「呼出中」であるため、プロセスは、入替え対象モジュールに走行を移そうとした時、つまりモジュールXからモジュールYへ突入する時、あるいはモジュールZから脱出する時、すでに入替え要求

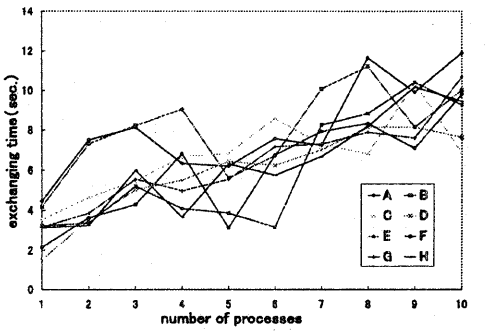


図 10 プロセス数と入替え時間の関係
(プロセッサ処理プログラムの場合,
プログラム部分が未使用または呼出中の時に入替え可能,
各モジュールの処理時間は表3の場合)

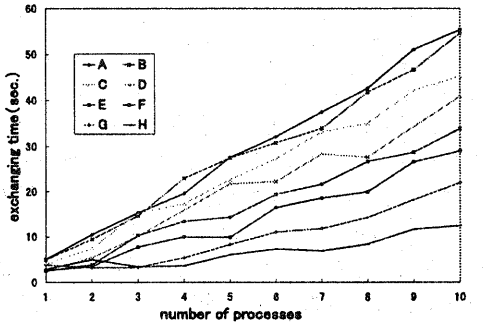


図 11 プロセス数と入替え時間の関係
(プロセッサ処理プログラムの場合,
プログラム部分が未使用の時のみに入替え可能,
各モジュールの処理時間は表3の場合)

があり、かつ入替え可能状態でない場合、移行の直前で停止させられる。入替え要求のタイミングやスケジューラの状態によって、入替え可能状態になるまでの時間にばらつきが生じ、さらに、図10の場合、READY状態における待ちの影響が加わるため、ばらつきが顕著になると推察する。そこで、標本数を50として測定した結果を図12に示す。十分にばらつきを吸収できていないが、考察を裏付ける結果となっている事がわかる。

次に、入替え可能条件が「未使用」のみである図9と図11について考察する。プログラムを共有するプロセス数が少ない時は、図5や図7と同様になっている。プロセス数が増加すると、大きく様子が異なる。これも図8や図10の場合と同様に、プロセッサ処理開始のためのREADY状態での待ちが発生し、入替え時間に大きな影響を与えるためである。また、type

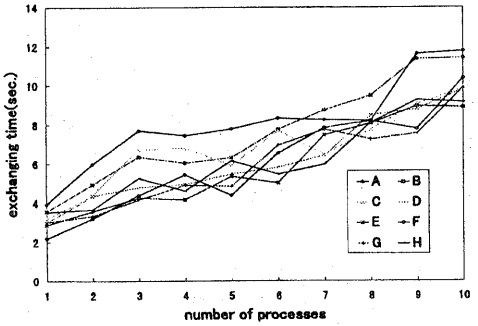


図 12 プロセス数と入替え時間の関係
(プロセッサ処理プログラムの場合,
プログラム部分が未使用または呼出中の時に入替え可能,
各モジュールの処理時間は表 3 の場合)

Hでもほぼ単調増加となるため、入替え対象モジュールの処理時間の影響以上に、READY状態での待ちの影響が大きいと言える。

図9においては、図5と同様に各typeとも、モジュールYとモジュールZの処理時間の和が一定(9秒)のため、各typeにおける差はほとんどない。一方、図11において、各typeで入替え時間に差が生じるのは、図7と同様に入替え対象モジュールの処理時間が違い、READY状態における待ちの影響度が異なるためである。

最後に、プログラムを共有するプロセス数が少ない場合と、プロセス数が多い場合の入替え時間について考察する。

プロセス数が一つの時は、プロセッサ処理によるREADY状態における待ちは発生しないため、**3.3**節で述べた場合と同様である。

入出力処理プログラムの場合の結果から推察すると、プログラムを共有するプログラムが多いとき、入替え時間は入替えができない実行状態のモジュールの総和とプロセス数の積になると思える。これは、READY状態における待ち時間の積算による影響である。

## 4 あとがき

我々が提案しているプログラムの部分入替え法について入替え条件を緩和した場合の評価を行ない、その際の問題点を述べた。入替え時間の評価結果を以下にまとめる。

(1) プログラムの性質に関係なく、プログラムを共有しているプロセス数が少ない時、入替え時間は最大モジュール処理時間の約半分となる

(2) 入出力処理プログラムの場合、プログラムを共有しているプロセス数が多くなると、入替え時間は、入替えができない実行状態であるモジュールの処理時間の総和に近づく

(3) プロセッサ処理プログラムの場合、プログラムを共有しているプロセス数が多くなると、入替え時間は、入替えができない実行状態であるモジュールの総和とプロセス数の積の値に近づく

以上の評価から、入替え可能条件を「未使用」のみとすると、入替え時間が長大化する問題があることが明らかとなった。

今後の課題として、入替え対象プロセス以外のプロセスが、同時走行している場合の入替え時間について評価を行なう必要がある。また、長大化した入替え時間の短縮化を計る必要がある。

## 参考文献

[1] Hiroshi Muramatsu, Masahiro Date, Hiroshi Yoshida, Masaharu Kitaoka, and Norio Kurobane : "Operating System SXO for Continuous Operation", IFIP 92, vol.1, pp.615-621, Jan. 1992.

[2] B.Snead, F.Ho, and B.Engram : "Operating System Features Real Time and Fault Tolerance", Computer Design, pp.177-185, Aug. 1984.

[3] 谷口秀夫 , 伊藤健一 , 牛島和夫 : "プロセス走行時におけるプログラムの部分入替え法 ", 電子情報通信学会論文誌 (D-I), vol.J78-D-I,no.5, pp.492-499, Mar. 1995.

[4] 谷口秀夫, 後藤真孝 : "走行中のプロセス間で共有されたプログラムの部分入替え法 ", 電子情報通信学会論文誌 (D-I), vol.80-D-I,no.6, pp.495-504, Jun. 1997.