# PSI-IIのGC（2）
## ―全体構成―

6B-4

川田　易治　（エスイーティ㈱）
立野裕和　（三菱電機㈱）
中島　克人　（㈱ICOT）

1．はじめに

PSI-IIでは、PSI-Iと同様、ファームウェアでマーク&コンパクション方式のGC（ガーベジ・コレクション）をサポート、高速化を図っている。PSI-Iのものに対し、PSI-IIでは多重論理空間でのGCを実現した他，プロセス単位でのGC機能（Collect _stack _garbage ）も追加した。

2．GC起動のタイミング

マルチプロセスをサポートするPSI-IIでは、個々のプロセスには3つのスタックエリア（グローバル、ローカル、トレール）が割当てられる。システムの制御管理等に使用されるシステムエリア、プログラムを格納する領域であるヒープエリアの2つは各プロセスから共用される。

この2つの共有エリアと各プロセスの3つのスタックエリアに対して割当てられたメモリの先頭にはGray Page と呼ばれる実メモリが1ページ（1kword）割当てられている。

プログラムの実行により1つまたは複数のエリアのメモリが伸長し、このGray Page と呼ばれるメモリ領域にまで達すると、実ページ割り付け要求ハードウェア割り込みが命令の切目に発生し、新たにそのエリアに対してメモリを与えるためのマイクロルーチンが起動される。通常は新たなページを1ページとGray Page とを割り付け、元の処理に復帰するのであるが割り当てる実ページが無い場合には、このマイクロルーチンはメモリ回収要求のトラップを起こし、制御をメモリ管理を受け持つソフトウェアに渡す。

このメモリ回収要求のトラップは、プロセス作成時のスタックエリアのページ割り当て時や特定のエリアに対するページ割り当て時のように、実ページを新たに消費するタイミングでなら、いつでも起こり得る。［1］

3．ソフトウェアでの処理

メモリ回収要求を受けるソフトウェアのためにPSI-IIでは以下の3つの組込述語が用意されている。

1) Collect _free_pages(^Page)

これは全プロセス、全エリアの未使用となった実ページ（エリアトップより先の部分）を回収し、実ページを管理するスタックに自由ページとして返却する組込述語であり、回収後の自由実ページ数を出力としてソフトウェアに知らせる。

2) Collect _garbage

全プロセスの3つのスタックエリアと、共有ヒープエリアのガベージコレクションを行なうマイクロルーチンである。

3) Collect _stack _garbage （Process)

1プロセスの3つの固有スタックエリアのみを対象としガベージコレクションを行なう。

メモリ回収要求のトラップを受けたソフトウエア（メモリーマネージャ）では，実ページがどれくらい必要かを計算した後，前述のcollect _free_pages を行う。これにより必要なページ数が回収できたのであれば，トラップの処理はこの時点で終了する。

それでも実ページの回収が必要ページ数に満たないとき，メモリーマネジヤはcollect _garbage(collect_stack_garbage）による実ページの回収をはかる。

4．collect_stack_garbage のインプリメンテーション

システム立上げ時のOSの主記憶への展開が終わった直後におけるメモリ使用状況の調査によれば、PSI-Iでは各プロセス平均して、グローバルスタックエリアは9割以上がゴミであった。プロセス固有のスタックエリアで最も多くメモリを使用するグローバルエリアにこの様にゴミが残りやすいと判明したため、PSI-IIでは新たにプロセス単位のGC（Collect _stack _garbage ）をサポートしている。このGCでは、共有エリアであるヒープ領域へのポインタはすべてアトミックと見なされて処理されるので、他プロセスへの影響がなく、通常処理を行う他のプロセスと並行してGC処理を行なうことも可能である。

Collect _stack _garbage はCollect _garbage のマイクロルーチンを大部分で共用しており、この組込述語の為に新たに要したマイクロコードの容量は数十ステップ程度である。

5．Collect _garbage のインプリメンテーション

PSI-I では全エリアをGC時には1枚の論理空間と見なして行なうGC方式を採用しているが、PSI-IIでは多重論理空間のためにそのままの方式を用いることができないために「ring法」を採用している［2］。

ring法では、マーキング時において、スタックから共有ヒープへのポインタは、ポインタのかかれていたスタック

Garbage Collector of PSI-II (2) -- Implementation --
Y.Kawada*1, H.Tateno*2, K.Nakajima*3
*1:SET, *2:Mitsubishi Electric Co.,Ltd, *3:ICOT

にセル、ポイントされたヒープのセル共々、ringを指すように向きを変える。又、ヒープ内での上向きポインタも同様に、ringに向きを変える。これは、上向きポインタのメンテナンスをマーキング時に行なうことにより、ポインタの向きを一方向に揃え、モリスの方法［３］でのコンパクションを1pass で終える為である。 ring法ＧＣで使用されるring（ワーク領域）は、プログラムのロード時や実行時に共用ヒープを指すポインタが生成される時に確保される分と、ローカルエリアがのばされる時に確保される分とがある。

ＧＣの実行以前にゴミとなったポインタなどもある為、使用されなかったring領域が存在する。ＧＣルーチンでは、全プロセスのコンパクション終了時にring量の再評価を行ない、不必要なringはＧＣ終了と共に解放している。

尚、collect _stack _garbage, Collect_garbage におけるスタックエリアのマーキングはPSI-Ⅰ同様静的に確保されているワーク領域を使った方法がとられ、スタックエリアのコンパクションもPSI-Ⅰと同じモリスの方法に基づくアルゴリズムが採用されている。

また、これら２つの組込述語では各エリアのコンパクションは行なうが、未使用となった実ページの回収は行なわない。この回収は、前述の組込述語（collect _free_ pages ）によって行なわれる。

## ６．ＧＣのモジュール構成

図１にＧＣ用マイクロルーチンのモジュール構成を示す。

（１）ＧＣの起動に必要な初期設定とエラーチェックを行う。

（２）プロセスの実行に必要な情報は、システムエリアにあるＰＣＢ（process control block)と呼ばれる場所に格納れている。マーキングルートは、このＰＣＢから読み出される。

（３）コンパクションは、ＧＣ対象エリア全てのセルのマーキングが終了した後に行われる。

（４）collect stack garbage では、ringを使用しないため再評価は不要である。

尚、ＧＣ用マイクロルーチン全体でのコード容量は約2.0Kである。

## ７．まとめ

PSI-IIに実装されたＧＣは、スタックエリアのマーキングとコンパクションに関しては、PSI-Ⅰと同じアルゴリズムを採用している。PSI-IIでは、多重論理空間を導入したことにより、ヒープ領域のマーキングにring法を採用したが、この点がPSI-Ⅰとの大きな相違である。

PSI-IIのＧＣシステムは、ソフトウエアで管理されている。これは、全プロセスＧＣ、１プロセスＧＣの２種類のＧＣ機能の選択をソフトウエアに任せることによって効率の良い、柔軟な処理が望めるからである。

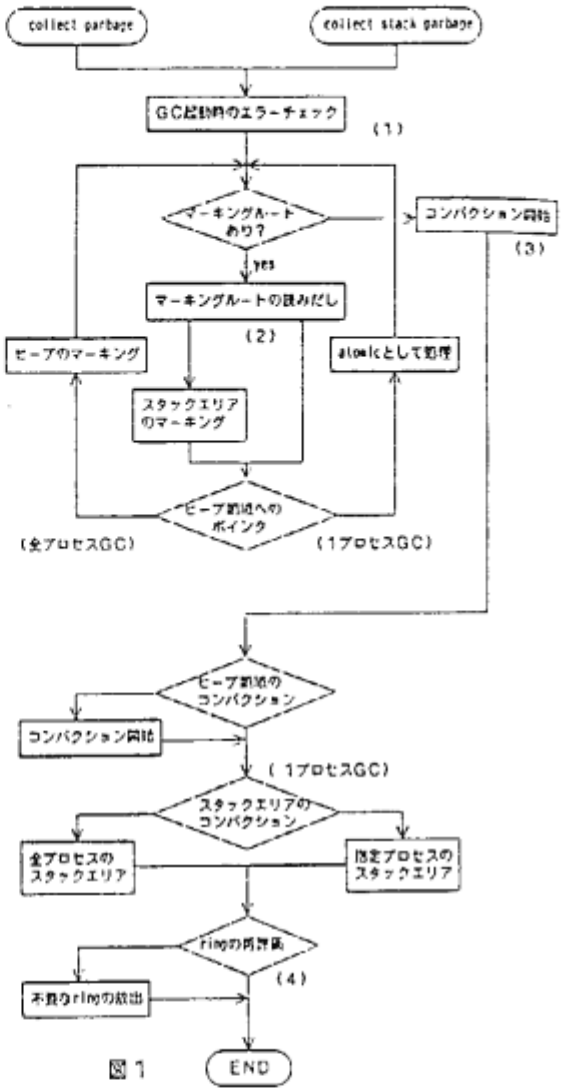

図１

## おわりに

本研究にあたり多くの貴重な助言をいただいたICOT及び関連メーカーの方々、またコーディング、テストにご協力いただいたメルコム サービス（株）長谷川氏、 菱電エンジニアリング（株）白水氏 に深く感謝します。

## 参考文献

［１］：吉田ほか、 マルチPSI 要素プロセッサPSI-IIのメモリ管理とプロセス管理 第３３回情報処理全国大会 １９８６

［２］：立野ほか、 PSI-IIのＧＣ（１） 第３５回情報処理全国大会 １９８７

［３］：Horis.F.L. A time-and Space-Efficient, Garbage Compaction Algorithm.of the ACM,21,8,1978