

TM-0351他

情報処理学会 第35回全国大会 論文集4-2
(並列推論システム)

©1987. ICOT

ICOT

Mita Kokusai Bldg. 21F
4-28 Mita 1-Chome
Minato-ku Tokyo 108 Japan

(03) 456-3191-5
Telex ICOT J32964

Institute for New Generation Computer Technology

TM-0351	PSI-IIのGC(1)－多重論理空間のGC－	立野裕和、池田守宏(三菱)、西川 宏 (松下技研)
TM-0352	PSI-IIのGC(2)－全体構成－	川田易治(エスイーティ)、立野裕和(三菱)、中島克人
TM-0353	PSI-IIにおけるESPサポート用ファームウェア	村澤 靖、池田守宏(三菱)、吉田裕之(沖)、近山 隆
TM-0354	PSI-IIにおけるKLO実行順序系組込述語の実現方法	中島 浩、立野裕和(三菱)、木村康則
TM-0355	PSI-IIのコンパイラ	近藤誠一(三菱)、佐藤泰典(沖)、近山 隆
TM-3556	PSI-IIの性能評価(1)－概要と速度性能－	中島 浩、近藤誠一、吉田裕之(沖)、 中島克人、稲村 雄
TM-0357	PSI-IIの性能評価(2)－機会命令評価－	近藤誠一(三菱)、中島克人、稲村 雄
TM-0358	PSI-IIの性能評価(3)マイクロ命令動的評価－	吉田裕之(沖)、中島 浩(三菱)、 中島克人
TM-0359	マルチPSI第2版のハードウェア	武田保孝、中島 浩、益田嘉直(三菱)、 瀧 和男
TM-0363	並列論理型OS-PIMOS(1)－資源管理方式－	佐藤裕幸、越村三幸、近山 隆、瀧 和男
TM-0365	並列論理型言語GHCによる論理回路設計 ルール検証プログラムの記述	佐藤令子、太細 孝(三菱)、瀧 和男
TM-0396	KL1の並列処理－実用規模KL1プログラムの動特性評価	細井 聡、岸本光弘、久門耕一、服部 彰 (富士通)

P S I - II の G C (1)

— 多重論理空間の G C —

6B-3

立野裕和 池田守憲 西川憲
(三菱電機㈱) (松下技研)

1. はじめに

第五世代コンピュータ・プロジェクトの一環としてマルチPSIフロントエンドプロセッサPSI-IIを開発した。

PSI-IIでは、プロセスごとに大きな論理アドレスを与え、なおかつ生成されるプロセス数の制限をなくするために、プロセス単位に独立した論理アドレスを与えている。我々は、これを多重論理空間と呼んでいる⁽¹⁾。

動的なメモリ管理を必要とする論理型言語の処理系などでは、ガベージコレクションが必須となるが、多重論理空間を採用した事により、その実現に困難が生じる。

本稿では、多重論理空間に対するガベージコレクション方式としてPSI-IIで実現したリング法を報告する。

2. PSI-II ガベージコレクタ概要

PSI-IIのガベージコレクタは一括型のマーク&コンパクション方式を採用している⁽²⁾。ガベージコレクション機能は、組込み述語としてソフトウェアに提供され、指定されたプロセスのスタック領域のみを対象とした「コレクト・スタック・ガーベッジ」と、共有ヒープ領域（プログラム等が置かれてる全プロセス共通の領域）を含む主記憶全体を対象とした「コレクト・ガーベッジ」の2種類が準備されている⁽³⁾。

3. 多重論理空間に於けるガベージコレクションの問題

3.1 多重論理空間

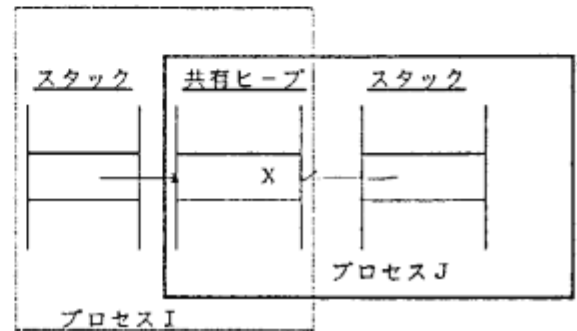
PSI-IIでのKLO処理系は、グローバル、ローカル、トレイルの3種のスタック領域と共有ヒープ領域及びソフトウェアとファームウェアの交信など特殊な用途に用いられるシステム領域の5つの領域を使用する。PSI-IIでは、プログラムの実行単位をプロセスと呼び、各プロセスのスタック領域には、プロセスごとに独立な論理空間を与え、ヒープ/システム領域には全プロセスで共通な論理空間を与えた。

各スタック領域には、プログラムの実行に伴い共有ヒープ領域へのポインタが生成される。プログラム実行中の主記憶イメージを図1に示す。

3.2 ガベージコレクション上の問題

通常のガベージコレクションアルゴリズムでは、対象とする主記憶領域が全てガベージコレクタから見えている事が前提となる。換言すれば、ガベージコレクションの対象となるアドレス空間は1種のみである。

しかし、多重論理空間では、図1に示したようにアドレス空間はプロセス単位に独立している。このようなポインタが現われた場合ガベージコレクタはポインタのメンテナンスを行なう為にスタック上の共有ヒープを指しているセルとヒープ上の指されているセルとの間にポインタを張る。しかし、多重論理空間では、あるプロセスから他のプロセスにポインタを張ることは出来ない。このために、各プロセスから共有されている領域に対して張られたポインタの扱い方が大きな問題となる。



プロセスI、Jが共有ヒープのセルを指している
図1

4. リング法によるガベージコレクション

4.1 基本的な考え方

プロセス間にまたがるポインタを張れないので、スタックからヒープへのポインタはガベージコレクションのマーキング時に全て一度中間テーブルを経由させる事にする。我々はこの中間テーブルをリングと呼んでいる。リングは全てのプロセスからもヒープからもアクセス出来る領域（実際には、前述のシステムエリア）にとる。

マーキング時にスタックからヒープへのポインタが現われると、ポイントされていたヒープのセルの内容をリングにコピーし、もとのヒープセルとそれをさしていたスタックのセルにはリングへのポインタを書く。

このようにしてマーキングが終了した後、まずヒープをコンパクションする。基本的にはスイープ・コンパクションの方法（Morrisのアルゴリズム）⁽⁴⁾で行なうが、ポイントされていたヒープセルにはリングへのポインタが書かれている。そこで、このようなセルをコンパクションする場合、リングから本来そのセルに書かれ

ていたデータを読み出す。リングには、リングを指していたセルの移動先を書き込む。リングから読み出したデータは、コンパクションしようとしているセルに書かれていたものとみなして Morris の方法にしたがった処理を行なう。

スタックのコンパクションはヒープのコンパクション後に行なう。ヒープへのポインタが書かれていたセルにはリングへのポインタが書かれているのでリングの内容をコピーするだけでヒープへのポインタのメンテナンスをおこなうことが出来る。

4.2 最適化

(1) リングの量を減らす

既にのべた方法ではリングとして必要な領域は、全スタック上のヒープへのポインタ数となる。これを避ける為に、同じヒープセルを指すスタックからのポインタは同じリングセルを指すようにすることでリング量はヒープ上のポイントされているセルの総数に出来る。

(2) コンパクションを1パスにする

コンパクションは Morris の方法で行なうが Morris の方法では、ポインタの向きが一方であれば GC 対象領域のスイープを1パスで行なうことが可能である。リング法を実現する為に、既にヒープ上のポイントされているセルの総数分の領域がとられている。この領域を使いスタック上からのポインタのみでなく、このポインタからたどる事の出来るヒープ上のポインタも含めてマーキング時にリング領域を使用してヒープ上のポインタの向きを一方に揃える事によってコンパクション時のヒープ領域のスイープを1パスで行なうことが可能である。

4.3 P S I - II に於けるリングの確保方式

リング法において必要なリングの数は、ポインタで指されているヒープ上のセルの総数である。但し、4.2 で述べたコンパクションを1パスで済ます最適化を実現する為にデータ構造におおじて1ポインタ当たり1語または2語のリングを確保している。リングの確保法は静的に見積もる方法と、プログラム実行時に動的に確保する方法の2つが考えられる。静的に、つまりシステムが立ち上がった時点でリング量を見積もることは困難である。なぜなら動的に生成され得るポインタの全てに対処する為には、ヒープ領域とほぼ同じだけのリングを確保しなければならない事になるからである。

動的に、即ちヒープへのポインタの生成時、もしくは指され得るヒープデータの生成時にリングを確保する方法は、通常処理に対するオーバーヘッドとなり得るが、我々は、以下のようにする事に依りそのオーバーヘッドを極小化している。

リングを確保するのは以下の場合である。

- (1) プログラムが実行中にヒープ上にメモリを割り付けそこを差すポインタを生成した場合
- (2) プログラムが実行中にヒープ上を差すポインタをあらたに生成した場合

(3) I P L 時にプログラムを主記憶上に展開する場合

(4) プログラム実行のための制御情報として処理系がヒープを差すポインタをスタック上の環境フレームに書く場合

上に示した内(1)~(3)は全て組み込み述語によって行なわれるので各組み込み述語の内部処理としてリングの確保が行なわれる。(4)はその頻度が大きい事から環境を積むスタック用にメモリを割り付ける際に割り付けるメモリ量の1/4をリングとして確保する。これは、環境フレームの最大1/4が、制御情報としてのヒープへのポインタが書かれるからである。

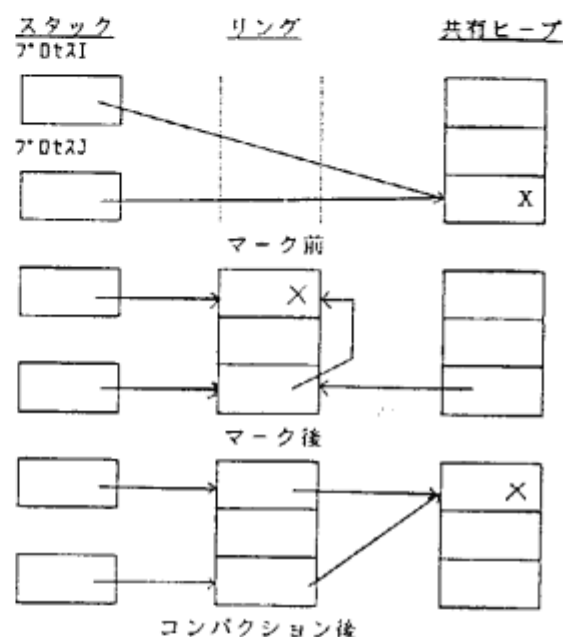


図 2

5. おわりに

本稿では、多重論理空間に於けるガベージコレクションアルゴリズムについてのべた。P S I - II に採用したガベージコレクタでは最適化を行ない、ヒープ領域をコンパクションする際に1パスのスイープでコンパクションを実現した。

最後に、リング法の検討に関し適切なご指導をいただいた I C O T の方々及び関連メーカーの方々に深く感謝する。

参考文献

- (1) 吉田 他 マルチ P S I 要素プロセッサ P S I - 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) 西川、池田 P S I のガベージコレクタ
ICOT TECHNICAL REPORT TR-213
- (3) 川田 他 P S I - II GC (2)
第35回情報処理全国大会 1987
- (4) Morris, F. L.: A time-and Space-efficient Garbage Compaction Algorithm
Comm of the ACM, 21, 8, 1978