

MRB-GC の評価 - アクティブセル量とヒープ消費 -

1Q-8

武井則雄¹, 平野喜芳¹, 西崎慎一郎¹, 森田京子¹, 木村康則², 近山隆³

1: (株) 富士通ソーシアルサイエンスラボラトリ

2: (株) 富士通研究所

3: (財) 新世代コンピュータ技術開発機構

1 はじめに

ICOTでは並列推論マシン(PIM)の開発を進めている。PIM上の言語KL1はフラットGHCをもとにした並列論理型言語である。KL1は副作用を持たない言語である為、メモリの消費速度が速くガーベジコレクション(GC)を頻繁に起こす。このため、GCの方法が実行時間に影響を及ぼす。PIM上のKL1処理系のGC方式として一括型GCとMRB方式によるインクリメンタルGCの併用を検討している。

本稿では、MRB-GCの評価の一つとして実行中のアクティブセル量の推移を測定し、MRB-GC方式を実装した処理系でのメモリ消費との関係を調べた結果を報告する。

2 KL1処理系のGCについて

KL1処理系において実行時の負担を軽くしてインクリメンタルにGCを行なう方法として、MRB(Multiple Reference Bit)方式によるGC[1]を提案している。この方式では、データへの参照数をポインタ側の1ビットのフラグに持たせ、'多数'か'単一'かで表現する。これにより不要となった単一参照データのみを回収対象とする。参照数管理がリファレンスカウント方式などと比較して少ないコストで実現可能である。しかし、一度多重参照となったデータは、不要となっても回収できない。

一括型GCの一つであるコピーイングGCは、GC時のアクティブなセル量にGC時間が比例する。一方MRB-GCは、処理時間中にMRBのメンテナンスを行ない回収をしている為、実行時間の上でオーバーヘッドを生じる。ここで、アクティブセルの割合が大きくなると、一括型GCのコストが高くなる為、MRB-GCによるオーバーヘッドを加えても両者を併用の方がGCコストが低減し、実行時間の点で有利となる。

以上から、MRB-GCと一括型GCの併用を考えた場合、アクティブセルの割合によって一括型GC、MRB-GCの有効性が左右されると考えられる。またアクティブセル量を調べることで、前者のGCコストや後者で回収できない割合などの類推の手掛りになると考える。以

下にアクティブセル量の推移とヒープ消費の測定結果の記述と両者の比較を試みる。

3 実験条件

3.1 処理系

実験に用いた処理系はPDSS[3]を改造したものをを用いた。PDSSはUNIX上に作られた単一プロセッサのKL1処理系で、エミュレータ部分はC言語でかかれており、MRB-GCをサポートしている。

3.2 実験方法

アクティブセルは、一定のリダクション毎に一括GCを起こして、その際にコピーされたデータの数をアクティブセルとしてその数を調べた。ヒープの使用状況は、実行中一定リダクション毎に使用状況をサンプリングして集計している。実行に必要なだけのメモリ量を十分に与え、途中一括GCは起こさずに終了させた。アクティブセル、ヒープ使用にはKL1データとゴール・レコード等の制御レコードが含まれる。

なお、一括GCではデレファレンスをしながらGCをする為、間接ポインタはコピーされない。MRB-GCではデータの消費の際にデレファレンスにより単一参照の間接ポインタのみ回収される。このため、アクティブセル量の測定においては回収済みとなったことによる回収時期の差と、多重参照の間接ポインタの量が以下の測定結果に現れる場合がある。

3.3 ベンチマークプログラム

評価に用いたベンチマークプログラムは以下の通りである。

プログラム名	機能
prime	1000までの素数生成
kl1cmp	KL1自身で書かれたKL1コンパイラ
semi	セミグループ

Evaluation of MRB-GC -Active Cells and Heap Consumption-

Norio TAKEI¹, Kiyoshi HIRANO¹, Shin'ichirou NISHIZAKI¹, Kyouko MORITA¹, Yasunori KIMURA², Takashi CHIKAYAMA³

1: Fujitsu Social Science Laboratory Ltd. 2: Fujitsu Laboratories Ltd.

3: Institute for New Generation Computer Technology (ICOT)

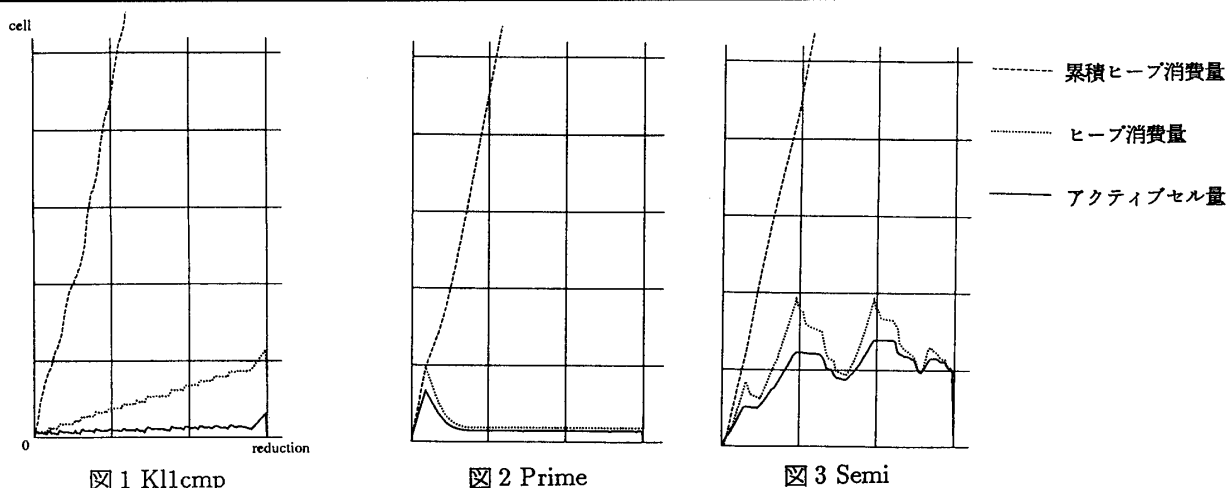


図1 Kllcmp

図2 Prime

図3 Semi

4 評価結果 - ヒープ使用状況との比較 -

図で、横軸はリダクション数を示し実行開始から終了までを表わす。縦軸はセルの量を示し、累積ヒープ消費量の傾きが全てのプログラムで一定になるよう調節した。「ヒープ消費量」はMRB-GCをした場合の使用状況を表わし、「累積ヒープ消費量」は実行に必要な累積のヒープ使用状況を表わしたもので、MRB-GCをしない場合のヒープ使用量に相当する。MRB-GCによる回収量は、ヒープ消費量と累積ヒープ消費量との差として現れ、回収できなかった量は、ヒープ消費量とアクティブセル量との差に現れる。MRB-GCの回収効率はアクティブセル量と累積ヒープ消費量、ヒープ消費量のそれぞれの差の比で示される。

Kllcmp(図1)では、累積ヒープ消費量、ヒープ消費量は単調増加している。しかし、後者の増加速度は前者の10%に満たない。アクティブセル量も増加傾向にはあるが、ヒープの消費に比べほぼ一定状態に近く、量も極少ない。また図より、MRB-GCによって回収できない「ゴミ」が累積して行く傾向にあるが、回収効率は10分の1以下で良いことが分かる。

Prime(図2)は、自然数を1000まで生成し、そのデータから素数を取り出す方法を取っている。自然数を生成するまでの三者の増加速度はほぼ同じである。その後、ヒープ消費量とアクティブセル量の推移は似ており、素数をふるい分ける段階からMRB-GCを行なうことで消費した不要なデータが回収され、緩やかに減少して一定の値(生成する素数の数)に近くなる。この間「ゴミ」は生成されず両者の差は小さい。累積ヒープ消費量はその後単調に増加し、これに比べヒープ消費量・アクティブセル量は極少ない。これより、MRB-GCの回収効率は良いものと言える。

Semi(図3)ではヒープ消費量の推移とアクティブセル量の推移は似た傾向を示し、「ゴミ」も溜まって行く様子は見られない。推移そのものは凹凸がはっきりしており、山と谷が交互に現れる。これは、ストリームによってメッセージのやりとりをしている為、その生成・消費による増減と考えられる。一方、累積ヒープ消費量はほぼ単調に増加しており、アクティブセル量はこれの5分

の1程度である。回収効率は5分の1程度でMRB-GCによる回収効率は良いものと言える。

5 考察

実験結果からMRB-GCを行なう場合と行なわない場合のGCの効率を比較する。Kllcmp,primeのようにMRB-GCの回収効率は良いが、アクティブセル量が極端に小さい場合は一括GC時のコピー量が増大する見込みはあまりなく、GCコストという面ではあまり有利にはならないことがあることが分かった。一方semiのように、MRB-GCをしない場合のヒープ消費量と比べある程度の量を持つ場合、与えるメモリ量の変化によって一括GC時にコピーするデータの総量が増加することもあり得る。つまり、MRB-GCによるGC全体のコストの低減ができる可能性が高くなることになる。

以上より、MRB-GCの回収効率という点ではプログラムによるとはいえ良いものと言える。GCコストの低減については、今回の結果からは直接判断できないが、メモリ量に対するアクティブセル比が40~50%を超えるとMRB-GCを行なった方がGCコストを低減できるという報告がある[2]。これより、semiのようにMRB-GCを行なうことでアクティブセル量に近いヒープ消費量で実行でき、アクティブセル量も少なくない場合、上記のようなGCコストの低減が得られると考える。

謝辞

日頃御指導頂いているICOT第4研究室内田俊一室長をはじめ、研究員の方々に感謝致します。

参考文献

- [1] Chikayama, T. and Kimura, Y. Multiple Reference Management in Flat GHC. Proc. ICLP'87 1987
- [2] 木村 他. KL1の多重参照管理によるガーベジコレクション. 情処. プログラミング言語研究会 1989.
- [3] 平野 他. 汎用計算機上のKL1処理系-PDSS-. 情処. プログラミング言語研究会 1989.