

KLIC への世代 GC の実装に対する予備的実験

6 U-8

吉川 隆英 近山 隆

東京大学 工学系研究科

{taka, chikayama}@logos.t.u-tokyo.ac.jp *

1 はじめに

KLIC[2] は可搬性と効率性の両立を目標に構築された、並列論理型言語 KL1 の並列処理系である。

KLIC では、自動メモリ管理を行っており、ガベージコレクション (GC) の実行時間は通常プログラム実行時間全体のうち無視できない時間を占める。従って、GC の性能は処理系の性能を左右する大きな要因の 1 つである。

KLIC では管理用データ構造もヒープ上にとるので、短寿命のデータが多い。そのため、プログラムによりアクセスされる頻度が極めて低い有効である時間が極めて長いデータがヒープ上に混在していると、これらも GC の度にコピーされるため、そのようなデータが多量に存在すると、全体としてはかなりのオーバヘッドとなる。

このような「寿命の長い」データに対処できる別な GC 方式を用いることにより GC の性能を向上させる事が考えられる。そのような GC 方式の代表的なものが「世代 GC」である。

本稿では、KLIC で世代 GC を実装するにあたってその効果を予測するためヒープ上のデータがどのように振る舞うかを調べ、その結果を示す。

2 測定方法

測定にあたり、KLIC の逐次核に手を入れた。GC を経験した回数によって、データに割り付けるメモリ空間を変更することにより、GC をある回数経験したデータの量、及び、ある回の GC でごみとなったデータの量などが測定できる。

3 評価方法

以下の小規模なベンチマークプログラムについて、

- ・ ある回数 GC を経験した後に回収されたデータがごみ全体の中で占める割合。
- ・ ある回数 GC を経験したデータのうちで、次の GC の後でも生き残るものの割合。

を測定した。

- hanoi.kl1
「ハノイの塔」。17 枚の円盤を移す手数を数えるプログラム。
- kkqueen.kl1
「N-Queen」。9 個のクイーンを盤面に配置するプログラム。
- primes.kl1
「エラトステネスのふるい」のアルゴリズムにより 100,000 以下の素数を全て生成し、その個数を調べるプログラム。
- random.kl1
KLIC に含まれる乱数生成器を用いて 40,000 個の乱数を発生させるプログラム。
- qlay.kl1
layered-streams を用いて「N-Queen」を解くプログラム。クイーンの数 9 とした。
- KLIC
KLIC に添付されるテストプログラムのうち life.kl1, puzzle.kl1, wave.kl1, turtles.kl1, mastermind.kl1, pascal.kl1, qlay.kl1 をコンパイルしたときに測定したデータを平均したもの。

測定には AT 互換機 (AMD K6-233MHz, Memory 128MB), FreeBSD2.2.5, KLIC は 3.002 版に修正を加えて用いた。

4 評価結果

結果を図 1, 2 に示す。

図 1 は、ある回数 GC を経験した後に回収されたデータがごみ全体でどのくらいの比率を占めているかを示している。この図から回収された全データのほとんどが 1 回目の GC で回収されていることが分かる。これは、KLIC が管理用のデータまでもヒープ上に

*Evaluation of performance of generational collection for KLIC

Takahide Yoshikawa and Takashi Chikayama
School of Engineering, the University of Tokyo
7-3-1 Hongo, Bunkyo-ku, Tokyo, 113-8656, Japan

割り当ててしまうため短時間でゴミとなってしまいうデータが非常に多いことを示している。

図2は、ある回数 GC を経験したデータのうちで、その次の GC でも回収されなかったものの割合を示している。プログラムの種類により多少異なるが、gc を数多く経験したデータは長く生き続けるという傾向を見て取ることが出来る。

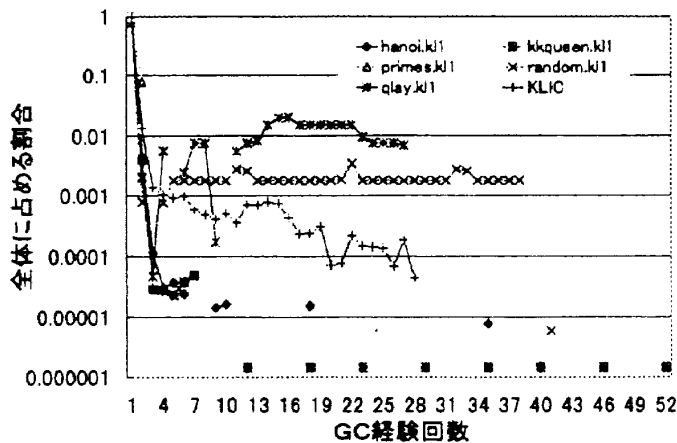


図 1: ある回数 GC を経験したゴミの割合。

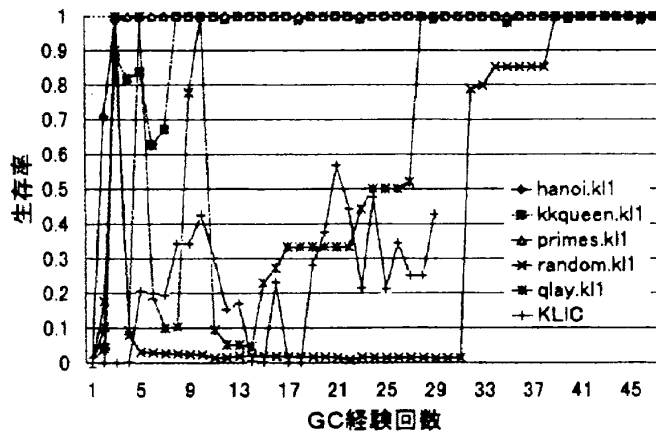


図 2: ある回数 GC を経験したデータの生存率。

以上のことから、KLIC 上でも GC を 2 回以上経験したデータを「長寿命なデータ」として別に取り扱うことが妥当であるといえる。

さらに、ゴミ全体の中で短寿命のものが相当量を占めるので、これらを素早く回収する為には、短寿命のデータに対する GC を短い周期で行うことが有効であると考えられる。

従って、KLIC の性能向上には、世代 GC の適用が有効であると考えられる。

5 おわりに

KLIC の逐次核に手を入れることによって実行時におけるヒープ上のデータの振る舞いを調べた。

その結果、KLIC では長寿命のデータと短寿命のデータとが混在しており、また短寿命のデータが多いため、これらをわけて GC を行う世代 GC 方式が非常に有効であることが確認された。

今後は、この結果を元に KLIC に世代 GC を実装していく予定である。

参考文献

- [1] Paul R. Wilson: Uniprocessor Garbage Collection Techniques,
<ftp://ftp.cs.utexas.edu/pub/garbage/bigsurv.ps>, 1992.
- [2] T. Chikayama, T. Fujise, Daigo Sekita: A Portable and Efficient Implementation of KL1, PLILP'94, 1994.
- [2] 関田 大吾, Inside KLIC,
<ftp://ftp.cs.utexas.edu/pub/garbage/bigsurv.ps>, 1992.