並行オブジェクト向けの分散GCの一方式

4V－7

渡部卓雄
北陸先端科学技術大学院大学
情報科学研究科

瀬尾明志
日本ユニシス（株）

1. はじめに

分散計算環境における大域的ガベージコレクションの一手法を提案する．本方式はリファレンスカウント方式とマーキング方式を併用したアルゴリズムを採用し，以下のような特長を持つ：(1)循環する参照を含むすべてのゴミを回収できる．(2)アプリケーションプログラムと並行に動作する．(3)比較的オーバーヘッドの高いマーキングフェーズの稼働率を低くおさえている．(4)リファレンス情報，マーキング，終了判定のメッセージに重みを用いて，メッセージ数を減少させている．(5)集中制御を必要としない．本方式によるGCをCM-5，nCUBE/2 上に実装し，他のアルゴリズムとの比較も行った．

2. 対象とする計算システム

本研究のGCアルゴリズムは分散メモリマルチプロセッサないしはネットワークで結合されたワークステーションクラスタを対象とする．ここでノード間の通信は非同期メッセージによるものとし，任意の2ノード間についてメッセージの受信順序は送信順序を保存する（FIFOnessを持つ）と仮定する．

3. 大域的参照の管理

大域的な参照の管理は，各ノード毎に輸入参照表と輸出参照表をそれぞれ一つ置くことによって行う．ノード外のオブジェクトへの参照は輸入参照表に，ノード外から参照されるオブジェクトは輸出参照表にそれぞれ登録される．輸入参照表は，いわゆる代理(proxy)オブジェクトとみなすこともできる．

あるノードAにおけるオブジェクトxの参照がノードBに送られるとしよう．このときxのローカルアドレスはAの輸出参照表に登録され，Aのノードidと輸出参照表のオフセットの組がxの大域的アドレスとしてBに送られる．一方，xの大域的アドレスを送られたBは，それを自分の輸入参照表に登録する．B内部では，輸入参照表のエントリをxのアドレスとみなす．

4. GCのアルゴリズム

4.1. 基本方針

本方式によるGCのシステムは，ノード間にまたがる参照を対象とするグローバルGCと，各ノード内のローカルGCに分けることができる．

ローカルGCは，大域変数，アクティブなオブジェクト，未処理メッセージ，輸出参照表をルートとし，これらから参照関係（の推移閉包）によって到達できないオブジェクトをゴミとして回収する．

一方グローバルGCでは，ノード外から参照されなくなった輸出参照表のエントリがGCの対象となる．グローバルGCによってオブジェクトそのものが占めていたメモリエリアが解放されることはない．

輸入／輸出参照表によって大域的参照を管理しているため，ローカルGCとグローバルGCは独立に行うことが可能であり，ローカルGCには任意の方式を用いることができる．以下，本稿ではグローバルGCについてのみ説明する．

グローバルGCはさらに，リファレンスカウント部とマーキング部に分けられる．以下にその概要を説明する．

4.2. リファレンスカウント部

リファレンスカウントによるグローバルGCは，アプリケーションの実行と並行して随時行われる．輸出参照表の各エントリには，ノード内のオブジェクトへの参照に加えて，ノード外からの参照数を格納するためのフィールドがある．このフィールドの値が0になったエントリはゴミとして回収される．

参照数の増減はノード単位で行われる．例えばあるオブジェクトへの参照が，そのオブジェクトの参照を持たないノードにコピーされるとき（そしてそのときのみ），参照数は1だけ増える．つまりノードの内部で他のノードのオブジェクトへの参照をコピーしても参照数は変化しない．一方，ローカルGCによって輸入参照表のあるエントリがゴミとなったときに，そのエントリが指しているオブジェクトの参照は1だけ減少する．したがって，参照数の増減を伝えるメッセージ数は実際の参照の増減の回数よりも低くおさえることができ，また参照数はノード数を超えることはない．

参照の増減はメッセージによって伝えられるが，ここでは非同期メッセージを仮定しているため，ナイーブな方法で同期を取るとメッセージ数が増大する．本方式では Weighted Reference Counting (WRC)[2] ないしは Generational Reference Counting (GRC)[4] を用いてメッセージ数の減少を図っている．

4.3. マーキング部

マーキングの開始

リファレンスカウント部によってゴミであると認識できない輸出参照表エントリ（複数のノードにわたって循環した参照関係がある）は，マーキング部によって回収される．ローカルGCによってゴミが十分回収されなかったノードがマーキングの開始宣言をする．これは適当な放送アルゴリズムや放送機構を用いて全ノードに伝えられる．

メッセージ中の参照の確保

各ノードは，マーキング開始メッセージを受信したら，そのノードに入ってくるメッセージのなかの参

照を獲得する．これにはいくつかの方法が考えられるが，実装するプラットフォームにあわせて最も良い方法を採用する．

CM-5のように未処理メッセージのクリア命令を持っている並列計算機では，それを用いる．

・ブルドージング[8]

・分散スナップショット[3]

・メッセージカウント[5]

・大域的な参照関係の記録

以下のようにして矛盾のない参照関係を記録する．

1) 輸出参照表の複製を作る．

2) 輸出参照表の複製の各エントリ（eとする）について，ノード内参照によってたどれる輸入参照表のエントリを見つけてeに記録する．

このようにして作った輸出参照表の複製のエントリを集めたものが大域的な参照関係となる．この参照関係は安定であるため，以下のグローバルマーキングは輸出参照表の複製を用いてアプリケーションと並行に行うことができる．

マーキング

各ノードは，輸出参照表（複製ではない）以外のルートからたどることの出来る輸出参照表の各エントリについて，その実体を持つノードにマーキングメッセージを送る．

マーキングメッセージは，(1)輸入参照表が参照しているオブジェクト名，(2)適当な自然数値の重み，(3)マーキングの起点となったノード名を含んでいる．

マーキングメッセージを受け取ったノードは，輸出参照表の複製のエントリのうち，(1)に相当するものにマークを付ける．マーク済みの場合は(3)の起点ノードに(2)の重みを返却する．マークしたエントリが指している輸入参照表の各エントリについて，その実体を持つノードにマーキングメッセージを送る．このとき重みを適当に分割する．

マーキングの起点となったノードは，返却された重みの合計が自分が発行した重みとなることで，マーキングが終了したことがわかる．全体の終了判定は，Spanning Tree を用いた一般的なアルゴリズムやバリア同期命令等を用いて行う．

ゴミの回収

マーキングが終了した時点で，マークされていない輸出参照表の複製のエントリはゴミである．対応する（複製でない）輸出参照表のエントリを解放しておけば，あとはローカルGCとリファレンスカウントによって自動的にオブジェクトも回収される．

5. 評価

5.1. 実装

本方式による分散GCをThinking Machines社のCM-5（64ノード）およびnCUBE社のnCUBE/2（256ノード）にそれぞれ実装し，評価を行った．この実装は，応用プログラムに相当する部分を人工的に作っている．このGCを実際の言語処理系（並行オブジェクト指向言語）に組みこんで，応用プログラムの下で評価する予定である．

メッセージクリアとマーキングの終了判定について2通りの実装をして比較した．第一の実装ではメッセージクリアに分散スナップショットを用い，終了判定に一般的な spanning tree による同期アルゴリズムを用いた．第二の実装では，CM-5 のメッセージクリア命令とバリア同期命令をそれぞれ用いた．

さらに比較のために，Langらによる分散GCアルゴリズム[6]を同一プラットフォーム上に実装した．この実装はグローバルGC以外の部分のコードは我々のGCの実装と同一のものを用いている．

5.2. ゴミの回収比率

リファレンスカウントとマーキングによるゴミの回収比率は以下の通りである．

	ref. counting	marking
CM5	63.8%	36.2%
nCUBE/2	67.2%	32.8%

以上の結果から，比較的オーバーヘッドの大きいマーキングの稼働率を減少させることは成功している．

5.3. メッセージ数

グローバルGCのマーキング１回に必要なメッセージ数を64ノードのCM-5上で比較した．

	我々の方法	Lang[6]
分散スナップショット	4164	8285
CM-5のバリア同期	289	4304

ノード数を変化させて比較測定した結果は[7]を参照されたい．

参考文献

[1] G. Agha, Actors, MIT Press, 1986.

[2] D. I. Bevan, "Distributed Garbage Collection using Reference Counting", in PARLE '87, LNCS 259, pp. 176-187, Springer-Verlag, 1987.

[3] K. M. Chandy & L. Lamport, "Distributed Snapshots: Determining Global States of Distributed Systems", ACM TOPLAS, 3(1), pp. 63-75, 1985.

[4] B. Goldberg, "Generational Reference Counting: A Reduced-Communication Distributed Storage Reclamation Scheme", in PLDI '89, pp. 313-321, 1989.

[5] 鎌田十三郎，松岡聡，米澤明憲，「超並列計算機上の高効率な大域的ガーベジコレクション」，in JSPP '94, pp. 33-40, 情報処理学会, 1994

[6] B. Lang, C. Queinnec & J. Piquer, "Garbage Collecting the World", in POPL '92, pp. 39-50, 1992

[7] 瀬尾明志，渡部卓雄，「並行オブジェクトのためのハイブリッド分散ガベージコレクションの一方式」，Research Report, IS-RR-94-7S, 北陸先端科学技術大学院大学, 1994, (WOOC '94にて発表．ftp.jaist.ac.jpよりコピー可)

[8] N. Venkatasubramanian, G. Agha & C. Talcott, "Scalable Distributed Garbage Collection for Systems of Active Objects", in Memory Management, LNCS 637, Springer-Verlag, pp. 134-147, 1992.