

大域 GC における WTC 方式を用いた マーキング終了検出

六沢 一昭

中島 克人

市吉 伸行 稲村 雄

沖電気工業(株) 総合システム研究所

三菱電機(株) 情報電子研究所

(財) 新世代コンピュータ技術開発機構

1 はじめに

大域 GC におけるマーキングの終了を WTC 方式を用いて検出する方式を述べる。メッセージの送受信数をカウントする方式(例えば [2]) が終了検出方式としてよく知られているが、この方式は基本的に「終了をチェックし、終了していない場合は再度チェックを試みる」方式であるため、「マーキングの終了がすぐには検出されず必ず遅れが生じてしまう」という問題があった。本方式では、WTC 方式 [1] を応用しマーキング自身に終了検出機能を持たせることによってこの問題を解決した。

2 計算モデル

- 局所メモリを持つ有限個のプロセッサ (PE) がネットワークで接続されている。
- 共有メモリはない。PE は非同期メッセージによって互いに通信する。
- メッセージは送信されてから受信されるまでに任意の時間がかかる。このため任意の時刻において、「送信はされたがまだ受信されていないメッセージ(“通信中のメッセージ”と呼ぶ)」が存在しうる。
- 他の PE の局所メモリ中のデータを参照するポインタ(“外部参照ポインタ”)を各 PE は持ちうる。

3 大域 GC とマーキング

システムにはひとつの“ルート”がある。システム内のすべてのデータについて以下の定義に従い有効/無効を決定する処理を“大域 GC”と呼ぶ。

有効 なデータはルートから到達できる。

無効 なデータはルートから到達できない。

データの有効/無効は以下の処理によって決定できる。

ルートを持っている PE(“マスタ PE”と呼ぶ) はルートからポインタをトレースし出会ったデータにマークを付ける。ポインタでないデータあるいはマークの付いているデータに出会うとそのトレースは終了し、ペ

クデータに出会うと分散する。外部参照ポインタに出会った場合はマークを付けるとともに外部参照ポインタの指す PE へ“マーキングメッセージ (mark)”を送信する。mark は外部参照ポインタが指すデータのアドレスを運ぶ。mark を受信すると mark が指すデータからマーク付けを始める。以上の「ルートからトレースしてマークを付ける」処理を“マーキング”と呼ぶ。マーキングによってマークの付いたデータは有効であり、マーキングの終了した時点でマークの付いていないデータが無効である。

マーキングの終了検出

マーキングの終了とは以下が成り立つことである。

- すべての PE がアイドル (マークするデータがない)。
- 通信中のメッセージが存在しない。

マーキングの終了検出は容易ではない。すべての PE がアイドルであっても通信中の mark が存在するかもしれないので終了が保証できないからである。

4 メッセージカウント方式の問題点

この方式は基本的に「適当な時刻に終了のチェックを開始し、終了していない場合は適当なタイミングでチェックを再度試みる」方式である。このため以下の問題点がある。

- 終了チェックの開始が遅過ぎると終了検出が遅くなり、早過ぎると無駄なチェックが増えてしまう。
- 再度のチェックまでの時間が長いと検出の遅れが大きくなり、短いと無駄なチェックが増えてしまう。

5 終了検出機能を持ったマーキング

マーキングの終了をチェックするのではなく、それ自身が終了検出機能を持ったマーキング方式を述べる。この方式は終了検出の方式である WTC 方式をマーキングに適用したものである。

すべてのアイドルでない PE と mark が重みを持ち (マスタ PE は負の整数、その他は正の整数)、合計がゼロになるように制御する。この結果、マスタ PE がアイドルで重みがゼロならばマーキングの終了していることが保証される。

まずメッセージの追い越しのないシステムにおける処理を述べ、次に追い越しのある場合について述べる。

A Scheme for Termination Detection of Marking Phase
in Global Garbage Collection Using Weighted Throw Counting
Kazuaki ROKUSAWA (OKI), Katsuto NAKAJIMA (MELCO),
Nobuyuki ICHIYOSHI, You INAMURA (ICOT)

5.1 メッセージの追いつきのない場合

5.1.1 マスタ PE の処理

ルートからマーキングを開始する。

外部参照ポイントに出会うと *mark*を送信する。*mark*には重みを付け、その分だけ自分の重みを減らす。

*mark*を受信した時は *mark*の重みを自分の重みに加え、*mark*が示すデータからマーキングを行なう。

return(後述)を受信すると 重みを自分の重みに加える。

request(後述)を受信したならば *supply*を返信して重みを供給する。自分の重みは供給した分だけ減らす。

マスタ PE が「アイドルかつ重みゼロ」になったならばマーキングの終了が検出される。

5.1.2 マスタ PE 以外の PE の処理

*mark*を受信するとマーキングを始める。

外部参照ポイントに出会うと 以下の処理を行なう。

自分の重みが2以上ならば *mark*を送信する。重みを2つの正の整数に分割し、一方を自分の新しい重みに設定し、もう一方を *mark*に付ける。

重みが1の場合は 分割ができないので、重みを要求するメッセージ(*request*)をマスタ PE へ送る。*request*には重みを付けない。*mark*の送信は「重みの供給」あるいは「*mark*の受信」によって重みが2以上になるまで待つ。

*mark*を受信した時は メッセージが持っていた重みを自分の重みに加え、マーキングを行なう。

*supply*を受信したならば 以下の処理を行なう。

アイドルならば *supply*の重みを返却メッセージ(*return*)でマスタ PE に返す。

アイドルでなければ 重みを自分の重みに加える。

アイドルになった場合は 自分の重みを *return*でマスタ PE に返却する。

5.2 マーキングが終了していることの説明

マスタ PE がアイドルで重みもゼロならばマーキングの終了していることを以下に述べる。

*request*を除くすべてのメッセージとマスタ PE 以外のアイドルでない PE は正の整数の重みを持つ。このためマスタ PE の重みがゼロであることは「マスタ PE を除くすべての PE はアイドル」で「*request*以外の通信中のメッセージも存在しない」ことを保証する。一方 PE は *request*の送信後 *return*を必ず送信するので、*request*をネットワークに残したままマスタ PE の重みがゼロになることもない。以上のことから、マスタ PE がアイドルで重みもゼロならばマーキングの終了が保証できる。

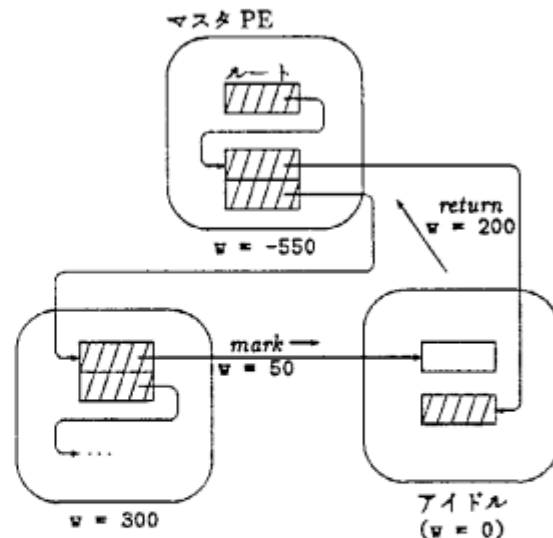


図 1: マーキングの一場面

5.3 メッセージの追いつきのある場合

メッセージの追いつきがある場合は上述した方式は正しく働かない。*request*が *return*によって追いつかれると *request*をネットワークに残したままマスタ PE の重みがゼロになってしまう可能性があるからである。これを防ぐには以下のいずれかの変更を行なえばよい。

- *request*の到着を確認してから *return*を送信する。
- *request*にも重みを付ける。

5.4 メッセージカウント方式との比較

本方式における検出の遅れは1メッセージの送受信にかかる時間(終了 ~ 最後の *return*受信)だけであり、これはメッセージカウント方式よりも明らかに小さい。

一方、マーキング処理自身は重みの割り付け / 回収を行なうためコストが高くなる。また *request*が頻繁に送信されるような状況が起こると終了そのものが遅くなってしまふおそれがある。

6 おわりに

WTC方式の応用により終了検出機能を持ったマーキング方式を述べた。この方式はマーキングの終了を直ちに検出することができる。

参考文献

- [1] K. Rokusawa, N. Ichiyoshi, T. Chikayama and H. Nakashima, "An Efficient Termination Detection and Abortion Algorithm for Distributed Processing Systems," ICPP88, Vol.I, pp.18-22, 1988.
- [2] F. Mattern, "Algorithms for Distributed Termination Detection," Distributed Computing, Vol.2, pp.161-175, 1987.