大規模分散システムのためのグループメンバシップ・プロトコル

3M－2

早原茂樹　芦原評　清水謙多郎
電気通信大学 情報工学科

1　はじめに

分散システムにおいて、複数のプロセスやプロセッサをメンバとして一括して制御、管理するグループ機構は、負荷分散やファイルレプリケーション等のシステムレベルのサービスから、分散データベース、グループウェア等のアプリケーションに至るまで、重要な用途を持つ[2]。

本論文では、グループは動的に変化し、グループの生成、消滅、メンバの加入、削除を繰り返すものとする。そのため、各メンバが、所属グループの最新のメンバシップを把握し、そのメンバシップを更新できる操作が必要となる。以下では、大規模広域分散システムへの適用を想定し、

(1) 大域的なロックやタイムスタンプを用いない、
(2) メッセージの送受信順序の不一致に対応する、
(3) 特定の制御メンバを必要としない(完全分散型)、
(4) 更新操作は非封鎖型である、

ことを特徴としたグループメンバシップ・プロトコルの方式を提案する。

2　グループメンバシップ・プロトコル

分散システムは、ノードの集合と論理的な通信路から構成される。各ノードは、分散システム内で大域的に識別され、他の全てのノードに対してメッセージを送信できる。メッセージは必ずしも送信順に受信されるとは限らず、メッセージが到着するまでの時間は、有限であることのみが保証される。また、メッセージは重複して受信されることはないものとする。これらは容易に実現可能である。

グループ G に対するメンバシップ更新操作には、*Insert*(加入) と *Delete*(削除) がある。これらの操作は、G 内のメンバが互いに更新メッセージをやりとりすることにより実行される。この更新メッセージを U で表わす。また、U には要求と応答があり、必要に応じて要求を $U^{(r)}$、応答を $U^{(a)}$ と表わす。

メンバ M_x が送信・受信した U の系列は、ログ L_x に保持される。また、所属グループに対する k 番目の操作を U_k で表わす。L_x 内に登録されている U_i と U_j について $(i<j)$、$U_i \xrightarrow{L_x} U_j$ は、U_i が U_j よりも先行したことを表わすものとする。また、M_x による U の送信操作を $send_x[U]$ により表わす。

複数のメンバで送受信される U の間の関係と順序保存について、以下のものを定義する。

情報同値: L_x と L_y が、全く同じ U の集合を含む。

順序同値: L_x と L_y に含まれる全ての U_i と U_j について、$U_i \xrightarrow{L_y} U_j$ ならば、$U_i \xrightarrow{L_x} U_j$ である。

順序保存: L_x と L_y に含まれる全ての U_i と U_j について、$send_y[U_i], send_y[U_j]$ であり $U_i \xrightarrow{L_y} U_j$ ならば、$U_i \xrightarrow{L_x} U_j$ である。

因果関係保存: L_x と L_y に含まれる全ての U_i と U_j について、$send_y[U_j]$ であり $U_i \xrightarrow{L_y} U_j$ ならば、$U_i \xrightarrow{L_x} U_j$ である。

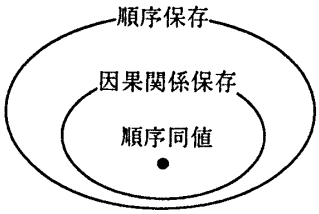

図 1: 更新メッセージ U の間の関係

順序同値ならば、全てのメンバのログ内の順序系列が一貫しているために、メンバシップリストの一貫性が保証される。しかし、大規模分散システムにおいて、全ての U の順序同値を実現するには、コストが大きく適さない。

そこで、メンバシップ更新操作が必ず因果関係にあること注目する。メンバシップ更新操作は、対象メンバが G に加入していれば削除要求の U が、そうでなければ加入要求の U が送信される。これは、同一対象メンバに対する $U^{(r)}$ どうしの関係を因果関係保存にすることで、ログには加入、削除の U が交互に登録されることになる。つまり、グループメンバシップ・プロトコルにおいて、$U^{(r)}$ どうしの関係が因果関係保存ならば、順序同値となる。

メンバシップが一貫したと判断する時点を全てのメンバから $U^{(a)}$ を受信した時点とするために、同一メンバから送信される $U^{(r)}$ と $U^{(a)}$ の関係を順序保存とする。これは、$U^{(r)}$ を送信し、全てのメンバから $U^{(a)}$ を受信するまでの間に、並行して他のメンバから送信された $U^{(r)}$ は、全てのメンバから $U^{(a)}$ を受信するまでの間に、必ず受信されることを保証する。

本プロトコルは、$U^{(r)}$ を送信し、全てのメンバから

Group Membership Protocol in Large Scale Distributed Systems
Shigeki Hayahara, Hyo Ashihara, Kentaro Shimizu
The University of Electro-Communications
1-5-1 Chofugaoka,Chofu-shi,Tokyo,182,Japan

$U^{(a)}$を受信するまでの間は、メンバシップについて一時的な矛盾を許す。そして、全てのメンバから$U^{(a)}$を受信した時点で、全てのメンバが把握するメンバシップが一貫する。

3 プロトコルの設計

3.1 データ構造

各メンバは、G内でメンバシップ更新操作が行われた回数であるグループシーケンスカウンタsと、未到着の$U^{(r)}$の個数を記録している要求待ちカウンタwを保持する。sはUに付加して送受信され、wは$U_t^{(r)}$を受信したときのsの現在の値とtの差が記録される。$U_s^{(r)}$を送信した場合、全てのメンバから応答を受信し、かつw=0になったときに、メンバシップが一貫したと判断される。

各メンバは、メンバシップを管理するための配列M_LISTを保持する。M_LISTの各要素は、Gに登録された各メンバに対応する。Gに登録されている現メンバ数$n(1\leq i\leq n)$に対して、M_LIST[i].nameは、メンバ名(下位通信レイヤにより位置付けが行われる), M_LIST[i].sは、そのメンバが登録されたときのsの値, M_LIST[i].opは、そのメンバがグループに加入しているか、そうでないかを表わす。

3.2 アルゴリズム

以下では、更新操作をOP、更新操作の対象として指定されたメンバをM_oで表わす。

```
初期値 s := 0 ; w := 0 ;
[U_s^(r) の送信]
  s := s+1 ;
  for each i, 1≤i≤n do
    send U_s^(r) to M_i
  od
[U_t^(r) の受信]
  if s = t then
    s := s+1 ;
    [M_LIST の更新]
  else if s < t then
    w := w+(t−s−1) ;
    s := t ;
    [M_LIST の更新]
  else
    if w ≠ 0 then
      w := w−1 ;
      [M_LIST の更新] ;
      if w = 0 then
        [M_LIST のリフレッシュ]
      fi
    else
      s := s+1 ;
      [M_LIST の更新]
    fi ;
  fi ;
  send U_s^(a)
[U_t^(a) の受信]
  if s < t then
    w := w+(t−s) ;
    s := t
  fi
[M_LIST の更新]
  if ∃i, M_LIST[i] = M_o then
    if M_LIST[i].s < t then
      M_LIST[i].s := t ;
      M_LIST[i].op := OP
    fi
  else if ∀i, M_LIST[i] ≠ M_o then
    n := n+1 ;
    M_LIST[n] に M_o を登録 ;
    M_LIST[n].s := t ;
    M_LIST[n].op := OP
  fi
[M_LIST のリフレッシュ]
  for each i, 1≤i≤n do
    if M_LIST[i].op = "Delete" then
      n := n−1 ;
      M_LIST[i] の抹消
    fi
  od
```

4 おわりに

[1]は、メンバシップの一貫性について必要とするレベルに応じたプロトコルを選択できる機構を提案しているが、本研究のようにメンバシップの因果関係に注目したプロトコルではない。[3]は、放送型通信アルゴリズムの受信順序について整理したものである。本研究は、その結果の一部を利用し、新たなグループメンバシップ・プロトコルについて論じたものである。[4]は、メッセージにタイムスタンプを格納することで、メンバシップの一貫性を実現しているが、完全な分散型でなく、一貫したという判断基準がないため、常に同一プロセスが発するメッセージのタイムスタンプを比較しなければならない。[5]は、大規模分散システムにおいて、メンバのレベルを分け、効率の良いグループ通信を説明しているが、メンバシップの一貫性を保証するものではない。本研究は、大規模分散システム内でグループを構成したメンバの動的な変化に対して、可用性と信頼性のあるメンバシップ管理が実現できるものと考える。

参考文献

[1] F.Jahanian, S.Fakhouri, R.Rajkumar, "Processor Group Membership Protocols:Specification, Design and Implementation", *Proc.13th.Int.Conf.Distributed Computing Systems*,2-11,1993.

[2] L.Liang, S.T.Chanson, G.W.Neufeld, "Process Groups and Group Communications:Classifications and Requirements", *IEEE Computer*,*23*,2,56-66,1990.

[3] 滝沢誠, 中村章人, "放送型通信アルゴリズム", 情処論,*34*,11,1341-1349,1993.

[4] X.Jia, H.Nakano, K.Shimizu, M.Maekawa, "Highly Concurrent Directory Management in the GALAXY Distributed System", *Proc.11th.Int.Conf.Distributed Computing Systems*,416-423,1991.

[5] O.Babaoglu, A.Schiper, "On Group Communication in Large Scale Distributed Systems", *Proc.14th.Int.Conf Distributed Computing Systems*,1994.