180
数理解析研究所講究録
第666巻 1988年 180-189

リーダー選出問題における時間最小アルゴリズムについて

京都大学工学部　大戸　豊　（Yutaka OHTO）
京都大学工学部　茨木　俊秀　（Toshihide IBARAKI）

あらまし

分散システムにおいては、集中型のシステムにはなかった種々の問題が生じる。その一つとして、コントロールトークンの再生において、その作業をつかさどるプロセッサを定めたり、またファイル等の資源管理を指示するプロセッサを決定することがしばしば要求される。これが、リーダー選出問題である。本報告では、リーダーが選出されるまでの時間を減少させることに重点をおき、時間最小の分散アルゴリズムを提案する。

1．まえがき

複数のプロセッサを通信リンクでつないで構成する"分散システム"は集中型システムに比べ、1）ＣＰＵやメモリ等の資源を共有できる、2）信頼性が向上する、3）負荷の分担が可能である、等の利点がある。ところが分散システムでは、集中型システムにはない新たな問題が発生してくる。分散システムには中央コントローラが存在しないため、例えば、システム内にコントロールトークンを流すことで各プロセッサの競合を制御する方式が考えられるが、そのコントロールトークンが消失した場合、新たにコントロールトークンをつくる"リーダー"と呼ばれる一つのプロセッサを、システム内から選び出す必要が生じる。あるいは、分散システム内のファイル等の資源管理をつかさどるプロセッサの決定が要求されることもある。これがリーダー選出問題である。

システムが非同期で、形態が一般グラフの場合、メッセージ数$\Theta(n \log n + m)$［n、mはそれぞれシステム内のプロセッサ数とリンク数］の分散アルゴリズムが提案されている[1,2,3,4]。この中で、文献[1]のアルゴリズムは時間$(n \log^* n)$を持つ。

ところで、リーダー選出問題におけるこれらの研究では、リーダー決定までに必要なメッセージ数の最小化が中心に議論されてきた。それは、メッセージの伝送にかかる通信コストが計算コストに比べ高価であり、いかにメッセージを少なくするかということが実用上重要であるからである。しかし、最近光ファイバーなどの進歩により、通信リンクの容量は増加し、その結果、通信コストは低下の傾向にある。またＬＡＮ（ローカルエリアネットワーク）においては通信コストはほとんど無視できるという状況にあり、メッセージ数より、リーダー選出までの時間の減少がより重要な場合も考えられる。

そこで本報告においては、時間を減少させることに重点を置き、リーダー選出問題の時間最小アルゴリズムを提案する。そのために、まず、すべてのプロセッサにおいてシステムの形態を示すグラフを構築する問題を考え、一つの時間最小アルゴリズムを与え、このアルゴリズムがグラフ構築までに必要とするメッセージ数は、$(2m^2-mn+m)$以上、$(2m^2-mn+2m)$以下であることを示す。グラフが構築できれば、その情報にもとずいてリーダーを選出できるから、このアルゴリズムはリーダー選出アルゴリズムでもある。本報告の後半では、グラフ構築問題の任意の時間最小アルゴリズムは、$m(m+1)$個のメッセージを必要とすることを示す。したがって、上のアルゴリズムはメッセージ数のオーダーの意味で最良である。さらに、上のアルゴリズムがリーダー選出問題のアルゴリズムとしても時間最小であることを示す。

2. 諸定義と仮定

ここでは、準備段階として本論文で想定する分散システムに関する諸定義および仮定を行う。これらは、分散システムに対し、通常用いられるものである（例えば[6]参照）。本報告においては、分散システムにおけるプロセッサおよびそれらを結ぶリンクをそれぞれグラフ$G=(V,E)$で表現する。

V：節点(プロセッサ)の集合。
E：枝(リンク)の集合。

また、$n=|V|$、$m=|E|$とする。以下、プロセッサを節点、リンクを枝と呼ぶ。

[定義1](分散システムにおける性質・仮定)

1) グラフGの形態にはあらかじめ何の制限もおかない。但し、有限性および連結性は仮定する。またアルゴリズムの実行中にGは変化しない。
2) 中央コントローラは存在しない。
3) 節点はメッセージを送ることによってのみ通信できる。通信は、一般に非同期で行なわれる。またメッセージは誤りなしに送られる。
4) 同じ枝を通ってきたメッセージの順序は変わらない(FIFO)。また枝上で反対向きのメッセージがお互いに影響を及ぼすことはない。
5) 各々の節点は"id"と呼ばれる一意的な値を持っており(以下の議論では、簡単のため節点とそのidを同一視して扱う)、また何個の節点とつながっているかという次数degを知っていて、それらを区別できる(すなわち接続枝(ポート)には番号1,2,・・・,degが付されている)。しかし、初期状態において各節点はこれら以外に何の情報も持たない。

6) 各々の節点は局所的メモリおよび、任意個のメッセージを保てるバッファを持つ。
7) 節点内での処理およびメッセージの発信受信に要する時間は、枝上の伝送の遅れと比較して無視できる。また、一つの枝上のメッセージの伝送時間は、それが発せられた時刻によって定まり、メッセージの内容にはよらない。
8) ２つ以上のメッセージが１つの節点に同時到着することはない（適当に順序をつけバッファに入れることができる）。 □

ところで、ｎ個の節点の中から１つをリーダーとして選出するには、各節点の持つ一意的なidにもとずいて、その中で最大のidをもつ節点を選べばよい[5,6]。そのようなアルゴリズムを評価する際、アルゴリズムが始まってからリーダーが決まるまでに必要な時間およびメッセージ数が問題になる。そのためには、アルゴリズムの起動およびリーダーが決まる、ということの定義を明確にしておく必要がある。

[定義2](アルゴリズムの起動) 静止状態にあるｎ個の節点のうちｐ個($1 \leq p \leq n$)がリーダー選出の必要性を知り、他にメッセージを送ることでリーダー選出アルゴリズムを起動する。リーダー選出の必要性を示すメッセージを受け取った節点は、そのことを他に知らせると同時に自己のリーダー選出アルゴリズムを起動する。□

[定義3](リーダー決定) リーダーが決まったとは次の２つをともに満足する時のことをいう。
1)最大のidをもつ節点が、自分のidが最大であることを知る。
2)その他のすべての節点は、自分のidは最大ではなく、しかも最大のidが(つまり、リーダーが)みつかったことを知る。□

３．アルゴリズムとその考え方

節点ｋ(ｋはid)において、ID_k、$deg_k(a)$、$ADJ_k(a)$(ただし、$a \in ID_k$)は、それぞれ、その時点で節点ｋがグラフＧ中に存在することを知っている節点のidの集合、節点ａ($\in ID_k$)の次数、節点ａ($\in ID_k$)に隣接している(つまり、枝で結合されている)ことが判明している節点の集合である。各節点ｋが最初に受けるメッセージは(i,p,d,ε)タイプであり、このメッセージは、1)節点ｉの次数はｄである、2)節点ｉはそのｐ番目の枝によって節点ｋに隣接している、という情報を持つとともに、アルゴリズムの起動を知らせる役目も果たしている。εはnull情報を意味する。もう一つのメッセージのタイプは、(i,p,d,j)であるが、これには、1)節点ｉの次数がｄである，2)節点ｉはそのｐ番目の枝によって節点ｊに隣接している，という情報が含まれている。したがって、これらのメッセージを

受けると、各節点kではデータID_k、$deg_k(a)$、$ADJ_k(a)$、$a \in ID_k$を更新する。また、各節点kは、(i,p,d,j)(あるいは(i,p,d,ε))のメッセージを受けたとき、ア)枝(i,j)の情報((i,p,d,ε)の場合は枝(i,k)の情報)をすでに知っている、イ)自分は葉節点である、の場合以外は、そのメッセージをそのまま((i,p,d,ε)の場合は(i,p,d,k)に直して)送られてきた以外のすべての方向へ送る。なお、ア)とイ)のどちらも成立しない場合の特別な場合として、i＝k、つまり自分の出したメッセージが一巡して戻ってきたという場合が考えられるが、このときには隣接する節点へはすでにkおよびその次数の情報は送られているので、メッセージは、(i,ε,ε,j)でよい。特に、p番目の枝の相手はjであるから、jに対してこのメッセージをp番目の枝を通して送る必要はなく省略できる。

各節点kでは、あるメッセージを受けて、その結果ID_k、および$a \in ID_k$に対する$deg_k(a)$、$ADJ_k(a)$からグラフの全体が構築できたとき、そのメッセージの他への発送をすませた後、アルゴリズムを停止する。このとき、ID_kにはグラフ内のすべての節点が入っているから、そのなかで最大のものをリーダーとすればよい。

リーダーが何であるかは、1)構築されたグラフにもとずいて自分で決定する、2)他節点から知らせてもらう、などの方法で、知ることができる。一番最初にリーダーがわかる節点は他節点から知らせてもらうことはできないから、自分で決定しなければならない。リーダーは最大のidをもつ節点であるから、リーダーを決定するには、グラフの全体が構築できればよい。節点kでのグラフの構築は、データID_k、$deg_k(a)$、$ADJ_k(a)$($a \in ID_k$)がすべて完成することと同値である。ここではまず全節点がグラフを構築するというグラフ構築問題を考え、その分散アルゴリズムを与える。

グラフ構築アルゴリズム　A

各節点$k \in V$はそれぞれ独立に以下のアルゴリズムを実行する。

(1)　起動前：$ID_k := \phi$.

(2)　起動：$ID_k := \{k\}$、$deg_k(k) :=$(自分の節点の次数)、$ADJ_k(k) := \phi$とする。メッセージ(k,p,$deg_k(k)$,ε)を、自分に隣接するp番目の枝($p = 1,2,\cdots,deg_k(k)$)を通して送る。[εはnull情報を示す。]

(3)　メッセージ(i,p,d,j)を受け取った時(p、d、jの一部がεである場合も含む)：

Step 0(起動)：

(a) if $ID_k = \phi$ [節点kはまだ起動していない。このとき受けとるメッセージは(i,p,d,ε)タイプのみである]

(b) then $ID_k := \{k\}$、$ADJ_k(k) := \phi$、$deg_k(k) :=$(自分の節点の次数)とし、メッセージ(k,q,$deg_k(k)$,ε)をq番目の枝

($q=1,2,\cdots,deg_k(k)$)を通して送る。

Step 1（データの更新）:

(a) if $j \neq \varepsilon$、$i \in ID_k$ and $j \in ADJ_k(i)$[メッセージ(i,p,d,j) or (j,p',d',i)をすでに受け取ったことがある]あるいは$j=\varepsilon$、$i \in ID_k$ and $k \in ADJ_k(i)$[メッセージ(k,p'',d'',i)をすでに受け取っている]

(b) then exit

(c) else[枝(i,j)の情報が新しく得られた]

$ID_k := ID_k \cup \{i\}$および、$d \neq \varepsilon$ならば$deg_k(i) := d$とする;

(i) $j = \varepsilon$の場合

$ADJ_k(k) := ADJ_k(k) \cup \{i\}$;

$ADJ_k(i) := \{k\}$;

Step 2へ。

(ii) $j \neq \varepsilon$の場合

$ADJ_k(i) := ADJ_k(i) \cup \{j\}$;

$ADJ_k(j) := ADJ_k(j) \cup \{i\}$;

Step 2へ。

Step 2（メッセージの発送）:

(a) if $deg_k(k) = 1$ [kは葉節点]

(b) then Step 3へ

(c) else [kは葉節点ではない]

if $i = k$[自分へ戻ってきたメッセージの処理]

then メッセージが送られてきた方向および、p番目の枝を除いて、すべての枝にメッセージ$(i,\varepsilon,\varepsilon,j)$を送る。

else[$i \neq k$]

もし $j = \varepsilon$ならば、$j := k$として、それ以外ならばjはそのままでメッセージ(i,p,d,j)を送られてきた以外の方向へ送る。Step 3へ。

Step 3（グラフ構築の完成テスト）:

(a) if $|ADJ_k(a)| = deg_k(a)$ for all $a \in ID_k$ [グラフの構築は完成]

(b) then halt[以後、メッセージを受けても何の動作もしない]

(c) else exit. □

なお、Step 3の(b)は、アルゴリズムの停止を明示するために加えたもので、メッセージの発送に関しては影響を持たないことを注意しておく。この部分がなくても、グラフの構築が完成していれば、Step 1 (a)(b)によって、新しくメッセージを送り出すことはないからである。

４．アルゴリズムＡの正当性

枝の情報とは、その枝の端点をｉ，ｊとしたとき、ｉとｊが隣接しているという情報のことを言う。グラフを構築するには、全節点の次数と全枝の情報をすべて受け取らなくてはならない。

［補題1］アルゴリズムＡによれば、すべての節点は有限時間内にすべての枝の情報を受け取ることができる。□

次に、節点の次数に関する情報を考える。各節点ｋはメッセージ(i,p,d,j)と(j,p',d',i)の一方のみを他へ発送する。その理由は、上の補題で述べたように、これらのメッセージは枝(i,j)について同等の情報を持つからである。しかし、前者には、ｉの次数のみが、また後者にはｊの次数のみの情報が含まれているので、これらのメッセージは、次数に関して同等の情報を持つわけではない。このため次の補題が必要である。

［補題2］アルゴリズムＡによれば、節点ｋが他の節点ｋ'から発せられた一つのメッセージ(最初(k',p',d',ε)の形しており、その後(k',p',d',j)の形に変更される)を受け取ったとき、そのメッセージが通ってきた経路上のすべての節点について、それらの存在と次数の情報をすでに受けとっている。□

補題２において、ｋ'の次数も当然ｋに伝わる。したがって、補題１と補題２をすべてのｋに適用すれば、次の定理が言える。

［定理1］有限時間内にアルゴリズムＡは停止し、すべての節点はグラフを構築できる。　□

５．時間量の評価

すべての節点でグラフを構築するアルゴリズムの時間最小性を次のように定義する。

［定義4］(時間最小性) 同一の起動条件の下で、任意の節点ｓにおいて、ｓが節点ｋの次数を知る時刻、および、枝ｅの情報を得る時刻が、任意のｋ、ｅについて、他のどのアルゴリズムと比べても遅くないとき、その分散アルゴリズムは時間最小であるという。□

時間最小性を達成するアルゴリズムの一つとして、アルゴリズムＡ*を考える。Ａ*は次のように動作する。

・起動はアルゴリズムＡと同様に行なう。
・メッセージを受けとったら、自分のメモリ内に新しく加えて記憶す

る。その後直ちに、

・現在自分がもっている情報をすべて、隣接する節点に(メッセージが送られてきた節点も含めて)送る。

明らかにこれ以上早く情報を伝達することは原理的にありえない。任意の枝の情報、および節点の次数の情報について、各節点に最初に到着する時刻をアルゴリズムAとA*で比較することで、次の定理が成り立つ。

[定理2] アルゴリズムAは時間最小である。□

システムは一般に非同期であり、上の結果は一般に成立するものであるが、グラフ構築に要する時間の簡単な評価を得るため、枝上の伝送にちょうど一単位時間かかる同期的な動作を仮定して時間量を求めてみる。Gの2節点sとtの距離dis(s,t)をsとtを結ぶ最小枝数の経路の枝数と定義し、Dをネットワークの直径、すなわち最も離れている2節点間の距離とする。

[補題3] システムが同期的に動作する場合、アルゴリズムAにおいて、節点sから発せられたメッセージ(s,p,d,ε)あるいは(s,p,d,s')が枝e=(i,j)を通るとすれば、dis(s,i)=dis(s,j)=Dの場合には、(D+1)単位時間に、それ以外の場合には、D単位時間以内に通っている。□

[定理3] システムが同期的に動作するとき、アルゴリズムAによってすべての節点がグラフを構築するまでに必要な時間は、一斉スタートの場合、L単位時間である。ただしLは次のように定義される。ある節点kに対し、dis(k,i)=D、dis(k,j)=Dをみたす枝(i,j)∈Eが存在するとき、L=D+1。その他の場合、L=D。□

6・メッセージ数の評価

次に、アルゴリズムAのメッセージ数を評価する。そのため、一つの枝(i,j)に関するメッセージ(i,p,d,j)あるいは(j,p',d',i)の伝送の様子に注目する。

[補題4] アルゴリズムAにおいて、一つの枝(i,j)に関するメッセージ(i,p,d,j)あるいは(j,p',d',i)がGの同じ枝を同方向に2回以上流れることはない。□

[補題5] アルゴリズムAにおいて、一つの枝(i,j)に関するメッセージはGの各枝に多くて2回(しかもそのときは向きが反対)流れるだけである。□

［補題6］ アルゴリズムＡにおいて、枝(i,j)と節点ｋ(≠i,j)を考えると、ｋに接続する枝の中で、(i,j)に関するメッセージをｋから外へ送り出さないものがちょうど一本存在する。□

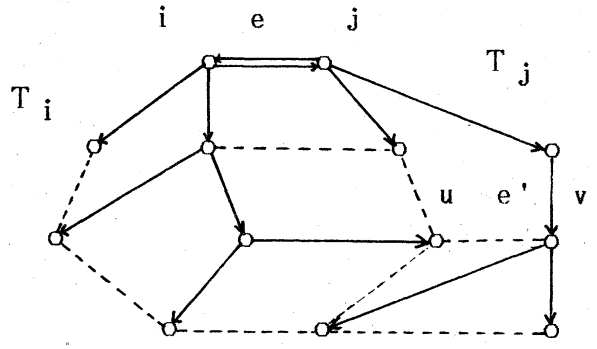

図１　　枝ｅ=(i,j)の情報を運ぶ木T_iとT_j

つまり、枝ｅ＝(i,j)に関するメッセージ(i,p,d,j)と(j,p',d',i)の伝搬形態に注目すると、(i,p,d,j)は節点ｊで作られ、ｊからｉ以外の節点へ送り出され、(j,p',d',i)はｉで作られ、ｉからｊ以外の節点へ送り出される。これらのメッセージが節点ｋ(≠i,j)に最初に到着する経路は、それぞれｊとｉを根とする２つの木T_jとT_iによって表わされる(図1)。T_iとT_jは互いに素でしかも、両者と枝(i,j)を合わせると、Ｇの全域木となっている。補題６によってT_iとT_jの枝には(i,j)に関するメッセージがちょうど一回流れる。ｅ'=(u,v)を(i,j)でもなければT_iとT_jのどちらにも属さない任意の枝とする(図１の点線の枝)。枝ｅ'にはその両端ｕとｖから(i,j)に関するメッセージが送られ途中で交差する。つまり、ｅ'には(i,j)に関するメッセージが２個流れる。最後に枝ｅ=(i,j)には、(i,p,d,ε)と(j,p',d',ε)の２個のメッセージが流れる(Step0(b))。これら全てをあわせると、メッセージ数は、

$$2(m-n+2)+(n-2)=2m-n+2$$

である。なお例外的な場合として、ｉから出たメッセージ(i,p,d,ε)に起動されて、Step0(b)によってｊで作られたメッセージ(j,p',d',ε)がｉに届く以前に、(i,p,d,j)がｉに到着する場合があり、このときはメッ

セージが最初に届いた経路が作る木は、jを根とする全域木一個である。このとき、メッセージ数は、

$$2(m-n+1)+(n-1)=2m-n+1$$

になる。アルゴリズムAのメッセージの送り方から、m個の枝(i,j)すべてについて他の枝とは独立にメッセージが送られるので、結局次の定理が成り立つ。

[定理4] アルゴリズムAのメッセージ数は、$2m^2-mn+m$以上、$2m^2-mn+2m$以下である。□

7. 時間最小のグラフ構築に必要なメッセージ数

以下の議論では、メッセージの単位として、枝一つおよびその両端点の情報(すなわち端点の次数、メッセージを送った枝番号)をとる。したがって、たとえば、一つの経路に関する情報は、このような単位に分解した形で送られるものとする。メッセージ数とは、このように分解されたメッセージの個数である。このとき、次の定理が成り立つ。

[定理5] 時間最小という条件のもとでは、全節点でグラフを構築する任意の分散アルゴリズムにおいて、メッセージ数m(m+1)は必要である。□

8. リーダー選出問題とグラフ構築問題

今までリーダー選出問題をグラフ構築問題におきかえた議論をしてきた。グラフが構築できれば、リーダーは選出できるから、アルゴリズムAを実行することで、有限時間内にリーダーを選出できる。しかし、リーダーは自分で決定しなくても、他から知らせてもらうことも可能であるから、全節点がグラフを構築しなくてもよい場合も考えられる。この場合、リーダー選出問題とグラフ構築問題では、メッセージ数、時間量に差がでてくる可能性あるが、次の定理が成り立つ。

[定理6] リーダー選出問題はどの節点においてもグラフ構築問題よりも早く解けることはない。□

謝辞

日頃ご討論いただく研究室の諸氏に感謝いたします。なお、本研究は一部、文部省科学研究費によるものである。

文 献

(1) Gafni,E.:"Improvements in the Time Complexity of Two Message-Optimal Election Algorithms", In Proc. 4th Ann. ACM Symp. Princ. Distr. Comp.:175-185(1985).

(2) Afec,Y. and Gafni,E.:"Time and Message Bounds for Election in Synchronous and Asynchronous Complete Network", In Proc. 4th Ann. ACM Symp. Princ. Distr. Comp.:186-195(1985).

(3) Gallager,R.G., Humblet,P.A. and Spira,P.M.:"A Distributed Algorithm for Minimum-Weight Spanning Trees", ACM TOPLAS 5:66-77(1983).

(4) Korach,E., Moran,S. and Zaks,S.:"Tight Lower and Upper Bounds for Some Distributed Algorithms for a Complete Network of Processors", In Proc. 3rd Ann. ACM Symp. Princ. Distr. Comp.:199-207(1984).

(5) LeLann,G.:"Distributed Systems - Towards a Formal Approach", in Information Processing 77:155-160(1977).

(6) Liestman,A.L. and Peters,J.G.:"Distributed Algorithms", Technical Report TR86-10, School of Computing Science, Simon Fraser University, June(1986).