25. 分散型アルゴリズムの基礎†

徳　田　雄　洋††

1. はじめに

個々の計算システムを通信によって結合したシステムを，**分散型計算システム**と呼び，この分散型計算システムの上で問題解決を行うことを目標として設計されたアルゴリズムを，**分散型アルゴリズム**と呼ぶ．分散型アルゴリズムは，一般に次の3つの特徴を持っている．

- システム全体は1つの集中制御の下にない．
- 結果として，並列に処理を行うことができる．
- 個々の計算システム間で，データ情報や制御情報を渡しながら，全体の計算が進行して行く．

分散型アルゴリズムは，計算機科学の歴史の比較的初期から早くも登場し，同一構造を持った有限状態機械を平面等に配置して，生命現象や複雑な計算現象の模倣を行うアルゴリズムの研究がJohn von Neumann以来行われてきた[1),23),34)]．

これらの研究は，ある意味で純粋に知的な探究であったが，近年VLSI技術と通信技術の急速な発達により，実用上の効率と信頼性の追求に基づく新しいタイプの分散型アルゴリズムの研究が活発化してきた．

今日の分散型アルゴリズムの対象となる情報処理の分野は，例えば次のように極めて広範囲である．

- 広域電子計算機網[36)]
- 分散型データベース[3),4),21)]
- ローカル・ネットワーク[31),32),35)]
- 多数の処理装置を含むVLSIシステム[14),20),25)]

適用分野の多様性とともに，分散型アルゴリズムの処理形態もさまざまであり，例えば以下のように各種異なるものが存在する．

- 個々の計算システムの上で主に計算現象が起こるものと，通過して行くデータの上で主に計算現象が起こるものとがある．
- 個々の計算システムのどの対の間でも直接の通信が可能なものと，特定の対の間でのみ直接の通信が可能なものとがある．
- 分散型システム全体の通信ネットワークの形状を前もって個々の計算システムが知っている場合とそうでない場合がある．

分散型アルゴリズムは一般に，データ参照の局所性を活用した高速のローカル処理が可能で，計算機資源の共有と負荷分散が図れ，個々の計算システムの小さな障害にも柔軟に対応でき，また自然な並列処理による処理の高速化が実現できるという長所を持っている．

しかしながら反面，分散型アルゴリズムは，通信のためのオーバヘッドの増大，分散化されたデータベース間の無矛盾性の保証，通信路や個々の計算システムの信頼性の保証，正当性の確認や検証の複雑さ，デッドロック等の現象の存在といった問題も含んでおり，アルゴリズムの世界で一般的市民権を得るには，もう少し時間がかかりそうである．

本稿では，分散型アルゴリズムの基礎的問題として**局所的排他制御問題**と**大域的終了性判定問題**の2つを取り上げ，いずれもEdsger W. Dijkstra[12),13)]らによる流儀で，問題の定義，解の提示，解の正当性の証明を述べる．

最初に示すDijkstraの局所的排他制御問題の解[12)]は，非常に強力な仮定を設けることによって，単純な正当性の証明が可能になっている．したがって実用上の見地からは，そのままの形での実現は困難であるが，その正当性の証明は教育的価値が高いと思われるので取り上げた．

次に示すDijkstraとScholtenによる大域的終了性判定問題の解[13)]は，グラフの上の分散型アルゴリズムとして一般性も高く，また実用上の見地からも有用である．

---

† Basic Distributed Algorithms by Takehiro TOKUDA (Dept. of Computer Science Yamanashi University).

†† 山梨大学工学部計算機科学科

2. 局所的排他制御問題

オペレーティング・システムの古典的排他制御問題の1つである，**食事する哲学者の問題**[10]を，一般の無向グラフの上で解決する問題である．

問題[12]（局所的排他制御問題）

有限な無向グラフ上の各点に哲学者1名，各線にフォーク1本をそれぞれ配置する．各哲学者は思索状態と食事状態を繰り返している．各哲学者は食事状態の開始時に，自分の点に接合している線すべての上のフォークを取り上げ，食事を行い，終了時にフォークをもとの位置に返却する．また各哲学者は，必要なフォークがすべてそろわなければ，食事状態に入れない．なお思索状態の時，各哲学者はフォークを全然使用しない．更に，各哲学者は自分と線で隣接している哲学者，すなわち隣人の状態のみを知ることができる．以上の条件の下で，どの哲学者も飢えることなく，思索状態と食事状態を円滑に繰り返すことができるような方式を設計せよ．（**図-1** 参照）

補　足

ⅰ）各哲学者は，思索状態と食事状態の他にも状態を持ってよい．各哲学者の状態遷移は一瞬のうちに起こると考えてよい．

ⅱ）各哲学者は，最初みな思索状態である．隣接する2哲学者の状態遷移の時刻は一致しないと考えてよい．

解（Edsger W. Dijkstra[12] の解）

各哲学者は，思索状態，待ち状態，食事状態の3状態を持ち，待ち状態と食事状態を合わせて，着席状態と呼ぶことにする．

各哲学者は，次のステップ [1]-[3] を繰り返せばよい．

[1]（思索状態）思索を行う．食事をしたくなったら，着席状態の隣人すべてに対し，外向きの矢を出し，ステップ [2] へ．

[2]（待ち状態）自分が隣人に対して出している外向きのの矢が，すべて消えたら，ステップ [3] へ．

[3]（食事状態）食事を行う．食事が終了したら自分へ向かっている矢をすべて消し，ステップ [1] へ．

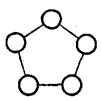


図-1　5人の食事する哲学者の問題を表わす無向グラフ

方　針

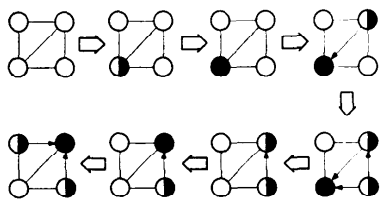
この Edsger W. Dijkstra の解の特徴は，外向きの矢を食事をしたい哲学者が隣人に出すことによって，待ち行列が形成されるところにある．すなわち，自分より以前に，食事状態または待ち状態の隣人に対し，外向きの矢を出すことによって，順番を記憶する．食事を終えた各哲学者は，自分へ向けられている矢を消して思索状態へもどるから，待ち行列の中で，自分の出している外向きの矢がすべて消えた人から，食事状態へ入れるという仕組みである（**図-2** 参照）．

定　理 1（排他制御）

隣接する任意の2哲学者 A，B について，AもBもともに食事状態にいることはない．

証　明

各哲学者から出ていく矢，向かっていく矢の本数と時間の関係をまとめると，**図-3** のようになる．補足と図-3 から次が成立する．



（○：思索状態，◑：待ち状態，●：食事状態）

図-2　4人の場合の状態遷移の例

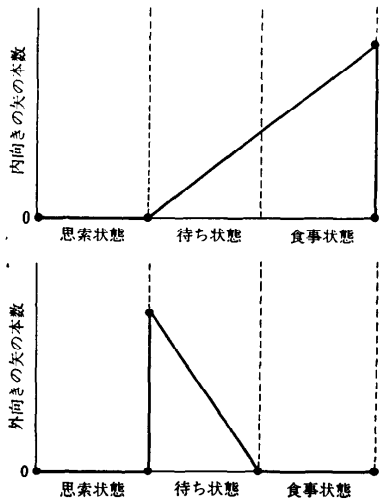


図-3　各哲学者に影響する矢の本数の変化の様子

(性質1) 隣接する任意の2哲学者A, Bについて, AもBも着席状態ならば, AからBへの矢またはBからAへの矢が存在する.

(性質2) 任意の哲学者Aについて, Aは待ち状態であるか, Aは外向きの矢を持っていないかのいずれか一方である.

さて, 哲学者Aは食事状態と仮定する. Bは着席状態か思索状態のいずれかである. Bが思索状態なら題意は成立するから, Bは着席状態としよう. Bが着席状態ならば, 性質1により, AからBへの矢またはBからAへの矢が存在する. 性質2からAは外向きの矢を持たない. よってBからAへの矢が存在する. すなわち, Bは外向きの矢を持っていることとなり, 性質2から, Bは待ち状態であることがわかる. すなわちBは食事状態にいない. (証明終)

定 理 2 (食事につけることの保証)

待ち状態の哲学者は, 必ず食事状態へ到達することができる.

証 明

まず, 次の性質を示す.

(性質3) 外向きの矢は, 巡回する道を構成することはない.

性質3が成立する理由は次の通りである. 矢は外向きにのみ発生できるから, 巡回する道を構成するためには, その最後の瞬間に, 自分へ向かう矢を持った哲学者が, さらに外向きの矢を出さなければならない. しかしながら各哲学者の動作は, 自分へ向かう矢を消した後でなければ, 外向きの矢を出すことはできない.

したがって性質3から, 外向きの矢によって構成される道の長さは有限で, 自分へ再びもどってくることはないことがわかる.

さて, 待ち状態の哲学者Aに対し, 次のような哲学者の集合 $W(A)$ を定義する.

$$W(A)=\{B \mid B \text{ は } A \text{ から外向きの矢印のみを1回以上たどって到達できる}\}.$$

性質3から, $W(A)$ の中には, 外向きの矢を一本も持たない哲学者が存在する. この哲学者の1人をCと呼ぶことにすると, Cは定義から食事状態であり, このCの食事終了時には, $W(A)$ の個数は1以上減少する. $W(A)$ の個数はいつも0以上であるから, 必ずいつか $W(A)$ の個数は0となる. そしてAは食事状態に到達する. (証明終)

3. 大域的終了性判定問題

分散型計算システムの比較的一般的なモデルと考えられる波及型計算システム[13]の上で, 個々の計算システムがすべて終了したかどうかを判定する問題を, Edsger W. Dijkstra と C. S. Scholten の方法[13] で解く.

大域的終了性判定問題の定義を述べる前に, まず一般の有限な有向グラフの上の波及型計算システムについてその定義を直感的に述べる.

[波及型計算システム]

(1) 有限な有向グラフの各点に計算システムを1つずつ配置し, 矢で直接結ばれた2点間では, 矢の方向と同一方向にも反対方向にも情報を送れるものとする. (なお点Aから点Bへの矢が存在する時, BをAの後者, AをBの前者と呼ぶことにする.)

(2) 有向グラフ中には, 入力矢を持たない点がただ1つ存在し, その点を開始点と呼ぶ. 開始点以外の有向グラフ中の点を内部点と呼ぶ.

(3) 各点は, 直接に矢で結ばれた点と, 矢の方向と同一方向または反対方向に情報の通信を行いながら, 個々の計算を進める.

(4) 各点は本来の計算以外に, 矢の方向と同一方向に青信号, 矢の方向と反対方向に赤信号を1回にいずれか1個を送ることができる.

(5) 波及型計算システム全体の計算は, 開始点がその後者すべてへ青信号を1つずつ送ることにより始まる.

(6) 各内部点は, 自分の前者から青信号を受け取ることにより, 各点の計算を始める. 計算を始めた内部点は, データの送受信, 青信号と赤信号の送受信を行って計算を進める. またどの内部点へも少なくとも1つの青信号は送られてくるものとする.

(7) 各内部点は, 自分の計算終了後は, 青信号と赤信号の送受信を1回以上行う. 特に各内部点が最後に送る信号は, いずれかの前者に対する赤信号であるものとする.

(8) 各矢の上を, 矢と同一方向に通過した青信号の総数と, 矢と反対方向に通過した赤信号の総数は, 近隣の点の上の計算が終了してしばらくすると同数となり, 以後無変化となる.

(9) すべての点の計算システムの計算が終了した時, 波及型計算システム全体の計算は終了したと言う.

問 題 (波及型計算システムの大域的終了性判定)

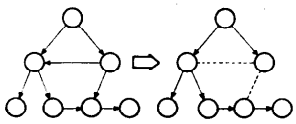


図-4 有向グラフの形が既知の場合の信号路（右図実線矢印）の設定の例

青信号と赤信号の送受信方式を設計し，その方式の下で，開始点のすべての後者から開始点へ赤信号が到着することが，波及型計算システム全体の計算が終了するための必要十分条件となるようにせよ．

方　針

波及型計算システム全体の有向グラフの形状が前もって既知の場合は，開始点を根とするスパン木を作っておき，この有向木に沿って，青信号と赤信号をやり取りすれば，容易に解決できる．しかし本問では，この全体の有向グラフの形状は既知でない．したがって何らかの手段によって，有向木の代用品を作り出さなければならない．（図-4 参照）

解（Edsger W. Dijktra と C. S. Scholten[13] の解）

青信号と赤信号の送受信方式は，以下の条件1から条件3までを満足するように行う．ただし，次の用語を用いる．

矢の信号不足量：矢の上を通過した青信号総数と赤信号総数との差．

点の入力不足量：その点へ向かう矢すべての信号不足量の総和．

点の出力不足量：その点から外へ向かう矢すべての信号不足量の総和．

（条件1）どの矢の信号不足量も0以上をいつも保つ．

（条件2）どの内部点も，

（入力不足量＞0　または　出力不足量＝0）

をいつも保つように，青信号や赤信号を送る．（すなわち，入力不足量＝0 で出力不足量＝0 の時に，自発的に青信号を送ることや，入力不足量＝1 で出力不足量＞0 の時に勝手に赤信号を送ることはしない．）

（条件3）青信号の送り先は任意の後者でよいが，赤信号の送り先となる前者は次のようにして決める．各点に記憶部を持たせ，点Aから点Bへ青信号を1つ送った時は，Bの記憶部に名前Aを1つ記入する．点Bが赤信号を前者へ送りたい時は，点Bの記憶部からなるべく最古でない名前Cを見つけ，Cへ赤信号を1つ送る．そしてBの記憶部から名前Cを1つ削除する．（図-5 参照）

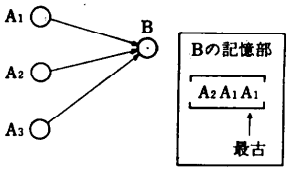


図-5 赤信号の送り先の前者を決める様子

定　理 3（必要性）

すべての点の計算が終了してしばらくすると，開始点の出力不足量は0となる．

証　明

赤信号を送れるためには，条件1から

（入力不足量≧1）

で，しかも条件2から，

（入力不足量＞1　または　出力不足量＝0）

が成立しなければならない．すべての点の計算が終了してしばらくすると，有向グラフ全体の入力不足量の総和が1つずつ減少して行き，ついに赤信号は送れなくなるから，上記の条件の否定が各内部点で成立する．すなわち

（入力不足量＝0）　あるいは

（入力不足量≦1　かつ　出力不足量≧1）

が成立する．つまり，すべての内部点で，

（入力不足量≦出力不足量）．

が成立し，開始点では入力不足量＝0 であることと合わせて，すべての点で，次が成立する．

（入力不足量≦出力不足量）．

有向グラフ全体の入力不足量の総和と出力不足量の総和は等しいから，結局，すべての点で，

（入力不足量＝出力不足量）

となる．したがって特に，開始点では，

（入力不足量＝出力不足量＝0）

となる．（証明終）

定　理 4（十分性）

開始点の出力不足量が（計算開始前を除き）はじめて0となった時，すべての点の計算は終了している．

証　明

まず，（入力不足量＞0　または　出力不足量＞0）の点のことを，作業中の点と呼ぶことにする．また，点Bの記憶部で名前Aが最古の名前である時，AからBへの矢のことを，作業中の矢と呼ぶことにする．

定義から，次の2つの性質は明らかである．

（性質1）作業中の矢は，作業中の点から作業中の点へと出ている．

(性質2) 作業中の内部点に対して，ちょうど1本の作業中の矢がはいっている．

さらに，次の性質3が成立する．

(性質3) 作業中の矢は巡回する道を構成することはない．

性質3の理由．作業中の矢が巡回する道を構成するためには，その最後の瞬間に，外向きの作業中の矢を持った点を，新しく作業中の矢が指すことが必要である．しかしながら，その点へ新しい作業中の矢を向けるためには，その点の入力不足量と出力不足量は0でなければならず，その点からの外向きの作業中の矢は存在しえない．(証明終)

さて，開始点は全体の計算が開始して以後，後者すべてから赤信号を受けとる直前までは，作業中の点であり，しかも開始点へ向かう作業中の矢は存在しないことに注意すると，上記，性質 1-3 から次が言える．

(性質4) どの時刻の作業中の点も，開始点から作業中の矢のみを0回以上たどって，到達することができる．

性質4からただちに，次が成立する．

(性質5) どの時刻の作業中の点も，開始点から信号不足量が正である矢を0回以上たどって到達することができる．

さて，定理4を性質5から示す．開始点の出力不足量が0となると，到達できる作業中の点は存在しない．したがって，すべての点で，

(入力不足量＝出力不足量＝0)

であり，すべての点における計算は終了している．(証明終)

4. おわりに

分散型アルゴリズムの例を，局所的排他制御問題と，波及型計算システムの大域的終了性判定問題についてそれぞれ1例ずつ示した．

食事する哲学者の問題については，確率的アルゴリズムによる分散型の解が，文献 16) と 27) にある．波及型計算システムについては，CSP[22] 流の通信方式の場合の終了性判定法が文献 29) に，Dijkstra と Scholten の終了性判定法を応用したグラフ上の結び目の判定法が文献 30) にある．

その他の分散型アルゴリズムの一部について，出典を示すと，次のようになる．グラフの上の最短路を求める方法が文献 7) に，グラフの上の最小重みスパン木を求める方法が文献 17) にある．また Dijkstra と Scholten とほぼ同時期に考案されたグラフ上の一般性の高い分散型アルゴリズムが文献 3) にある．実際的排他制御法が文献 26)，28) に，計算機網のためのアルゴリズムが文献 9) に，分散型システムのデッドロックの検出法が文献 19) にある．

なお，分散型アルゴリズムを扱う会議としては，例えば，ACM SIGACT-SIGOPS 主催の ACM Symposium on Principles of Distributed Computing と IEEE Computer Society 主催の International Conference on Distributed Computing Systems がある．

本稿の作成にあたり査読者，川合慧，高木明啓，徳田英幸，武舎広幸，渡辺治の各氏から多数の有益な助言を頂いたことを感謝する．

参考文献

1) Balzer, R.: An 8-state minimal time solution to the firing squad synchronization problem, Information and Control, Vol. 10, No. 1, pp. 22-42 (1967).
2) Berg, H. K. (ed.): Special Issue on Distributed System Testbeds, IEEE Computer Magazine, Vol. 15, No. 10 (1982).
3) Bernstein, P. A. and Shipman, D. W.: The correctness of concurrency control mechanisms in a system for distributed databases (SDD-1), ACM Trans. Database Systems, Vol. 5, No. 1, pp. 52-68 (1980).
4) Bernstein, P. A. and Goodman, N.: Concurrency control in distributed database systems, ACM Computing Surveys, Vol. 13, No. 2, pp. 185-221 (1981).
5) Brinch Hansen, P.: Distributed Processes: A Concurrent Programming Concept, Comm. ACM, Vol. 21, No. 11 (1981).
6) Chandy, K. M. and Misra, J.: Asynchronous distributed simulation via a sequence of parallel computations, Comm. ACM, Vol. 24, No. 4, pp. 198-206 (1981).
7) Chandy, K. M. and Misra, J.: Distributed computation on graphs: shortest path algorithms, Comm. ACM, Vol. 25, No. 11, pp. 833-837 (1982).
8) Chang, E. J. H.: Echo algorithms: depth parallel operations on general graphs, IEEE Trans. Soft. Eng., Vol. SE-8, No. 4, pp. 391-401 (1982).
9) Dalal, Y. K. and Metcalfe, R. M.: Reverse path forwarding of broadcast packets, Comm. ACM, Vol. 21, No. 12, pp. 1040-1048 (1978).
10) Dijkstra, E. W.: Hierarchical Ordering of Sequential Processes, Acta Informatica, Vol.

1, pp. 115-138 (1971).
11) Dijkstra, E. W.: Self-stabilization in spite of distributed control, Comm. ACM, Vol. 17, No. 11, pp. 643-644 (1974).
Also In: Selected Writings on Computing: A Personal Perspective (by E. W. Dijkstra). Springer-Verlag (1982).
12) Dijkstra, E. W.: On the interplay between mathematics and programming, Lecture Notes in Computer Science, Vol. 69, pp. 35-46 (1979).
13) Dijkstra, E. W. and Scholten, C. S.: Termination Detection for Diffusing Computations, Information Processing Letters, Vol. 11, No. 1, pp. 1-4 (1980).
14) Foster, M. J. and Kung, H. T.: Recognize Regular Languages With Programmable Building-Blocks, VLSI 81 (edited by J. P. Gray), Academic Press (1981).
15) Francez, N.: Distributed Termination, ACM Trans. Prog. Lang. and Systems, Vol. 2, No. 1, pp. 42-55 (1980). [Corrigendum: Vol. 2, No. 3, p. 463 (1980). Technical Correspondence: Vol. 3, No. 1, pp. 112-113 (1981)].
16) Francez, N. and Rodeh, M.: A distributed abstract data type implemented by a probabilistic communication scheme, Proc. 21st IEEE Symp. Foundation of Computer Science, pp. 373-379 (1980).
17) Gallager, R. G., Humblet, P. A. and Spira, P. M.: A distributed algorithm for minimum-weight spanning trees, ACM Trans. Prog. Lang. and Systems, Vol. 5, No. 1, pp. 66-77 (1983).
18) Gehringer, E. F., Jones, A. K. and Segall, Z. Z: The Cm testbed. IEEE Computer Magazine, Vol. 15, No. 10, pp. 40-53 (1982).
19) Gligor, V. D. and Shattuck, S. H.: On deadlock detection in distributed systems, IEEE Trans. Soft. Eng., Vol. SE-6, No. 5, pp. 435-440 (1980).
20) Haynes, L. S. (ed.); Special Issue on Highly Parallel Computing, IEEE Computer Magazine, Vol. 15, No. 1 (1982).
21) Ho, G. S. and Ramamoorthy, C. V.: Protocols for deadlock detection in distributed database systems, IEEE Trans. Soft. Eng., Vol. SE-8, No. 6, pp. 554-557 (1982).
22) Hoare, C. A. R.: Communicating sequential Processes, Comm. ACM, Vol. 21, No. 8, pp. 666-677 (1978).
23) Kobayashi, K.: On minimal firing time of firing squad synchronization problem for poly-automata networks, Theoret. Comp. Sci., Vol. 7, No. 2 (1978).
24) Kohler, W. H.: A survey of techniques for synchronization and recovery in decentralized computer systems, ACM Computing Surveys, Vol. 13, No. 2, pp. 149-183 (1981).
25) Kung, H. T., Sproull, R. and Steele, G.: VLSI Systems and Computations, Spring-Verlag (1981).
26) Lamport, L.: Time, clocks, and the ordering of events in a distributed system, Comm. ACM, Vol. 21, No. 7, pp. 558-565 (1978).
27) Lehmann, D. and Rabin, M. O.: On the advantages of free choice: a symmetric and fully distributed solution to the dining philosophers problem, Proc. 8th Symp. Principles of Prog. Lang., pp. 133-138 (1981).
28) Le Lann: Distributed systems — toward a formal approach, IFIP 77, pp. 150-160 (1977).
29) Misra, J. and Chandy, K. M.: Termination Detection of Diffusing Computations in Communicating Sequential Processes, ACM Trans. Prog. Lang. and Systems, Vol. 4, No. 1, pp. 37-43 (1982).
30) Misra, J. and Chandy, K. M.: A Distributed Graph Algorithm: Knot Detection, ACM Trans. Prog. Lang. and Systems, Vol. 4, No. 4, 678-686 (1982).
31) Shoch, J. F. and Hupp, J. A.: The "worm" programs—Early Experience with a Distributed Computation, Comm. ACM, Vol. 25, No. 3, pp. 172-180 (1982).
32) Shoch, J. F., Dalal, Y. K., Redell, D. D. and Crane, R. C.: Evolution of the Ethernet Local Computer Network, IEEE Computer Magazine, Vol. 15, No. 8, pp. 10-27 (1982).
33) van Dam, P. and Stankovic, J. (eds.): Special Issue on Distributed Processing, IEEE Computer Magazine, Vol. 11, No. 1 (1978).
34) von Neumann, J.: Theory of Self-Reproducing Automata (edited by A. W. Burkes), University of Illinois Press (1966).
35) Wilkes, M. V.: The Impact of Wide-Band Local Area Communication Systems on Distributed Computing, IEEE Computer Magazine, Vol. 13, No. 9, pp. 22-25 (1980).
36) 野口正一，木村英俊，大庭弘太郎：情報ネットワークの理論，岩波講座情報科学第5巻，岩波書店 (1982).

(昭和57年12月23日受付)