1R−8 最大値問題に対するメッシュバス上での$O((\log\log n)^2)$アルゴリズム

堀川 豊　岩間 一雄
九州大学工学部

1. まえがき

メッシュバス計算機（MBUS，図2）は，メッシュ計算機（MC，図1）の局所通信機能をバスによるグローバル通信機能に置き換えた並列モデルである．その物理的実現性はMCに比べてそれほど劣らないと考えられ，上記のようなモデル上の制約による計算時間の自明な下限も存在しない．実際，グラフ問題に対しては，連結成分等の基本的問題に対して，PRAMモデルに匹敵する高速アルゴリズムが実現されることが知られている[1]．

本稿では，最大値問題に対する$O((\log\log n)^2)$時間の確率アルゴリズムを考える．n^2個の各プロセッサに与えられたn^2個の整数の中から最大の整数を探す問題である．PRAM上では$O(\log\log n)$時間のそれ以上改良できない決定性アルゴリズムが知られているが[2,3]，それに比べてもそれ程悪くない．PRAM以外のいわゆる実用的モデル上で，$\log n$より小さい計算時間で実現した例はほとんどないと思われる．

プロセッサ間の物理的距離が小さいという利点がMBUSの特色ではあるが，逆に，1度に通信できるデータの数が$2n$（バスの数＝プロセッサ数の約平方根）に限定されるという欠点がある．従って，必要なデータ通信の絶対量が多い問題，例えば全てのプロセッサが1度はデータをバスに乗せる必要のある問題では対数時間を実現することはできない．もちろんそのような問題は数多く存在し，最近ラウティング問題に対してかなり良い下限が証明された[4]．本稿で扱う問題は必要なデータ通信量の少ない問題の範疇に入り，MBUSに向いた問題であることは確かである．

2. メッシュバス計算機モデル

図2に示されるように，MBUSは内部にローカルメモリを有するn^2個のプロセッサ$P_{i,j}(1\le i\le n)(1\le j\le n)$と2次元メッシュ上に配置されたバスからなる．

バスには同時書き込みを許す．同時書き込みが生じたときはビットごとのOR演算が実行される（いわゆるwired-ORバス）と仮定する．例えばある行バス上のプロセッサが論理値0または1を有するとする．1を有するプロセッサがその行バスに1を書き込む作業を行うと，1個でも1のプロセッサがあればバスの値が1になるので，いわゆる論理ORを計算できる．次に1を有するプロセッサが自分のプロセッサ番号（各プロセッサに唯一）をバスに書き込んだ場合を考えてみよう．2つ以上のプロセッサが1を有している場合には，バス上で実現できる値が少なくとも1つのプロセッサ番号と異なっているはずである．このことを点検することにより，1を有するプロセッサがただ1つであるかどうかの判定を定数ステップで行なうことができる．

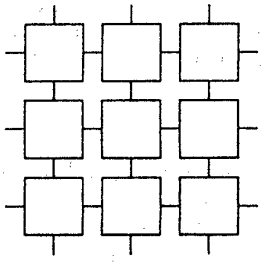

図1．メッシュ計算機

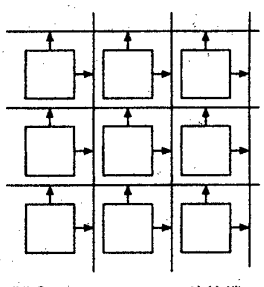

図2．メッシュバス計算機

An O((log log n)^2) Mesh-Bus Algorithm for Finding the Maximum
Yutaka HORIKAWA , Kazuo IWAMA
KYUSHU UNIVERSITY

3. アルゴリズムの概要

n^2個のデータ（簡単のため，全ての値は異る）はn^2個のプロセッサのローカルメモリにそれぞれ1個ずつ与えられ，アルゴリズムがスタートする．以下にその概略を示す．

アルゴリズム *FIND-MAX*（概要）
ステージ1．乱数を利用して，各行においてランダムにプロセッサを1台選ぶ（選ばれない場合もある）．
ステージ2．ステージ1で選ばれたn個（以下）のプロセッサのデータの中の最大値D_{maxrow}を求める．
ステージ3．ステージ1と同様に各列でプロセッサを1台選ぶ．
ステージ4．ステージ2と同様に最大値$D_{maxcolumn}$を求める．
ステージ5．D_{maxrow}と$D_{maxcolumn}$を比較して，大きい方をD_{max}と置き，すべてのプロセッサへ通信する．D_{max}より大きいデータを持つプロセッサがなければ終了．そうでなければ，D_{max}より大きいデータを持つプロセッサについてステージ1〜5を繰り返す．

各ステージをもう少し詳しく見てみる．ステージ1では，最初，各行のn台のプロセッサは全て“生きている”ので，それぞれのプロセッサに確率$\frac{1}{n}$で“手を挙げさせる”と，平均的には各行で1台が選択されることを期待できる．各プロセッサは0から1の間の乱数を発生させ，それが$\frac{1}{n}$より小さい時に行バスにそのプロセッサ番号を書く（手を挙げる）．2章で述べた方法により1台のプロセッサのみ手を挙げたかどうか判る．1台も挙げなかったら同じことを繰り返し，2台以上挙げた場合はそれらが今度は$\frac{1}{2}$の確率で手を挙げることを繰り返すという手法により，定数ステップでかなりの行で1台のプロセッサを選択できると考えられる．選択できなかった行はそのままにする．ステージ3も同様である．

ステージ2はいわゆるリーグ戦方式による．第i行で選択されたデータは対角線プロセッサ$P_{i,i}$に移動され，その値は第i行，第i列全てのプロセッサに伝えられる．こうして各プロセッサ$P_{i,j}$は$P_{i,i}$から伝えられる行データと$P_{j,j}$から伝えられる列データを有することになる．それらの大小比較を行なうと，ある行全てのプロセッサにおいて行データが列データより大きいような行がただ1つ存在する．そのような行は，2章の方法によって特定できるので，その行の対角線プロセッサのデータをD_{maxrow}とする．定数ステップで十分である．

以上の解析によれば，全体でも定数ステップで最大値が求まることになる．実際の計算機実験によれば，ランダムな入力データに対しては上記の繰り返しを3回行なうことによって，かなりの確率で最大値が求まることが判明した．しかし，どのような入力に対しても（高い確率で）定数回で終了するとはいえない．例えば，値の大きなデータがある行に集中していたとする．すると，第1回目の繰り返しの後で行き残るデータがその行に集中してしまい，2回目以降の1台のプロセッサの選択ができなくなる．列方向にも考えているから大丈夫と思われるかもしれないが，例えば，ある$\sqrt{n}\times\sqrt{n}$のエリアに大きな値が集中している場合にはどうにもならない．

4. アルゴリズムの詳細

まずある行に（事前に分かっていない）k個のプロサッサが生き残っている時，その中からランダムに1個を選ぶアルゴリズムを与える．

アルゴリズム *PICK-ONE*
ステージ1．$p=\frac{1}{n}$と置く．
ステージ2．生きているプロセッサは確率pで手を挙げる．1つ以上手を挙げれば，手を挙げなかったプロセッサは死ぬ．1つも手を挙げなかった場合は全て生き残る．

ステージ3. $p > \frac{1}{\log n}$ ならステージ4へ．そうでなければ $p = p^{\frac{1}{5}}$ としてステージ1へ．

ステージ4. 確率 $\frac{1}{2}$ でステージ2と同様の操作を行なう．

ステージ5. ステージ4を $c\log\log n$ 回繰り返し（定数 c は補題1参照），そこで生き残っているプロセッサが1個なら成功．2個以上なら失敗．

PICK-ONE の計算時間は，ステージ2の繰り返しが $O(\log\log n)$ で，またステージ4の繰り返しも $O(\log\log n)$ なので全体でも $O(\log\log n)$ である．

以下の議論では，2項分布のすそ野の可能性を議論するために，便利な以下のようなチェルノフの近似式をしばしば用いる．

$$Pr[X \leq (1-\theta)pn] \leq e^{-\frac{\theta^2 pn}{2}}$$

$$Pr[X \geq (1+\theta)pn] \leq e^{-\frac{\theta^2 pn}{2}}$$

補題1. *PICK-ONE* が失敗する確率を $\frac{1}{n}$ 以下にする定数 c の値が存在する．

（証明の方針）ステージ3が終了した段階で，生き残っているデータの数が $(\log n)^{\frac{5}{4}}$ 以下であれば，その後の $\frac{1}{2}$ ずつ減らしていくプロセスを，例えば $\frac{13}{4}\log\log n$ 回実行すれば平均的には，

$$(\log n)^{\frac{5}{4}} \times (\frac{1}{2})^{\frac{13}{4}\log\log n} = (\log n)^{\frac{5}{4}-\frac{13}{4}} = (\log n)^{-2}$$

個程度まで減るはずである．それでも2個（以上）残る確率は，上記のチェルノフの近似式を適用すると

$$e^{-\frac{(\log n)^4\cdot(\log n)^{-2}}{2}} = e^{-\frac{(\log n)^2}{2}}$$

となり，この値は，$\frac{1}{n}$ より十分小さい．

次にステージ2の繰り返しの後，$(1-\frac{1}{n})$ より十分高い確率で生きているプロセッサの個数が $(\log n)^{\frac{5}{4}}$ 以下になっていることを示す．ある時点で生きているプロセッサの個数 t が $n^{(\frac{4}{5})^k} > t > n^{(\frac{4}{5})^{k+1}}$ であったとしよう．すると，確率 p が $p = n^{-(\frac{4}{5})^k}$ のときは1台も手を挙げず，次の繰り返しの $p = n^{-(\frac{4}{5})^{k+1}}$ のときにかなりの台数のプロセッサが手を挙げることが（平均的に）期待される．$p = n^{-(\frac{4}{5})^{k+1}}$ のとき手を挙げるプロセッサの台数の平均値は最大でも $n^{(\frac{4}{5})^k}\cdot n^{-(\frac{4}{5})^{k+1}} = n^{\frac{1}{5}(\frac{4}{5})^k}$ であり，その後に続く $p = n^{-(\frac{4}{5})^{k+2}}, n^{-(\frac{4}{5})^{k+3}}, \cdots$ のときはどのプロセッサも手を挙げず，$p = n^{-(\frac{4}{5})^{k+10}}$ のときになって初めて（$\frac{1}{5} > (\frac{4}{5})^{10}$ に注意）再び何台かが手を挙げるというように生きているプロセッサの台数が減少していく．このような平均的な振舞いが阻害される要因は次の2点がある．(1) 手を挙げた台数が平均値より異常に多い．(2) 平均的には何台か手を挙げるはずなのに手を挙げない．

仮に $t = n^{(\frac{4}{5})^k}$ としよう．このとき，$p = n^{-(\frac{4}{5})^{k+1}}$ で手を挙げれば上記の様に手を挙げる台数の平均値は非常に小さく心配ない．仮にこのとき手を挙げずに次の $p_1 = n^{-(\frac{4}{5})^{k+2}}$ で手を挙げたと仮定する．この時の台数の平均値は $t_1 = n^{(\frac{4}{5})^k}\cdot n^{-(\frac{4}{5})^{k+2}} = n^{\frac{9}{25}(\frac{4}{5})^k}$ となる．この値は次の繰り返しの確率 $p_2 = n^{-(\frac{4}{5})^{k+3}}$ で平均的に1台手を挙げる台数，つまり $t_2 = n^{(\frac{4}{5})^{k+3}} = n^{\frac{64}{125}(\frac{4}{5})^k}$ よりは小さく，その t_2 まで増える確率は上記のチェルノフの近似式を適用すれば，

$$Pr\left[X \geq n^{(\frac{4}{5})^{k+3}}\right] \leq e^{-\frac{1}{2}\left(1-n^{\frac{19}{125}(\frac{4}{5})^k}\right)^2\cdot n^{\frac{9}{25}(\frac{4}{5})^k}}$$

となり，十分大きな定数 a に対し e^{-at_1} で押さえられる（実際はもっともっと小さい値になる）．以上，$p = n^{-(\frac{4}{5})^{k+1}}$ で手を挙げないことを仮定したが，この確率もチェルノフの近似式を用いると，

$$Pr[X \leq 0] \leq e^{-\frac{1}{2}n^{\frac{1}{5}(\frac{4}{5})^k}}$$

となり，同様に小さいものになることは容易に判る．つまり，平均的な振舞いが壊れる程まで平均からずれる確率は上記(1),(2)とも非常に小さく，プロセッサの台数が $\log n$ 程度まで減少した時点のみ考えておけば十分である．その場合でも確率は $e^{-a\log n}$ で押さえられ，この値は $\frac{1}{n}$ よりは十分小さくなる（a の値が大きいから）．□

いよいよ3章で与えたアルゴリズムの厳密化を行なう．

アルゴリズム *FIND-MAX*

ステージ1. 各行で *PICK-ONE* を実行する．

ステージ2. 以前と同様．

ステージ3. 各列で *PICK-ONE* を実行する．

ステージ4. 以前と同様．

ステージ5. 以前と同様．

定理1. *FIND-MAX* が $c(\log\log n)^2$ 時間に終了しない確率が $\frac{1}{n}$ より少なくなる定数 c が存在する．

（証明の概略）ステージ1と3に $O(\log\log n)$ 時間かかるので，ステージ1～5の繰り返し回数が $O(\log\log n)$ であることを示せばよい．最初の繰り返しを見てみよう．ステージ1では $\frac{1}{n}$ 以下の失敗確率で各行から1台のプロセッサを選べる．そこで，このような1台のプロセッサを選ぶ作業が成功する行の数が $n^{\frac{9}{10}}$ 以下（つまり $n^{\frac{1}{10}}$ の“余裕”をみていることになる）になる確率は（$\frac{1}{n}$ から比べれば）無視できる程度小さい（実際は列に対しても同様のことがいえるがこのことは必要ない）．

次に，ステージ5の後で生きているプロセッサの数が $(n^2)^{\frac{2}{3}} = n^{\frac{4}{3}}$ 以下になる確率が十分高いことを示す．上記で各行（列）から1台選ばれた少なくとも $n^{\frac{9}{10}}$ 個のプロセッサは n^2 個全体からまんべんなく選ばれると仮定してよいから，結局 n^2 個の全てのプロセッサが確率 $n^{\frac{9}{10}}\cdot n^{-2} = n^{-\frac{11}{10}}$ で手を挙げると仮定してよい．この場合，大きい方から $n^{\frac{4}{3}}$ 個のデータを有するプロセッサの中から1台でも手を挙げれば生きているプロセッサは $n^{\frac{4}{3}}$ 個以下になる．実際に計算を行なってみると $n^{\frac{4}{3}}$ 個のプロセッサが確率 $n^{-\frac{11}{10}}$ で手を挙げることになり，1台も挙げない確率はチェルノフの近似式を用いてやると，

$$Pr[X \leq 0] \leq e^{-\frac{1}{2}n^{\frac{7}{30}}}$$

となり，無視できる程度に小さい．

この議論を一般化して，全体で m 台のプロセッサが生きているとき，ステージ1～5の1回の繰り返しでそれが $m^2 3$ 以下に十分高い確率で減少することを言えばよい．このとき，これら残った m 個がある $\sqrt{m}$ 行 $\times\sqrt{m}$ 列に集中している場合が最悪であることに注意し．上と同様の“余裕”を使って計算する．1台のプロセッサを選ぶ作業が成功する行の数が $(\sqrt{m})^{\frac{9}{10}} = m^{\frac{9}{20}}$ であると考えて，生き残っている m 個のプロセッサで大きい方から $m^{\frac{2}{3}}$ 個のデータを有するプロセッサが確率 $m^{\frac{9}{20}}\cdot m^{-1} = m^{-\frac{11}{20}}$ で手を挙げると平均的には $m^{\frac{7}{60}}$ 個が手を挙げることになり，1台も挙げない確率はチェルノフの近似式を用いてやると，

$$Pr[X \leq 0] \leq e^{-\frac{1}{2}m^{\frac{7}{60}}}$$

となり，m の値が大きいときは $m^{\frac{2}{3}}$ 以下にならない（失敗の）確率は上とほとんど同様に無視できるほど小さい．m が少なくなるとそうはいかないが，それから後も $\log\log n$ 回の定数倍程繰り返し回数をのばせば，補題1の証明の最初の部分と同様の議論によって，失敗確率が $\frac{1}{n}$ 以下になることを示せる．□

参考文献

[1] K. Iwama and Y. Kambayashi. An $O(logn)$ parallel connectivity algorithm on the mesh of buses. *Proc. 11th IFIP World Computer Congress*, pp.305-310,1989.

[2] Y. Shiloach and U. Vishkin. Finding the maximum, merging, and sorting in a parallel computation model. *J. Algor.*, Vol.2,pp.88-102.1981.

[3] L. Valiant. Parallelism in comparison problems. *SIAM J. Comput.*, Vol.4,pp.248-355,1975.

[4] K. Iwama and E. Miyano and Y. Kambayashi. Routing problems on the mesh of buses. *Proc. 3rd International Symposium on Algorithms and Computation*, 1992.