Vol.2011-AL-135 No.8
2011/5/16

間違えても大丈夫な凸包構成アルゴリズム

岡 本 吉 央[†1]　ステファン ランガマン[†2]

平面に与えられた n 点から成る集合の凸包を計算する問題を考える．しかし，計算において，高々e 回の基本演算における間違いが存在する可能性があるものとする．この設定において，計算量が $O(n \log h + ne \log \log h)$ となるアルゴリズムを与える．ただし，h は凸包の端点数である．また，この問題に対して $\Omega(n \log h + ne)$ という計算量の下界も与える．

Planar Convex Hulls Against Lies

Yoshio Okamoto[†1] and Stefan Langerman[†2]

Consider the problem of computing the convex hull of a given set of n points in the plane, while at most e errors can occur during the computation. Under this setup, we provide an algorithm running in $O(n \log h + ne \log \log h)$ time, where h is the number of extreme points of the convex hull. Further, we give a lower bound of $\Omega(n \log h + ne)$ for this problem.

1. はじめに

コンピュータ・サイエンスでは，理論においても実践においても故障耐性に関して大きな関心を払っている．特に，故障の組合せ的性格をアルゴリズムの中で用いられる基本演算の出力の誤りとして捉える研究の流れがあり，整列問題[2),8),9)]，最小値発見問題[11)]，最小値最大値同時発見問題[1),4),6)] が研究されてきた．そこで考えている計算モデルは，各内点が入力 2 数の大小比較を行う計算木であり，葉がアルゴリズムの出力に対応する．Pelc によるサーベイ論文[10)] を参照のこと．

本研究では，計算幾何の問題に取り組む．その例として，平面における凸包構成問題を考える．平面上の n 個の点の集合に対して，その**凸包**とはそれらの点をすべて含む最小の凸集合のことである．**凸包構成問題**では，与えられた点集合の凸包の端点を時計回り順で出力する．計算モデルは再び計算木であるが，各内点では「3 点 p, q, r に対して，p は q, r を通る直線の上側にあるか？」という種類の質問を行うこととする．誤りは計算木の任意の内点で起こりうるが，任意の計算過程においてその数は高々e であるとする．すなわち，計算木において根から葉に至る任意のパスにおいて，誤りの起こりうる内点の数は高々e である．これは，アルゴリズムと敵対者の間の 2 人ゲームであると見ることもできる．すなわち，アルゴリズムが計算木を与え，敵対者がその内点のいくつかで誤りが起こりうると指定するのであるが，根から葉へ至る任意のパスにおいて高々e 個の内点しか指定できないとするのである．アルゴリズムが e を事前に知っていることを仮定する．このモデルにおける計算時間は根から葉に至るパスの長さの最大値である．

本論文では，平面上に与えられた n 個の点に対する凸包を構成するアルゴリズムで計算時間が $O(n \log h + ne \log \log h)$ となるものを与える．ここで，h は凸包の端点数であり，e は計算において起こりうる誤りの数である．このアルゴリズムは Chan[3)] による出力依存アルゴリズムの考え方に基づく．

また，(上で挙げた計算モデルにおける) 任意のアルゴリズムが凸包構成問題を解くための計算時間が $\Omega(n \log h + ne)$ になるという下界も証明する．これは，出力依存凸包構成問題に対して知られている下界[7)] と誤りの起こりうる状況での最小値発見問題に対して知られている下界[11)] の帰結である．

2. 基本技法 — 反復

誤りの起こりうる状況でのアルゴリズム設計に対して知られている技法は少ししかない．基本的な戦略は，誤りのない状況で正しく動作するアルゴリズムに変更を加えて，誤りの起こりうる状況に対処することである．

問題 P に対して，誤りのない状況で正しく動作するアルゴリズムを A とし，誤りは A における (true か false を出力する) 条件判定でのみ起こると仮定する．このアルゴリズム A に変更を加えて誤りの起こりうる状況に対処させるためには，A における各条件判定を $2e+1$ 回反復させればよい．そうすると，起こりうる誤りの数は e 以下なので，これら $2e+1$ 回の条件判定の出力で多数決をすれば，それは正しい条件判定の結果となる．アルゴリズム A

†1 北陸先端科学技術大学院大学
Japan Advanced Institute of Science and Technology
†2 ブリュッセル自由大学
Université Libre de Bruxelles

における条件判定の数が高々t回であるならば，このような変更を加えたアルゴリズムにおける条件判定の数は$O(te)$となる．

3. Graham のように

計算時間が$O(n\log h + ne\log\log h)$となるアルゴリズムを示す前に，計算時間が$O(n\log n+ne)$となるアルゴリズムを示す．これは，計算時間が$O(n\log h+ne\log\log h)$となるアルゴリズムの中でサブルーチンとして使用されるが，単体でも$h=\Omega(n)$のときに$O(n\log h+ne\log\log h)$時間のアルゴリズムよりより上界を与える．

定理 1. 平面上に与えられたn個の点の凸包を構成するための$O(n\log n+ne)$時間アルゴリズムが存在する．ここで，eは計算において起こりうる誤りの数である．

ここでは，点集合の凸包の上側を計算するアルゴリズムのみを記述する．漸近的にはこれで十分である．

証明. Graham スキャン[5]の次のような変更を考える．

ステップ 1： 与えられた点をx座標によって(e個の誤りに対処しながら)整列する．

ステップ 2： Graham スキャンを実行する，しかし，各条件判定は$2e+1$回反復する．

アルゴリズムの正当性は直ちに分かる．計算時間について，ステップ 1 は Bagchi[2] か Long[9] のアルゴリズムを用いると$O(n\log n+ne)$時間で実行できる．ステップ 2 は，通常の Graham スキャンで$O(n)$時間かかるが，各条件判定を$2e+1$回繰り返すので，合計$O(ne)$時間費やす．よって，アルゴリズム全体の計算時間は$O(n\log n+ne)$である． □

4. Chan のように

次に，計算時間が$O(n\log h+ne\log\log h)$となるアルゴリズムを示す．

定理 2. 平面上に与えられたn個の点の凸包を構成するための$O(n\log n+ne)$時間アルゴリズムが存在する．ここで，hは凸包の端点数であり，eは計算において起こりうる誤りの数である．

再び，点集合の凸包の上側を計算するアルゴリズムのみを記述する．

証明. 次のような，Chan の出力依存最適アルゴリズム[3]の変更を考える．平面上に与えられるn個の点の集合をPとする．

ループ： 各$H=2^{2^0},2^{2^1},2^{2^2},\ldots$に対して次を実行する．

ステップ 1： $m=\lceil n/H\rceil$として，Pを高々H個の点から成るm個のグループ$P_1,\ldots,P_m$に分割する．

ステップ 2： 各$i=1,\ldots,m$に対して，P_iの凸包を定理 1 のアルゴリズムを用いて(e個の誤りに対処しながら)計算する．

ステップ 3： Pの左端の点から，$P_1,\ldots,P_m$上で Jarvis の行進を実行して，Pの凸包の端点を(e個の誤りに対処しながら)H個発見する．Pの右端の点が見つかったら，終了する．そうでない場合は，ループの先頭に戻り，次のHの値から続ける．

Chan のアルゴリズムの正当性と同様に，このアルゴリズムの正当性も分かる．

計算時間の算定のため，$H=2^{2^t}$となるループの実行を考える．ステップ 1 は定数時間で実行可能である．ステップ 2 は各$i=1,\ldots,m$に対して$O(|P_i|\log|P_i|+|P_i|e)$時間費やすので，全体で，計算時間は

$$\sum_{i=1}^{m} O(|P_i|\log|P_i|+|P_i|e) = O(mH\log H+mHe) = O(n\log H+ne)$$

となる．

ステップ 3 は少し注意が必要である．Pの左端の点を見つけるために，$P_1,\ldots,P_m$の左端の点を見て，その中で最も左にある点を見つければよい．$P_1,\ldots,P_m$の左端の点は既に分かっているので，これは$O(me)$時間で実行可能である．同様に，Pの右端の点も$O(me)$時間で計算できる．

Jarvis の行進の各ステップでは，現在の点から各P_iに対する上側接線を見つけ，その中から最も傾きの大きいものを選んでいく．各ステップでP_iの上側接線を見つけるためには，P_iの左端の点から始めて，P_iの上側凸包の端点を順に調べていく．このように順に見ていくこと(Chan の論文[3]に書かれている Welzl のアイディアであるが)によって，P_iの各点は高々一度しか調べられない．よって，反復の技法を用いることで，このループにおけるP_iの上側接線全体は$O(|P_i|e)$時間で計算できる．上側接線が分かれば，凸包の次の端点は$O(me)$時間で発見できる．したがって，ステップ 3 全体の計算時間は$O(Hme)+O(ne)=O(ne)$となる．

ループ全体では，ステップ 1 から 3 までで$O(n\log H+ne)$時間を費やす．アルゴリズムが終了するのは，tが$2^{2^{t-1}}<h\le 2^{2^t}$を満たすときである．よって，アルゴリズム全体の計算時間は

$$\sum_{t=0}^{\log_2\log_2 h} O(n\log 2^{2^t}+ne) = O(n\log h+ne\log\log h).$$

となる．□

5. 下　　界

次の定理は計算量の下界を与える．

定理 3. 平面上に与えられた n 個の点の凸包を構成する任意のアルゴリズムの計算時間は $\Omega(n \log h + ne)$ である．ここで，h は凸包の端点数であり，e は計算において起こりうる誤りの数である．

証明. 知られている 2 つの結果を結び付ける．

- 凸包構成問題を (誤りのない状況において) 解く任意のアルゴリズムの計算時間は $\Omega(n \log h)$ である[7]．
- n 個の数字の中の最小値を (誤りが e 個ある状況において) 見つけるアルゴリズムの計算時間は $\Omega(ne)$ である[11]．与えられた n 個の数 $a_1, \ldots, a_n$ の中の最小値を見つけたいとする．一般性を失わずに，これらの数はすべて異なり，正であるとする．このとき，平面上の $n+1$ 個の点 $(a_1, a_1^2), \ldots, (a_n, a_n^2), (a_1 + \cdots + a_n, 0)$ を構成する．これらの点の凸包の端点数は 3 であり ($n \geq 2$ のとき)，左端の点が $a_1, \ldots, a_n$ を与える．したがって，最小値発見問題を凸包構成問題に線形時間で還元できた．

よって，凸包構成問題の計算時間は

$$\max\{\Omega(n \log h), \Omega(ne)\} = \Omega(n \log h + ne)$$

となる，ただし，任意の非負実数 a, b に対して $\max\{a, b\} \geq (a+b)/2$ が成り立つということを利用した．□

6. 最　後　に

上界 $O(n \log h + ne \log\log h)$ と下界 $\Omega(n \log h + ne)$ の間にはギャップがある．上界は Chan のアルゴリズム[3] を変更することで得られた．よって，Kirkpatrick と Seidel による別の出力依存最適アルゴリズム[7] の変更を考えたくなるかもしれない．しかし，それを直接行うと，$O(ne \log h)$ という上界しか得られない．$O(n \log h + ne)$ という上界を得るためには，完全に異なるアイディアが必要になるのかもしれない．

参　考　文　献

1) Aigner, M.: Finding the Maximum and Minimum, *Discrete Applied Mathematics*, Vol.74, No.1, pp.1–12 (1997).
2) Bagchi, A.: On Sorting in the Presence of Erroneous Information, *Inf. Process. Lett.*, Vol.43, No.4, pp.213–215 (1992).
3) Chan, T.M.: Optimal Output-Sensitive Convex Hull Algorithms in Two and Three Dimensions, *Discrete & Computational Geometry*, Vol.16, No.4, pp.361–368 (1996).
4) Gerbner, D., Pálvölgyi, D., Patkós, B. and Wiener, G.: Finding the Maximum and Minimum Elements with One Lie, *Discrete Applied Mathematics*, Vol.158, No.9, pp. 988–995 (2010).
5) Graham, R.L.: An Efficient Algorithm for Determining the Convex Hull of a Finite Planar Set, *Inf. Process. Lett.*, Vol.1, No.4, pp.132–133 (1972).
6) Hoffmann, M., Matoušek, J., Okamoto, Y. and Zumstein, P.: Minimum and Maximum against k Lies, *SWAT* (Kaplan, H., ed.), Lecture Notes in Computer Science, Vol.6139, Springer, pp.139–149 (2010).
7) Kirkpatrick, D.G. and Seidel, R.: The Ultimate Planar Convex Hull Algorithm?, *SIAM J. Comput.*, Vol.15, No.1, pp.287–299 (1986).
8) Lakshmanan, K.B., Ravikumar, B. and Ganesan, K.: Coping with Erroneous Information while Sorting, *IEEE Trans. Computers*, Vol.40, No.9, pp.1081–1084 (1991).
9) Long, P. M.: Sorting and Searching with a Faulty Comparison Oracle, Technical report, University of California at Santa Cruz (1992). Available at `http://www.soe.ucsc.edu/research/reports`.
10) Pelc, A.: Searching Games with Errors — Fifty Years of Coping with Liars, *Theor. Comput. Sci.*, Vol.270, No.1-2, pp.71–109 (2002).
11) Ravikumar, B., Ganesan, K. and Lakshmanan, K.B.: On Selecting the Largest Element in Spite of Erroneous Information, *STACS* (Brandenburg, F.-J., Vidal-Naquet, G. and Wirsing, M., eds.), Lecture Notes in Computer Science, Vol.247, Springer, pp.88–99 (1987).