数理解析研究所講究録
第263巻 1976年 23-39

On Bounds of the Number of Comparisons to Select the t-th Largest of n Elements

Kohei NOSHITA
(Department of Computer Science
University of Electro-Communications
Chofu, Tokyo 182)

Abstract

In this paper we shall show some results on the bounds of $V_t(n)$, where $V_t(n)$ denotes the number of binary comparisons necessary to select the t-th largest element of the given n elements.

Let $v(G,t)$ be the number of comparisons necessary to select the t-th largest element of G, where G is a partially ordered set of elements (poset, in short). The partial order is naturally defined as representing the orders among elements obtained from comparisons.

Let $\alpha(x,G)$ ($\beta(x,G)$) be the number of elements larger (smaller, respectively) than x in a poset G.

Theorem 1

Let G be a poset and G' be the poset obtained from G by deleting every element x such that either $\alpha(x,G) > t$ or $\beta(x,G) > n-t+1$. Let $S = \{x \mid \beta(x,G) > n-t+1\}$. Then $v(G,t) = v(G',t')$ for $t' = t-|S|$.

Theorem 2

$V_t(n) \geq \min\{V_{t-1}(n-1), V_t(n-1)\}+2$
for $2 \leqq t \leqq \lceil n/2 \rceil$.

Theorem 3

$V_5(10) \leqq 16$.

Theorem 4 (Exact values of $V_t(n)$)

n	$V_1(n)$	$V_2(n)$	$V_3(n)$	$V_4(n)$	$V_5(n)$
6	5	7	8	8	7
7	6	8	10	10	10
8	7	9	11	12	12
9	8	11	12	(14)	14
10	9	12	(14)	(15)	(16)

The parenthesized values are the best upper bounds known until now. See the table on page 215 in the Knuth's book [2].

Theorem 5 (Improvement of the Hadian-Sobel algorithm)

There is a selection algorithm, which gives the following upper bound.

$$V_t(n) \leqq n-t+(t-1)\lceil \log_2(n-t+2)\rceil-(t-\lfloor t/2\rfloor-1)$$

for n such that

$$2^k+t-2 < n \leqq 2^k+t-2+2^{k-\lfloor t/2\rfloor-1}$$

where $k \geq 3$.

§1. はじめに

本稿では，選択アルゴリズム（selection algorithm）の基本的な性質を調べ，選択に要する比較回数の上下限に関するいくつかの結果を示す．

選択の問題とは，ソーティングの問題の1種であり，任意に与えられた（大小の順序がつけられる）n個の要素のうち，t番目に大きい要素を，2要素の大小比較（binary comparison）の演算を用いて選択する（良い）アルゴリズムを見つけることである．本稿では特に，最悪の場合に要する比較の回数を最小にするアルゴリズムの考察に限定する．この問題は，既にKnuthの本[2]にかなり詳しく最近の結果までふくめて紹介されているので，ここでも術語や従来の結果は，特に断らない限り，同書を参照するものとする．

さて$V_t(n)$で，与えられたn個の要素のうちt番目に大きい要素を求めるのに（最悪の場合に）必要な比較回数（の最小値）を表わすことにする．

本稿では，$V_t(n)$に関する基本的ないくつかの性質を調べ，その応用として，小さいnに対する$V_t(n)$の正確な値を求める．また$V_5(10)$に対する新しい上限値を与える．これらは，[2]の215頁にでている$V_t(n)$の上限表のいくつかの項目の改良ないしは正確な値を与えるものである．最後に$V_t(n)$

の上限を改良する1つのアルゴリズムを示す.

§2. 基本的な性質といくつかの $V_t(n)$ の値

入力として与えられるn個の要素は，1からnまで番号をふったn個の数として考え，その番号で各要素を識別するものとする．各要素の値は，すべて相異なるものとしておく．

2要素の比較によって判明した大小の順序は，半順序として考えることができるので，いくつかの要素でなるこの半順序集合（posetとよぶことにする）をグラフで図示することにする．これは，普通のHasse図であるが，間接的（推移的）な順序関係を示す枝は，省略して描く．なお大きい要素は，上に描くことにする．特に入力のn個の要素は，孤立したn個の点からなるposetである．

どんな選択アルゴリズムも，このposetを入力として，比較演算を用いてt番目の要素を求める決定木（decision tree）[2]として表現できるが，少し拡張解釈して，1つのposet Gを入力として，G中の2要素の比較を繰返して，t番目の要素を指定するものとする．

（Def 1）

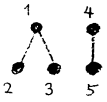
$v_A(G,t)$ は，poset Gと順位tに対する選択アルゴリズムAが（最悪の場合に）必要な比較回数とする．

(Def 2)

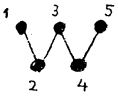
$$v(G,t) = \min_{A} \{ v_A(G,t) \} .$$

定義より, $V_t(n) = v([\overset{1}{\bullet}\overset{2}{\bullet}\overset{3}{\bullet}\cdots\overset{n}{\bullet}], t)$.

(Ex 1)

$v($  $, 2) = 3$.

(Ex 2)

$v($  $, 3) = 3$.

poset G に対して, $\#G$ でその節点(要素)の個数を表わすことにする. 便宜上, $t \leq 0$ または $t > \#G$ なる t に対して

$$v(G,t) = 0$$

とおく.

さて任意の poset G_1, G_2 がその要素の番号を除いて同じ形(同型)なら, G_1 に対する選択アルゴリズムにおいて, 要素の番号に適当な置換を施せば, G_2 に対する1つの選択アルゴリズムに変換できる. 従って選択アルゴリズムや各定義は, 各 poset の節点に番号がふられていないと仮定しても本質的な差異はない.

(Def 3)

2つの poset G_1, G_2 に対して, 要素の間に1対1の対応 f があって, 任意の要素 $a, b \in G_1$ に対して,

$$a < b \quad \text{ならば} \quad f(a) < f(b)$$

が G_2 で成立つ時，

$$G_1 \trianglelefteq G_2$$

とかく．（ここで $a \in G$ という記法で，a が G の要素であることを表す．）

(Def 4)

$$G_1 \equiv G_2 \quad \text{iff} \quad G_1 \trianglelefteq G_2 \text{ かつ } G_2 \trianglelefteq G_1 .$$

(Lemma 1)

任意の G_1, G_2, t に対して，$G_1 \trianglelefteq G_2$ ならば，

$$v(G_1, t) \geqq v(G_2, t) .$$

証明：$v(G_1, t)$ を実現するアルゴリズム A と simulate する (G_2, t) に対するアルゴリズム B を考えればよい．■

(Th 2)

任意の poset G と慣性 t に対して，

$$S_1 = \{ \nu \mid \#\{\mu \mid \nu > \mu\} > n - t + 1 \}$$

$$S_2 = \{ \nu' \mid \#\{\mu' \mid \nu' < \mu'\} > t \}$$

とする．いま G から S_1, S_2 に属すすべての要素を除いてできる poset を G' とすると，

$$v(G, t) = v(G', t') .$$

ただし $t' = t - \# S_1$ とする．

証明：$v(G', t')$ を実現するアルゴリズムを A とする．

Aは、そのまま(G, t)に対するアルゴリズムBにもなっているから,

$$v(G', t') \geq v_B(G, t) \geq v(G, t) .$$

いまGの中で S_1 (S_2) に属す要素に(Gの順序に矛盾することなく)線形の順序をつけ,各要素の値は,G'のどの要素よりも大きく(小さく)与える"oracle" [2] をつくることができる。すると $v(G, t)$ を実現するアルゴリズムCは,(G', t') に対するアルゴリズムDに変換できる。よって

$$v(G, t) \geq v_D(G', t') \geq v(G', t') .$$

以上をあわせて $v(G, t) = v(G', t')$ が成立つ ■

なお別証明が [4] で与えられている。この定理は,特定の(G, t)に対するすべてのアルゴリズムを数えあげる時,有用なものであり,ここでは特に"削除定理"とよぶことにする。

(Lemma 3°) [2]

$$V_t(n) \geq V_t(n-1) + 1$$

$$V_t(n) \geq V_{t-1}(n-1) + 1 .$$

(Th 4)

$$V_t(n) \geq \min\{V_{t-1}(n-1), V_t(n-1)\} + 2 .$$

証明: つぎのいずれかが成立つことを証明する.

$V_t(n) \geq V_{t-1}(n-1) + 2$,

$V_t(n) \geq V_t(n-1) + 2$.

$V_t(n)$ を実現するアルゴリズム A を考える。A は，次のような j 個の比較ではじまるものと仮定してよい．

第 1 回目の比較：　要素 (1) と (2)

第 2 回目の比較：　要素 (3) と (4)

第 j 回目の比較：　要素 $(2j-1)$ と $(2j)$.

ただし $1 \leqq j \leqq \lfloor n/2 \rfloor$. これは，第 k 回目 $(1 \leqq k \leqq j)$ の比較でできる 2 つの poset が同型であること，各比較に関わる要素が共通していないことによる（比較の結果できる 2 つの poset に対して，A が別の 2 要素を選ぶとしても，poset が同型であるから，$V_t(n)$ を実現するアルゴリズム A′ を作り，上の様に標準化できる）. さらに同様の考察により，(1) > (2) であると仮定してよい．ここまでで，

のような poset になる．A において，第 $(j+1)$ 回目の比較は，要素 (1) か (2) かのいずれかが 1 方の対象に選ばれるものと仮定することができる．いま A は，要素 (1) を選んだとする．この時要素 (1) に最大の値 (∞) を与える "oracle" をつくることができる．アルゴリズム A において

て，要素(1)に関わる比較を全部除いてできるアルゴリズムをBとすると，Bは，$(n-1)$個の要素のうち$(t-1)$番目の要素を選択するアルゴリズムになっている．アルゴリズムAは，要素(1)について少くとも2回比較しているから，

$$v_A([\overbrace{\bullet\bullet\cdots\bullet}^{n}], t) - 2 \geq v_B([\overbrace{\bullet\bullet\cdots\bullet}^{n-1}], (t-1)) \geq V_{t-1}(n-1)$$

よって $V_t(n) \geq V_{t-1}(n-1) + 2$.

もしアルゴリズムAが，上で要素(2)を選んでいれば，最小の値を用いる同様の議論を用いると，

$$V_t(n) \geq V_t(n-1) + 2 .$$ ∎

(Ex 3)

$$V_3(7) = 10 .$$

Floydによって，この事実はすでに報告されている[6]．ここでは，(lemma 1)と(削除定理)の応用の例として，(7,3)に対するすべてのアルゴリズムを表現するminimax法に基く treeをしらみつぶしで証明した．これは，かなり大きなサイズの“tree search”であるが，強力な“定理”の援用で“手”で計算できる範囲におさめることができた1例である．“計算機”でこのtree searchを実行させるのは，容易ではない．なぜなら，単純な方法では，グラフの同型性判定の(高能率な)アルゴリズムが必要になるからである．

(Cor 5)

$V_3(8) = 11$.

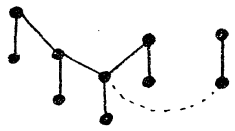
$V_5(10)$ に関しては, [2] によると $V_5(10) \leqq 17$ となっている. 次にこの上限を改良する.

(Th 6)

$V_5(10) \leqq 16$.

証明: 16回の比較で選択する1つのアルゴリズムを示す.

Step 8 の直後:

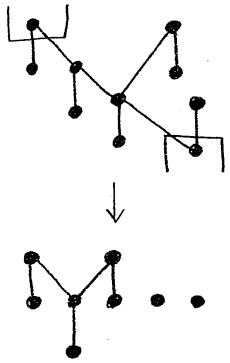


rank(5)

(8回の比較でここまで(対称的に)くることができるのは, 明らかである.)

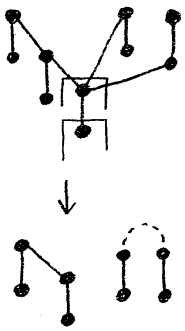
Step 9 :

Case 1.



rank(4)

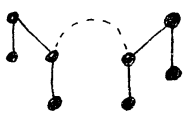
Case 2.



rank(5)

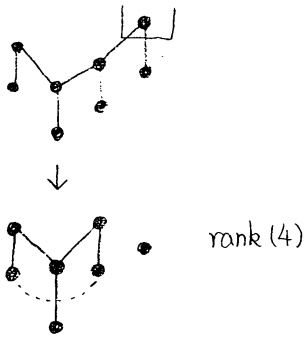
<u>Case 1</u>. については, あと7回の比較で十分である ([3] における証明を参照).

<u>Case 2</u> Step 10:

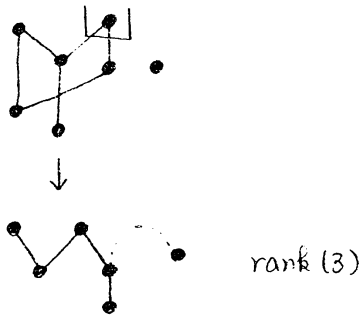


rank(5)

Step 11 :

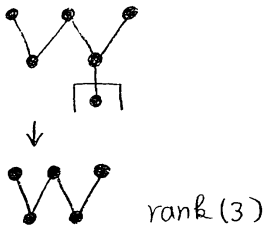


Step 12 :

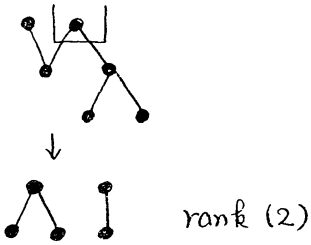


Step 13 :

Case a.



Case b.



OK (cf. (Ex 2))　　OK (cf. (Ex 1))

いずれの場合も 16 ステップで選択できる. ∎

(Th 7)

小さい n に対する $V_t(n)$ の値は, 次の通りである.

	$V_1(n)$	$V_2(n)$	$V_3(n)$	$V_4(n)$	$V_5(n)$	$V_6(n)$
6	5	7	8	8	7	5
7	6	8	**10***	**10***	**10***	8
8	7	9	**11***	**12**	**12**	**11***
9	8	11	12	(14)	**14**	**14**
10	9	12	(14)	(15)	**(16)**	**(16)**

ここで ( ) で括っているものは, いままで知られている最善の上限である. 太字が本稿で求めた結果である. 印*の

ついているものについては，（Ex 3）の注釈参照．その他のものは，既に正確な値として知られている[2]．なお

$V_3(9)=12$ は，[5] による．

$V_5(9)\leqq 14$ は，[3] による．それが正確な値であることは，$V_4(8)=12$ とともに，（Th 4）による．

$V_4(9)=13$ または 14 である（14らしい）．$V_4(9)=14$ が確定すると，$V_5(10)=16$ も確定する（Th 4）．

さて，ここで述べた一連の結果は，一般に，中間値 (median) を求める問題について，

$$V_{\lceil n/2\rceil}(n)=2n-4$$

と予想がたてられそうであるが，現在知られている上限と下限との間隙は，まだ相当大きい．

§3. Hadian-Sobel アルゴリズムの改良

任意の n，t に対して，n に線型的な $V_t(n)$ の上限を与えるアルゴリズムがよく知られているが[1]，順位 t が小さい時には，Hadian-Sobel のアルゴリズムがはるかに能率的であることも知られている[2]．本節では，この Hadian-Sobel のアルゴリズムを改良する．

（Th 8）

任意の n，t（$3\leqq t\leqq\lceil n/2\rceil$）に対して，

$$2^k+(t-2) < n \leqq 2^k+(t-2)+2^{k-\lfloor t/2 \rfloor -1}$$

ならば（$k \geqq 3$），

$$V_t(n) \leqq (n-t)+(t-1)\lceil \log_2(n-t+2) \rceil - (t-\lfloor t/2 \rfloor -1)$$

である．

上の範囲の n に対して，Hadian-Sobel のアルゴリズムが与える上限より，

（$t-\lfloor t/2 \rfloor -1$）回

改良されることになる．他の n については，そのままである．

証明：上の上限式を与える1つのアルゴリズムを与える．

$r=\lfloor t/2 \rfloor +1$ とおく．また $n=2^k+(t-2)+m$（ただし $1 \leqq m \leqq 2^{k-r}$）とする．

(1) n 個中（$t-2$）個の要素を（Hadian-Sobel のアルゴリズムと同様に）最後のステップのためにとっておく．

(2) 残りの（$n-t+2$）個の要素と（2^r+1）個の集合 $S_1, S_2, \cdots, S_{2^r}, R$ に分割する．ここで S_i $(i=1,\cdots,2^r)$ は，2^{k-r} 個，R は，m 個の要素をもつものとする．

（S_i の要素の個数は，2の巾であることに注意）

[Step 1] 各 S_i $(i=1,\cdots,2^r)$ に対して，標準的な（バランスしている）knockout tournament を実施して，最大の要素 σ_i を選抜する．

いま $M_1=\{\sigma_1,\sigma_2,\cdots,\sigma_{2^{r-1}}\}$, $M_2=\{\sigma_{2^{r-1}+1},\cdots,\sigma_{2^r}\}$ とおく．

[Step 2]　各 M_i $(i=1,2)$ に対して，標準的な knockout tournament を実施して，最大の要素を選択する．

ここで，次のような M_i の要素を σ_{j_i} とする．[Step 2] の tournament において，σ_{j_i} は，第1回戦で負けた（小さい）要素であり，その比較で勝った要素は，第2回戦で負けた要素であり，$\cdots$，その比較で勝った要素は，最終戦で負けた要素である，ようなものである．たとえば図1では，○で囲んでいる要素が各々 σ_{j_i} である．ここで $1 \leqq j_1 \leqq 2^{r-1}$，$(2^{r-1}+1) \leqq j_2 \leqq 2^r$ である．

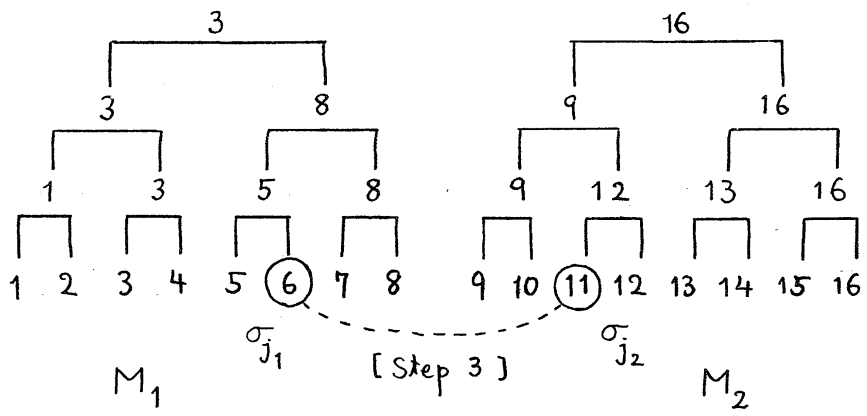


図1．Step 2 における knockout tournament （$r=4$）

[Step 3]　σ_{j_1} と σ_{j_2} を比較する．

対称性より，[Step 3] で σ_{j_2} が負けた要素と仮定して差支えない．各 σ_{j_i} $(i=1,2)$ は，自分より大きい要素が，少くとも $\lfloor t/2 \rfloor$ 個あるから，σ_{j_2} は，決して t 番目に大

き い要素になれない．したがって S_{j_2} の要素をすべて除くことができる．

［Step 4］ S_{j_2} のかわりに R を考える．まず R において，標準的な knockout tournament を実施し，最大の要素を選択する．［Step 2］で実施した M_2 に対する tournament で残っている要素の集合において，いま求めた R の最大の要素に関する置換え選択（replacement selection）を実施する．これで選択された最大の要素と（［Step 2］で既に得られている）M_1 の最大の要素を比較する．

この比較で大きい方の要素は，この段階まで考慮した (2^k+m) 個の要素の最大要素であるから，除くことができる．

［Step 5］（これ以後は Hadian-Sobel アルゴリズムと同様）最初にとっておいた $(t-2)$ 個の要素を1つずつ取り上げ，置換え選択を実施して，$(t-2)$ 個の要素を除く．最後に，残った要素の中から最大の要素を選べば，それが望みの t 番目に大きい要素である．

以上でアルゴリズムの記述は，おしまいである．このアルゴリズムが正しいことは，容易に検証できる．

次にこのアルゴリズムで要する比較回数を計算する.

[Step 1] $2^r \times (2^{k-r} - 1)$ 回

[Step 2] $2^r - 2$ 回

[Step 3] 1 回

[Step 4] (高々) $(m-1) + (r-1) + 1$ 回

[Step 5] (高々) $(t-2) \times k + (k-1)$ 回

併せて定理で述べている $V_t(n)$ の上限式を得る. ■

特に $t = 3$ とすると,

$$2^k + 1 < n \leqq 2^k + 2^{k-2} + 1$$

をみたす n に対して,

$$V_3(n) \leqq n - 4 + 2\lceil \log_2(n-1) \rceil.$$

となり, [5] の下限の結果とあわせて, 上のような n に対して, $V_3(n)$ の上下限の差は, 1 回となる (次頁(Note)参照).

§4. あとがき

本稿では, 選択アルゴリズムのいくつかの基本的な性質を調べ, 小さい n に対する $V_t(n)$ の正確な値を求めた. また $V_5(10) \leqq 16$ を示した. これらの結果は, Knuth の本 [2] の $V_t(n)$ の表の改良を与えている. 最後に, Hadian-Sobel のアルゴリズムの 1 つの改良を示した.

(Note)

最新の Knuth からの私信 [6] には，次の事実が伝えられている．

(1) D. Kirkpatrick が，すべての $n \geq 50$ に対して，$V_3(n)$ の正確な値を決定したこと．

(2) 同じく，§4. で述べた Hadian-Sobel のアルゴリズムの改良を独立に示したこと．比較回数の式は，本稿のものと同じであるが，さらに n の範囲に対する制限を弱めたこと．

§5. 引用文献

[1] Blum, M. et al, "Time Bound for Selection," JCSS, Vol. 7 (1973), 448-461.

[2] Knuth, D.E., The Art of Computer Programming, Vol. 3 (Sorting and Searching), Addison-Wesley (1973).

[3] Noshita, K., "Median Selection of 9 Elements in 14 Comparisons," Info. Proc. Letters, Vol. 3, No. 1 (1974), 8-12.

[4] 野崎昭弘，私信（1975年3月）．

[5] Pratt, V. & F.F. Yao, "On Lower Bounds for Computing The i-th Largest Element," IEEE 14th SWAT (1973), 70-81.

[6] Knuth, D.E., 私信（1975年5月）．

■