HOW GOOD IS THE ADVERSARY LOWER BOUND ?

Peter Ružička, Juraj Wiedermann

Computing Research Centre
Dúbravská 3, 885 31 Bratislava

Czechoslovakia

ABSTRACT

In this paper we discuss the strength of the adversary argument in establishing lower bounds on the complexity of certain sorting-type problems. The relationship between adversary argument and so called information theory argument is indicated and the efficiency of adversary argument relative to the type of comparisons involved in the computation of a problem is investigated. The results concern the effect of polynomial comparisons on lower bounds. In certain cases (MIN and MERGE problems) by using polynomial comparisons we are able to obtain assymptotically the same lower bounds as those established when comparisons without arithmetics are used.

INTRODUCTORY FACTS

1.1 (Motivation) Only for relatively few problems there are algorithms known to be optimal in the worst-case. Thus, one of the principal goals in the complexity theory is to determine tight upper and lower bounds for the inherent difficulty of specific problems. Of a special interest there is inherent difficulty of combinatorial calculations which frequently arise in computing.

While the upper bounds on the computational complexity of a problem are relatively easily established by exhibiting an algorithm to solve the task and analyzing its cost, it is usually much more difficult to determine tight lower bounds. The difficulty arises from the fact that only properties common to the whole class of algorithms, which solve the given problem, can be exploited. There are known just few strategies used to determine lower bounds of comparison problems.

1.2 (Problem and computational model) In this paper we shall concentrate on sorting-type decision problems defined in the following way: let X be a set of n elements, S be a subset of all linear orderings of X and P be a partition on S. The decision problem $D(S,P)$ is the problem how to determine for a given ordering $w \in S$ that partition set from P to which w belongs.

For example, if the $MAX(n)$ is considered, then using our notation the problem can be described as $D(S, \{P_1, \ldots, P_n\})$ where

$$P_i = \{ a_{k_1} \ldots a_{k_n} \in S \mid a_{k_i} = \max \{ a_1, \ldots, a_n \} \} ,$$

i.e. P_i is the set of those orderings from S in which the maximal element is in the i-th position. Thus, the problem to find the maximal element of X is identical with the problem to determine for an arbitrary ordering that partition set to which the ordering belongs.

We use a comparison tree as a general model to express decision algorithms. Each interior node of a rooted tree is associated with a comparison operation, and output (result) takes place at the leaves of the tree. Computation proceeds from root, node by node, until a leaf is reached. The path of greatest length indicates a worst-case complexity of a problem.

We shall discuss decision problems not only under the more traditional assumption that only binary comparisons (without arithmetics) are allowed but also under a rarer assumption that more general comparisons (with certain arithmetics) are permitted. Moreover, in comparisons only so called ignorant arithmetics are allowed, i.e. such by means of which the maximum cannot be realized.

1.3 (Information theory argument) One of the most efficient techniques for establishing lower bounds is the one using the information theory argument [2]. The main idea behind is that every algorithm using binary comparisons which is capable of giving r different results must also be capable of distinguishing them by r different computations and thus it must perform at least $\lceil \log_2 r \rceil$ comparisons. A lower bound which is derived using information theory argument will be called information theory bound (IT_B).

In the case of binary comparisons Fredman [3] discussed, in a precise manner, the strength of the information theory argument for a class of decision problems. He proved that information theory argument is powerful for the case

when P is the partition of S into singleton sets. He showed, however, that in almost all cases the information theory argument is fairly weak by proving that there is a gap of $n.\log_2 n + O(n)$ comparisons between ITB and optimal solution for almost all decision problems $D(S,P)$ where S is the set of all n! orderings and P is a partition of S into two sets.

When certain arithmetics are permitted in computation, the complexity of decision problems can be significantly lowered. For example, to solve the problem due to Chase (see [3, p. 356]) whether an arbitrary ordering of a set of n elements has even or odd parity, only 1 comparison using multiplications and subtractions suffices while $O(n.\log_2 n)$ binary comparisons are necessary. Also using a comparison tree program which allows comparisons between exponential functions over X, the maximum of n integers can be computed with $\lceil \log_2 n \rceil$ comparisons, while n-1 binary comparisons are necessary. In both examples the information theory lower bound has been attained using more general comparisons. However, there still remains a problem if there exists such a decision problem for which ITB would not be attainable by using any allowable comparisons.

The strength of information theory argument in the case of more general comparisons was investigated by author [7]. In this paper the effect of polynomial comparisons on the complexity of certain decision problems is exploited. We show that in the case of minimum problem for n integers the ITB is not attainable by using polynomial comparisons.

1.4 (Adversary argument) Utilization of the adversary argument is another method in obtaining lower bounds. On the contrary to the information theory argument which uses only information about the number of distinguishable results of a problem, the adversary reacts to the computational process of an algorithm. The intuition behind the adversary argument is that the adversary, using certain pattern of input data, forces the algorithm to follow some long path of computation tree. The length of the path constitutes an adversary bound (AB). This operational strategy of adversary is denoted further as constructive one.

In our opinion the reason for treating adversary argument is that constructive way of determining lower bounds will enable us to penetrate deeper into the problem than a nonconstructive one, and so it would be of a considerable help in understanding the decision problem.

In this paper we characterize adversary argument from three points of view. Firstly, the relationship between ITB and AB is determined by proving that for

each decision problem an adversary reaching ITB can be constructed. Secondly, since the essential feature of ITB is that it remains invariant as long as comparisons with ignorant arithmetics are used, the effectiveness of adversary technique relative to the arithmetic operations involved was studied. The case of comparisons using linear functions was investigated by Yao [8], more general case was treated by Dobkin and Lipton [1]. By applying adversary argument we prove for certain problems that assymptotically it is not possible to achieve savings in the number of comparisons by utilizing more general polynomial comparisons. Finally, we prove the existence of a decision problem for which the optimality cannot be proved by using adversary technique.

EFFICIENCY OF ADVERSARY ARGUMENT

2.1 As follows from Fredman's results the complexity of many nonlinear decision problems can be fairly well lower estimated by using information theory argument. By using this argument a tight lower bound for sorting has also been established which is optimal for small values.

In most cases lower bounds on the complexity of linear decision problems with binary comparisons have been proved using adversary argument. The following theorem claims that using adversary techniques tight lower bounds for nonlinear problems can also be achieved.

Theorem

For each decision problem $D(S,P)$, an adversary determining the lower bound greater than or equal to $\lceil \log_2 |P| \rceil$ can be constructed.

Proof:

Let us consider a decision problem $D(S,P)$ where $P = \{P_1,\ldots, P_k\}$ and an arbitrary comparison tree T which solves the problem $D(S,P)$. Denote nodes and leaves of the tree T as $N_1,\ldots, N_p$ and $L_1,\ldots,L_r$, respectively. With each node N (leaf L) we associate a subset $S_N(S_L)$ of S such that S_N (S_L) will be the set of all such orderings from S which fulfil the sequence of comparisons on the path from the root of T to N (L). To the initial node (root of the tree T) we associate the entire set S. The strategy of adversary will be the following:

Suppose we are in a node N_i, $1 \leqslant i \leqslant p$, with a comparison "f:0".

Using the comparison we can split the set S_{N_i} into two parts

$$\{ a_{k_1} \cdots a_{k_n} \in S_{N_i} \mid f(a_{k_1}, \ldots, a_{k_n}) > 0 \} \quad \text{and}$$

$$\{ a_{k_1} \cdots a_{k_n} \in S_{N_i} \mid f(a_{k_1}, \ldots, a_{k_n}) \leqslant 0 \}.$$

Nodes corresponding to these sets are S_{N_d} and S_{N_e}, respectively.

Thus we have $S_{N_i} = S_{N_d} \cup S_{N_e}$. As the following step in the computation the adversary chooses the node N_d if it holds

$$|\{ v \mid S_{N_d} \cap P_v \neq 0, \quad 1 \leqslant v \leqslant k \}| \geqslant |\{ v \mid S_{N_e} \cap P_v \neq 0, \quad 1 \leqslant v \leqslant k \}|$$

otherwise it chooses the node N_e.

Informally, the adversary strategy keeps information about a number of partition sets it has to distinguish using bisection.

The theorem follows from the observation that for every accepting leaf L of the tree T it holds $S_L \subseteq P_v$ for some $1 \leqslant v \leqslant k$.

2.2 In general case, the use of more general comparison operations does increase the power of computation. However, Yao conjectured [8] that the complexity of any decision problem $D(S,P)$ where S is the set of all n! orderings of a set with n elements remains unchanged even when linear comparisons are used. But if some restriction is placed on the set S (i.e. not all n! orderings are possible), then decision problems in which linear comparisons speed up the computation are also known.

It is also interesting to investigate the effect of polynomial comparisons on the complexity of certain problems. Firstly, we concentrate on the maximum / minimum problem. Rabin showed [4], that for the number of comparisons necessary to determine maximum of n real numbers, $n-1$ is the lower bound even if we permit the use of analytical functions in comparisons. When considering integers, Rabin's result cannot be exploited because the proof of it is based on the properties of real numbers. However, Reingold obtained the result [6] that the maximum of n integers can be computed with $\lceil \log_2 n \rceil$ comparisons between exponential functions. If we restrict ourselves to linear comparisons, then $n-1$ comparisons are always necessary to compute the maximum of n integers. Friedman

has raised the question whether the minimal integer from the set of n integers can be computed with logarithmic or linear number of polynomial comparisons. The following result claims that polynomial functions in comparisons do not reduce assymptotically the minimum number $MIN_p(n)$ of comparisons necessary for computing the minimum of the set of n integers.

Theorem

$$MIN_p(n) = O(n)$$

We sketch the idea of the proof. Consider an arbitrary set $\mathcal{Y}$ of n integers $\{i_1,\ldots,i_n\}$ and an arbitrary comparison tree T which solves the minimum problem for $\mathcal{Y}$. During the computation, the algorithm T successively performs comparisons in the form $f(a_1,\ldots,a_n) : 0$ where f is a polynomial of n variables with terms arranged in the usual lexicographical order based on some order of elements in $\mathcal{Y}$. The first term with nonzero coefficient is called the leading term. We shall construct a deterministic "responding" strategy which will force the tree algorithm T to make a large number of comparisons. Our responding strategy is the following:

Let R be a partial ordering of $\mathcal{Y}$ as established by our strategy in the previous computation and let $f:0$ be a comparison in consideration. Consider the lexicographical arrangement of terms of f based entirely on the partial ordering R. Two possibilities can occur: either the leading term of f is uniquely determined or it is not. In the latter case, by specifying the order among certain number of elements of $\mathcal{Y}$, exactly one term from the set of potential candidates for leading term is choosed (and these binary relations among elements of $\mathcal{Y}$ are then added to R). In both cases, the answer to the comparison question will be "$f > 0$" if the leading term is positive, otherwise the answer will be "$f \leqslant 0$".

In order to estimate this responding strategy of adversary, we use the following property:

FACT:

There exists an integer constant $c > 0$ such that for an arbitrary polynomial $f(a_1,\ldots,a_n)$ of n variables with integer coefficients the set

$$\{(i_1,\ldots,i_n) \mid i_1 > i_2 > \ldots > i_c \;,\; i_c > i_k \quad \text{for} \quad c < k \leqslant n$$

and there exists exactly one term g of polynomial f such that

$$f = f' + g \quad \text{and} \quad |g(i_1,\ldots,i_n)| > f'(i_1,\ldots,i_n)\}$$ is nonempty.

Now, the assertion of the theorem follows from the FACT because from one polynomial comparison determined by the previous strategy at most two binary comparisons between two elements in the leading term different from the elements with relations in R can be abstracted.

2.3 Analogously we are able to prove similar result for the $MERGE_p(n)$ problem in which two ordered sequences of n distinct numbers are merged into one ordered sequence of 2n numbers. Let us consider the set $\mathcal{J}$ of 2n integers $\{i_1,\ldots,i_{2n}\}$ where it holds $i_{2k} > i_{2k+2}$, $i_{2k-1} > i_{2k+1}$ for $1 \leqslant k \leqslant n-1$. Again, consider the same lexicographical arrangement and the same responding strategy as in the proof of the previous theorem with the exception that initially the partial ordering R given by the MERGE problem is also considered. Let Q be the sequence of polynomial inequalities generated by a comparison tree program solving the merging problem based on the previously described responding strategy. Suppose that there exists an element, say i_k, such that it does not occur in any inequality in Q as an element with the smallest index in the leading term. Then it must occur in the so called "critical" position (i.e. that position in the term where the decision about the order of this term in the arrangement is made) in the leading term in at least k/2 inequalities of Q. The number of "critical" positions is proportional to n while the number of i_k elements, for which the order using critical positions is determined, is proportional to n^2 by which we conclude

Theorem

$$MERGE_p(n) = 2n - O(\sqrt{n})$$

2.4 Instead of polynomial functions we shall now consider restricted polynomial functions

$$f(a_1,\ldots,a_n) = \sum_{i=1}^{n} \sum_{s=1}^{maxexp} c_{i,s}.a_i^s$$

where maxexp denotes maximal exponent. In this case using the idea of previously described adversary strategy we are able to prove that

- for the minimum of n integers it holds $MIN_{RP}(n) = n-1$
- for the merging problem it holds $MERGE_{RP}(n) = 2n-1$

- for the minimum and maximum problem of n integers it holds

$$MINMAX_{RP}(n) \geqslant n - 2 + \lceil \log_2 n \rceil$$

- for the second smallest element of n integers it holds

$$SEC_{RP}(n) = n - 2 + \lceil \log_2 n \rceil$$

LIMITATIONS OF ADVERSARY ARGUMENT

3.1 In the previous section our considerations concerned mostly positive aspects of adversary argument applicability. We already saw that for each sorting-type decision problem ITB can be determined in a constructive way and that for certain decision problems adversary responds sensitevily also to general comparisons of algorithm.

Now we want to point out the fact that for certain problems using adversary argument the complexity of optimal algorithms solving those problems cannot be achieved.

3.2 Let us consider the general selection problem $V(k,n)$ which is the problem to determine the minimum number of binary comparisons that are required to determine the k-th largest of n elements of a linear ordered set X. Consider algorithms which solve the $V(k,n)$ problem for arbitrary n and k, $1 \leqslant k \leqslant n$ and which are of fixed strategy. An algorithm A solving a problem $D(S,P)$ is of a fixed strategy if it holds

1. there exists a comparison tree T for the problem $D(S,P)$ such that for all $w \in S$ the sequence of comparisons performed by T is identical with the sequence of comparisons performed by A

2. for each comparison c from A there exists $w \in S$ such that if w is the input of A, then A must perform c at some step of its computation

Using the notion of fixed strategy we are able to overcome the case in which the process of constructing optimal comparison trees is made using backtracking, and to ensure that even using any kind of coding we still be able to path through all branches of program containing comparison operations (in the case of fixed size of the problem).

3.3 The main result of this section is formulated in the following theorem:

Theorem

There is no algorithm A with fixed strategy for computing $V(k,n)$ which is optimal in the worst-case for all n and k, $1 \leqslant k \leqslant n$.

Proof:

Suppose that A is optimal for $k = 1$ and consider the way in which this algorithm works for an arbitrary k, $1 \leqslant k \leqslant n$. The algorithm compares maxima of partial ordering (so called "local" maxima) until it finds such an element x which is greater than $n-k$ other elements. The algorithm A must proceed in comparing local maxima because in another case, when performing some "nonlocal" comparison, it has to do it in the case of the MAX problem too, as it follows from the fixed strategy. And in this latter case it could not be optimal for $k = 1$.

To find the element x the algorithm A performs at least $\lceil \log_2 (n-k+1) \rceil$ comparisons. Furthermore, the algorithm A knows that x can be among k greatest elements and in the worst-case, if x is not the k-th greatest element (suppose that in determining this fact it does not need any comparison as it is true for the case $k = 1$), then the algorithm must determine the $(k-1)$-th element of the set $X - \{x\}$, and so at least $V_{k-1}(n-1)$ comparisons are further needed. If A needs $V'_k(n)$ comparisons for computing the k-th largest element, then it holds

$$V'_k(n) \geqslant \lceil \log_2 (n-k+1) \rceil + V_{k-1}(n-1)$$

for $1 \leqslant k \leqslant n$, $V_0(n) = 0$. For $k = 3$ we get

$$V'_3(n) \geqslant n - 3 + \lceil \log_2(n-1) \rceil + \lceil \log_2 (n-2) \rceil .$$

Following Kirkpatrick [3] it holds

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

for $4.2^s < n-1 \leqslant 5.2^s$ and thus for infinitely many n the algorithm A is not optimal because it holds

$$V'_3(n) > V_3(n)$$

for $n = 4.2^s + 3$, ...

We conclude with remark that for certain problems solved by algorithms with fixed strategy the adversary argument cannot be efficiently exploited because the complexity of these problems can strongly depend on all problem parameters (as there are k and n in the case of the $V(k,n)$ problem).

References

1. Dobkin, D., Lipton, R., On the Complexity of Computations under Varying Sets of Primitives. Automata Theory and Formal Languages 2nd GI Conference, Lecture Notes in Computer Science 33, Springer-Verlag (1975), 110-117.

2. Knuth, D.E., The Art of Computer Programming, Vol. 3, Sorting and Searching, Addison-Wesley, Reading, Mass., 1973.

3. Fredman, M.L., How Good is the Information Theory Bound for Sorting? Theoretical Computer Science 1 (1976), 355-361.

4. Kirkpatrick, D.G., Topics in the Complexity of Combinatorial Algorithms. Department of Computer Science, University of Toronto, Technical Report No. 74, 1974.

5. Rabin, M.O., Proving Simultaneous Positivity of Linear Forms, JCSS 6 (1972), 639-650.

6. Reingold, E.M., Computing the Maxima and the Median, 12th Annual Symposium on Switching and Automata Theory (1971), 216-218.

7. Ružička, P., The Influence of Arithmetics on the Complexity of Comparison Problems, Computing Research Centre, Bratislava, Technical Report (1977), 10-19 (in Slovak).

8. Yao, A. Ch., On the Complexity of Comparison Problems Using Linear Functions, 16th Annual Symposium on Foundations of Computer Science,(1976), 85-89.