AD-A055 890 COLLEGE OF WILLIAM AND MARY WILLIAMSBURG VA DEPT OF M--ETC F/G 12/1
ON THE OPTIMALITY OF LINEAR MERGE.(U)
JUN 78 P K STOCKMEYER, F F YAO N00014-76-C-0673

UNCLASSIFIED TR-16 NL

| OF |
AD
A055 890

END
DATE
FILMED
8-78
DDC

FOR FURTHER TRAN

(12)

AD A055890

AD No.
DDC FILE COPY

On the Optimality of Linear Merge

Paul K. Stockmeyer and F. Frances Yao

DDC
RECEIVED
JUN 28 1978
F

Technical Report 16

June 1978

This document has been approved for public release and sale; its distribution is unlimited.

78 06 27 001

Unclassified
Security Classification

DOCUMENT CONTROL DATA - R & D

(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)

1. ORIGINATING ACTIVITY (Corporate author)

College of William and Mary
Department of Mathematics and Computer Science
Williamsburg, Virginia 23185

2a. REPORT SECURITY CLASSIFICATION: Unclassified

2b. GROUP

3. REPORT TITLE

(6) On the Optimality of Linear Merge. (9) Technical Rept.

4. DESCRIPTIVE NOTES (Type of report and inclusive dates)

Technical Report 16, June 1978

5. AUTHOR(S) (First name, middle initial, last name)

(10) Paul K. Stockmeyer and F. Frances Yao

6. REPORT DATE: Jun[illegible]78

7a. TOTAL NO. OF PAGES: 12

7b. NO. OF REFS: 5

8a. CONTRACT OR GRANT NO.: (15) N00014-76-C-0673, NSF-MCS-77-05313

8b. PROJECT NO.: NR 044-459

c.

d.

9a. ORIGINATOR'S REPORT NUMBER(S): Technical Report 16

9b. OTHER REPORT NO(S) (Any other numbers that may be assigned this report)

10. DISTRIBUTION STATEMENT

Approved for public release; distribution unlimited

(12) 14 p.

11. SUPPLEMENTARY NOTES: (14) TR-16

12. SPONSORING MILITARY ACTIVITY

Mathematics Program
Office of Naval Research
Arlington, Virginia 22217

13. ABSTRACT

Let $M(m,n)$ be the minimum number of pairwise comparisons which will always suffice to merge two linearly ordered lists of lengths m and n. We prove that $M(m,m+d) = 2m+d-1$ whenever $m \geq 2d-2$. This generalizes earlier results of Graham and Karp ($d = 1$), Hwang and Lin ($d = 2,3$), Knuth ($d = 4$), and shows that the standard linear merging algorithm is optimal whenever $m \leq n \leq \lfloor 3m/2 \rfloor + 1$.

DD FORM 1 NOV 65 1473 (PAGE 1)

S/N 0101-807-6801

Unclassified
Security Classification

78 06 27 001

AD A058 197

On the Optimality of Linear Merge

Paul K. Stockmeyer */
Department of Mathematics and Computer Science
College of William and Mary
Williamsburg, Virginia 23185

F. Frances Yao **/
Computer Science Department
Stanford University
Stanford, California 94305

Abstract. Let $M(m,n)$ be the minimum number of pairwise comparisons which will always suffice to merge two linearly ordered lists of lengths m and n. We prove that $M(m,m+d) = 2m+d-1$ whenever $m \geq 2d-2$. This generalizes earlier results of Graham and Karp $(d = 1)$, Hwang and Lin $(d = 2,3)$, Knuth $(d = 4)$, and shows that the standard linear merging algorithm is optimal whenever $m \leq n \leq \lfloor 3m/2 \rfloor + 1$.

ACCESSION for
NTIS White Section
DDC Buff Section
UNANNOUNCED
JUSTIFICATION
BY
DISTRIBUTION/AVAILABILITY CODES
Dist. Spec.
A

*/ Research supported by the Office of Naval Research under contract N00014-76-C-0673, NR044-459, and by the William and Mary faculty research leave program.

**/ This research was supported in part by National Science Foundation grant MCS-77-05313.

1. Introduction.

Suppose we are given two linearly ordered sets A and B consisting of elements

$$a_1 < a_2 < \ldots < a_m$$

and $b_1 < b_2 < \ldots < b_n$,

respectively, where the $m+n$ elements are distinct. The problem of merging these sets into a single ordered set by means of a sequence of pairwise comparisons is of obvious practical interest, and several algorithms have been devised for handling it.

An intriguing theoretical problem is to determine $M(m,n)$, the minimum number of comparisons which will always suffice to merge the sets in a decision tree model [5]. Evaluating this function in general seems quite difficult, and values are known for only a few special cases, including $m \leq 3$ ([1], [2], and [4]). In one direction, an upper bound for $M(m,n)$ is provided by a simple procedure variously referred to as the normal, standard, linear, or tape merge algorithm. Here the two smallest elements (initially a_1 and b_1) are compared, and the smaller of these is deleted from its list and placed on an output list. The process is repeated until one list is exhausted. It is easy to see that this algorithm requires $m+n-1$ comparisons in the worst case, so that

$$M(m,n) \leq n+m-1 \ .$$

Although better algorithms exist for many cases, R. L. Graham and R. M. Karp independently observed that this algorithm is optimal when $|n-m|$ is 0 or 1 . That is, they showed that

$$M(m,m) = 2m-1$$

and $M(m,m+1) = 2m$.

Later Hwang and Lin [3] proved that

$$M(m,m+2) = 2m+1 \quad \text{for} \quad m \geq 2$$

and $M(m,m+3) = 2m+2 \quad \text{for} \quad m \geq 4$,

while Knuth [5, p. 204] verified that

$$M(m,m+4) = 2m+3 \quad \text{for} \quad m \geq 6 .$$

In this paper we generalize these results by proving that

$$M(m,m+d) = 2m+d-1 \quad \text{for} \quad m \geq 2d-2 .$$

Intuitively, this means that the standard merge algorithm is optimal, in the worst-case sense, whenever $m \leq n \leq \lfloor 3m/2 \rfloor + 1$.

2. Oracles.

A lower bound for $M(m,n)$ will be produced by means of an "oracle", the proof technique utilized for example by Knuth [5, Section 5.3.2]. In his formulation, when presented with a comparison a_i vs. b_j , an oracle announces which is larger and simultaneously chooses a strategy for answering further questions so as to force a large number of additional comparisons to be made. A useful lower bound is obtained from an oracle that has an effective strategy for dealing with any comparison it might encounter.

In addition to an oracle that provides a lower bound for $M(m,n)$, oracles are needed to furnish lower bounds for two other functions. Let $/M(m,n)$ be the number of comparisons required to merge two lists for which, unknown to the merger, a_1 is in fact greater than b_1 . An oracle

for this function must therefore make all pronouncements consistent with $a_1 > b_1$. Similarly, let $/M\backslash(m,n)$ be the number of comparisons required when a_1 is greater than b_1 and a_m is less than b_n, again unknown to the merger. Occasionally we shall use the notation $M\backslash(m,n)$ to denote the number of comparisons required to merge two lists when a_m is less than b_n. This is not another new function, though, since by symmetry we have $M\backslash(m,n) = /M(m,n)$.

To illustrate these definitions, suppose $m = 2$ and $n = 4$. It is well known that $M(2,4) = 5$. However, there is a way to perform this merge in only 4 comparisons if in fact $a_1 > b_1$, by first comparing a_1 with b_2. If $a_1 > b_2$, the problem reduces to $M(2,2)$; otherwise, comparing a_1 with b_1 reduces the problem to $M(1,3)$. Thus $/M(2,4) \leq 4$.

3. An Example.

We illustrate the use of Knuth's oracles, and the strategies available to them, by verifying that $M(4,7) \geq 10$. Assume that oracles for achieving $M(m,n)$ and $/M(m,n)$ exist whenever $m+n \leq 10$ (see [5]). We consider four cases.

(i) First, suppose a merge algorithm begins by comparing a_1 with b_1. The oracle declares that $a_1 > b_1$, and requires that subsequent comparisons merge $\{a_1,a_2,a_3,a_4\}$ with $\{b_2,b_3,\ldots,b_7\}$, using an $M(4,6)$ oracle. Thus $M(4,7) \geq 1+M(4,6) = 1+9 = 10$ in this case.

(ii) If a merge algorithm begins by comparing a_1 with b_j, with $j \geq 2$, a more complex strategy is needed. The oracle declares that $a_1 < b_j$, and requires that later comparisons merge $\{a_1\}$ with $\{b_1\}$ and $\{a_2,a_3,a_4\}$ with $\{b_1,b_2,\ldots,b_7\}$, with the restriction that all future

pronouncements are consistent with $a_1 < b_1 < a_2$. These restrictions ensure that information gained in merging one subproblem is of no help in the other, even though b_1 is in both. The situation is illustrated in Figure 1. The top row is A, the bottom B, with smaller elements to the left. The dotted lines represent the restrictions the oracle imposes on itself, and the subproblems are encircled. With this strategy, the oracle can force at least $1 + M\backslash(1,1) + /M(3,7) = 1+1+8 = 10$ comparisons to be made in this case as well. Thus any algorithm which initially uses a_1 requires at least 10 comparisons.

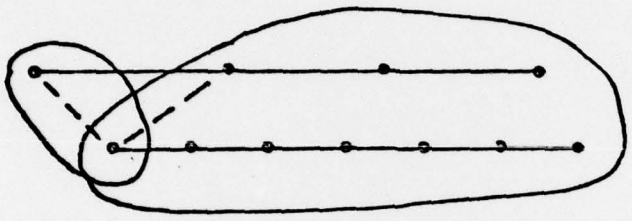

Figure 1. $a_1 < b_j$, $j \geq 2$.

(iii) An algorithm that first compares a_2 with b_j, with $j \leq 3$, can be handled in a manner similar to (ii). The oracle declares that $a_2 > b_j$ and requires that future comparisons merge $\{a_1\}$ with $\{b_1, b_2, b_3, b_4\}$ and $\{a_2, a_3, a_4\}$ with $\{b_4, b_5, b_6, b_7\}$, under the restrictions $a_1 < b_4 < a_2$. See Figure 2. The number of comparisons required in this case is at least $1 + M\backslash(1,4) + /M(3,4) = 1+3+6 = 10$.

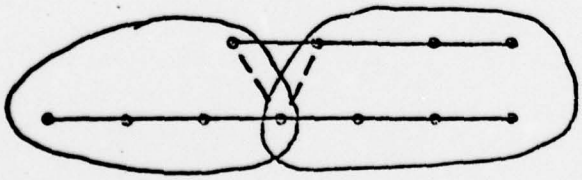

Figure 2. $a_2 > b_j$, $j \leq 3$.

(iv) If the first comparison is a_2 vs. b_j with $j \geq 4$, a simpler strategy will work. The oracle declares that $a_2 < b_j$, and insists that later comparisons merge $\{a_1, a_2\}$ with $\{b_1, b_2, b_3\}$ and $\{a_3, a_4\}$ with $\{b_4, b_5, b_6, b_7\}$ as in Figure 3. The number of comparisons required is at least $1 + M(2,3) + M(3,4) = 1 + 4 + 4 = 10$.

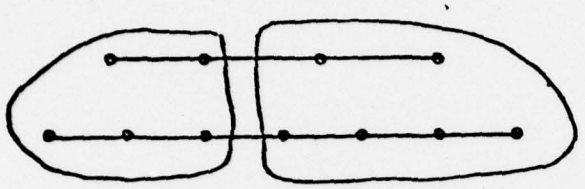

Figure 3. $a_2 < b_j$, $j \geq 4$.

We have shown that any merge algorithm that begins with a comparison using either a_1 or a_2 requires at least 10 comparisons for this problem. By symmetry, the same is true for a_3 and a_4. Having considered all cases, we conclude that $M(4,7) \geq 10$.

The two types of strategy illustrated above endow an oracle with sufficient power to prove our main result in the next section. In the "simple" strategy, the oracle answers the query and divides the merge problem into two disjoint unrestricted problems. In the "complex" strategy, there is an element of B in both subproblems, which are handled by suitably restricted oracles. Oracles for the functions $/M(m,n)$ and $/M\backslash(m,n)$ use the same strategies, with one or both subproblems inheriting the restrictions of the original. A subproblem may have one list empty in degenerate cases, as in case (i) above. In all cases, though, each subproblem contains fewer elements than the original problem, so that inductive proofs can be used.

4. The Main Result.

The proof of our theorem is simplified by first establishing a few preliminary results.

Lemma 1.

(i) $/M\backslash(m,n) \leq /M(m,n) \leq M(m,n)$.

(ii) $/M(m+1,n+1) \geq /M(m,n)+2$.

(iii) $/M\backslash(m+1,n+1) \geq /M\backslash(m,n)+2$.

Proof. Part (i) is obvious; any merge algorithm performs at least as well on more restricted problems. In part (ii), an oracle for $/M(m+1,n+1)$ can make all pronouncements consistent with $b_1 < a_1 < b_2 < a_2$, and force $\{a_2, a_3, \ldots, a_{m+1}\}$ to be merged with $\{b_2, b_3, \ldots, b_{n+1}\}$. Then the comparisons a_1 vs. b_1 and a_1 vs. b_2 can not be avoided. The proof of part (iii) is similar.

We are now ready to prove the main result. Although we are really interested only in part (a), bounds for all three functions must be proved simultaneously, as each oracle requires the help of at least one other.

Theorem 1.

(a) $M(m,m+d) \geq 2m+d-1$ for $m \geq 2d-2$

(b) $/M(m,m+d) \geq 2m+d-1$ for $m \geq 2d-1$

(c) $/M\backslash(m,m+d+2) \geq 2m+d$ for $m \geq 2d-1$.

Proof. If (b) and (c) are true for the threshold values $m = 2d-1$, then they are also true for $m > 2d-1$ by repeated application of Lemma 1 (ii) and (iii). Also, if (b) is true for $m \geq 2d-1$ then Lemma 1 (i) implies that (a) is also true for $m \geq 2d-1$. Thus it is sufficient to prove the

theorem for the threshold values of m only, that is,

$$M(2d-2,3d-2) \geq 5d-5 ,$$

$$/M(2d-1,3d-1) \geq 5d-3 ,$$

and $$/M\backslash(2d-1,3d+1) \geq 5d-2 .$$

The proof is by induction on d. The starting values for $1 \leq d \leq 3$ are given in Knuth [5, p. 203].

Part (a). Suppose an algorithm begins by comparing a_i with b_j, where $i = 2k-1$ and $j \leq 3k-2$, for some integer k satisfying $1 \leq k < d$. The oracle proclaims that $a_i > b_j$ and follows the simple strategy, yielding

$$\begin{aligned} M(2d-2,3d-2) &\geq 1 + M(2k-2,3k-2) + M(2(d-k),3(d-k)) \\ &\geq 1 + (5k-5) + (5(d-k)-1) \\ &= 5d-5 . \end{aligned}$$

If $i = 2k-1$ and $j \geq 3k-1$, the oracle announces that $a_i < b_j$ and uses the complex strategy, with b_{3k-2} in both subproblems. This leads to

$$\begin{aligned} M(2d-2,3d-2) &\geq 1 + M\backslash(2k-1,3k-2) + /M(2(d-k)-1,3(d-k)+1) \\ &\geq 1 + /M(2k-1,3k-2) + /M\backslash(2(d-k)-1,3(d-k)+1) \\ &\geq 1 + (5k-4) + (5(d-k)-2) \\ &= 5d-5 . \end{aligned}$$

This settles the case where i is odd. Reversing the order of the elements in A and B maps all points of A with even subscripts onto those with odd. Thus by symmetry we have handled the even case as well.

Part (b). Suppose the first comparison of an algorithm is a_i vs. b_j with $i = 2k-1$ and $j \leq 3k-2$, where $1 \leq k \leq d$. The oracle proclaims that $a_i > b_j$ and uses the complex strategy, with b_{3k-1} in both subproblems. In this case we have

$$/M(2d-1,3d-1) \geq 1 + /M\(2k-2,3k-1) + /M(2(d-k)+1,3(d-k)+1)$$

$$\geq 1 + (5k-5) + (5(d-k)+1)$$

$$= 5d-3 .$$

If $i = 2k-1$ and $j \geq 3k-1$, the oracle announces that $a_i < b_j$. The simple strategy yields

$$/M(2d-1,3d-1) \geq 1 + /M(2k-1,3k-2) + M(2(d-k),3(d-k)+1)$$

$$\geq 1 + (5k-4) + 5(d-k)$$

$$= 5d-3 .$$

Now suppose $i = 2k$ and $j \leq 3k$, with $1 \leq k < d$. Choosing $a_i > b_j$, the oracle follows the complex strategy, leading to

$$/M(2d-1,3d-1) \geq 1 + /M\(2k-1,3k+1) + /M(2(d-k),3(d-k)-1)$$

$$\geq 1 + (5k-2) + (5(d-k)-2)$$

$$= 5d-3 .$$

Otherwise, if $i = 2k$ and $j \geq 3k+1$, the simple strategy with $a_i < b_j$ produces

$$/M(2d-1,3d-1) \geq 1 + /M(2k,3k) + M(2(d-k)-1,3(d-k)-1)$$

$$\geq 1 + (5k-1) + (5(d-k)-3)$$

$$= 3d-3 .$$

Part (c). Assume an algorithm begins a_i vs. b_j with $i = 2k-1$ and $j \leq 3k-1$, where $1 \leq k \leq d$. The oracle picks $a_i > b_j$ and follows the simple strategy, yielding

$$\begin{aligned} /M\backslash(2d-1,3d+1) &\geq 1 + /M(2k-2,3k-1) + M\backslash(2(d-k)+1,3(d-k)+2) \\ &\geq 1 + /M\backslash(2k-2,3k-1) + /M(2(d-k)+1,3(d-k)+2) \\ &\geq 1 + (5k-5) + (5(d-k)+2) \\ &= 5d-2 \ . \end{aligned}$$

The case $i = 2k-1$ and $j \geq 3k$ is the mirror image of this case.

If $i = 2k$ and $j \leq 3k+1$, with $1 \leq k < d$, the simple strategy works again. The oracle declares $a_i > b_j$, and we have

$$\begin{aligned} /M\backslash(2d-1,3d+1) &\geq 1 + /M(2k-1,3k+1) + M\backslash(2(d-k),3(d-k)) \\ &\geq 1 + /M\backslash(2k-1,3k+1) + /M(2(d-k),3(d-k)) \\ &\geq 1 + (5k-2) + (5(d-k)-1) \\ &= 5d-2 \ . \end{aligned}$$

Finally, the case $i = 2k$ and $j \geq 3k+1$ is contained in the mirror image of this case.

In conclusion, we note that Knuth [5, p. 206] has made several conjectures concerning the behavior of $M(m,n)$, such as

$$M(m+1,n+1) \geq M(m,n)+2 \ .$$

In view of Theorem 1, it seems reasonable to add

$$M(m+2,n+3) \geq M(m,n)+5$$

to the list.

Also, it would be interesting to know the precise range of m and n for which the linear merge algorithm is optimal. No instances have been found outside the range $m \leq n \leq \lfloor 3m/2 \rfloor + 1$, but cases as small as $m = 7$, $n = 12$ remain open.

References

[1] R. L. Graham, "On sorting by comparisons," in Computers in Number Theory (A. O. L. Atkin and B. J. Birch, editors), Academic Press, London, 1971, pp. 263-269.

[2] F. K. Hwang, "Optimal merging of 3 elements with n elements," to be published.

[3] F. K. Hwang and S. Lin, "Some optimality results in merging two disjoint linearly-ordered sets," Bell Telephone Laboratories internal memorandum, 1970.

[4] F. K. Hwang and S. Lin, "Optimal merging of 2 elements with n elements," Acta Informatica 1 (1971), pp. 145-158.

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

14 KEY WORDS	LINK A		LINK B		LINK C	
	ROLE	WT	ROLE	WT	ROLE	WT
MERGE OPTIMALITY LINEAR MERGE						